Through analogy, novel situations and problems can be understood in terms of familiar ones. There is converging evidence that analogy-making lies at the very core of human cognition. Conversely, successful analogy-making requires the resources of an entire cognitive architecture. This book describes a computational model of analogy-making called AMBR (Associative Memory-Based Reasoning). AMBR is based on a hybrid symbolic-connectionist multi-agent cognitive architecture called DUAL. Macroscopic behavior in DUAL emerges from the interactions of simple processing agents in dynamic coalitions. Unlike the mainstream models of analogy-making, AMBR uses a decentralized representational scheme for problems and situations. The dynamic emergent processing of these decentralized representations is consistent with the context-sensitive and constructive nature of human memory. Both DUAL and AMBR were developed by Boicho N. Kokinov and his graduate students at New Bulgarian University. This book is a revised and expanded version of the author's Ph.D. thesis written under Prof. Kokinov's supervision at NBU. It will be of interest to cognitive modelers and cognitive scientists more generally.

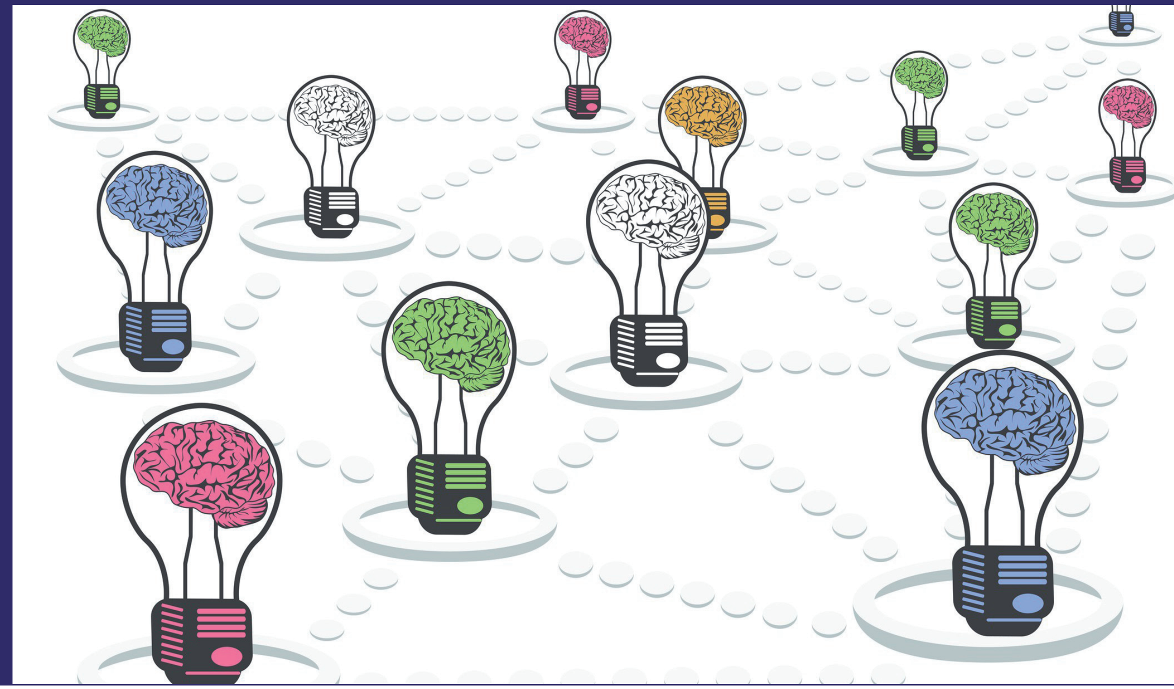
Alexander A. Petrov

Alexander A. Petrov is Director of the Laboratory for Cognitive Modeling and Computational Cognitive Neuroscience, and Associate Professor of Psychology at the Ohio State University, USA. He has over 30 peer-reviewed publications on models of analogy-making, perceptual learning, neural networks, and other topics. http://alexpetrov.com

978-3-659-26248-7

AMBR: Associative Memory-Based Reasoning

Petrov



Alexander A. Petrov

Associative Memory-Based Reasoning

A Computational Model of Analogy-Making in a Decentralized Multi-Agent Cognitive Architecture

LAP LAMBERT Academic Publishing

2013. Saarbrücken, Germany.

Alexander A. Petrov

Associative Memory-Based Reasoning

Alexander A. Petrov

Associative Memory-Based Reasoning

A Computational Model of Analogy-Making in a Decentralized Multi-Agent Cognitive Architecture

LAP LAMBERT Academic Publishing

Impressum / Imprint
Bibliografische Information der Deutschen Nationalbibliothek: Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

Bibliographic information published by the Deutsche Nationalbibliothek: The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at http://dnb.d-nb.de.

Coverbild / Cover image: www.ingimage.com

Verlag / Publisher:
LAP LAMBERT Academic Publishing
ist ein Imprint der / is a trademark of
OmniScriptum GmbH & Co. KG
Heinrich-Böcking-Str. 6-8, 66121 Saarbrücken, Deutschland / Germany
Email: info@lap-publishing.com

Herstellung: siehe letzte Seite /
Printed at: see last page
ISBN: 978-3-659-26248-7

Zugl. / Approved by: Sofia, New Bulgarian University, Ph.D. thesis, 1998

In loving memory of Boicho Kokinov
who introduced me to cognitive science
and taught me how to be a scientist

In Memoriam

Boicho Kokinov (1960–2013)

This book is dedicated to my teacher, Ph.D. advisor, and dear friend Boicho Kokinov. He is the author of the AMBR model described in the book and was working on the Afterword at the time of his sudden and premature death. The fatal blow that stopped Boicho's noble hearth also opened a deep wound in the hearts of the hundreds of people whose lives were enriched by this remarkable man.

Boicho Nikolov Kokinov was a Bulgarian scientist who led the development of cognitive science in Eastern Europe after the end of the Cold War. He was:

- Co-founder and president of the Bulgarian Cognitive Science Society
- Member of the governing board of the [worldwide] Cognitive Science Society
- Co-founder and vice-rector of New Bulgarian University (NBU)
- Co-founder and director of the Department of Cognitive Science and Psychology at NBU

- Co-founder and director of the Central and East European Center for Cognitive Science at NBU
- Founder and director of the Annual International Summer School in Cognitive Science at NBU
- Co-organizer of numerous conferences in workshops in Bulgaria and Europe
- Member of the editorial board of various scientific journals and grant panels.

In addition, he has authored over 70 peer-reviewed publications, edited several books, and advised over a dozen graduate students.

Boicho was a deep and original thinker. This book describes one of his scientific contributions. His true passion, however, was building communities of people. He cared deeply about the common good and had a remarkable gift to inspire everyone who met him. Boicho was a great teacher and an effective leader who worked tirelessly to reform the higher-education system in Bulgaria and to establish and promote cognitive science in Eastern Europe.

The focal point of this effort is New Bulgarian University—an institution to which Boicho devoted 20 years of his career. He played a major role in the establishment and accreditation of NBU as a whole and especially the Department of Cognitive Science and Psychology and the associated Central and East European Center for Cognitive Science. Founded in 1991, New Bulgarian University introduced for the first time in Bulgaria a model of higher education that emphasizes student choice, flexible program requirements based on credit hours, and close integration into the international academic community. These features are widely used in Western Europe and North America but contrast sharply with the rigid curricula of the traditional state-owned Bulgarian universities. Boicho was a vocal and eloquent proponent of the new system and under his leadership the Department of Cognitive Science and Psychology became (and still is) one of the strongest and most successful departments at NBU.

The Annual International Summer School in Cognitive Science showcases Boicho's exceptional community-building skills. He organized the first such school in 1994 with no funding and minimal institutional support from an immature university that did not even have its own building at the time. Through his personal connections and with no resources other than his individual charisma and the sheer power of persuasion, Boicho convinced a few prominent scientists to travel to Sofia at their own expense, pay for their hotel and food, and teach an intensive course for free! I was among the students at this first school and remember vividly the sense of excitement we had at the opportunity to partake in science at this highest level. For their part, the instructors said they had never met such an enthusiastic group of students and that our interactions were very stimulating for them too. And so the idea caught on, carefully nurtured by Boicho's astute leadership. With each consecutive year, the prestige of the summer school—and that of the Cognitive Science Center at NBU with it—grew steadily. The list of speakers swelled to a virtual catalog of the preeminent figures in cognitive science. For example, as of 2013 it includes four Rumelhart Prize winners. Thus people are only half-joking when they say, "Now that I am invited to the Bulgarian Summer School, I know I have finally made a name for myself."

Boicho was beloved by all. That is why the sudden news of his fatal heart attack on May 10th, 2013, sent shockwaves not only at NBU but throughout the cognitive science community worldwide. An email avalanche ensued, expressing first disbelief, then shock, and finally sadness and appreciation of his contributions. Commemorative sessions were held at the Cognitive Science Conference in Berlin and the Analogy Conference in Dijon later that year. As the study of analogy-making had been a major focus of Boicho's research interests throughout his career, his departure was felt especially strongly in the tightly knit community of analogy researchers. He had published extensively on the topic, edited several collections of articles, and hosted both the First (1998) and Second (2009) International Conferences on Analogy. All analogy researchers knew him well and—consequently and predictably—liked him and were inspired by him. The Third (2013) Conference was dedicated to him by an unanimous decision of the Program Committee. At the meeting itself, nearly all speakers began their presentations by sharing memories and expressing their admiration for him. The conference ended with two minutes of silence in his honor. Boicho is sorely missed by all.

New Bulgarian University honored his beloved co-founder with the academic equivalent of a state funeral. Hundreds of people attended the memorial service and there was an outpouring of sadness, admiration, gratitude, and recognition of Boicho's achievements and personality. The entire university leadership was in attendance, including the Rector, all Deans, and many members of the Governing Board. The Department of Psychology cancelled all classes on that day. Practically all faculty members and many staff members attended the service. There were also at least 150 students—graduate and undergraduate, past and present. United in their grief and heartfelt love for Boicho, it was not uncommon for students to cry on the shoulders of their professors and vice versa. Everybody without exception expressed their gratitude and admiration for Boicho. Remarkably many people stated in no uncertain terms that their lives had been transformed by him in one way or another, always for the better.

I count myself in that number. My life was profoundly affected by my apprenticeship and friendship with Boicho. It was from his lips that I heard the phrase "cognitive science" for the first time in my life, approximately 20 years ago. Had it not been for him, I might well have ended up in some other scientific discipline or even left academia altogether. He taught me how to be a scientist and, more importantly, how to be an upright person. Boicho's legacy includes dozens of students with stories similar to mine. Many of his former students are now faculty at NBU or various universities throughout Europe, US, and Canada. His many colleagues and collaborators were affected in no less profound ways.

One of the special gifts that Boicho bestowed upon me was the opportunity to observe and absorb his attitude towards life. He always strove to make the world a better place and to help his fellow travelers along the way. Where other people would complain, make excuses or accusations, Boicho would always say, "Let's do something about that. Will you help me?" The effects were spectacular. Deep down, this is why so many people loved him so much. He was always there on the front line, ready to help, ready to shoulder most of the burden. This is what made him such an effective and inspiring leader.

Even though his life was cut tragically short, he accomplished more in 52 years than most people ever accomplish. It is absolutely, undisputedly true that Boicho left the world a far better place than he found it. NBU, the cognitive science community, and hundreds of individuals are in his debt.

His was a life well lived.

Rest in peace, dearest Boicho. We will carry the torch forward and strive to be worthy of your generous gifts.

Sofia, Bulgaria *Alexander A. Petrov*
August 2013

Preface

This book is based on my Ph.D. thesis, which was completed in July 1998 under the supervision of Dr. Boicho Kokinov at the New Bulgarian University (NBU) in Sofia, Bulgaria. The book describes a cognitive model of analogy-making developed in an effort to understand the mental processes that take place when a person perceives one situation as structurally similar to another. The model is called AMBR and is based on a hybrid symbolic-connectionist cognitive architecture called DUAL. Both DUAL and AMBR were proposed by Boicho Kokinov in the late 1980s and are still in active development. I joined the AMBR research group in the early 1990s as a graduate student in cognitive science at NBU.

I could not have asked for a better doctoral advisor than Boicho. He lavished his time and attention on me throughout my studies at NBU. We had countless conversations, discussing every single idea in this book. DUAL and AMBR are Boicho's creations and he should have been listed as first author. With his usual grace and modesty, however, he declined to have his name appear on the title page. This book is dedicated to him in appreciation of his mentorship, friendship, and support. Unfortunately, Boicho did not live long enough to see the published book but he did see the complete draft.

This book documents the status of the AMBR project as of 1998. I have resisted the temptation to revise the original dissertation text too much, although I did smooth out the roughest edges, particularly in Chapter 2 and Section 5.7. A new Afterword traces the development of DUAL and AMBR after 1998, whereas a lengthy new postscript to Chapter 2 provides pointers to the recent analogy-modeling literature at large. Appendix C is also new. The number of bibliographic references has more than tripled. Of course all faults in the text are entirely my responsibility. Corrections for errors discovered after the book goes to print will be posted on my personal web site, http://alexpetrov.com.

Although my research took a different turn after my graduation from NBU, I always kept a keen interest in analogy-making and followed the developments in this field. Cognitive science has advanced considerably in the intervening 15 years. Cognitive neuroscience in particular has made great strides and today we can incorporate much stronger neurological constraints into our models compared to 1998.

I am currently developing an analogy-making model that is a neural network quite different from AMBR. But this is a story for another book.

This is a book about AMBR — a treasure trove of Boicho's original ideas. AMBR definitely deserves careful study and is of potential interest to cognitive modelers and cognitive scientists more generally. Graduate students can use the text as an example of one possible way to write a Ph.D. thesis. Current members of the AMBR research group may be interested to learn about a precursor of their work.

I give heartfelt thanks to all who contributed to the work described here. To Boicho, first and foremost and always — Thank you! I am forever in your debt and I will cherish the memory of you as long as I live. Rest in peace, my dear friend. To my professors and colleagues at the Central and Eastern European Center for Cognitive Science at NBU, who taught me so much and supported me in every way. As they are too many to enumerate here in full, I will single out just two names — Encho Gerganov and Vassil Nikolov. To Bob French, whose book about the Tabletop model was never far from my desk while I was writing my thesis. Thank you, Bob, for the stimulating discussions and for your hospitality in the summer of 1996. To the members of my Dissertation Committee, who made time in their busy schedules during the First Analogy Conference in July 1998 to read and comment upon a long thesis — Kenneth Forbus, Dedre Gentner, Keith Holyoak, John Hummel, Pentti Kanerva, Zdravko Markov, and Naum Yakimoff (chair). Thank you for your good will and encouragement. Your feedback and recommendations are now incorporated into the text. To Alexander Doumas, Keith Holyoak, John Hummel, Andrew Lovett, and Robert Morrison, for their critical comments on the Postscript to Chapter 2. To Georgi Petkov, who rose to the occasion and wrote the Afterword that Boicho meant to write but could not. To my mother and in loving memory of my father, who nurtured me with constant devotion and who were my first and most important teachers. Last but not least, very special thanks to my wonderful wife Petya and my adorable daughter Vicky.

Columbus, Ohio, USA *Alexander A. Petrov*
October 2013

Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Motivation

The key intuition underlying the research presented in this dissertation is that the mechanisms giving rise to human analogy-making are central to cognition. Analogy is not just a specific technique for problem solving and argumentation occasionally called upon when the more reliable methods such as deduction and proof do not work. As we view it, analogy is a manifestation of the fundamental cognitive ability to relate new information to old knowledge and to flexibly manipulate both until they fit into a harmonious whole. As such, it highlights a number of issues that are absolutely central to cognition in general—the organization of memory, manipulation of complex structured representations, dynamic relevance, flexible allocation of resources, perception and categorization, generalization, learning, etc. Research on analogy, therefore, transcends the boundaries of the specific phenomenon and goes deeply into the core of intelligence.

The main instrument for the research presented in this book in the methodology of cognitive modeling. The aim is to analyze analogy-making in computational terms and to construct a working artifact in the form of a computer program. The behavior of the model is then compared to empirical data collected by psychological experimentation. The criterion for success is whether the model contributes to our theoretical understanding of the hidden mechanisms of human cognition.

This book describes a computational cognitive model called AMBR (*Associative Memory-Based Reasoning*). It provides a detailed account of its mechanisms and demonstrates its operation by reporting the results of numerous simulation experiments performed with a computer implementation of the model. Throughout the book, an attempt has been made to formulate the implications of AMBR for our understanding of human cognition as well as to compare it to other models presented in the literature.

The research reported here is part of a larger research project launched by Boicho Kokinov approximately ten years ago (Kokinov, 1988, 1990, 1994a; Kokinov, Nikolov, & Petrov, 1996; Kokinov & Hadjiilieva, 1997; Kokinov, 1998). The long-term goal of the project is to give a unified account of deductive, inductive, and analogical reasoning by realizing them with the same set of mechanisms.

As analogy is in a sense representative of cognition in general, a model of analogy-making should be based on a full-fledged cognitive architecture. We do not expect that a small "analogy machine" based on a few simple assumptions could explain such complex phenomenon. Neither do we expect that this could be done by some all-encompassing "magic formula." Instead, we conceptualize analogy as an emergent product of the collective effort of many interdependent mechanisms. The claim is that these same mechanisms can be used for other cognitive tasks too. As a consequence, modeling analogy-making requires a solution to a number of issues about knowledge representation, organization of memory, allocation of computational resources, perception, etc.

AMBR is based on a cognitive architecture that is a first step towards this very distant goal. The architecture DUAL (Kokinov, 1994a,b,c) provides a framework for building dynamic emergent computational models of cognitive phenomena. AMBR is one such model.

1.2 Main Ideas of Dual and Ambr

1.2.1 Overview of Dual

DUAL is a general-purpose cognitive architecture that comprises a unified description of mental representation, memory structures, and processing mechanisms. All these aspects of the architecture are organized around the principles of hybridity, emergent computation, dynamics, and context sensitivity.

DUAL is hybrid—it consists of complementary aspects. Moreover, it is hybrid in two ways. On one hand, it hinges upon the symbolic/connectionist distinction and the integration between the two. On the other, there is the declarative/procedural distinction and integration thereof. DUAL is also emergent, dynamic, and context-sensitive. All processing and knowledge representation in the architecture is carried out by small entities called *Dual agents*. There is no central executive that controls the whole system, allocates resources, resolves conflicts, etc. Instead, there are small-scale DUAL agents and local interactions between them. The global behavior of the system emerges from the self-organizing pattern of these interactions. An important feature of DUAL's operation is that it is constantly changing in response to influences from the environment. This is possible due to the emergent nature of the processing and the lack of rigid centrally imposed algorithm.

In a little more detail, each DUAL agent is a hybrid entity serving both representational and processing purposes. Each agent is relatively simple and has access only to local information, interacting with a few neighboring agents. It has a *micro-frame* storing declarative and procedural knowledge. Its *symbolic processor* can perform simple manipulation on symbols (discrete compositional entities) and to pass them to other agents. The complementary aspect of the processor is engaged in spreading *activation* (continuous additive quantity) between agents. Thus they can also be conceptualized as nodes in a network.

The *speed* of the symbolic processing performed by a given DUAL agent depends on its *activation level*. Active agents work rapidly, less active agents work slowly, and inactive agents do not work at all. In this way, each agent contributes to the overall computation in the system to a different extent. As activation levels change continuously, the speed of the symbolic processing changes accordingly. This is a key factor for the dynamic emergent computation that is characteristic of DUAL.

The *long-term memory* of the architecture consists of the total population of all permanent DUAL agents. The active subset of them plus some temporary agents constitute the *working memory* of the system. The contents of the working memory changes dynamically, reflecting changes in the environment and the internal course of computation. This is another factor for flexibility and context-sensibility.

1.2.2 Main Ideas of Ambr

AMBR is a dynamic emergent model built on the basis of DUAL. In its general form it is conceived as an integrated model of deductive, inductive, and analogical reasoning (Kokinov, 1988). All three kinds of reasoning are viewed as slightly different versions of a single uniform reasoning process. The overall approach is that reasoning establishes correspondences between two problems, schemes, or situations, and transfers some elements from one to the other, with due modification. The model explains deduction, induction, and analogy in terms of the relationships between the two descriptions that happen to be put in correspondence in each particular case. In this way, analogy can be viewed as the most general case, with deduction and generalization at the two extremes—where the *source* and the *target* are related in a special way, one of them being a specific instance of the other.

The research reported in this book concentrates on analogy-making. Therefore, AMBR is presented and discussed here as a model of analogy-making regardless of the fact that some of the considerations may have broader scope.

The models of analogy-making typically decompose it into separate "stages" or "phases." For example, one possible decomposition includes: (*i*) representation of the target problem, (*ii*) retrieval of a source analog from memory, (*iii*) mapping the two descriptions, (*iv*) transfer from the source to the target, (*v*) evaluation of the analogical inferences, and (*vi*) learning and generalization. Some researchers (e.g., Forbus et al., 1998; Gentner, 1989) argue that the stages of analogy-making are relatively independent and thus are susceptible to piecemeal exploration. Others (e.g., Chalmers, French, & Hofstadter, 1992) oppose this view claiming that the process of analogy-making is inseparable in principle because of the high degree of interdependence among its components.

AMBR agrees with the second position. In this model the components of analogy-making are conceptualized as *subprocesses* that overlap in time and influence each other. The long-term goal of the AMBR project is to develop an *integrated* model of all these subprocesses on the uniform foundation of DUAL. At present, however, only two of them are implemented in detail. The version of AMBR that is reported in this book is an integrated model of analogical access and mapping. These two subprocesses and the computational mechanisms that implement them in the model are discussed in detail. Special emphasis is put on the ways that they can interact and on the dynamic emergent nature of the computations.

Another feature of AMBR that is central to this book is that the model uses *decentralized representations of situations*.[1] Each DUAL agent is relatively simple and cannot represent much. Therefore, a whole *coalition* of agents is needed for the representation of each episode, schema, or even proposition. AMBR coalitions are emergent and have fuzzy boundaries. The members of a given coalition can come in or out of it dynamically and to participate in it with varying intensity. There is no centralized data structure enumerating all agents belonging to a coalition. This allows for greater flexibility and integration of the various subprocesses of analogy-making. In particular, the mapping process can begin before the whole coalition is accessed from memory. The correspondences established by the active elements of a situation can then influence the activation of their coalition partners. As a consequence, the episodes that better map to the target tend to be preferentially accessed. This organization has a number of advantages that are discussed in the book.

The current version of the model relies on six computational mechanisms to carry out the tasks within its scope. These are: *(i) spreading activation, (ii) marker passing, (iii) constraint satisfaction, (iv) structure correspondence, (v) rating, and (vi) skolemization*. Each of them serves a concrete function in the model. Spreading activation defines the working memory of the system, provides dynamic estimates of the relevance of each item, serves as a power supply for the symbolic processing, and underlies the relaxation of the network constructed by the constraint satisfaction mechanism. Marker passing is used for assessing semantic similarity, inheritance of properties, and carries out various information needed by other mechanisms. It also provides justifications for some of the hypotheses used by the constraint satisfaction mechanism. The latter underlies the process of mapping two structured descriptions and is a major instrument for achieving global consistency of the local activities in the model. The structure correspondence mechanism provides additional justifications for new hypotheses and dynamically modifies the topology of the constraint satisfaction network. The rating mechanism is responsible for promoting winner correspondences and for elimination of losers. Finally, skolemization uses general semantic knowledge to augment the description of a situation upon necessity.

[1] This term should not be confused with the *distributed representations* in neural networks.

1.3 The Ambr Family

Ambr is not the name of a single model but a generic name of a whole succession of models. Each of them builds upon the previous one and takes a few steps further in the long-term project. The major milestones along this road are the following:

Kokinov (1988) puts forth the conjecture that deduction, induction, and analogy can be conceptualized as different manifestations of a uniform reasoning process and gives it the name *Associative Memory-Based Reasoning.* An associative mechanism using spreading activation is proposed for the purposes of memory retrieval and estimation of relevance. The knowledge representation scheme is detailed in (Kokinov, 1989).

Kokinov (1994a) presents a much more elaborate version of AMBR. It will be denoted AMBR1 when reference to the particular version is important. It adopts Holyoak and Thagard's (1989) constraint satisfaction mechanism for the purposes of the mapping process. Unlike its precursor, however, the constraint satisfaction network (CSN) in AMBR1 is constructed dynamically by the joint operation of the marker passing and structure correspondence mechanisms. The CSN is integrated with the long-term memory of the model, which allows for interactions between the different subprocesses in analogy-making. AMBR1 uses centralized representation of situations—there is a frame containing a slot for each situation element.

Kokinov (1994a,b,c) also singles out the architecture DUAL as something different from the specific model AMBR. The main architectural principles of DUAL are established: multi-agent approach (Minsky, 1986), lack of central executive, hybridization at the micro-level, dynamic emergent computation, context sensitivity, etc. There is a computer implementation of the architecture and the model. It is used for simulation experiments.

In a M.Sc. thesis supervised by Boicho Kokinov, Petrov (1997) develops a detailed specification of DUAL and resolves some ambiguities of the original proposal. An exact and general mechanism for determining symbolic processor's speed on the basis of the activation level is specified. The connectionist aspect is identified as an energy supplier for the symbolic one. The notion of coalitions and the meso-level of description are explicated. A new portable implementation of the architecture is developed in Common Lisp (Steele, 1990) with CLOS (Keene, 1989).

There are improvements of AMBR too. This version of the model (Petrov, 1997) is denoted AMBR2A. It introduces decentralized representations of episodes and designs the machinery for maintaining them. In particular, there are *secretaries* that register the hypotheses for each element and assist the construction of the constraint satisfaction network. The activation function of AMBR1 is replaced with a better one. The knowledge base is expanded considerably and more extensive simulation experiments are performed.

Petrov's (1998) Ph.D. thesis extends the model further. The 1998 version[2] is denoted AMBR2 and is the one described in this book. The machinery for analogical

---

[2] When it is important to differentiate between the 1997 and 1998 versions, they are denoted AMBR2A and AMBR2B, respectively. Typically, however, we denote them collectively as AMBR2.

mapping is specified in full. The mapping process can identify winner correspondences, thus setting the stage for the transfer process. New mechanisms for using semantic knowledge for augmenting the description of episodes are added to the model. The existing mechanisms and the computer implementation are improved and extended. The knowledge base is enlarged considerably, both by elaborating the existing descriptions and by adding new concepts and episodes. This larger knowledge base is used for new simulation experiments.

1.4 Outline of the Book

The subsequent chapters are summarized as follows:

Chapter 2 — *Background* — reviews some empirical data about analogy-making reported in the literature. It also presents briefly a selection of models and discusses their strengths and weaknesses.

Chapter 3 — AMBR *in Broad Strokes* — presents a concise and relatively self-contained description of the cognitive architecture DUAL and the model AMBR2.

Chapter 4 — *Knowledge Representation* — describes the knowledge representation scheme in detail. It contrasts the advantages and disadvantages of centralized and decentralized representation of situations. It also introduces the domain used for the simulation experiments.

Chapter 5 — AMBR *Mechanisms at Work* — provides a rigorous and systematic description of current AMBR mechanisms. The operation of the model is illustrated on a concrete example by showing how the mechanisms apply to a particular target problem. The chapter contains diagrams and transcripts from actual program runs.

Chapter 6 — *Simulation Experiments* — reports results of simulation experiments involving ten target problems and more than 1200 runs of the program. These data are used to compare qualitatively the performance of AMBR with the regularities observed in human analogy-making.

Chapter 7 — *Possibilities for Future Extensions of* AMBR — discusses the limitations of the current version and suggests ways in which the model could be extended in the future. In particular, it gives some ideas about modeling the subprocess of analogical transfer. Also, it introduces a research project aimed at adding perceptual capabilities to DUAL and AMBR. Finally, it presents the TEXTSCREEN micro-domain that can be used as a testbed for this project.

Chapter 8 — *Conclusion* — concludes the book with a summary of its main points and a discussion of the contributions of this project.

A new *Afterword* traces the AMBR development after 1998.

Appendix A provides a sample of full-fledged agent definitions.

Appendix B gives simplified propositional representations of all episodes used in the experiments.

Appendix C describes the exact relationship between symbolic speed and connectionist activation in DUAL. It also outlines the “suspendable” extension of Lisp that we developed to implement variable-speed symbolic computations.

Chapter 2
Background

2.1 The Phenomenon of Analogy

Analogy has been the focus of much cognitive research. (For reviews[1] see Gentner, 1989; Goshwami, 1992; Holyoak & Thagard, 1995; Keane, 1988). Still, there is no universally accepted definition. Michalski (1989) explained analogy as a superposition of induction and deduction. By contrast, Holyoak and his collaborators (Gick & Holyoak, 1983; Holyoak & Thagard, 1995; Hummel & Holyoak, 1996) considered schema induction as a consequence of a successful analogy. There are, however, some ideas that have received widespread support. The following excerpt from Gentner (1989, p. 201) provides a starting point:

> Analogy is a mapping of knowledge from one domain (the base) into another (the target), which conveys that a system of relations that holds among the base objects also holds among the target objects. Thus, an analogy is a way of focusing on relational commonalties independently of the objects in which those relations are embedded.

The importance of *structure*, or system of relations, has been demonstrated in many studies (Gentner & Landers, 1985; Gentner & Toupin, 1986; Clement & Gentner, 1991). Objects from the two situations are seen as counterparts when they fulfill similar roles in the respective relational structure. The degree of this structural overlap or *quasihomomorphism* (Holland et al., 1986; Holyoak & Thagard, 1989) determines to a large extent the soundness of an analogy. Central to the mapping process is the *principle of systematicity*: People prefer to map connected systems of relations governed by higher-order relations with inferential import rather than isolated predicates (Gentner, 1983, 1989).

Analogy-making involves a *mapping process* that aligns structured descriptions of the two episodes and establishes a set of correspondences. There does not need to be any resemblance between individual elements of the two descriptions. Various theorists have suggested, however, and empirical evidence confirms, that object

---

[1] Most references in this chapter are to sources published prior to 1998. Section 2.2.6 provides pointers to the more recent literature.

and predicate *semantic similarity* influence the mapping process, with high similarity leading to greater ease of mapping (Gentner & Toupin, 1986; Holyoak & Koh, 1987; Ross, 1987). This is especially clearly seen when objects and roles are "cross mapped" (Gentner & Toupin, 1986; Ross, 1987, 1989).

Semantic similarity is much more important for *analog access*—the process of finding and accessing a suitable analog from long-term memory. There is considerable evidence that this process relies more on semantic commonalties and less on structural commonalties than mapping does. For instance, people often fail to access potentially useful analogs if they have too little semantic overlap with the target problem (Gick & Holyoak, 1980, 1983; Ross, 1989). Spontaneous analogies from remote domains seem especially difficult (Seifert, McKoon, Abelson, & Ratcliff, 1986; Keane, 1987).

The *multiconstraint theory* of Holyoak and Thagard (1989, 1995) is an influential synthesis of these and many other experimental findings. According to this theory, analogy-making is governed by a combination of the following three constraints: (*i*) *structural consistency*—the pressure to identify and use an isomorphism between the descriptions of the two situations, (*ii*) *semantic similarity*—the pressure to map elements with some prior semantic similarity (e.g., joint membership in a semantic category), (*iii*) *pragmatic centrality*—the pressure to give preference to elements that are deemed especially important to goal attainment, and to try to maintain correspondences that can be presumed on the basis of prior knowledge. All three constraints are conceptualized as "soft"—they do not operate as inviolable rules but rather as competing pressures (Hofstadter, 1984).

According to the multiconstraint theory, all three constraints play a role throughout the course of analogy-making (Thagard, Holyoak, Nelson, & Gochfeld, 1990; Holyoak & Thagard, 1995). However, the constraints affect the different subprocesses to a different degree. Thus, semantic similarity seems to dominate the analog access but the other two constraints also play a role. Structural consistency exerts its major impact in the mapping process. Later stages of analogy-making are very sensitive to pragmatic pressures. More specifically, they are very important during *analogical inference* (or *transfer*) and *evaluation*—the processes of augmenting the target description and verifying the consistency of the inferences.

There are a number of other factors that also influence the course of analogy-making. For example, there is evidence for an *order effect* on analogical mapping (Keane, 1994). It is faster and more accurate when the order of presenting the target elements to the subject encourages a correct initial correspondence, which can then constrain subsequent mappings. There is also evidence for *priming effects* on analogy-making (and problem solving in general, Kokinov, 1990, 1994a). In these experiments, exposure and work on selected problems affected the performance on a later problem. The magnitude of this effect decreased with time. Other data from the same lab (Kokinov & Yoveva, 1996; Kokinov, Hadjiilieva, & Yoveva, 1997) indicate *context effects* on problem solving.

All these empirical findings must be taken into account when building and evaluating cognitive models of analogy-making. The following section presents a brief overview of some of these models.

2.2 Models of Analogy-Making

Analogy-making is a very complex phenomenon and it is very difficult to encompass all of it at once. As a consequence, most models in the field could be characterized by the ancient maxim "Divide and conquer!" That is, analogy-making is usually conceptualized in terms of separate stages or phases. While this book advocates a different and more interactionist approach, this conceptualization is necessary for expository purposes. Thus, one possible division includes:

- **Perception** (representation building) of the target problem;
- **Retrieval** of an appropriate analog (or base) from long-term memory;
- **Mapping** the base onto the target to find corresponding elements;
- **Transfer** of knowledge from the base to the target;
- **Evaluation** of the imported knowledge within the target framework;
- **Learning and generalizing** the new experience for use in the future.

These stages are supposed to be relatively independent from one another and thus susceptible to piecemeal exploration. Different researchers focused their attention on different aspects of analogy-making, each building a model that highlights some issues at the expense of others.

In contrast, the AMBR project advocates the strategy of integration. This does not mean that we overlook the time-honored "Divide and conquer!" On the contrary, we think it has given rise to quite a lot of knowledge which could (and should) serve as a springboard for any further research. Out of the many models reported in the literature (Anderson & Thompson, 1989; Carbonell, 1983; Evans, 1968; Halford, Wilson, & Phillips, 1998; Hall, 1989; Holland, Holyoak, Nisbett, & Thagard, 1986; Kedar-Cabelli, 1988; Kolodner, 1993; Veloso, 1994), the following sections discuss those which have directly influenced our work.

2.2.1 SME and MAC/FAC

The Structure Mapping Engine (Falkenhainer, Forbus & Gentner, 1986, 1989; Forbus & Oblinger, 1990; Forbus, Ferguson, & Gentner, 1994) is a computer implementation of Dedre Gentner's Structure Mapping Theory (1983). It is designed as a domain-independent analogical matcher. It takes two inputs: a base description and a target description. Both are in predicate calculus. SME relies on purely syntactic operations to produce a set of correspondences. The underlying intuition is that syntax can capture meaning. The model uses semantic knowledge only insofar as it distinguishes identical from non-identical symbols and never mixes symbols of different types. Entities (individual objects and constants) are mapped onto other entities, functions onto functions, and relations onto relations. Special priority is given to higher order relations, thereby operationalizing the systematicity principle postulated by the theory.

A serious limitation of the model is that it depends on identicality of predicate names. In the widespread version of the program, the matching algorithm requires that the predicates at the top of the relational structure are identical. These *local matches* are then recursively expanded to subordinate levels of the structure. Thus, the identicality restriction applies most forcefully precisely at the level which is most important according to the theory—the high order relations.

The shortcomings of the identicality restriction became apparent when SME was used as a building block for bigger systems (Falkenhainer, 1988, 1990a). Subsequent versions of the Mapping Engine (Falkenhainer, 1990a) have relaxed this restriction by applying *minimal ascension* through an `is-a` hierarchy and/or using role information (i.e., the dependencies which a given element satisfies). In our view, these are important improvements. Still, most applications of SME reported in the literature (e.g., Forbus, Gentner, & Law, 1994) use the "default" rigid identicality.

More generally, the weakness of SME is that it relies very heavily on the *form* of the represented knowledge. For instance, the model differentiates between attributes and relations.[2] Yet, the only difference is that attributes are predicates with one argument while relations have two or more. Logically, each attribute can easily be transformed into an equivalent relation and vice versa, e.g., `hot(X)` $\leftrightarrow$ `temperature-of(X,high)`. SME thus requires that some other part of the system represents (and, perhaps, iteratively re-represents) the two situations in a common representational format (Forbus, Gentner, Markman, & Ferguson, 1998). There have been numerous demonstrations that SME can, in fact, operate as a module in large reasoning systems (e.g., Falkenhainer, 1988, 1990b; see Forbus, 2001, and Forbus et al., 1998, for reviews). More research is needed to clarify the interface between a putative analogy module and the rest of cognition.

Despite its limitations, the Structure Mapping Theory and SME are very influential pioneering work and their importance cannot be questioned. Gentner (1983) was the first to advocate that analogy depends on *structure* in a period when all kinds of similarities were explained by feature overlap (Tversky, 1977). Nowadays the importance of structure and systematicity is taken for granted. In general, the essence of a situation—the part that should be mapped—is a high-level coherent whole, not a collection of isolated low-level similarities.

SME is a key component of the MAC/FAC model of similarity-based retrieval (Forbus, Gentner, & Law, 1994). The model explains retrieval in terms of a two-stage process. During the first (MAC) stage, a cheap filter is used to weed out the majority of episodes in long-term memory. This filter calculates dot products over *content vectors*—flat enumerations of the functors participating in the respective episode description. The second (FAC) stage then takes the output of MAC and subjects the candidates to more expensive processing. It uses SME to assess the structural overlap between the candidate and the probe. Selection in both cases is based on comparing numerical scores against predefined thresholds.

---

[2] The original version of SME (e.g., Falkenhainer, Forbus & Gentner, 1986) ignored attributes altogether. This is no longer the case in the current version. Also, the "modes" defined in the early publications are eliminated. Now SME always operates in what used to be called "literal similarity mode." (Kenneth Forbus, personal communication, July 1998.)

This computational scheme has a number of engineering advantages. It also accounts for several behavioral phenomena regarding analog access (Forbus et al., 1994). However, it is questionable whether the cognitive system uses two different representations for each memory item. One obvious problem that arises in such dual representation scheme is how to maintain consistency between the two representations of the same episode. Moreover, both representations seem too rigid and static. As argued later in this book (e.g., section 4.5), such centralized representations of situations cannot explain well the flexibility of analog access.

It is not clear how MAC/FAC could account for the context and priming effects in analogy-making (Kokinov, 1994a; Kokinov & Petrov, 2001). SME and MAC/FAC are context-sensitive insofar as their operation depends on the contents of working memory, which in turn can depend on context (Kenneth Forbus, personal communication, July 1998). This is a valid point but WM content by itself does not suffice to account for the multiplicity of context influences on human cognition. For example, items that are only related by association but do not participate in shared predicates will be ignored by SME even if they are present in WM. Such associations can play an important role in memory access (e.g., Godden & Baddeley, 1975).

2.2.2 IAM

The Incremental Analogy Machine (Keane & Brayshaw, 1988; Keane, Ledgeway, & Duff, 1994) is another model of the mapping stage in analogy-making. It starts by identifying a *seed group* in the base situation and picks up a *seed element* from that group. It is similar to SME in many respects and is subject to the same limitations. For instance, IAM relies on syntactic criteria for choosing the seeds. The seed group is the group of predicates having the most higher order connectivity between its elements. The seed element is sought among the relations that take multiple arguments.

The main idea of the model is to establish a *seed match* relating the seed element to some target element and then use this match to incrementally grow a whole set of coherent matches. The *match rules* that carry out this task are sensitive to the structural, semantic, and pragmatic constraints on analogical mapping. The seed is used for disambiguation of ambivalent cases. All decisions are made sequentially, which requires backtracking when a commitment is inappropriate.

The backtracking algorithm allows IAM to work in limited working memory and produces order effects. Both properties are psychologically desirable (Keane et al, 1994). On the other hand, backtracking amounts to exhaustive search which casts doubts on IAM's abilities to scale up. It seems to us that mapping should be done as a combination of sequential and parallel processes.

2.2.3 ACME and ARCS

The Analogical Constraint Mapping Engine (ACME, Holyoak & Thagard, 1989) is another influential pioneering model. It is an important precursor of AMBR. ACME introduced the notion of *constraint satisfaction* to the analogy literature. The model uses a massively parallel connectionist algorithm to find a globally consistent set of correspondences between the two descriptions being mapped. The main idea is to build a *constraint satisfaction network* (CSN) with nodes representing hypothetical correspondences and positive and negative links enforcing the constraints. After a relaxation process, the network settles in a state representing a (potentially suboptimal) solution to the constraint satisfaction problem.

Like the models discussed above, ACME starts with a propositional description of the two situations. It then translates these representations in connectionist terms using a centralized symbolic algorithm. There is no genuine interaction between the symbolic and connectionist components. Therefore, ACME can be considered as a precursor of hybrid models but in itself it does not constitute such a model.

A weakness of the model is that it constructs too many hypotheses—all elements from the target are paired with all elements from the base, with the restriction that objects must map to objects, one-place predicates to one-place predicates, etc. Most of these hypotheses are completely implausible and have to be suppressed later. In addition, the size of the resulting network is too demanding for the working memory of the system (Keane et al., 1994; Kokinov, 1994a; Halford, Wilson, & Phillips, 1998; Hofstadter, 1995; Hummel & Holyoak, 1997).

ACME has other limitations that are discussed at various places in this book. Still, the idea of constraint satisfaction has been adopted in AMBR and is the foundation of one of its main mechanisms. The model has certainly influenced our work. There are many differences between the two models, however, as presented in detail in section 5.4.1.

A complementary model—ARCS (Analog Retrieval by Constraint Satisfaction)—applies the constraint satisfaction idea to the task of analog retrieval (Thagard, Holyoak, Nelson, & Gochfeld, 1990). The model first scans the whole episodic memory and looks for episodes having element(s) similar to some target element(s). It constructs a node for each tentative correspondence between a source episode and the target. More nodes hypothesize correspondences between individual propositions. ARCS then sets appropriate excitatory and inhibitory links and relies on the relaxation procedure to decide which analog best satisfies the constraints.

The model uses a semantic knowledge base for estimating the degree of semantic similarity between various entities. These estimates, however, are static. To illustrate, synonyms always count for 0.6, superordinates for 0.3, etc. As Kokinov (1992b, 1994a) has argued, this approach fails to reflect the dynamic and context-sensitive nature of human similarity judgements.

ARCS is broadly similar to MAC/FAC in that it uses a semantically based preliminary screening to identify candidate analogs and then applies the mapping machinery (although running in economical mode) to do more careful analysis. In effect, both models put a limited matcher inside the retrieval module. This creates redundancy when the retrieved episode is passed to the main mapping engine. We argue that there are better ways for integrating the two subprocesses in analogy-making.

2.2.4 LISA

Hummel & Holyoak (1997) proposed an integrated model of analogical access and mapping called LISA (Learning and Inference with Schemas and Analogies). This is a structure-sensitive connectionist model and as such combines the advantages of the symbolic and subsymbolic approaches to cognitive modeling. The model represents propositions as distributed patterns of activation over units representing semantic primitives. The distributed representation brings flexibility and generalization capabilities. LISA uses *dynamic binding* to combine these representations into propositional structures[3] (see also Shastri & Ajjanagadde, 1993). Thus it achieves the structure sensitivity that is crucial for analogy-making. The cost for this is that LISA must operate within inherent capacity limits given by the size of the *phase set* required for the dynamic binding. Hummel and Holyoak (1997; and also Halford et al., 1998) argue that similar limitations arise in human reasoning.

A key innovation is that LISA treats analogical mapping as a form of learning. The model establishes correspondences by gradually learning weights of the *mapping connections* between various elements. This allows the model to arrive at globally consistent mappings without the need of massively parallel constraint satisfaction. Moreover, analog access and mapping are integrated—they are treated as processes of guided pattern classification.

Due to these powerful and flexible mechanisms, LISA has been able to simulate various empirical phenomena with considerable success (Hummel & Holyoak, 1997, 2003, 2005). It advances the research on analogy in many ways. Still, the model is not without its problems.

One open question involves the size of the descriptions that the model can handle. Due to the distributed representations, quite complex machinery is required to maintain even a simple proposition. Things become even more complicated with hierarchical structures. Although the representational scheme can in principle support descriptions of arbitrary complexity, higher-order predicates require the so-called parent-daughter distinction of proposition (P) units. The neurological plausibility of this distinction seems very problematic.

Another shortcoming is that LISA uses what we call *centralized representations of situations*. Each situation could be in one of three modes (driver, recipient, or dormant) and all elements are simultaneously flipped from one mode into the other. This implies that LISA, like ARCS and MAC/FAC, treats the episodes in the long-term memory as units—they are either retrieved wholesale or not at all. As argued in Section 4.5.1, this approach has certain disadvantages.

---

[3] See Section 2.2.6 for details, a concrete example, and critique of dynamic binding.

2.2.5 Copycat and Tabletop

All models cited so far start from a hand-coded representation of the target problem "implanted" into their working memory. In other words, they bypass the task of building an appropriate representation of the target situation. There are strong arguments, however, that this latter perceptual aspect is crucial to analogy-making (Chalmers et al., 1992). Without it, a large and important part of the overall task is done by the coder of representations (typically a human programmer) instead of the model. All models discussed so far suffer from this limitation. The current version of AMBR makes no exception.

The intimate interplay between perception and analogy-making is the defining feature of the work of Douglas Hofstadter, Melanie Mitchell, and Robert French (Hofstadter, 1984, 1995; Mitchell, 1993; French, 1995). Their models—Copycat and Tabletop—constitute an important bridge over the gap that separated research on analogy-making from that on perception. Both models build their own descriptions of the problems they work with. For Copycat, the problems involve letter strings in a micro-domain; Tabletop deals with arrangements of objects on a table. The perceptual activity goes in parallel with the process of building correspondences between different elements of the situation. Thus the two processes can influence each other.

Fundamental to Copycat and Tabletop is the notion of *statistical emergence*: the program's macroscopic behavior emerges from the interaction of a large number of low-level activities in which probabilistic decisions are made. There is no central executive that controls the operation the system. Instead, all processing is done by small entities called *codelets* that create, mediate, and respond to various *pressures*. This provides for great flexibility. The model presented here shares many of these ideas, although in a different form.

Both Copycat and Tabletop lack any episodic memory and do not address the problem of accessing a source analog from a large pool of past episodes. Thus, they leave an indispensable component of analogy-making out of their scope, just as models such as LISA and AMBR do with perception.

2.2.6 Postscript

Most references in this chapter so far (and throughout the book more generally) are to sources published prior to 1998. They reflect the state of the art at the time that AMBR2 was being developed. Of course, research on analogy-making did not stop in 1998. This postscript, written in the summer of 2013, provides pointers to the more recent literature. This is not a comprehensive review. We focus on the new developments in analogy modeling, particularly models descending from the foundational work cited above. Even within this circumscribed scope, however, the references below are just a sample from a much larger literature.

Analogy research is a thriving multi-disciplinary field (for reviews, see e.g., Dunbar & Blanchette, 2001; Gentner, 2010; Halford, Wilson, & Phillips, 2010; Holyoak, 2005, 2013). There have been two international conferences on analogy (Gentner, Holyoak, & Kokinov, 2001; Kokinov, Holyoak, & Gentner, 2009) and a third is scheduled for 2013 (http://leadserv.u-bourgogne.fr/analogy2013/). A workshop on "Analogies: Integrating Multiple Cognitive Abilities" (Schwering, Krumnack, Kühnberger, & Gust, 2007) led to a special issue of the journal of *Cognitive Systems Research* (Schwering, Kühnberger, & Kokinov, 2009).

Cognitive neuroscience, propelled by advances in functional magnetic resonance imaging (fMRI) and other techniques, has seen especially rapid growth since 1998. There is mounting evidence that the prefrontal cortex is heavily involved in analogy and relational reasoning (e.g., Boroojerdi et al., 2001; Bunge et al., 2005; Christoff et al., 2001; Morrison et al., 2004; Waltz et al., 1999; Wright et al., 2008; See Krawczyk, 2012, for review).

The intervening 15 years have also seen big advances in the field of computational models of analogy (see, e.g., French, 2002; Gentner & Forbus, 2011; Holyoak, 2005; Kokinov & French, 2002, for reviews). Many brand new models have been proposed (e.g., Blank, 1997; Eliasmith et al., 2012; Eliasmith & Thagard, 2001; Halford, Wilson, & Phillips, 1998; Jani & Levine, 2000; Kanerva, 1998; Könik et al., 2009; Larkey & Love, 2003; Leech, Mareschal, & Cooper, 2008; Lu, Chen, & Holyoak, 2012; O'Donoghue, Bohan, & Keane, 2006; Petkov & Kokinov, 2006; Petrov & Anderson, 2005; Rachkovskij, 2004; Ramscar & Yarlett, 2003; Rasmussen & Eliasmith, 2011; Salvucci & Anderson, 2001; Schwering, Krumnack, Kühnberger, & Gust, 2009; Wilson, Halford, Gray, & Phillips, 2001). In addition, many classic models have been developed and refined (e.g., Doumas, Hummel, & Sandhoffer, 2008; Grinberg & Kokinov, 2003; Hummel & Holyoak, 2003; Lovett, Tomai, Forbus, & Usher, 2009; Marshall, 2006; Nestor & Kokinov, 2004).

The structure-mapping research program is going strong (see Gentner, 2010; Gentner & Forbus, 2011, for recent reviews). For 30 years since the pathbreaking articles in the early 1980s (e.g., Forbus, 1984; Gentner, 1983), the research group led by Dedre Gentner and Ken Forbus has consistently produced a steady stream of publications that have left an indelible mark in the field. Here we can mention just three strands of their research program, with the understanding that there is much more where these three come from. The web site of the Qualitative Reasoning Group at Northwestern University (http://www.qrg.northwestern.edu/) is a good starting point for exploring their modeling work.

First, the Structure Mapping Engine (SME) is being incorporated as a key component in a variety of large-scale integrated AI systems (Forbus, 2001; Forbus et al., 1998). An early example of such system is Falkenhainer's (1988, 1990b) PHINEAS, which learns physical theories by analogy with previously understood examples. In addition to SME, PHINEAS incorporated other modules that had themselves been used in other projects—QPE, an implementation of Forbus' (1984) *qualitative process theory*, and DATMI (Decoste, 1991)), a measurement interpretation system. A more recent example is the Companion cognitive architecture (e.g., Forbus, Klenk, & Hinrichs, 2009; Forbus & Hinrichs, 2006; Klenk & Forbus, 2009a).

This architecture is motivated by the hypothesis that analogical processing is central to human reasoning and learning. Accordingly, it uses SME for the "inner loop" of cognition, where it performs analogical mapping and structural alignment between representations. Companion uses MAC/FAC (Forbus, Gentner, & Law, 1994) for similarity-based retrieval and SEQL (Kuehne, Forbus, Gentner, & Quinn, 2000) for generalization. A notable feature of Companion is that it is not an empty architectural shell nor a special-purpose problem solver, but a general knowledge-rich agent that can acquire or learn domain knowledge by building on an extensive pre-existing ontology. Different Companion applications have used various Cyc ontologies (http://www.cyc.com/platform/overview; Lenat, 1995). These are massive knowledge bases containing hundreds of thousands of concepts and millions of assertions describing various relationships among these concepts. In terms of sheer KB size and problem complexity, no model in the analogy literature can rival the SME-based systems. According to Gentner and Forbus (2011, p. 273), "To date the Companions architecture is the only one that has been tested in experiments in which the inputs were produced by groups other than the researchers, and where the results were independently evaluated by other organizations."

Companion also features *coarse-grained parallelism*—it is implemented as a distributed system that allocates individual nodes of a computer cluster to a small number of semi-independent, asynchronous processes (or *agents*). For example, a Companion system may have an Executive agent that prioritizes the work on the Companion's goals, a Session Reasoner that carries out domain reasoning, and an Analogical Tickler that monitors the state of the working memory and continually retrieves cases from the large knowledge base and presents them to the user and the Session Reasoner. This coarse-grained parallelism is an interesting counterpoint to the much finer-grained parallelism in the DUAL architecture.

A second strand in the structure-mapping research program is its emphasis on learning. In her experimental work, Dedre Gentner has had a long-standing interest in learning and development (e.g., Christie & Gentner, 2010; Gentner, 1989, 2010; Gentner, Loewenstein, & Hung, 2007; Gentner & Toupin, 1986). She currently is a co-Principal Investigator of the Spatial Intelligence and Learning Center (http://spatiallearning.org/) funded by the National Science Foundation. The early models focused on the mapping and retrieval stages of analogy-making, although PHINEAS (Falkenhainer's 1988, 1990b) was a pioneering exploration of how to learn abstract knowledge via cross-domain analogies. More recent models increasingly emphasize learning. This follows the general trend in the field of symbolic artificial intelligence. Early AI systems focused on deductive processing, search, and problem solving, but today statistical learning techniques are widely used across all areas of AI (Russell & Norvig, 2009). Relational learning is the current frontier (Forbus, 2010). While much can be done with classifiers, broadening the expressiveness of what can be learned is crucial for many tasks, as well as for capturing the range of human learning. SME and MAC/FAC are key components of several insightful recent models of concept learning via analogical generalization and near-misses (e.g., Kuehne, Forbus, Gentner, & Quinn, 2000; McLure, Friedman, & Forbus, 2010), as well as learning new domain theories through multiple cross-domain analogies (e.g., Klenk & Forbus, 2009a, 2009b).

A third strand in the structure-mapping research program is its recent emphasis on sketch understanding and high-level perception. A key tool for this research is the open-domain[4] sketch understanding system CogSketch. Its authors (Forbus et al., 2011, p. 648) argue eloquently for the importance of sketching in human cognition:

> Sketching enables people to externalize and communicate ideas. People draw maps, the structure of complex systems, and sequences of sketches illustrating how a process unfolds. The power of sketching is such that visual languages are invented do depict otherwise abstract ideas (e.g., electronic circuit schematics, software modeling diagrams, parse trees). Sketching is fascinating scientifically because it engages visual, spatial, and conceptual knowledge and skills. Consequently, understanding how people understand and communicate with sketches should provide important insights for understanding human cognition more generally. Moreover, if we can use models of sketch understanding to create software that can participate in sketching in human-like ways, there are potentially significant practical benefits.

This argument is reinforced by data from cognitive linguistics. Lakoff and Johnson's (1980) seminal book *Metaphors We Live By* is full of persuasive linguistic examples that suggest that many of our fundamental concepts are organized in terms of *orientational metaphors*. For instance, virtue is up, whereas depravity is down. This is manifested in expressions such as: "He is *high*-minded. She is *upright*. That *low* trick would be *beneath* me." (Lakoff & Johnson, 1980, p. 16). Furthermore, our experience with physical objects (especially our own bodies) provide the basis for an extraordinarily wide variety of *ontological metaphors* that cast events, activities, emotions, ideas, etc., as entities and substances (p. 25f):

> We are physical beings, bounded and set off from the rest of the world by the surface of our skins, and we experience the rest of the world as outside us. Each of us is a container, with a bounding surface and an in-out orientation. We project our own in-out orientation onto other physical objects that are bounded by surfaces. Thus we also view them as containers with and inside and an outside. Rooms and houses are obvious containers. . . . But even when there is no natural physical boundary that can be viewed as defining a container, we impose boundaries—marking off territory so that it has an inside and a bounding surface—whether a wall, a fence, or an abstract line or plane. . . .
>
> [For example,] we conceptualize our visual field as a container and conceptualize what we see as being inside it. . . . Thus we can say: "The ship is *coming into* view. I have him *in* sight. That's *in* the *center of* my *field* of vision." . . .
>
> We use ontological metaphors to comprehend events, actions, activities, and states. Events and actions are conceptualized metaphorically as objects, activities as substances, states as containers. A race, for example, is an event, which is viewed as a discrete entity. . . . Thus we can say of a race: "Are you *going to* the race? Did you *see* the race? *Halfway into* the race, I ran out of energy. He's *out of* the race now."

These ideas are related to Barsalou's (1999) *perceptual symbol system* hypothesis and to the rapidly expanding (sic!) field of situated cognition (e.g., Robbins & Aydede, 2008). They suggest ways in which abstract thought and language can be grounded in the perceptual-motor circuits in the brain (e.g., Eliasmith, 2013). Other

[4] CogSketch is available from http://www.qrg.northwestern.edu/software/cogsketch/index.html.

researchers (e.g., Chalmers et al., 1992; Hofstadter, 1984, 1995; Holyoak & Thagard, 1995; Hummel & Stankiewicz, 1998; Lakoff & Johnson, 1999; Lovett, 2012; Mitchell, 1993; French, 1995) have also pointed out the importance of analogy in building a bridge between perception and cognition.

For all these reasons, tools such as CogSketch are important additions to our armamentarium. The key scientific hypothesis embodied CogSketch is that perceptual processing produces *qualitative* spatial representations (Forbus et al., 2011). These representations can then be used for matching (via SME), retrieval (via MAC/FAC), and generalization (via SEQL). To create a sketch, the user simply begins drawing. CogSketch captures the sketch's *ink*, but the user must use certain segmentation tools to help the software segment the ink into separate objects. The user can also provide optional *conceptual labels* for various pieces of ink or the spaces defined by the ink. For example, a sketch can consist of a circle labeled "Sun," a larger concentric circle labeled "orbit," and a small filled circle labeled "planet." The conceptual labels are selected from the OpenCyc knowledge base (http://www.cyc.com/platform/overview) that contains over 58,000 concepts from a broad variety of domains. Given a sketch, CogSketch automatically computes qualitative spatial relations between objects. These relations include topology (e.g., containment, intersection, contact) and relative position.[5] CogSketch combines this relational information with the conceptual labels to produce a qualitative representation of the sketch. As it depends on the user's initial segmentation and labeling of the image, CogSketch is not an autonomous model of scene perception and object recognition. But note that in human-to-human sketching too, "recognition is a catalyst, not a requirement" (Forbus et al., 2011, p. 649). When people sketch to each other, they typically also talk and call out what the squiggles on the page are meant to represent.

In a recent Ph.D. dissertation advised by Ken Forbus, Andrew Lovett (2012) extended CogSketch in two ways. First, the new system—Perceptual Sketchpad—constructs *hybrid* representations that combine qualitative and quantitative (metric) information. Second, Perceptual Sketchpad constructs three-level *hierarchical* representations. Starting with the entry-level objects segmented by the user, it can, on demand, parse an object into a set of edges or aggregate several objects into a larger-scale group. The resulting *hierarchical hybrid representations (HHRs)* span three levels of abstraction—edges, objects, and groups—and contain qualitative and quantitative information at each level.

Lovett's (2012) system can also encode and manipulate procedural knowledge in the form of Spatial Routines for Sketches (SRS). This is a modeling framework inspired by Ullman's (1987) *visual routines*. SRS implements a set of cognitive operations such as mental rotation, grouping by proximity, addition or removal of parts or objects, shape deformation, etc. An important class of operations involve perceptual comparisons via SME. The operations can be combined to create a spatial routine for performing some spatial task or describing a particular strategy for solving geometric analogy problems. A spatial routine is analogous to a computer

[5] Petrov, Van Horn, and Todd (2011) provide experimental evidence that two particular qualitative relations—collinearity and bisection—can play important roles in object recognition.

program—each operation takes an input, processes it, and produces an output. For example, a differencing routine (or strategy) takes two HHRs as inputs and produces an HHR describing how the first input should be changed to become identical with the second.

Using the SRS framework, Lovett (2012) builds models of three problem solving tasks: (*i*) Evans' (1968) geometric analogy problems (e.g., Lovett, Tomai, Forbus, & Usher, 2009), (*ii*) Raven's Progressive Matrices (RPM; Raven, Raven, & Court, 2000) test of fluid intelligence (Lovett, Forbus, & Usher, 2010), and (*iii*) Dehaene, Izard, Pica, and Spelke's (2006) visual oddity task (Lovett & Forbus, 2011). Each of these models has sufficient information-processing resources to solve many (but not all) problems in its respective domain. This is evidence for the generality and expressive power of the SRS framework. Moreover, the performance of each model is compared to psychological data (Lovett, 2012). Problems that are hard for the model are also difficult for people. A variety of "ablations" can be performed on the model, blocking the ability to do specific operations such as grouping objects together. These ablations cause selective failures to solve particular problem types. Lovett (2012) performed ablation analyses to evaluate the difficulty of a problem along three dimensions: encoding and abstraction, working memory load, and control processes. He then used linear regression to assess how well these factors account for human accuracy and response-time patterns across the problem types. A detailed examination of Lovett's work is beyond our present scope but these models certainly deserve careful study.

An important feature of the SRS models is that many of the visual routines use structure mapping to compute perceptual comparisons (Lovett, 2012). In other words, the process of (high-level) perception is heavily dependent on the mapping process. More broadly, Gentner and Forbus (2010) suggest that "similarity computations appear to be 'inner loop' core operations of cognition, i.e., they are used *throughout* cognitive processes" (p. 272, quotes in original, emphasis added). Thus, analogy researchers seem to be reaching a consensus that high-level perception (and memory retrieval and perhaps other cognitive processes) cannot be modeled as encapsulated modules separate from the mapping process. This is a partial[6] resolution to the theoretical controversy of whether perception is separable from mapping (e.g., Fodor, 1983; Forbus et al., 1998) or not (e.g., Chalmers et al., 1992; French, 1995; Hofstadter, 1984, 1995; Kokinov, Bliznashki, Kosev, & Hristova, 2007; Mitchell, 1993; Nestor & Kokinov, 2004). One lesson that we draw from Andrew Lovett's (2012) thesis is that, once one tries to build a detailed model that can perform a *visual* task such as Raven's Progressive Matrices or the oddity task, the modularity of high-level perception is revealed to be untenable. On the other hand, when the interactionist view is pushed to its logical extreme (e.g., Lakoff, 1987; Lakoff & Johnson, 1999; Linhares, 2000), it denies the very existence of objects and facts independent of human cognitive capacities. On this view, instead of external single bounded entities that can be segmented on a CogSketch image and labeled in terms of an OpenCyc ontology, objects are seen as "internal units of description"

[6] The resolution is only partial because the converse question—whether mapping is separable from perception—remains contentious. This controversy is discussed in more detail in Section 3.2.1.

(Linhares, 2000, p. 268). Note that this does not deny the existence of a reality that exists even if we close our eyes to it. Rather, it denies that the *segmentation* of the world into separable units, the imposition of *boundaries* between those units, is a matter of objective fact. This is reminiscent of the so-called ontological relativity thesis in philosophy of language and the accompanying arguments for the indeterminacy of translation (e.g., Quine, 1968)

Douglas Hofstadter (e.g., 1984, 1995) has always been a trenchant defendant of the interactionist view. Since 1998, he has continued to articulate it in new books (e.g., Hofstadter & Sander, 2013) written in his trademark style. In addition, several new Ph.D. students developed models under his supervision. These include Marshall's (1999, 2006) Metacat—a continuation of Hofstadter (1984) and Mitchell's (1993) Copycat project. Open-source software for running Metacat (and, by subsumption, Copycat) is available at http://science.slc.edu/ jmarshall/metacat/. Among other things, Metacat adds an episodic memory to the original Copycat architecture. This makes it even more closely related to the DUAL architecture described in this book. Other research by Hofstadter's group includes two models of the Letter Spirit micro domain (McGraw, 1995; Rehling, 2001) and a model (Foundalis, 2006) that can solve some (but by no means all) of Bongard's (1970) challenging pattern-recognition problems (see http://www.foundalis.com/res/bps/bpidx.htm for a comprehensive collection). Unfortunately, these fascinating models are beyond our present scope.

The LISA research program is also going strong (see Hummel & Holyoak, 2005, for review). Recall from Section 2.2.4 that LISA is a structure-sensitive connectionist model that uses a hybrid representational scheme. Objects, concepts, and relations are represented as distributed patterns over *semantic feature* units, whereas propositions are represented as hierarchical arrangements of localist *proposition (P)* and *subproposition (SP)* units. A defining characteristic of LISA is its reliance on *dynamic binding* by synchrony of firing. The easiest way to explain this is by a concrete example. Consider the proposition `taller-than(john, sally)`. It involves two objects (`john` and `sally`) and two roles (`more-tall` and `less-tall`). Each of these four entities is represented as a distributed pattern over the semantic feature units. For instance, the features of `john` include {male, adult, human, height-6, ...}, whereas those of `more-tall` include {more, dimension-height, dimension-additive, ...}.

It is critically important to represent that `john` occupies the `more-tall` role in the proposition, whereas `sally` occupies the `less-tall` role. Without such *role-filler binding*, it is impossible to differentiate `taller-than(john, sally)` from `taller-than(sally, john)`. In LISA, fillers are bound to roles by virtue of the synchrony of firing of the units that represent them. Thus, `john`, `more-tall`, and all their associated feature units fire in synchrony during a certain *phase* of the so-called *phase set*. Similarly, `sally`, `less-tall`, and all their associated feature units also fire in synchrony during another phase. The model cycles through all phases in the phase set, driven by complex oscillatory circuits (see also Shastri & Ajjanagadde, 1993; von der Malsburg, 1995). Hummel and Holyoak (2003, p. 225) and Knowlton, Morrison, Hummel, and Holyoak (2012, p. 375) cite neurophysiological evidence (e.g., Singer & Gray, 1995) that the frequency of these oscillations is approximately 40 Hz, which is in the gamma band of the electroencephalographic (EEG) spectrum.[7] In other words, one sweep through the entire phase set takes ≈ 25 ms. The number of distinguishable phases in this cycle limits the working-memory capacity of the system (see also Halford et al., 1998, 2010). Hummel and Holyoak (2003, p. 225) estimate that "the maximum amount of information that can be processed together during analogical mapping [is] 4–6 role bindings or roughly 2–3 propositions." In the above example, this implies that `john`, `more-tall`, and their associated features are activated for ≈ 5 ms.[8] This role-filler binding is then inhibited while the other binding (`sally` and `less-tall`) is activated for ≈ 5 ms. The system then can process a few more subpropositions (= role-filler bindings) during the remainder of the phase set. Then the cycle repeats: `john` and `more-tall` fire again, followed by `sally` and `less-tall`, and so forth. At the level of the semantic features, this scheme produces sets of mutually desynchronized patterns of activation, one for each subproposition. LISA has elaborate control mechanisms for coordinating this activity and for bringing propositions in and out of the phase set.

LISA's hybrid representational scheme combines the flexibility of distributed semantics with the structure-sensitivity of explicit propositions. The resulting model is very powerful and has been remarkably successful. It can account for a number of phenomena in analogical reminding and mapping (e.g., Hummel & Holyoak, 1997; Krawczyk, Holyoak, & Hummel, 2005), analogical inference and schema induction (e.g., Hummel & Holyoak, 2003), similarity judgment (Taylor & Hummel, 2009), and other domains (see Hummel & Holyoak, 2005, for review).

A successor model called DORA (Discovery Of Relations by Analogy; Doumas, Hummel, & Sandhoffer, 2008) makes a foray into the important and difficult problem of how a cognitive architecture can discover relational concepts from examples and represent them as explicit structures (predicates) that take arguments (fillers) bound to distinct roles. The mechanisms for role-filler binding reduce the problem of learning multiplace (e.g., binary) relations to the problem of learning single-place (unary) properties (or attributes). These single-place role-filler bindings are then linked together to form complete relational structures. To achieve this linkage, DORA extends the time-dependent mechanisms of LISA so that role-filler bindings are represented not by *synchrony* (as in LISA) but by *systematic asynchrony* (see also Love, 1999). The idea is that roles fire immediately *before* their corresponding fillers, rather than simultaneously. In the above example, `more-tall` would fire first, followed by `john`, followed by `less-tall`, and finally `sally`. This segregates the roles and fillers in time and thereby works around a technical limitation

[7] John Hummel (personal communication, 18 October, 2013) advises caution on this point because "it is not currently known what exactly is oscillating at 40 Hz. Is it one neuron, a population of neurons, or multiple desynchronized populations?"

[8] John Hummel (personal communication, 18 October, 2013) expresses reservations on this point: "We have tried to stay as close to the neurophysiology as possible here, but I would be uncomfortable saying that LISA relies on spike timing in the 5 ms range, especially since the synchrony LISA exploits is actually burst rather than spike synchrony."

of the synchronous firing scheme. In LISA, predicates (such as `more-tall`) and objects (such as `john`) must be different data types. "Because object semantics fire at the same time as the predicate semantics to which they are bound, the only way to know whether a given unit represents an object or a feature is to assume that the two are represented by separate (non overlapping) pools of semantic feature units." (Doumas et al., 2008, p. 8). By contrast, DORA can use the temporal order of firing to disambiguate roles from fillers. As a result, it can encode predicates and objects with a common set of semantic units. "The capacity to treat role and filler semantics equivalently and still specify their bindings dynamically makes all of DORA's other operations possible." (Doumas et al., 2008, p. 8). Note that the asynchronous firing reduces the effective WM capacity in half.

In our opinion, DORA can learn only comparative relational concepts such as `taller-than`. To learn them from individual examples, DORA utilizes a *neural comparator circuit* that is invoked whenever two or more objects in the phase set have features that vary along the same metric dimension (e.g., height, size, color, etc.). The comparator circuit itself is hardwired into DORA in advance and is not learned. The relevant semantic features are given a priori as well. DORA's comparator activates the semantic unit `more` in synchrony with the larger value along the dimension, the semantic unit `less` in synchrony with the smaller, and the semantic unit `same` if the two values are equal. The resulting `more`, `less`, or `same` semantics, along with the semantics describing the dimension itself (e.g., `height`) provide the relational semantics for DORA's emerging relational predicates. (Doumas et al., 2008). In other words, the semantics of the unary predicate `more` is grounded in the internal procedural knowledge of the system as defined (and implemented) by the comparator circuit. DORA then links these single-placed predicates into multiplaced relations such as `taller-than`. The mechanisms that accomplish this linkage are complex and involve mapping, comparison, recruitment of new units, Hebbian learning, and intricate executive control.

Doumas et al. (2008, pp. 32–33) argue that DORA can also learn non-comparative relational concepts such as `chase`. We find their argument vague and unconvincing because it is not cached out fully in mechanistic terms. As far as we can tell, fleshing out the detailed steps that are necessary to carry out the proposed computation would expose that the argument implicitly relies on the existence of a "chase recognition circuit" that activates the proper semantic features in the proper order. This leads to some deep epistemological questions that are beyond our present scope.

DORA is an important milestone in analogy research because of its pioneering attempt to address in mechanistic terms the fundamental theoretical question of the origin of relational concepts. Despite its many limitations, it succeeds in rendering explicit how hard this problem is even for the most mundane relations. Much further research is needed in this foundational and challenging field. In broadest terms, DORA is built on the insight that relational concepts can be grounded in the regularities of internal procedural knowledge. The system must be able to *perform* comparisons before it can *reason* about comparative relations. This is a very fruitful idea that opens fascinating connections to Barsalou's (1999) perceptual symbol systems and Lakoff and Johnson's (1980) orientational metaphors discussed above.

Another major development in the LISA research program is that LISA has emerged as the main framework for interpreting neuropsychological, fMRI, and EEG data in computational terms (see Knowlton et al., 2012, for review). This research is still in its infancy. In an oft-cited article (Morrison et al., 2004), for example, the effects of frontotemporal lobar degeneration in patients with neurodegenerative disease was modeled by reducing certain LISA parameters from their default values. Different parametric manipulations degrade the model performance in characteristic ways, some of which correlate with patterns of behavioral deficit in various special populations. Much more research is needed to develop a full-fledged mechanistic explanation.

LISA is promoted as being neurologically plausible (Hummel, 2011; Knowlton et al., 2012; Morrison et al., 2004). While this claim is rarely challenged in the analogy research community, LISA's binding mechanism—synchrony of firing—is very controversial in the neuroscientific community. The so-called *binding problem* is a hotly debated topic in neuroscience (see, e.g., Feldman, 2013; Singer, 2007; Singer & Gray, 1995, Treisman, 1999; von der Malsburg, 1995, for reviews; see also the special issues of the journals *Neuron*, Roskies, 1999, and *Visual Cognition*, Müller, Elliott, Herrmann, & Mecklinger, 2001). The objective complexity of the issues is compounded by terminological confusion, as the phrase "the binding problem" is used in reference to at least four distinct problems with different computational and neural requirements (Feldman, 2013). Most data come from experiments on *visual feature binding*, whereas LISA depends on a form of *variable binding* in Feldman's (2013) terminology. Synchrony is a popular candidate solution to all four problems (e.g., Hummel, 2011; Shastri & Ajjanagadde, 1993; Singer, 2007; von der Malsburg, 1995). The binding problems are far from settled, however, and there are powerful arguments against the synchrony hypothesis (e.g., Cer & O'Reilly, 2006; Shadlen & Movshon, 1999; O'Reilly, Busby, & Soto, 2003).

Two criticisms of the synchrony hypothesis are particularly convincing. The first is the so-called *decoding problem* (Shadlen & Movshon, 1999). Binding by synchrony is useful only if some decoding (or "read-out") mechanism in the brain can distinguish synchronous from asynchronous signals at the relevant time scale, which according to Hummel and Holyoak (2003, p. 225) is on the order of 5 milliseconds per phase. The problem is that, according to standard biophysical estimates of neuronal function and synaptic transmission (Kandel, Schwartz, & Jessell, 2000; Koch & Segev, 2000), *this is just too quick for the neural hardware.* In other words, standard cortical neurons (including prefrontal neurons) cannot accomplish in 5 ms the complex decoding operations implicit in LISA's mechanisms.[9] We cannot develop this argument here because it requires detailed knowledge of neurophysiology that is beyond the scope of this book. For our purposes, it is sufficient simply to enumerate some of the most relevant facts. Firing-rate codes (as implied by LISA's continu-

[9] John Hummel (personal communication, 18 October, 2013) claimed that "the necessary 'read-out' process is nothing more complicated than postsynaptic neurons being sensitive to the synchrony (at some temporal time scale) of spike arrival. ... These mechanisms are not all complex. Just temporally-sensitive sums." However, he is reluctant to make firm commitments about how or whether LISA's precise timing parameters map onto the timing properties of neural hardware.

ous activation values) are useless at this time scale because much longer temporal windows (e.g, 50 ms) are needed to estimate a firing rate reliably. Thus, the putative decoding circuits must differentiate between synchronous and asynchronous firing at the level of individual spikes, not spike trains. But synchronous spikes stand out as special only if they do not arise frequently by chance. Shadlen and Movshon (1999) argue convincingly that, given typical fan-in factors and background noise levels in the cortex, synchronous spikes that carry binding information cannot be discriminated reliably from spurious coincidences among unrelated spikes. Furthermore, there is the problem of the transmission delays across anatomically distant cortical areas. The duration of a typical spike (or "action potential") is on the order of 1 ms and the time constant of a typical chemical synapse is 1–5 ms or longer (Kandel et al., 2000, Table 10-1 on p. 176). Thus, the putative millisecond precision is lost whenever a signal is relayed across a single synapse. Moreover, axonal conduction delays are quite variable (because they depend on axon diameter and myelination) and are also in the millisecond range (Swadlow & Waxman, 2012). Thus, if a neuron in area A is sending phase-tagged spikes to two other neurons in distant areas B and C, respectively, the latter two neurons will not receive the spikes synchronously. This is analogous to a situation in which a person simultaneously sends Christmas cards to two friends living in different cities. It is highly unlikely that the cards will arrive simultaneously because of the different distances to the two destinations and because of the different conduction velocities along the way. The problem is that the desynchronization produced by such informationally irrelevant biophysical delays is comparable to the temporal precision needed to carry binding information in LISA. The above list illustrates (but by no means exhausts) the serious biophysical obstacles that must be overcome in order to decode phase-tagged spikes successfully. DORA's systematic asynchrony imposes even greater implementational demands. In particular, the putative decoding mechanisms in DORA must be able to detect not only simultaneous co-occurrences (which bind roles or fillers to features within a phase) but also cross-correlations of events at fixed time lags (which disambiguate roles from fillers across consecutive phases). We are extremely skeptical whether cortical neurons can perform such operations reliably at a millisecond time scale.

The second criticism of the synchrony hypothesis identifies a timing problem from the other direction—*binding by synchrony is too slow for cognition*. For concreteness, consider a phase set with 6 phases of 5 ms each. Thus, the total duration of one sweep through the phase set is 30 ms. This temporal scale is necessary for the purposes of dynamic binding. Note, however, that the mapping process in both LISA and DORA requires the integration of information *across* the phases in the phase set. This is accomplished by the so-called self-supervised learning (SSL) mechanism that requires at least 3 sweeps through the phase set to perform one update of the so-called mapping connections (Doumas et al., 2008, Appendix A). Thus, it takes $3 \times 30 = 90$ ms to perform one SSL update on 3 binary relations (6 role-filler bindings) in LISA. Given that even simple mapping tasks typically involve multiple comparisons among multiple relations, a system that needs 90 ms to process 3 binary relations once will quickly accumulate overall response times that exceed the characteristic latencies of human cognition. The problem is even more acute in

DORA because it requires twice as many phases as LISA to process the same material, given that roles and fillers in DORA do not share phases. For reasons such as these, "temporal phase coherence is no longer considered a major contender in feature binding, in part because it would be much too slow to account for the experimental data. It is much more relevant in variable binding where most other models do not apply" (Feldman, 2013, p. 7). The above argument suggests that temporal phase coherence is much too slow for analogy-making also. A related argument is known as the 100-step program constraint on cognition (Feldman & Ballard, 1982; Newell, 1990).

The proponents of the synchrony hypothesis (e.g., Doumas & Hummel, 2005; Hummel, 2011; Hummel & Holyoak, 1997, 2003; Knowlton et al., 2012) embrace these limitations and claim that they correspond to the well documented limitations in human working-memory capacity (e.g., Cowan, 2000). However, our present concern is not about the limited number of items that can be held in WM but about the time it takes to process them. Concretely, the concern is not that LISA can only maintain 6 distinct role-filler bindings at a time but that it takes 90 ms to process them once. This is arguably too slow to model human performance.

Can this problem be resolved within the LISA framework? There seem to be only two possible approaches: reducing the duration of each elementary processing step or increasing the amount of informational content that can be processed in one step. The first approach does not seem viable because the current LISA proposal already exceeds the temporal resolution of the neural hardware as discussed above. The second approach hinges on the concept of *chunking* that is central to the WM literature but is almost absent from the discussions of synchrony-based role-filler binding. LISA supports a form of chunking in the so-called parent-daughter distinction of proposition (P) units. "When a P unit is active in daughter mode (i.e., as an argument in another proposition), it functions as a chunked representation of its propositional content" (Hummel & Holioak, 1997, p. 457). This is indeed a powerful mechanism, provided that the parent-daughter distinction can be given a neurologically plausible interpretation. However, it is far from clear whether this form of chunking is general enough to resolve the problem discussed here because, "This chunked representation is useful for mapping only insofar as mapping connections have already been established between the P unit and others: Because it cannot express its semantics directly, a P unit in daughter mode can only affect the recipient analog through learned mapping connections" (ibid.). But humans are quite capable of processing complex relational information at comparatively high speeds even when the content is novel—witness the time it took you to read the current sentence. LISA needs time to process each P unit in parent mode in order to learn the mapping connections that can support subsequent speedups when the same P unit is processed in daughter mode. So it is far from clear whether there are overall time savings when both modes are accounted for. Further research is necessary to establish whether LISA can achieve the information throughput that human cognition can achieve.

It is notable in this regard that the existing implementation of LISA already makes predictions about the total number of WM cycles needed to solve a given problem. When multiplied by the assumed oscillation frequencies, the number of cycles can easily be converted to response times. Note also that this conversion can be performed even without detailed knowledge of how LISA's mechanisms map onto neural hardware. The model makes definite predictions in terms of number of cycles. We appeal to LISA modelers henceforth to report descriptive statistics about the number of cycles observed in their simulations. If oscillations in the gamma frequency band in human EEG are in any way relevant to LISA's binding mechanism (Knowlton et al., 2012), the model makes predictions about overall response times. While we are not aware of any reports[10] of such predictions, we suspect that they would reveal that LISA is unacceptably slow relative to human performance. DORA is two times slower than LISA.

In conclusion, the LISA framework constitutes an important and influential proposal about how relational reasoning can be implemented in the brain. It has generated and continues to generate numerous insights into the mechanisms of analogy-making. As such, it definitely deserves careful study and further development. On the other hand, LISA's foundational mechanism—binding by synchrony—remains controversial and is subject to some very serious objections. DORA pushes the envelope still further. In many ways, these models' contributions are not so much in giving specific answers to various theoretical questions, but in revealing the true complexity of these questions.

The proponents of the synchrony hypothesis acknowledge that much research remains to be done: "Temporal structure in the form of oscillatory activity is in fact prominent in the brain, although no direct evidence yet connects such activity to the coding of propositions" (Knowlton et al., 2012, p. 376). John Hummel (personal communication, 18 October, 2013) cautions that the above analysis "rests on various assumptions about the timing properties of neurons and what they map onto, both in LISA's operation and in the neurophysiological data." He concludes that, "although it is by no means clear that LISA is consistent with the timing properties of real neurons, neither is it so clear that it is inconsistent. Admittedly, this lack of clarity is a limitation of the theory as it currently stands."

Given that the neurological plausibility of the synchrony-based framework is by no means obvious, the scientific community needs to explore alternative approaches in addition to continued work on LISA and DORA. An alternative approach in fact exists and seems very promising. Instead of synchrony of firing, binding can be implemented by conjunctive *tensor products* of various kinds (e.g., Eliasmith, 2013; Gayler, 2003; Gayler & Levy, 2011; Halford et al., 1988, 2010; Kanerva, 1988;

---

[10] Viskontas et al. (2004) report LISA latencies alongside human RTs. However, the LISA latencies are relative to an assumed pre-processing stage whose timing is not factored into the number of iterations plotted on Figures 3B, 4B, and 4D in Viskontas et al. (2004). As we need absolute (as opposed to relative) LISA latencies to perform the analysis suggested here, this data set is not immediately relevant for our present purposes. For the record, the time units ("iterations") on these plots are implementation-dependent. To convert to more meaningful units, 600 such iterations correspond to the time to process one binary proposition (two role-filler bindings) once (John Hummel, personal communication, 23 October, 2013). If the comparison with the human response times is taken at face value, RT differences of ≈ 1.5 seconds (across conditions within subjects) correspond to the time (600 iterations) needed to process one binary proposition in the simulation.

Plate, 2003; Smolensky, 1990; Smolensky & Legendre, 2006; Stewart, Bekolay, & Eliasmith, 2011). A closely related proposal is based on *coarse-coded conjunctive distributed representations* (CCDRs, Cer & O'Reilly, 2006; O'Reilly, Busby, & Soto, 2003; Petrov, Huang, & O'Reilly, 2013). These systems are free from the rigid temporal constraints that undermine the synchrony-based accounts. Furthermore, tensor products allow multiple relations to be represented and processed in parallel. The degree of parallelism is still limited by the throughput capacity of the neural hardware and large problems must still be solved piecemeal in a sequence of processing steps. However, the throughput limitations are not nearly as crippling as those in LISA and the flow of control is not as sequential. This modest amount of parallelism is consistent with the behavioral results from working-memory experiments because the WM capacity limitations may stem from various forms of interference rather than from a limited number of slots (e.g., Halford et al., 1988; Nairne, 1990; Neath & Nairne, 1995; O'Reilly, Braver, & Cohen, 1999; see Nairne, 2002, for a review and critique of the standard model of short-term memory).

Tensor products and CCDRs have been used in various models of relational reasoning and analogy-making (e.g., Blank, 1997; Cer & O'Reilly, 2006; Eliasmith & Thagard, 2001; Eliasmith et al., 2012; Halford et al., 1988; Kanerva, 1998; Petrov, Huang, & O'Reilly, 2013; Plate, 2003; Rasmussen & Eliasmith, 2011). Unfortunately, these models are quite technical and beyond the scope of this brief review, but we cannot fail at least to mention two especially promising lines of research. It is notable that both of them are based on full-fledged cognitive architectures. The first is the Leabra architecture developed by Randy O'Reilly and his collaborators (e.g., Jilk, Lebiere, O'Reilly, & Anderson, 2008; Kriete & Noelle, 2011; Kriete, Noelle, Cohen, & O'Reilly, 2013; O'Reilly, 1998, 2006; O'Reilly, Braver, & Cohen, 1999; OReilly, Busby, & Soto, 2003; O'Reilly, Hazy, & Herd, in press; O'Reilly & Munakata, 2000; O'Reilly, Petrov, Cohen, Lebiere, Herd, & Kriete, in press). The second is the Semantic Pointer Architecture developed by Chris Eliasmith and his collaborators (e.g., Eliasmith, 2013; Eliasmith & Anderson, 2003; Eliasmith et al., 2012; Stewart, Bekolay, & Eliasmith, 2011).

Tensor products and CCDRs have received their share of criticism too, including the charge that they are doomed to fail because they allegedly violate the so-called role-filler independence (Doumas & Hummel, 2005; Hummel, 2011). In our opinion, this particular charge is unfounded, but undoubtedly there are many open questions and much need for further tests of this alternative framework.

In conclusion, the field of analogy modeling has made enormous strides since 1998. Several sophisticated research programs have been developed, most of them centered around a cognitive architecture. Still, there are many open questions and unresolved problems, particularly with respect to the neurological substrate of relational reasoning. We fully agree with Gentner and Forbus' (2011, p. 273) assessment that, "Coming up with unified models that are both capable of human-like performance on realistic tasks and have a clear, biologically plausible implementation, remains an open problem."

Chapter 3
AMBR in Broad Strokes

The aim of this chapter is to present a concise and relatively self-contained description of the AMBR model and the DUAL architecture.

It is impossible to speak about AMBR without first mastering the DUAL terminology presented briefly below. DUAL is a general cognitive architecture (cf. Anderson, 1983; Newell, 1990) that is the foundation of the model. It was proposed by Kokinov (1994a, 1994b, 1994c). A detailed description of the 1998 version of the architecture can be found in Petrov (1997), and a shorter one in Kokinov and Petrov (2001). The Afterword traces the DUAL research since 1998.

3.1 Dual Cognitive Architecture

3.1.1 Main Ideas of DUAL

DUAL is a general-purpose cognitive architecture that comprises a unified description of mental representation, memory structures, and processing mechanisms. All these aspects of the architecture are organized around a small set of principles:

- **Hybridity.** DUAL is hybrid—it has complementary aspects. Moreover, it is hybrid in two ways. On the one hand, it hinges upon the symbolic/connectionist distinction and the integration between the two. On the other, there is the declarative/procedural distinction and integration thereof. The four aspects derived from these two pairs are merged together and coexist at every level of granularity in the architecture.
- **Emergent computation.** All processing and knowledge representation in the architecture is carried out by a cohort of small entities called *Dual agents*. There is no central executive that controls the whole system, allocates resources, resolves conflicts, etc. Instead, there are small-scale DUAL agents and local interactions between them. The global behavior of the system emerges from the self-organizing pattern of these interactions.

- **Dynamics and context-sensitivity.** An important feature of DUAL's operation is that it is constantly changing in response to influences from the environment. This is possible due to the emergent nature of the processing and the lack of rigid centrally imposed algorithm.

3.1.2 Basic Terms and Levels of Description

The basic structural and functional unit of DUAL is the *Dual agent*. Due to its importance, the DUAL agent has synonymous names: *micro-agent* or simply *agent*. Other names like *node* and *unit* are used to bring connotations from other theories, notably semantic networks and connectionism. It is important to note that throughout this book all the aforementioned terms refer to the same concept: the DUAL agent.

DUAL agents are the smallest building blocks of DUAL. Technically, there is nothing in the architecture but agents of various kinds. They interact with one another and thus combine into larger complexes. The *interactions* between agents are very important in DUAL because they keep the architecture together. They are often reified and called *links*, especially in contexts where the agents are called *nodes*.

A major architectural principle of DUAL is that larger structures emerge from the interaction of smaller ones. Thus, one can consider building blocks of increasing size. DUAL agents are at the base of this hierarchy, followed by *coalitions*, and *formations*. There is no sharp boundary between the latter terms. As a rule of thumb, a coalition consists of a relatively small number (e.g., less than 20) of interconnected DUAL agents while formations are much bigger.

DUAL-based models are complex systems and must be analyzed at different levels of granularity. It is useful to distinguish the following three levels:

The microlevel (agent level) deals with DUAL agents. Relevant topics here include the internal structure of a agent, its information-processing abilities, and the differences among agents of different types.

The mesolevel (coalition level) deals with *coalitions* of DUAL agents. A coalition is a set of agents and a pattern of interactions among them. Coalitions have two very important properties: they are *emergent* and *dynamic*. Thus, the mesolevel deals with the interactions between DUAL agents, the emergence of non-local phenomena out of local activities, and the dynamics of the organization of DUAL agents into coalitions.

The macrolevel (system level) deals with *formations* of DUAL agents and with whole *models*. Formations consist of big populations of agents and define the macroscopic structure of DUAL models. It is at this level where psychological concepts such as *working memory*, *mapping*, and *analogy* start to play the lead. They help describe the overall behavior of DUAL-based models and to compare them with other cognitive models and with humans.

These three levels are not independent. In fact, it is impossible to tell them apart. To illustrate, any analysis of coalitions crucially depends on the properties of their individual members. Conversely, a large part of the description of a DUAL agent is devoted to its interactions with other agents. Changes made at one level propagate to neighboring levels, recursively. For expositional convenience, however, each level is discussed in a separate section below.

3.1.3 Dual at the Microlevel

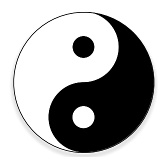
At this lowest level of granularity, the entity of main interest is the *Dual agent*—its internal organization and operation, as well as the interactions with its peers. Micro-agents are very important in DUAL because everything in the architecture ultimately boils down to them and their interactions. They are the "building blocks" that compose larger structures—coalitions, formations, and systems.

A very fundamental property of DUAL agents is that they are *hybrid* entities. They bring together ideas that are usually considered in opposition. In DUAL, opposites are not treated as irreconcilable antagonists but rather as complementary aspects of a harmonious whole.

Moreover, DUAL agents are hybrid in two ways. On one hand, they have both connectionist and symbolic aspects; on the other, they serve both as representational and processing units. These two dimensions are orthogonal and thus form the four aspects shown in Table 3.1.

Table 3.1 Different aspects of DUAL agents. (Compare with Table 3.2.)

	Representation	Processing
Connectionist aspect	Activation level	Spreading activation
Symbolic aspect	Symbolic structures	Symbol manipulation



From the connectionist perspective, each DUAL agent is a unit in a neural network. It has an *activation level* attached to it and continuously spreads activation to other agents. From the perspective of the classical symbolic approach to cognitive modeling, DUAL agents are *symbols*—they stand for something else. Concretely, they represent various concepts, objects, relations, etc. In addition to this representational aspect there is a procedural one: agents manipulate symbols. They can receive symbols from other agents, store them in local memories, transform them (thus producing new symbols) and so on.

DUAL agents interact constantly with one another. These interactions are very important because they are the fabric that weaves agents into larger complexes. DUAL interactions are relatively simple—they always involve only two micro-agents. One of them takes the initiative and either *reads* or *sends* some information to the other. Combined with the connectionist/symbolic distinction, this makes the four aspects summarized in Table 3.2.

Table 3.2 Different aspects of DUAL interactions. (Compare with Table 3.1.)

	Type *read*	Type *send*
Connectionist aspect	Activation level	Spreading activation
Symbolic aspect	Symbolic structures	Symbol exchange

As mentioned earlier, it is often convenient to speak of *links* instead of interactions. In particular, we can speak of the attributes of a link, notably its *weight* and *label*. We can also discuss different types of links, draw diagrams with circles and arrows, etc. For instance, the phrase "a population of interacting DUAL agents" translates into "a network of interconnected nodes." Throughout this book, both phrases mean the same thing.

3.1.3.1 Microframes

Each DUAL agent is a micro-frame. More precisely, it is the symbolic, representational aspect of a DUAL agent that is a microlevel frame. It has *slots* which in turn may have *facets*. Slots and facets are placeholders—they are filled up with *fillers*. Many fillers are references to other micro-frames and thus link the given DUAL agent to its peers. Consider the example on Figure 3.1. It shows the agent representing the concept `cup`. This frame has five slots, one of which has two facets.

```
cup
  :type :concept
  :subc (liquid-holder 1.0)
  :instance ((cup-1 0.3) (cup-5 0.2))
  :a-link (saucer 0.5)
  :slot1
     :type :relation
     :c-coref (cup-md-china 0.5)
```

Fig. 3.1 An example of a micro-frame. See text for details.

There are two major kinds of slots: *general slots* and *frame-specific slots* (or *G-slots* and *S-slots* for short). The former have predefined semantics that is invariant for all micro-frames. There are different kinds of general slots depending on their *label*. For example, the slot `type` is filled by a *tag* denoting the type of the agent. The slot labeled `subc` denotes that the concept (or *class*) represented by this frame is a subclass of another concept, the slot `instance` is filled by (a list of) references to specific instances of the concept, and so forth. Note that each individual reference has a *weight*.

In contrast to general slots, *frame-specific slots* do not have invariant semantics. Thus, `slot1` in `frame1` may mean something very different from `slot1` in `frame2`. Frame-specific slots also have labels but these are only arbitrary identifiers whose sole purpose is to distinguish one anonymous slot from the other. S-slots (and only they) have *facets*. Facets can be conceived of as slots within slots. The same set of labels applies to both G-slots and facets.

3.1.3.2 Connectionist processing

DUAL employs a dual representation scheme. Facts are represented symbolically by micro-frames, while their *relevance* to the particular context is represented by connectionist means. Each DUAL agent (and hence each micro-frame) has an *activation level* attached to it. There is an automatic process of *spreading activation* that continuously restructures the knowledge base, making some nodes more accessible and others completely inaccessible. Thus, each DUAL agent can be viewed as a node in a connectionist network. It has an *input zone*, *activation function* and *output function* (Rumelhart & McClelland, 1986).

The output of a micro-agent influences the input zones of the agents that are interacting with it. The former acts as a sender in the interactions and the latter—as receivers. Using the node-and-link terminology, we can say that the node sends activation to its *neighbors* via links. The phrase "there is a link from agent X to agent Y" means that agent X has a slot (or facet) filled up by a reference to Y. Each link has a weight that controls what portion of the sender's output is allotted to the particular receiver. Weights are usually normalized so that the sum of the weights of all outgoing links equals one.

The connectionist aspect of DUAL agents influences the symbolic one by determining the agent's *availability*. The notion of availability contributes very much to the hybrid nature of DUAL agents—it ties together all four aspects from Table 3.1. Like the agent itself, availability has declarative and procedural aspects:

Visibility. A DUAL system may consist of thousands of agents, each of which contains some particular small piece of knowledge. At any given moment, however, only a small fraction of this large knowledge base is visible. The symbolic processes that take place in the architecture can operate only on visible declarative elements. In addition, more active (and hence more visible) data elements are more attractive to the procedural machinery and thus are more likely to be taken into consideration.

Speed. The availability of a DUAL agent determines not only the visibility of its declarative aspect but also the speed of its procedural aspect. Very active agents work rapidly and thus determine the system's overall line of computation, low-active ones work slowly, and inactive ones do not work at all. As the pattern of activation over the network of agents changes, the speed of individual processors changes accordingly, making the computation performed by DUAL-based models dynamic and context-dependent.

3.1.3.3 Symbolic processing

A great deal of the information processing in the architecture is symbol manipulation — deterministic construction, transformation, storage, and exchange of symbolic structures. We use the general term *symbolic processing* to refer to these activities. They are carried out by the *symbolic processors* of DUAL agents. Each agent has such processor. It also has *local memory* to support the processor's work. Part of the local memory is *permanent*; the rest is *volatile memory*. The former keeps the micro-frame with its slots, facets, and fillers. The latter consists of an *input zone* and a *buffer*. Thus, a typical symbolic transaction involves receiving a symbolic structure into the input zone, comparing it with old symbols stored in the buffer, and sending it with due modifications to some of the agents referenced in the micro-frame.

The *speed* of the symbolic processor depends on the connectionist activation level of the respective DUAL agent. The exact rule for determining the speed is based on an energetic analogy that is described in detail in Appendix C. The main idea is that each symbolic operation requires the symbolic processor to do certain amount of *work* to carry it out. Doing work requires *energy*, which is supplied to the symbolic processor by the connectionist aspect of the agent. The speed of the computation depends on the *power* (i.e. on the rate of energy supply and consumption), which in turn is linearly related to the activation level.

3.1.4 Dual at the Mesolevel

DUAL agents are simple, they cannot do much in isolation. Therefore, they depend on one another and form coalitions. A *coalition* is a set of agents and a pattern of interactions among them. It is the entity of main interest at DUAL's mesolevel.

Coalitions have three very important properties: they are *decentralized, emergent,* and *dynamic*. None of these properties is present at the level of individual DUAL agents (the micro-level). There are "tight" coalitions and "loose" coalitions depending on the intensity of the interactions among their members. Tight coalitions are characterized by heavily weighted links and by intensive exchange of symbolic structures within the coalition. By contrast, loose coalitions are characterized by relatively weak links, often temporary ones, and by little or no symbolic interchange. There is a range between these two extremes. Moreover, coalitions do not have clear-cut boundaries. An agent can be involved in many of them at once, and to a different extent. Coalitions can "recruit" new members, either permanently or temporarily. They may share members and thus "flow" gradually from one into another.

Recall that DUAL agents can be seen as representational units—each of them stands for some single entity. By extension, coalitions of agents represent composite entities like propositions and situations. In the DUAL knowledge representation scheme even a simple proposition is represented by a number of agents. In such cases we say that there is a *meso-frame* that consists of several *micro-frames*.

Meso-frames can be quite complex, much more complex than any of the participating micro-agents). In this way, the expressive power of the DUAL representations is not limited by the restriction that each agent can have only a few slots. Coalitions are limited only by the connectionist mechanism that controls the activation level of their individual members and hence indirectly restricts the number of agents that can be active at a time.

The connectionist mechanism is also responsible for determining which parts of a meso-frame are *relevant*. It is possible, especially in loose coalitions, that only part of their members are active enough to pass the threshold. Thus, only part of the declarative knowledge stored in the meso-frame will be visible. In other circumstances, another part of the knowledge will be brought to the fore. This makes DUAL meso-frames dynamic and context-dependent.

From a processing point of view, coalitions are important in DUAL because it is at their level where non-local computation emerges. Each individual DUAL agent contributes somehow to the collective performance by doing its small and local-specific job. Each agent runs at its own speed and in parallel with other agents. To succeed in its task, the agent usually depends on other members of its coalition. It cooperates with them and competes with the agents from other coalitions. The net result of all these activities is that the coalition as a whole accomplishes some computation that is beyond the reach of any individual agent. This accomplishment has resulted from an *emergent* process—it is not carried out by any centralized processor following a rigid routine.

It is important to note that the interaction pattern among the participants in a coalition changes dynamically over time. New agents join in, others stay back, fall out and so on. In the node-and-link terminology, the *topology* of the network changes via dynamic addition and/or removal of nodes and links. This *computational dynamics* plays a key role in the overall flexible and context-sensitive behavior of DUAL-based models (Kokinov et al, 1996).

3.1.5 Dual at the Macrolevel

To summarize the story so far, at the microlevel we speak in terms of *Dual agents*, at the mesolevel—of *coalitions*. Now, at the highest level of granularity we speak of *Dual formations* and *systems*. A DUAL formation consists of a big population of agents—on the order of hundreds or thousands in number. A DUAL system consists of all agents that are present at a given instant of time, regardless of whether they are active or inactive, permanent or temporary, etc.

Most of the agents and, therefore, most of the knowledge and processing in the architecture reside in the *Dual network*. Most agents in this network are permanent but additional temporary ones may be created during the computation and added to the total pool. Similarly, most links are permanent but additional temporary ones may be established. Thus, the topology of the network is relatively stable but not absolutely frozen.

The collection of all permanent nodes and links in the DUAL network comprise the *long-term memory* (LTM) of the architecture. It contains the system's knowledge (both declarative and procedural) about the world. The LTM is big—even for simple domains and situations one needs hundreds or thousands of agents.

In any given moment, however, only a small portion of this large formation is actually needed. DUAL provides special mechanisms, the most important of which is spreading activation, for effectively determining which agents (and coalitions) are *relevant* to the particular task and context. Recall that each agent has an activation level that is the system's estimate of its current relevance. So, by definition the *working memory* (WM) of the architecture consists of the set of all agents whose activation level exceeds a certain predefined threshold.

The working memory is the locus of almost all processing in DUAL and, therefore, we will consider it in more detail. An agent can enter the WM in two ways: permanent agents enter it whenever they become active enough to pass the threshold; temporary agents must be explicitly created and linked to the network by a specialized *node constructor*. Agents stay in the working memory as long as their activation level is maintained above the threshold. When a permanent agent "drops out" of WM, it returns back to *dormancy* and could reenter the WM later. Temporary agents, however, have no second chance.

In sum, the contents of the working memory may be expressed by the formula:

WM = active portion of LTM + temporary agents

The activation in the network originates from two special agents. The so-called *input node* models the influence of the environment.[1] The *goal node* is, in a very rudimentary sense, the medium of the "intentions" of the system. The human user attaches some agents to these nodes and thereby initiates the spread of activation in the DUAL network. The activation then propagates via the links and brings some agents from LTM to WM. There is a *decay* process which limits the total amount of activation and hence the size of the working memory.

3.2 Associative Memory-Based Reasoning

We now turn to the presentation of AMBR—a cognitive model built on the basis of the DUAL architecture. "AMBR" is an acronym for "Associative Memory-Based Reasoning" (Kokinov 1988, 1990, 1994a, 1997). The model has been conceived with very broad scope. Much of the work on it is still in progress. The current version is numbered AMBR2. Previous versions were AMBR1 (Kokinov 1994a) and AMBR2A (Petrov, 1997). We fully recognize the fact that the model as it currently stands and is reported here is incomplete. Here and now AMBR2 is an integrated model of analogical access and mapping. We view this version as only an intermediate stage of a bigger project.

[1] In future models it will be replaced by a whole formation—the *visual array*.

3.2.1 Main Ideas of AMBR

Since its initial conception (Kokinov, 1988) the AMBR model has advocated a set of ideas about human reasoning in general and analogy-making in particular. They have been distilled by Kokinov (1997) into the following three principles:

- **Integration.** The reasoning process cannot be partitioned into a sequence of independent stages performed by specialized module-like components. Rather, there are *subprocesses* that run together and each of them is potentially influenced by the rest. Each computational mechanism is responsible not only to produce its immediate result but also to create appropriate guiding pressures for other mechanisms. That is why AMBR is designed as an integrated model based on a parallel emergent architecture.
- **Unification.** Analogy is not a specific mode of reasoning. Rather, deduction, induction (generalization), and analogy are slightly different versions of the same uniform reasoning process. The same computational mechanisms are used in all cases—there is some sort of perceptual processing that builds internal representation of the problem being solved, there is some (sub)process that accesses relevant information from long-term memory, there is some (sub)process that tries to map the new problem to previous knowledge, etc. Deduction, induction, and analogy all fit into the same framework, the differences being in the outcome of the processing but not in the processing itself. Thus the term *deduction* applies to cases when the new problem happens to match with a general old schema, *induction* goes the other way around, and *analogy* applies when the two situations are at approximately equal level of abstraction. Conceptualized in this way deduction and induction are just two extremal (and hence very important) points on the analogy continuum. Therefore AMBR is designed as a general model of reasoning with emphasis on analogy-making.
- **Context-sensitivity.** Human reasoning is context-sensitive. Its outcome depends not only on the task and long-term memory knowledge but also on the environmental setting, recent activities of the reasoner, etc. AMBR is designed with the explicit aim to reflect this context-sensitivity of human thinking.

This book focuses on the first point from this list—integration. Deduction, induction, context, and priming effects are treated elsewhere (e.g., Kokinov, 1990, 1992, 1994a; Kokinov & Petrov, 2001; Kokinov & Yoveva, 1996). Our present goal is to explicate the principle of integration of subprocesses in more detail.

As discussed in Chapter 2, theories of analogy-making frequently partition the process into a sequence of stages (e.g., Gentner, 1989). The computational models that stem from these theories typically involve separate "engines," each of which works on its own and dovetails with the next. The output from the retrieval module is fed into the mapping module, whose output in turn is fed to the transfer module, etc. This "pipeline paradigm" is illustrated in Figure 3.2. Each module influences the next only through the data structures it passes to it (cf. Fodor, 1983).

A problem with this approach is that it depends on the tacit assumption that all these stages are separable. While this assumption definitely merits careful consider-

Fig. 3.2 Schematic description of the "pipeline" paradigm of analogy-making. The whole process is broken into a sequence of independent stages. They can interact only through the data structures (not shown in the figure) that each of them feeds to the next.

ation, it is questioned by a number of researchers (Chalmers, French, & Hofstadter, 1992; Kokinov, 1994a; Hummel & Holyoak, 1997). One question that has been especially controversial[2] is whether perception (representation building) and mapping are separable or not. This is a special case of the larger controversy about the putative modularity of the cognitive architecture (e.g., Anderson & Lebiere, 1998; Fodor, 1983; Rumelhart & McClelland, 1986). There is psychological evidence for various interaction effects between various components in Figure 3.2 (e.g., Catrambone & Holyoak, 1989; Hummel & Holyoak, 1997; Kokinov, 1994a; Kokinov & Petrov, 2001; Loewenstein, Thompson, & Gentner, 1999; Medin, Goldstone, & Gentner, 1993). Together with some conceptual arguments (Chalmers et al., 1992), this evidence rules out a strict serial pipeline. To our knowledge, no mainstream analogy researcher defends the extreme position depicted in Figure 3.2. What is being defended instead (e.g., Forbus, Gentner, Markman, & Ferguson, 1998) is the *reentrant modular* scheme depicted in Figure 3.3. It allows for the possibility to reenter an earlier stage of the pipeline and repeat the processing in light of intermediate results produced on a later stage. This interleaves the stages iteratively and can account for many interaction effects. The so-called *map-analyze cycle* (Falkenhainer, 1988, 1990b; Forbus et al., 1998) is a closely related idea.

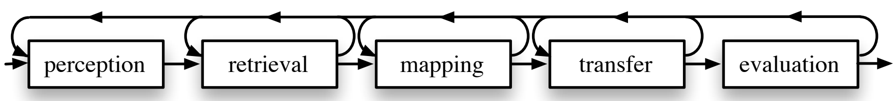


Fig. 3.3 Schematic description of the reentrant modular paradigm of analogy-making. The flow of control is more flexible than the strict pipeline in Figure 3.2. An earlier stage can be reentered and repeated in light of intermediate results produced on a later stage. This iterative interleaving can account for various interaction effects. Compare with Figure 3.4.

For example, a model can start with a hand-coded description of a target situation. This description is used as a memory probe to retrieve, via MAC/FAC, one or more candidate source episodes from long-term memory. One of these sources is then mapped to the target via SME. Up to this point, we have purely feed-forward processing consistent with the pipeline in Figure 3.2. Consider a case, however, in which the target contains a statement such as `hot(water-12)` that

[2] This controversy has subsided since 1998. In Section 2.2.6 (p. 19) we mention some recent developments in connection with Lovett's (2012) structure-mapping models of geometric analogies.

remains unmapped because the source uses a relational representation such as `temperature-of(milk-33, high-T)`. This can trigger some mechanism for *re-representation* (which can be considered a form of high-level perception) that uses domain knowledge to recast `hot(water-12)` as `temperature-of(water-12, high-T)`. This modified target description is then recycled back to the retrieval stage in Figure 3.3 and a new source episode may be retrieved from LTM, mapped to the target, and so forth. This map-analyze cycle can be repeated several times (Falkenhainer, 1988, 1990b; Forbus et al., 1998).

It is instructive to compare the above example with Hofstadter (1984) and Mitchell's (1993) Copycat system. Copycat emphasizes the intimate interplay between high-level perception and analogical mapping. Indeed, Chalmers et al. (1992, Abstract) argue that "perceptual processes cannot be separated from other cognitive processes even in principle." And yet, Copycat's internal organization can be reinterpreted to emphasize the similarity with the reentrant scheme in Figure 3.3. The basic elements of Copycat's processing are the so-called *codelets*—small pieces of code, each designed to perform a particular type of task. There are many codelets waiting to run at any given time, in a pool (called the *Coderack*) from which one codelet is chosen stochastically at every cycle. Now, the codelet types can be classified into categories that approximately map onto the traditional decomposition in terms of perception, mapping, and transfer. Mitchell (1993, Appendix C) lists several such categories, including "description-building, correspondence-building, and rule-building" codelets, among others. Given that only one codelet is allowed to run at any given time, each Copycat run can be reinterpreted as a very fine-grained interleaved sequence of perception, mapping, and transfer. In this interpretation, the differences with the view advocated by Forbus et al. (1998) are merely qualitative, not categorical. A Copycat run typically involves hundred of codelet steps, whereas structure-mapping models typically perform only a handful of iterations through the map-analyze cycle. The important similarity, however, is that both frameworks *interleave* perception, mapping, and transfer, albeit at very different grain sizes.

Chalmers, French, and Hofstadter (1992,) try to resist this interpretation by pointing out that "the situation-preception and mapping processes take place simultaneously... Codelets of both types are in the pool together." They also note that Copycat "makes no important distinction between structures built for the purpose of situation-perception ..., and those built for the purpose of mapping... Both types of structures are built up gradually over time, and both contribute to the program's current understanding of the overall situation." (Chalmers et al., 1992, p. 206). Also, these authors emphasize that the grain size of individual codelets is orders of magnitude finer than that of a problem-solving episode—most Copycat runs involve hundreds or even thousands of individual codelet steps. Thus, the sequentiality at the codelet level can be ignored and the macroscopic process analyzed in terms of parallel subprocesses. In fact, Hofstadter (1984, 1995) refers to this style of computation as *parallel terraced scan*.

This is true enough but also is, at least in part, a matter of interpretation. Moreover, the distribution of codelets in the Coderack undergoes macroscopic shifts during a run. Description-building codelets fire in large numbers early on, following by a "wave" of correspondence-building codelets, and finally rule-building codelets. This can be interpreted as a blurred version of Figure 3.3. In some ways, Copycat even imposes explicit sequential constraints on the flow of control. Certain types of codelets, for example, can run only after certain types of structures (e.g., successor groups) have been discovered. There are *processing events* at the macroscopic level too.[3] This is clearly seen on Copycat's showcase example—the celebrated **xyz** problem (Hofstadter, 1984, 1995; Mitchell, 1993). It involves the initial construction of a straightforward problem description that is later found to be incapable of generating a valid solution. In response to a snag like this, the system triggers certain emergency measures: the so-called *computational temperature* is clamped to a high value, *breaker* codelets are released, etc. On a successful run, these measures cause the dismantling of the original description and replacing it with an alternative. In the vocabulary of Forbus et al. (1998), this strongly resembles backtracking to the perceptual stage and then repeating the mapping with the new representation.

In our opinion, these two perspectives are not irreconcilable opposites, one of which must be wrong if the other is correct. Rather, we prefer to think of them as two complementary lenses for looking at the world and at the cognitive architecture. Through the modular lens, the world appears to consist of clear-cut objects belonging to well-defined categories. The mind appears modular or quasi-modular. Authors who habitually look through this lens (e.g., Forbus et al., 1998) tend to write about *domain-general engines* and *stages*, use formal logic in their models, and draw box-and-arrow flowcharts in their publications. The other lens is kaleidoscopic and not always quite focused. Through it, the world appears blurred and in flux. The mind is holistic, interactive, and creative. Authors who habitually look through this lens (e.g., Hofstadter, 1995; Linhares, 2000) tend to write about *fluid concepts* and *slippages*, aim to persuade through evocative metaphors rather than formal arguments, and eschew flowcharts in their publications.

We do not know what the world (and the mind) really are in themselves, unmediated by lenses or interpretative schemes (Kant, 1781/1997; Quine, 1968). We must use some lens at any one time, but we can switch among alternative lenses if we choose to do so. This is an excellent way to avoid theoretical myopia and we consciously try to practice it in our research whenever possible. One remarkable fact is that, even through the modular lens, the world does not appear *completely* modular. The reentrant connections in Figure 3.3 illustrate that. Conversely, even through the fluid lens, the world does not appear *completely* seamless. Copycat's macroscopic processing events illustrate that. The situation is summarized well by the Yin-Yang symbol on the left. Notice the conspicuous presence of a white dot in the middle of the black domain and vice versa. This prevents either side from winning decisively over the other. We prefer not to take sides but acknowledge that both points of view are informative, fruitful, and worthy of careful investigation.

[3] Copycat's successor Metacat (Marshall,1999, 2006) adds several components to the architecture. One of them is the *Temporal Trace* that stores an explicit temporal record of the most important processing events of this kind. Hitting a snag is a very salient event.

With this important qualification, most of this book describes the view seen through the fluid lens. AMBR adopts an explicitly interactionist approach and treats analog access, mapping, and transfer as parallel *subprocesses* rather than serial stages. These subprocesses are still ordered in time as suggested by the pipeline approach—for instance mapping can only start after at least a few agents are accessed from LTM. However, there is no requirement that a stage must end before the next one could start. On the contrary, subprocesses overlap considerably and can interact. This results in the cascade illustrated in Figure 3.4.

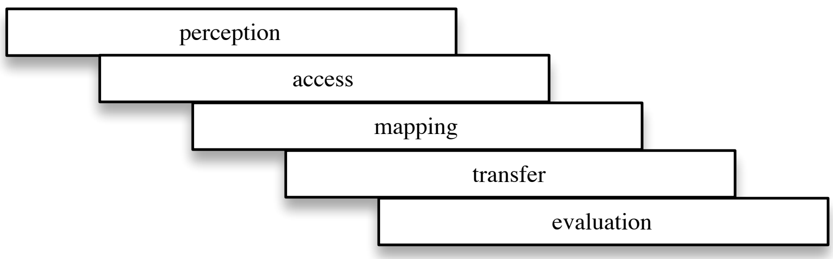


Fig. 3.4 Schematic description of the interactionist paradigm of analogy-making. There are subprocesses that overlap in time and can influence each other. Compare with Figure 3.3.

The interactionist approach seems problematic at first sight because each stage (or subprocess for that matter) depends on the result of the previous one. Indeed, how could the target problem be mapped to the source when it has not yet been even retrieved from memory?! It seems a logical necessity that the mapping comes after the retrieval. Similarly, the perceptual stage should come first, the transfer should follow the mapping, and so on.

AMBR overcomes this difficulty by representing information in smaller chunks. The model does not represent episodes as big units that are either manipulated wholesale or not at all.[4] Instead, it represents them as coalitions of small elements susceptible of piecemeal manipulation. This allows each subprocess to begin as soon as the previous one has produced some partial results. The analogy with Copycat's parallel terraced scan is obvious. The difference is that Dual postulates parallelism among the agents, whereas Copycat's codelets run one at a time.

As soon as AMBR's perceptual mechanisms have built internal representations of a few elements of the target problem, the access subprocess starts looking in the long-term memory for information that relates to these new elements. The concepts, propositions, episodes, etc. that are accessed in this way can now influence the perception of the target. In addition, they trigger the mapping subprocess which starts constructing the first tentative correspondences. If a promising candidate correspondence emerges, it could influence both perception and access. Gradually, all subprocesses are at work and more and more is perceived, accessed, mapped, transferred, and so forth.

[4] Therefore we prefer the term *analog access* to *retrieval*.

This is the upward motion of the "wave" of the reasoning process. Sooner or later the wave recedes. A stable representation of the target problem has been built and the perceptual mechanisms go off stage. A source episode wins the competition with alternative episodes from memory and the access subprocess diminish. One by one, all subprocesses terminate roughly in the order they have started. In this way there is something that could be characterized roughly as a sequence of stages. However, the boundaries between the AMBR "stages" are fuzzy and each one could in principle interact with any other.

Before closing this section we must make one final disclaimer. The 1998 version of the model implements only two of the subprocesses drawn in Figure 3.4: access and mapping. Thus AMBR2 avails itself to the same simplifying assumption that was criticized above. It artificially separates these two subprocesses from the rest. We admit this is a major flaw of the 1998 version. We hope, however, that the model is open-ended enough so that the missing components could be added without forcing radical changes in the existing ones. Chapter 7 contains some preliminary efforts in this direction. Until then, the exposition concentrates on what was actually implemented and running as of 1998.

3.2.2 Ambr Protagonists: Concepts, Instances, and Hypotheses

As any model based on the DUAL architecture, AMBR consists of nothing but agents of various kinds. They represent the knowledge and do all information processing in the model. Therefore the natural way to begin the presentation of AMBR is to introduce the various types of agents used by it.

Each AMBR agent is a DUAL agent and as such has a micro-frame (see Section 3.1.3.1). The micro-frame is a bundle of labeled slots, one of which serves to designate the type of the agent. The label of this slot is `type` and it is filled by a list of tags such as `:concept`, `:instance`, `:hypothesis`, `:temporary`, etc. These tags are used in conjunction with one another to account for the variety of agents employed by the model. For example, the type slot of some agent can be filled by the list `(:temporary :instance :relation)` thus stating that the agent in question is a temporary agent representing an instance of some relation.

There are rules that restrict the combinations among different type tags. For instance all agents of type `:hypothesis` are also `:temporary`. Therefore, despite the big number of possible type combinations there are only three major types of AMBR agents: *concept-agent*, *instance-agent*, and *hypothesis-agent*. These major types have subdivisions as illustrated in Figure 3.5.

Concept-agents (or *concepts* for short) represent classes of entities. The taxonomy of classes is represented by `subc` and `superc` links between the concepts. Some concepts are classes of *objects* such as `teapot` and `liquid-holder` while others represent relations such as `temperature-of` and `cause`. A concept agent may also have references to some of its instances, to be associatively related (via

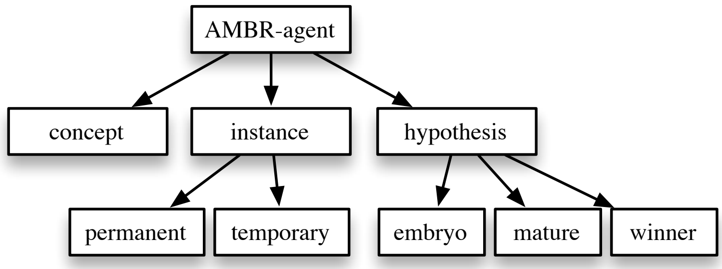


Fig. 3.5 Main types of AMBR agents.

`a-link`) to other concepts, etc. All concepts are permanent agents and form the backbone of AMBR's semantic memory.

Instance-agents (or *instances* for short) represent individual instances. Each instance agent has an `inst-of` slot filled by a reference to the concept agent representing the class of the instance (Figure 3.6). There are a several other slots with appropriate labels that relate the instance to other instances, concepts, or hypotheses. Concept and instance-agents are sometimes collectively called *entity-agents*.

```
liquid-holder:
  :type (:concept :object)
  :subc container
  :superc (teapot bottle cup)
  :a-link liquid

teapot:
  :type (:concept :object)
  :subc liquid-holder
  :instance (teapot-1 tpot-73)
  :hypoth teapot<->bottle

teapot-1:
  :type (:instance :object)
  :inst-of teapot
  :situation sit-ABC
  :hypoth (teapot-1<->bottle-3
           teapot-1<->bottle-4)
```

Fig. 3.6 Example of concept-agents, instance-agents, and some of the links between them. Each micro-frame has additional slots (not shown in the figure). All connectionist aspects are omitted.

Concepts and instances alike are characterized by one more tag in their type list—`:object`, `:relation`, or `:situation`. These tags are mutually exclusive. An `:object` tag means that the micro-frame represents some object or a class of objects. All agents in Figure 3.6 belong to this category. In contrast, the `:relation` tag is used to designate micro-frames that represent some relation. Such micro-frames usually have S-slots (see Section 3.1.3.1) that represent the arguments of the relation. The AMBR representation scheme allows to represent both *specific propositions* such as `made-of(teapot-1, metal-1)` and *general propositions* such as `made-of(teapot, metal)`. The details of the knowledge representation scheme are given in the next chapter.

Situation-agents (or *situations* for short) are a special kind of instance-agents. They are distinguished by the tag `:situation` in their `type` slots. Contrary to the name of the tag, such agents do *not* represent whole situations. Rather, they represent the spatio-temporal contiguity of a coalition of instances. Most instance agents are *affiliated* to some situation. The medium of this affiliation is a slot labeled `:situation` filled by a reference to the respective situation-agent. In the example above, the agent `teapot-1` is affiliated to `sit-ABC`. The other elements of this situation (both objects and propositions) will have the same reference in their respective slots. Thus the situation-agent that they all refer to represents the fact that all these instances have been perceived or inferred or remembered on the same occasion. On the other hand, there need not be any links from the situation agent to its elements. This is very important for the *decentralized representation* of situations in AMBR. It is a whole coalition of instance-agents that represent a particular problem, scene, episode, and so forth. Each participant is linked to only a few other elements and no one "knows" the entire situation as one encapsulated unit.

The mechanisms for analogy-making try to establish *correspondences* between instances from different situations, between their respective concepts, and so on. These correspondences are represented in the model by correspondence-agents (not shown in Figure 3.5), the most important type of which are the so-called *hypothesis-agents* (or *hypotheses* for short). Each hypothesis represents a tentative correspondence between two entities based on one or more *justifications*. The justification of a hypothesis is the reason for its creation and maintenance by the system. Each AMBR hypothesis must have a justification. (This is one big difference between AMBR and ACME.) The justification is either semantic or structural, represented by a concept or hypothesis agent respectively.

The hypothesis-agents are organized in a *constraint satisfaction network (CSN)*. Coherent hypotheses are connected with excitatory links while contradictory ones inhibit each other. This is the main instrument for achieving global consistency based on local computations. This approach follows the ACME model of Holyoak and Thagard (1989) but there are important differences (discussed later in this book). Hypothesis agents have a special activation function (cf. section 3.1.3.2) that gives them competitive power in the CSN.

Hypothesis agents are constructed on the initiative of their justifying agent. In the beginning of their life cycle they are created as *embryo hypotheses*. Those embryos that do not coincide with an existing hypothesis establish themselves and become *mature hypotheses*. They compete with the other hypotheses in the CSN and become either *winner* or *loser hypotheses*.

The presentation of the last few pages emphasized mostly on the symbolic declarative aspect of AMBR agents. Like all agents in the DUAL architecture, however, they are hybrid entities and have connectionist and procedural aspects as well (see Table 3.1). Different types of agents have different procedural knowledge and thus participate in the various computational mechanisms in the model.

3.2.3 AMBR *Mechanisms*

This section outlines the six basic mechanisms used in AMBR2: spreading activation, marker passing, constraint satisfaction, structure correspondence, rating, and skolemization. The presentation is intended to give a broad and relatively self-contained overview of these mechanisms and to show how they fit together. Chapter 5 provides a rigorous and much more detailed coverage.

3.2.3.1 Spreading activation

As stated earlier, each AMBR agent has a connectionist aspect and acts as a unit in a neural network (Section 3.1.3.2). It receives activation from the agents that interact with it, transforms this connectionist input according to its activation function, and in turn outputs activation to other agents along weighted links. Thus there is a pattern of activation over the whole population (or network) of agents. This activation originates from some special agents (Section 3.1.5) and then propagates the network. There is a decay factor and various thresholds that restrict the spread of activation.

This mechanism is of paramount importance in AMBR. It provides a dynamic estimate of the relevance of each individual agent. These estimates are then used by other mechanisms for various purposes. It defines the working memory of the model by bringing some agents above the threshold while keeping irrelevant ones away. This is the foundation of the access subprocess in analogy-making. Spreading activation also underlies the relaxation of the constraint satisfaction network.

Activation plays another very important role in AMBR (and DUAL in general). It is the energy supply for the symbolic aspect (Appendix C). More active agents work faster and are more visible to other agents (Section 3.1.3.3). In this manner, changes in the pattern of activation affect everything else in the model. This makes it dynamic, emergent, and context-sensitive (Kokinov, Nikolov, & Petrov, 1996).

3.2.3.2 Marker passing

Marker passing (MP) is the symbolic counterpart of spreading activation. It has been developed within the semantic network tradition (Quillian, 1966; Fahlman, 1979; Charniak, 1983; Hendler, 1988, 1989). In its most basic form it is a tool for answering the question, "Given two nodes in the network, is there a path between them?" The idea behind the marker passing is simple: the two *nodes of origin* are marked, they mark their neighbors, which in turn mark their neighbors, and so forth.

AMBR markers originate in instance-agents and are then passed by concept-agents upward in the class hierarchy. That is, markers can go only through links labeled `:inst-of` and `:subc`. For example, a marker can originate from `teapot-1` and then pass through `teapot`, `liquid-holder`, `container`, `artifact`, etc. Another marker starting from `bottle-7` could go through `bottle` and meet the first one in the concept-agent `liquid-holder`. The latter will detect this *marker intersection* and create a hypothesis that `teapot-1` corresponds to `bottle-7`. The concept node becomes the justification of the new hypothesis. In this way, the marker passing gives rise to semantically grounded hypotheses and triggers the constraint satisfaction mechanism.

The markers accumulate in the local buffers (see section 3.1.3.3) of concept-agents and provide a record of all instances of the particular class that are active at the moment. This information is then used by other mechanisms for various purposes.

3.2.3.3 Constraint satisfaction

The marker-passing and structure-correspondence mechanisms create hypotheses on the basis of local information only. The constraint satisfaction mechanism is responsible to achieve consistency at the level of whole coalitions. To that end, AMBR builds a *constraint satisfaction network (CSN)* with appropriate links between hypotheses. The pattern of activation in the CSN then gradually reaches a stable state in which a set of hypotheses emerge as winners while all others are suppressed.

In contrast with ACME (Holyoak & Thagard, 1989), the constraint satisfaction network in AMBR is tightly interconnected with the main network. This allows seamless integration with other mechanisms in the model. For example, suppose a particular hypothesis wins the competition and becomes highly active. Part of this activation spreads to the concept-agents involved in it. When the concepts become more active they process markers faster, which will tend to generate more hypotheses of the same kind. If the hypothesis is about instances, it will activate them and they in turn will support the other instances of the same coalition, etc.

Another important property of the constraint satisfaction network in AMBR is that it is built in a decentralized and incremental fashion. Individual hypotheses come one by one in the order of their creation, which reflects the system's current estimates of the relevance of the elements involved. This decentralized creation process raises the question of how to avoid duplication of hypotheses and to establish the links needed for the relaxation algorithm. This is the responsibility of the hypotheses themselves aided by the so called *secretaries*.

Each instance- or concept-agent has a secretary associated with it. The secretary is not a separate agent but a part of the respective agent itself. The job of the secretary is to keep track of the hypotheses involving the agent in question. It records them in the `:hypoth` slot of the agent (cf. Figure 3.6) and handles *hypothesis registration requests*.

Whenever an embryo hypothesis is born it contacts the secretaries of its two elements and requests registration. The secretaries receive these requests, consult their records, and send *secretary answers* to the hypothesis. There are several kinds of answers but basically they all belong to one of the following two major types. If the new hypothesis is a duplicate of an existing one, it is commanded to *resign* in its favor. The resigning hypothesis hands over its justification to the favorite and then fizzles out. In this way many hypotheses in the CSN have several justifications even though each of them is born with only one. The links to and from justifications are excitatory and connect the CSN with the main network.

The second major type of secretary answer is *establish*. It is sent to hypotheses that represent some novel correspondence. When the embryo hypothesis receives such answer it becomes mature and enters the competition with other mature hypotheses. The answer contains a list of the alternative hypotheses registered at the secretary. They are the rivals of the new one and it creates symmetrical inhibitory links with them. In this way each mature hypothesis becomes incorporated in the network. When it achieves this status it starts generating its own "child" hypotheses via the structure correspondence mechanism.

3.2.3.4 Structure correspondence

The structure correspondence (SC) mechanism generates new hypotheses on the basis of existing ones. It is also responsible for the excitatory links between coherent hypotheses. Either way, it fosters the systematicity of the mapping that emerges out of the constraint satisfaction network (Gentner, 1983).

There are several types of structure correspondence in AMBR: bottom-up, top-down, weak, etc. They are explained in detail in Chapter 5. This section only conveys the general idea by means of assorted examples.

Suppose there is a mature hypothesis involving two instance agents, e.g., `teapot-1<->bottle-3`. The bottom-up SC will create a new embryo hypothesis at the level of concepts. Namely: `teapot<->bottle`. If the instances are affiliated to situations, the structure correspondence mechanism will construct an embryo hypothesis about them too, e.g., `sit-ABC<->sit-XYZ`. These new hypotheses are likely to coincide with ones created earlier by some other agent. In these cases the secretaries of, e.g., `teapot` and `sit-XYZ` will detect the duplication and the redundant hypotheses will be forced to resign in favor of the older ones. Still, excitatory links between `teapot-1<->bottle-3` and the respective concept- and situation-level hypotheses will be established. This creates the pressure that instances of the same concept and/or the same situation are mapped consistently to instances of the other concept or situation, and vice versa.

The top-down SC applies when there is a mature hypothesis involving propositions. For instance, suppose that the agent `made-of-1` represents the proposition that `teapot-1` is made of `metal-1`. Suppose further that `made-of-3` states that `bottle-3` is made of `glass-3`. Then the hypothesis `made-of-1<->made-of-3` will generate the hypotheses `teapot-1<->bottle-3` and `metal-1<->glass-3`. (It will also generate bottom-up hypotheses such as `made-of<->made-of`, etc.)

The hypothesis `teapot-1<->bottle-3`, however, has probably been constructed already by the marker passing mechanism (because both are liquid holders). The secretaries will detect this duplication and the SC-generated embryo will resign in favor of the MP-generated mature hypothesis. In the end the latter will have two justifications: semantic and structural. Each additional justification improves the competitiveness of this hypothesis in the CSN.

3.2.3.5 Rating and promotion

Another responsibility of the secretary is to rate the relative success of each hypothesis on its secretary list. It checks at regular intervals who is the current *leader* among the hypotheses. That is, which one has the greatest activation level. The secretary maintains *ratings* for each hypothesis. Ratings are numerical values indicating how long the particular hypothesis has led the competition. When a hypothesis maintains a leading status long enough, it is *promoted* to winner status.

Thus, the rating mechanism promotes current leaders into final winners. This is done via a kind of competitive learning algorithm. The secretary performs *rating surveys* at regular intervals. Each survey detects the leader and increases its rating at the expense of the ratings of its competitors. The magnitude of the change is proportional to the margin between the activation levels of the leader and the second best hypothesis. When a particular rating reaches some critical level, the rating mechanism triggers the *promotion mechanism* for the respective hypothesis.

In addition to promoting winners, the rating mechanism also eliminates *losers*. When a particular rating drops too low and the activation level of the respective hypothesis is also low, the hypothesis is sent a *fizzle message* that causes it to drop out of the system. Non-leader hypotheses that maintain a reasonably high activation level are kept as plausible alternatives to the leader. In this way the constraint satisfaction network is trimmed of very implausible hypotheses without ruling out any possibility a priori. This adds another dimension to the dynamics of the CSN—its topology changes both by adding and removing nodes and links.

Still another function of the rating mechanism is to trigger the *skolemization mechanism* upon necessity.

3.2.3.6 Skolemization

AMBR skolemization is a technique for augmenting the description of some particular episode on the basis of general semantic information. This is an advanced topic that is discussed in detail in Chapter 5. This section provides an example that conveys the overall idea.

Suppose that the target situation contains a teapot and its material is explicitly represented: `teapot-1` is made of `metal-1`. Suppose further that `teapot-1` is mapped to `bottle-3` belonging to some other situation. The description of the latter, however, lacks explicit proposition about the material of `bottle-3`. Thus there is no counterpart of the target proposition `made-of(teapot-1, metal-1).`

The semantic memory, however, contains a *general proposition* that bottles are (usually) made of glass. These general proposition is represented by an instance of the relation `made-of`. This instance is not affiliated to any situation (cf. section 3.2.2) and one of its arguments is a concept-agent. For example, it might be of the form `made-of(bottle, prototype-glass)`. This proposition is handled by AMBR mechanisms in the usual way—it emits a marker, that marker intersects in the concept-agent made-of with the marker emitted by the specific proposition in the target, the marker intersection gives rise to a hypothesis, etc. Suppose that this *general hypothesis* wins the competition in the constraint satisfaction network (for lack of a better alternative).

The rating mechanism detects that the leading hypothesis involves a general proposition and triggers skolemization. The latter will construct a *skolem proposition* that concretizes the general proposition. In the example above, the mechanism will create *skolem instances* of the concepts `made-of` and `glass`. No instance of bottle is needed because the recipient situation already has one as indicated by the marker from `bottle-3` stored in the local buffer of `bottle`. The final outcome of the skolemization is that the material of `bottle-3` is taken by default to be `sk-glass-3`, where `sk-glass-3` is a skolem instance of the concept `glass`. This new agent affiliates to the situation containing `bottle-3`. It then emits a marker, which will intersect in the concept material with the marker originating from `metal-1`. This will create the semantically-grounded hypothesis `metal-1<->sk-glass-3` which enters the competition with high chances of success as `teapot-1` is already mapped to `bottle-3`.

3.2.4 Overview of a Run

This final section pulls everything together and shows how the computational mechanisms described above can be applied to the task of analogy-making.

In the present version of AMBR, the work on a problem begins with a hand-coded representation of the target situation. Some of the agents that participate in the (decentralized) description of this situation are attached to the special nodes that are sources of activation in the model. The goal element(s) are attached to the goal node; some of the other elements are attached to the input node, thus mimicking the perceptual mechanism. The input list can also include elements that do not belong to the target situation, thus modeling the external context. It is possible that target elements are presented to the system not simultaneously but incrementally, giving rise to various order effects.

Once the target elements are connected to the source nodes, the associative mechanism begins to operate. The activation spreads through the long-term memory and brings relevant conceptual and episodic information to working memory. Shortly after, the marker-passing mechanism joins in as instance-agents emit markers upon entering the WM. The markers begin propagating the active portion of the network.

Marker intersections provoke the construction of hypothesis-agents and thereby trigger the constraint-satisfaction mechanism. After consulting the secretaries, the hypotheses initiate the structure-correspondence mechanism. The secretaries register more and more hypotheses and rate their relative success.

Gradually, a number of agents enter the working memory. The activation does not spread unrestricted, however, and the intensity of memory access declines as the decay of activation prevents the nodes that are too remotely relevant from passing the threshold. Usually, two or three situations are retrieved in full and a few others only partially. These are the candidates for base analogs. In addition, the relevant concept-agents are also active and ready to guide the mapping.

The associative mechanism never stops completely because agents occasionally get in or out of the working memory. Moreover, the associative mechanism is responsible for controlling the speed of the symbolic aspect as well as for settling the constraint satisfaction network.

Meanwhile, the marker-passing mechanism has generated several hypotheses. In turn, they have created additional hypotheses via the structure-correspondence mechanism. The CSN has thus become fairly elaborate and winning correspondences begin to emerge. The hypotheses standing for such correspondences are promoted to winners. This makes them even more active and provides strong support for the respective entities in the main network. In this way, the base situation that best matches the target is fully and unambiguously accessed. All its elements enter working memory. The skolemization mechanism adds even more elements if such are needed to better match the target.

Sooner or later all secretaries of the target promote their winners. The mapping constructed by the model can be read from the set of winner hypotheses. (In fact, the system maintains a "working answer" throughout the whole run. It is often unnecessary to wait for the end.) The mechanisms for transfer should have been triggered at that time. They are not yet implemented in the current version of the model, however.

It should be emphasized that everything described so far happens as a result of a dynamic emergent process. There is no central executive that controls the operation of the system. Instead, a multitude of micro-agents interact with their immediate neighbors and their local activities give rise to macroscopic phenomena that an external observer could interpret as analog access, mapping, etc.

Chapter 4
Knowledge Representation

This chapter is devoted to knowledge representation in AMBR. It shows how the DUAL representation scheme (Kokinov, 1988, 1994a; Petrov 1997) is actually put to work in the model.

4.1 Domain

This section introduces the domain used for the simulation experiments reported in this book. It should be noted that AMBR mechanisms do not depend on this particular domain. It is simply a convenient testbed for the model.

The domain involves simple everyday tasks in a kitchen. It deals with concepts such as `water`, `high-temperature`, `baking-dish`, and `food`. Typical situations include heating and cooling liquids, boiling eggs, etc. For example, the knowledge base contains the following episode:

There is a teapot and some water in it. The teapot is made of metal and its color is black. There is also a hot-plate. The teapot is on the plate. The temperature of the plate is high.

The goal is that the temperature of the water is high.

The outcome of this arrangement is that the temperature of the teapot is high because it is on the hot plate. In turn, this causes the temperature of the water to be high.

Equipped with episodes of this kind, AMBR is then presented with situations in which some of the objects necessary for achieving the goal are missing. For instance, the goal is to heat some milk in a teapot but no heating source is mentioned. Another kind of problem is to give all the objects in place and then ask what will happen, etc.

We admit that these problems are very modest by human standards. AMBR does not attempt to solve the radiation problem or to understand the Rutherford atom. Indeed, its abilities are even more modest than suggested by the description above. Despite appearances, AMBR has no idea about what "real world" water actually looks like. It has so little "knowledge" that in fact it works in a micro-domain and this should be taken into account when evaluating its performance (Chalmers, French & Hofstadter, 1992). We argue, however, that reliance on such micro-domains is methodologically justified.

4.2 Desiderata

As simple as it is, AMBR's domain reveals a number of requirements that the knowledge representation scheme must meet to allow successful problem solving. We believe that the same requirements hold for any domain and become increasingly important for more complex ones.

4.2.1 Rich Descriptions of Episodes

The episodes stored in the long-term memory should be described in enough detail. This is not crucial for the mapping process but is absolutely necessary for transfer and evaluation. In particular, the causal structure should be quite elaborated. In the example above, the hot plate is important for heating the water but the color of the teapot is not. Without enough causal information the model could assume that in order to heat milk it should put it in a black teapot.

There is an additional complication—the rich representation of the source analog hinders its mapping to the target problem. The description of the latter is normally quite incomplete and, therefore, there are many elements in the source that do not have any counterpart in the target. Paradoxically, if there are two potential source analogs in the LTM, the one with sketchier description will map better to the target even though it may well be less useful for solving the problem. In the extreme case, a source analog that has absolutely nothing more than the target will achieve perfect match but zero utility.

One way around this obstacle is to partition the source descriptions into initial conditions, goals, solutions, etc. The target problem could then be mapped selectively to the appropriate sections of the base (e.g., Holyoak & Thagard, 1989). This approach, however, seems too rigid as it precludes any possibility of mapping elements from different compartments. Human problem solving does not observe such boundaries. For instance, the goal of one situation could map to some unintended side effect in another.

4.2.2 Semantic Knowledge

Most analogy models use semantic knowledge for two purposes only—as a source of constraints on mapping and for similarity-based retrieval of episodes from long-term memory (e.g., Falkenhainer, Forbus & Gentner, 1986; Holyoak & Thagard, 1989; Keane & Brayshaw, 1988; Kokinov, 1994a; Hummel & Holyoak, 1997). It is clear, however, that human analogy-making uses semantic knowledge in much more diverse ways than that.[1] Even in our tiny domain the general fact that plates are heat sources and as such are used to heat things is of obvious importance when asking how to heat water. Still, such knowledge goes unused if the model deals exclusively with finding correspondences between two episodes.

One challenge for the research community is to design mechanisms for using semantic knowledge in analogy-making. The skolemization mechanism proposed in Section 5.7 is a step in this direction. Note that one of the implications of this approach is that it blurs the boundary between deductive and analogical reasoning (which, according to our views, is quite fuzzy anyway; Kokinov, 1988).

4.2.3 Flexibility and Re-Representation

A third desideratum closely related to the first two is that episode representations should be flexible. Facts that follow from general rules need not be stored explicitly in each episode. They may be omitted from the representation and added again later upon necessity. The model should be able to switch between alternative representations such as `hot(X)` vs. `temperature-of(X,high-T)` or `left-of(X,Y)` vs. `right-of(Y,X)`.

There is a tacit assumption in analogy research about the asymmetry between source and target situations. It is quite often taken for granted that the target must conform to the source, whereas the latter remains static. In our view the process of analogy-making consists of bringing both situations closer to each other by modifying either one when appropriate. In this way one and the same base episode can map to various targets and each mapping entails reconceptualization of the base.

4.3 Representation of Concepts and Instances

With full awareness that AMBR agents are nothing but ungrounded symbols (Harnad, 1990), we follow the common AI practice to use mnemonic names like `milk`, `taste-of`, and `cause`. Those names are irrelevant for the model itself; the program would work just as well (or not-so-well) had the agents been named `ag001`,

[1] Moreover, research on memory suggests that remembering old episodes is often a matter of reconstruction rather than rote retrieval. Section 5.7.1 provides pointers to this literature.

`ag002`, etc. Indeed, the first version of AMBR (Kokinov 1994a) used such uninformative names. It was very instructive from a philosophical point of view as it laid bare how little "knowledge" the program actually had. It was not very practical, however, because it hindered enormously the process of developing, tuning, and documenting the model. Moreover, the use of English words as labels is not completely devoid of philosophical justification. According to the *causal-historical theory of reference* (e.g., Kripke, 1980; Devitt, 1999), the reference of natural-kind words such as "water" is transmitted by a causal chain that goes back to one or more events akin to a naming ceremony. Philosophers interested in this have emphasized the "reference borrowing" links in the chain: in acquiring a word or concept from others we borrow their capacity to refer, even if we are ignorant of the referent (Kripke, 1980). For instance, I use the words "mule" and "hinny" competently, even though I would be at a loss to discriminate the two animals in real life. I do not live on a farm and so I borrow the reference of these particular words from the appropriate domain experts. Analogous borrowing occurs for terms for medical diagnoses, brain areas, and many, many other things. So, it can be argued that, in an *extremely* passive and rudimentary manner, the AMBR model participates in the English speaking community. It borrows from its human creators the reference of *all* symbols in its lexicon, just as human speakers borrow the reference of *some* (or, perhaps, even *most*) of the words they use to communicate with each other. A critic (e.g., Searle, 1980) might insist that the *other* words—the ones I do have firsthand experiential grounding for—make all the difference. Still, in the process of developing, tuning, and documenting the model, I need to "communicate" with it, again in an extremely one-sided and rudimentary manner. It is far easier for me to "teach" AMBR a few hundred English words than to learn their AMBR translations (`ag001`, `ag002`, ...) myself. Notice that the claim here is not that AMBR *understands* English but that, in a one-sided and rudimentary manner, it *participates* in the English-speaking community. This community consists mostly, though not exclusively,[2] of human beings who shoulder the symbol-grounding burden.

As introduced in Chapter 3, AMBR uses *concept-agents* to represent classes of entities in the micro-domain and *instance-agents* to represent individual instances. The taxonomy of classes is represented by `subc` and `superc` links between concept-agents. Each class may be linked to zero, one, or more super- or subclasses, different links possibly having different weights. Similar links—`inst-of` and `instance`—relate instance-agents to concept-agents. Figure 4.1 illustrates.

Some instance-agents are temporary. They do not belong to the long-term memory of the system. They are constructed by some inference or (putative) perceptual mechanism and "live" as long as they stay in the working memory (Section 3.1.5). In the current version of AMBR, temporary instance-agents are used to represent the target situation and for Skolem instances. In contrast, permanent instance- agents are used for all LTM episodes. Concept-agents are always permanent.

"Top-down" links from concepts to instances deserve special attention. These `instance` links play a key role for analog access in AMBR. As discussed in Petrov

[2] The Google search engine, for example, has a strong claim for such participation.

```
liquid-holder:
  :type (:concept :object)
  :subc (container 1.0)
  :superc ((teapot 0.3)
           (bottle 0.3)
           (cup 0.2) )
  :a-link (liquid 0.5)

teapot:
  :type (:concept :object)
  :subc ((liquid-holder 0.8)
         (kitchen-equipment 0.2) )
  :instance ((teapot-1 0.2)
             (tpot-73 0.1) )

teapot-1:
  :type (:instance :object)
  :inst-of (teapot 1.0)

agent007:
  :type (:temporary :instance :object)
  :inst-of (teapot 1.0)
```

Fig. 4.1 Example of concept-agents, instance-agents, and some of the links between them. Each micro-frame has additional slots (not shown in the figure). Note that each reference has a weight used for spreading activation. Compare with Figure 3.6.

(1997, section 4.1.3), however, it is both psychologically implausible and computationally disadvantageous to maintain links to *all* instances of a given concept. Instead, there are such links to only *some* of them. This “privileged set” varies as a function of time (though much more slowly compared to other events in the model). Thus, at any given moment each concept supports only a few of the vast number of instances potentially available in the episodic memory.

The exact mechanisms for this are open for discussion and are not implemented in the current AMBR version. The main idea is to give priority to recently used instances, prototypes, or other salient agents without excluding anyone a priori. At present, there is an implemented tool for generating (static) variants of the knowledge base. The simulation experiments reported in this book are based on hundreds of such variants of the same “core” knowledge base. The set of `instance` links of each concept is generated by random sampling. The instance agents have unequal odds of being included in the sample, thereby approximating the mechanism suggested above.

Some instance-agents are distinguished by the tag `:prototype` in their `type` slots. These *prototype instances* are used as arguments in the so-called general propositions (discussed below).

4.4 Representation of Propositions

Individual AMBR agents are small and their micro-frames cannot represent much. Therefore, even relatively simple units of the representation such as propositions need be represented by a *coalition* of agents (Section 3.1.4). In the case of propositions, such coalitions are small and very tight.

```
color-of
  :type (:concept :relation)
  :subc physprop-rel
  :slot1
    :subc (physprop-rel . :slot1)
    :c-coref object
  :slot2
    :subc (physprop-rel . :slot2)
    :c-coref color

color-of-1
  :type (:instance :relation)
  :inst-of color-of
  :slot1
    :inst-of (color-of . :slot1)
    :c-coref teapot-1
  :slot2
    :inst-of (color-of . :slot2)
    :c-coref green-1

teapot-1
  :type (:instance :object)
  :inst-of teapot
  :c-coref (color-of-1 . :slot1)

green-1
  :type (:instance :object)
  :inst-of green
  :c-coref (color-of-1 . :slot2)
```

Fig. 4.2 A coalition of four micro-frames representing the proposition `color-of-1 (teapot-1, green-1)`. All connectionist aspects are omitted.

There is an agent that represents the *head* of the proposition. In Figure 4.2, this is the micro-agent `color-of-1`. It has the tags `:instance` and `:relation` in its `type` slot and is an instance of the concept `color-of`. The arguments of the relation are represented by S-slots in the heading micro-frame. Each S-slot has several facets (see Section 3.1.3.1).

The arguments (or roles) of the relation are bound to the actual entities involved in the particular instance of that relation by *conceptual coreferences* (or `c-coref`'s for short). In Figure 4.2, the first S-slot of the micro-frame `color-of-1` has a facet labeled `c-coref` and this facet is filled by a reference to the agent `teapot-1`. In a nutshell, the existence of `c-coref` links between two micro-frames (or their slots) means that the two frames represent two complementary aspects of the same entity. In our example, these links represent the fact that `teapot-1` and the first argument of `color-of-1` are one and the same thing. Similarly, the second argument of the relation is bound to (a reification of) the particular shade of green that happens to be the color of `teapot-1`.

Note that S-slot labels (`slot1`, `slot2`, etc.) in any proposition are absolutely arbitrary and by no means serve to define the arguments within the relation. In the example above, it is *not* crucial that `slot1` is the object and `slot2` the color. The slots in the instance-agent `color-of-1` could just as well be labeled `slot5` and `slot6` (or even `slot2` for the object and `slot1` for the color). Moreover, two instances of the same relation could use entirely different labels. Each S-slot has a `inst-of` or `subc` facet that points to the corresponding slot in the parent concept. This gives distinct advantages over a positional notation (in which interpretation of arguments depends on their order in the proposition). An AMBR proposition effectively has a *set* of arguments, not an ordered tuple.[3] Thus it is possible that two slots in a “child” inherit from the same slot in the “parent”, or that some parent slot is left unused, etc. As we shall see, this provides for great flexibility in analogical

```
each-snowdrop-is-white
  :type (:instance :relation)
  :inst-of color-of
  :slot1
    :SUBC (color-of . :slot1)
    :c-coref snowdrop
  :slot2
    :INST-OF (color-of . :slot2)
    :c-coref prototypical-snowdrop-white

snowdrop
  :type (:CONCEPT :object)
  :subc flower
  :c-coref (each-snowdrop-is-white . :slot1)

prototypical-snowdrop-white
  :type (:PROTOTYPE :instance :object)
  :inst-of white
  :c-coref (each-snowdrop-is-white . :slot2)
```

Fig. 4.3 Example of a general proposition. Note the use of tags in the TYPE slots and SUBC vs. INST-OF facets. Compare with the specific proposition shown in Figure 4.2.

[3] Analogously, connectionist systems (e.g., Hummel & Holyoak, 1997; Plate, 2003; Smolensky, 1990) represent propositions as sets of *role-filler bindings*.

mapping. It is possible to map propositions with different number of arguments and to map two arguments from one proposition to a single argument in another.

The proposition illustrated in Figure 4.2 is a *specific proposition*—it relates two specific instance agents. Such propositions typically encode episodic information. In addition, AMBR's knowledge base contains *general propositions* encoding semantic information. Their arguments are concepts or prototype instances. For example, the proposition in Figure 4.3 represents the general fact that each snowdrop is white. The skolemization mechanism uses such general propositions to create specific *Skolem propositions* about specific exemplars of the general class (Section 5.7).

4.5 Representation of Situations

This section compares two alternative strategies for representing situations (problems, episodes) for the purposes of analogy-making. It considers their advantages and disadvantages and presents the approach adopted in AMBR2.

4.5.1 Centralized Representation: Pros and Cons

We speak of *centralized representation* of a situation when there is an explicit data structure enumerating all elements belonging to it. The data structure may be a list, frame, or something else. The criterion is whether the system has a means to go through all members of the situation and only those members, without searching.

Centralized representations simplify the mechanisms of the model. Each situation has distinct identity and can be handled as a unit. It can be put in explicit competition with other situations, inspected for members with a given property, etc.

These computational advantages explain the widespread use of centralized representations in analogy models. For instance, MAC/FAC (Forbus, Gentner, & Law, 1994) maintains two data structures for each episode (*memory item* in original terms). *Content vectors* are used for cheap preliminary screening based on dot products. The SME analogical matcher (Falkenhainer, Forbus & Gentner, 1986) then takes s*tructured descriptions* to produce a numerical score for each episode that has passed the first (MAC) stage. ACME (Holyoak & Thagard, 1989) and ARCS rely on predicate calculus descriptions to construct hypotheses for a constraint satisfaction network. The Incremental Analogy Machine (Keane & Brayshaw, 1988) starts with a predicate calculus description and looks for the group of predicates that have the most higher-order connectivity between its elements. It then picks up a *seed* from this *seed group* and goes to the description of the other situation searching for a *seed match*, etc.

LISA (Hummel & Holyoak, 1997) is a very interesting case. It employs distributed representation of concepts, localist representation of propositions (*P* and *SP units* in LISA terms), and centralized representation of situations (or *analogs*). Each situation can be in one of three modes: *driver*, *recipient*, or *dormant*. The propositions in the driver are selected to become active in the *phase set* one at a time according to a fixed schedule specified by the human user. Recipient and dormant propositions respond to the patterns generated by the driver. Only recipient units, however, participate in analogical mapping. Thus in order to enter the mapping, an analog from LTM must first switch from dormant to recipient mode. This transition occurs in a stop-and-go fashion—all members of the situation are simultaneously flipped from one mode into the other. In the 1998 implementation of the model this was done by the human user (Hummel, personal communication, January 1998).

The first version of AMBR (Kokinov, 1994a) also used centralized representations of situations. There was a micro-frame standing for each situation as a whole. This micro-frame was called *head* and brought together all agents that comprised the representation of the situation. There was one S-slot for each element—object or relation. The head was linked to *all* elements and some elements were linked back to the head, thus creating a network like the one schematized in Figure 4.4. In addition to the "vertical" links between the head and its elements, there were many "horizontal" links between the elements themselves (not shown in the figure).

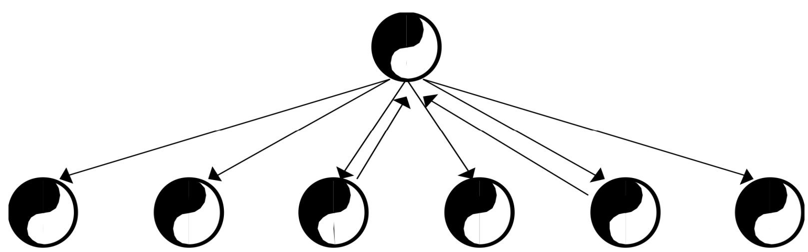


Fig. 4.4 Schematic outline of centralized representation of a situation as used in the first version of AMBR (Kokinov, 1994a). There is one *head* connected to all elements of the situation. Compare with Figure 4.5.

This representational decision provided ready solutions to many issues faced by the model. It was clear who was "responsible" for the situation. To begin working on a problem, for example, it was sufficient to put the head on the goal list. To decide which base analog "won," it was sufficient to compare the activation levels of the heads. The task of mapping one problem to another was reduced to a task of establishing slot-to-slot correspondences between two micro-frames. After the correspondences had been found, it was clear which elements of the source were unmapped and thus were potential candidates for transfer, etc.

However, each of these advantages can be viewed as a disadvantage at the same time. From a psychological point of view, it is controversial whether each episode in the LTM has such distinct and clear-cut identity. To illustrate, it is comfortable to suppose that *Hamlet* and *West Side Story* are salient and well-defined chunks for many people. It is acceptable to suppose that the radiation problem (Dunker, 1945) is a sufficiently self-contained chunk for some psychologists and a few of the participants in their experiments (Gick & Holyoak, 1983). The problems used to test AMBR, however, deal with mundane episodes such as boiling a pot of water. Most of the situations fall into this final category and it is far from clear whether they are represented in such neat and orderly manner. The centralized representational scheme seems incompatible with the numerous cases of omission, intrusion, blending, and interference in human memory recall (Kokinov, 2003, 2006; Kokinov & Petrov, 2001).

Second, centralized representations tend to be too static and inflexible because it is difficult to add or remove elements dynamically. They are also "flat" in the sense that all members participate on an equal footing. Special measures (e.g., differentiated link weights) are needed to make some elements more salient or pragmatically more important than others. Even in these cases, however, an item is either *always* in the situation or not at all. With bigger situations (cf. Section 4.2.1) this could lead to a mild version of the frame problem (McCarthy & Hayes, 1969).

In addition to these considerations, which in our view apply to all cognitive models, there are other problems with centralized representations that are particular to AMBR. The head micro-frame becomes unwieldy because it has too many slots. Even the simple situations used in the simulation experiments so far require at least a dozen S-slots in the head. For realistic situations this number would be on the order of one hundred. When the number of slots is that big, however, McCarthy & Hayes' (1969) frame problem appears again—it becomes necessary to specify which of the many elements are relevant to the task at hand. Furthermore, head frames violate the architectural requirement that DUAL agents should have only a few slots. Worst of all, the fan-out effect makes the connectionist mechanism very inefficient. Even when the head is very active it fails to activate its children because the weight of each individual link is very small (due to normalization). When (and if) this finally happens, there comes another problem—the coalition becomes so stable that it never leaves the working memory because the reverberation is stronger than the decay.

In response to these problems, the newer versions of AMBR (starting with AMBR2A, Petrov, 1997) have abandoned the centralized representation used by their predecessor. The shift to decentralized representations poses problems in its own right but also offers a number of substantial improvements.

4.5.2 Decentralized Representation: Pros and Cons

We speak of *decentralized representation* of a situation when there is no explicit data structure enumerating all elements belonging to it. This term should not be confused with the *distributed* representations prevalent in connectionist research. AMBR uses localist representations of individual elements and decentralized representations of situations.

The main idea of decentralized representations is to represent the situation as a coalition of micro-frames without designating any of them as a center (Figure 4.5). It is possible, though not required, that some (salient) coalitions have a head, but

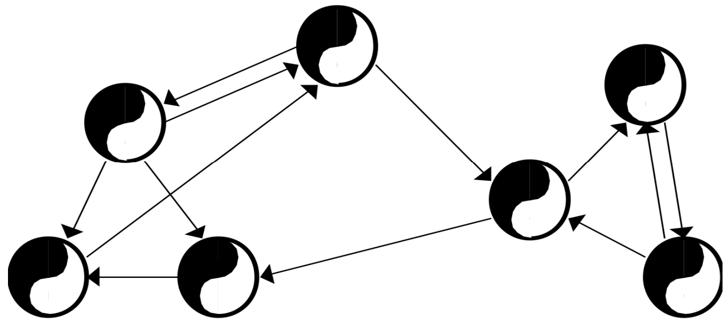


Fig. 4.5 Schematic outline of decentralized representation of a situation. There are many interconnected agents, none of which is in privileged position with respect to the others. Compare with Figure 4.4.

even in these cases the head is *primus inter pares.* It is not special in any way and do not have access to all elements of the situation.

With decentralized representations, the principal unit of analysis is the *coalition* (or *meso-frame*, see Section 3.1.4). This is an emergent entity which allows for great flexibility. It is easy to add new elements as they do not have to be "registered" anywhere. Thus, it would be easier to design a perceptual mechanism that incrementally builds such representations. As there are no fixed and predefined representation rules, each particular situation can be described in a way that is most suitable for it. Each micro-frame (including the head, if any) can have only a few slots and yet it is possible to represent big situations.

Decentralized representations can be rich and detailed enough to support analogical transfer and evaluation. Thus they meet the criterion presented in Section 4.2.1. At the same time, they can map successfully to impoverished and incomplete targets. This can be achieved when the mechanisms for access and mapping cooperate in the following way: The target problem acts as a driver and activates selected elements of several situations in the long-term memory. The full description of each of these potential source analogs can be very rich. At first, however, only a small fraction of the coalition members enter the working memory. These are the elements that are semantically similar to the target, plus their closest coalition partners. Thus the working memory contains descriptions of comparable complexity—the impoverished target and two or three *partially activated* sources. This commensurability is favorable for the mapping mechanisms and they start building correspondences. If a source analog matches the target well, its elements receive additional support and become more active. In turn, this gives them resources to bring more coalition partners into the working memory. The analog that has emerged as the winner can unfold its rich representation. Now the working memory contains an impoverished target and an elaborate source. The task for the mapping mechanisms thus becomes more difficult but they are aided by the initial correspondences that have had time to stabilize. When most (but not necessarily all) target elements have found their counterparts, the transfer and evaluation subprocesses can begin. They can rely on the rich representation of the source to create plausible inferences in the target.

Of course, the advantages of decentralized representation come at a price: situations no longer have guaranteed and easily available identity. This is good from psychological point of view, as it offers possibilities for modeling complex analogies, blends, and so forth (Grinberg & Kokinov, 2003; Kokinov & Zareva-Toncheva, 2001; Turner & Fauconnier, 1995). From computational point of view, however, decentralization of representations increases the complexity of the mechanisms that operate on them. Classical top-down algorithms must give way to a decentralized and emergent mode of processing. The individual elements have to take the initiative and do the job themselves instead of being passive data manipulated from outside. This poses difficult issues about synchronization, coherence, conflict resolution, resource allocation, etc.

4.5.3 Ambr Situations in Detail

We close this chapter with a brief description of the concrete representational scheme used in the 1998 version of AMBR.

As stated already, AMBR situations are represented as coalitions of agents. All situation elements are instance-agents, permanent or temporary (see Section 3.2.2). Most of them represent the objects and propositions in the situation. There are, however, a few agents that stand for different *states* within the situation. States loosely bind several elements together and are useful for explicating the causal structure of the situation. For instance, there usually are an initial state, goal state, and end state. They are distinguished by tags in their `modality` slots—`:init`, `:goal`, or `:result`. The initial state is often divided into overlapping substates.

States are instance-agents with S-slots that point to some of the members of the state. Thus they resemble propositions with arguments (cf. Figure 4.2). Not all agents that could be considered members of a particular state need be explicitly mentioned, however. An element could be listed in zero, one, or more states, including states of different types (e.g., `:init` and `:goal`). Each element may have one or more state-related tags.

In turn, states are themselves arguments of relations such as `cause`, `follows`, and `prevents`. If we turn back to the example from Section 4.1, the situation presented there could have an initial state that lists the three objects involved: `teapot-1`, `water-1`, and `hot-plate-1`. Another state-like agent combines the propositions that the teapot is on the plate and the plate is hot. This state is a left-hand argument of a causal relation stating that under these circumstances the teapot is also hot, etc.

Each situation in AMBR2 has a *situation-agent* associated with it. This is the most centralized aspect of the current representational scheme. Still, the situation agent is not equivalent to the *head* from previous versions. It bears very little representational content—it only embodies the spatio-temporal contiguity of the elements of the scene or episode. Situation agents are ordinary instance-agents. Their sole peculiarity is the tag `:situation` in their type slot.

Each individual member has a `situation` slot filled by a reference to the situation agent. We say that the instance is *affiliated* to the situation. The situation agent itself, however, has at most associative links to a few members.[4] It is not possible to reach the members if given only the situation agent. It is possible, though, to determine whether two instances belong to the same situation. To that end, each element should enter the working memory on its own and "claim membership." This arrangement resembles the relationship between instance-agents and concepts.

[4] These a-links are used in DUAL for spreading activation only. They are ignored by the symbolic aspect.

Chapter 5
AMBR Mechanisms at Work

5.1 Sample Problem

This chapter presents a detailed description of AMBR mechanisms on the basis of a concrete example. We will use a target problem from the domain outlined in Section 4.1. It is about cooling milk in a teapot:

Target problem CM1[1] (Cooling Milk, variant 1): *There is a teapot and some milk in it. The teapot is made of metal.*
The goal is that the temperature of the milk is low.

This situation is represented in the current knowledge base by eleven instance-agents. Seven of them are illustrated in Figure 5.1. The representation also contains agents for the init and goal states, etc. Note that no cooling object (such as a refrigerator) is included in the original description of the problem.

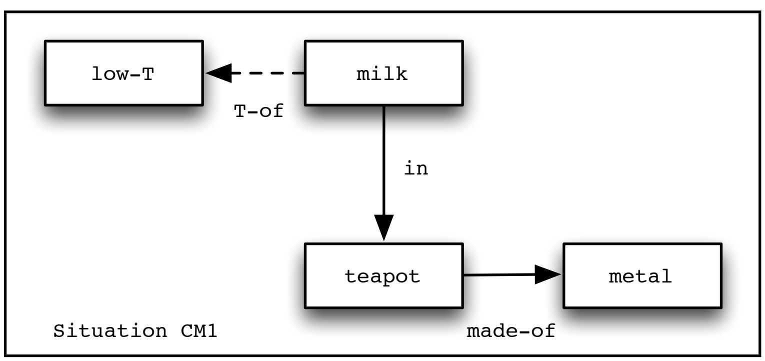


Fig. 5.1 Schematic representation of the target situation described in the text. Objects are depicted as boxes and propositions as arrows. Not all elements of the actual representation are shown.

[1] The episodes in the knowledge base are labeled by trigrams such as CM1 and FDO. These trigrams appear in the transcripts listed in this chapter.

In one of the many runs performed with the model, this problem happened to match to a long-term memory episode related to heating food in an oven (Figure 5.2). This particular run will serve as an illustration of the various mechanisms of the model.

Base situation FDO (Food on a Dish in an Oven): *There is a baking dish and some food on it. The shape of the dish is rectangular. There is also an oven. The dish is in the oven. The temperature of the oven is high.*

The goal is that the temperature of the food is high.

The outcome is that the temperature of the food is high. The fact that it is on the dish and the dish is in the oven entails that the food itself is in the oven. In turn, this causes the food to be hot, as the oven is hot.

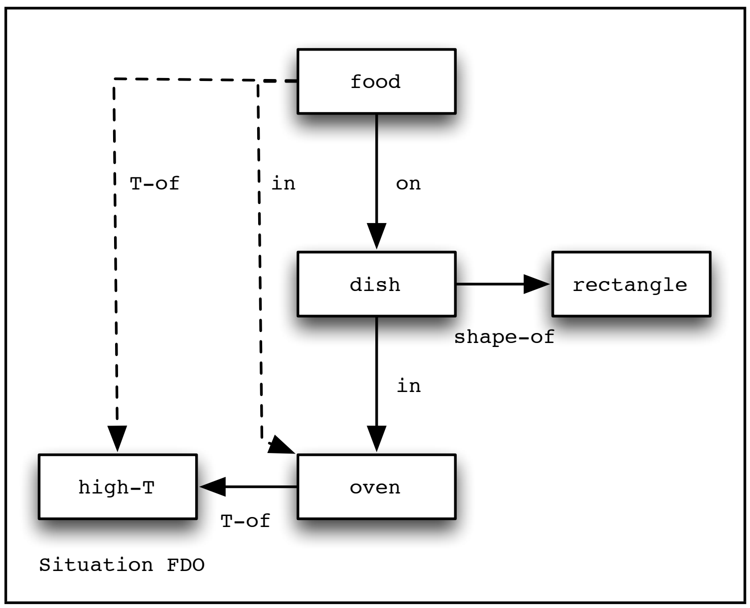


Fig. 5.2 Schematic representation of the base situation described in the text. The propositions explicating the causal structure of the situation are not shown.

Note that this description is much more elaborate than that of the target. It contains 21 agents (not all shown in the figure). As discussed in Section 4.2.1., it is typical that the source analog is much richer than the target. In particular, there are many agents representing the causal links. For example, it is represented that the propositions `on(food, dish)` and `in(dish, oven)` taken together are the cause for `in(food, oven)`.

5.2 Spreading Activation

5.2.1 Purpose

As stated earlier the spreading activation mechanism is of paramount importance in AMBR (and DUAL in general). It is responsible for computing dynamic estimates of the *relevance* of each particular memory item. It defines the *working memory* of the system by bringing some agents above a threshold while keeping irrelevant ones away. Recall the formula from Section 3.1.5:

WM = active portion of LTM + temporary agents

Given that all information processing occurs in the working memory, spreading activation defines which agents take part in each particular computation. Moreover, it serves as energy supply to the symbolic aspect and thus determines the *speed* of each symbolic processor (Appendix C).

At a more global level, spreading activation is the basis of the *access subprocess* in analogy-making. It is responsible for accessing concepts and instances (and hence situations) that are relevant to the target. It also assures the relaxation of the constraint satisfaction network, which in turn is a key factor for the *mapping subprocess*. Various *context* and *priming effects* are also directly expressible in terms of that mechanism (Kokinov, 1988 1994a, 1995).

5.2.2 Spreading Activation in AMBR

The connectionist aspect is a general architectural feature of DUAL (Section 3.1.3.2). This section is devoted to the particular design used in AMBR.

AMBR uses a modified version of the Grossberg activation function (Grossberg, 1978; Holyoak & Thagard, 1989). The function is chosen to meet the following design requirements:

- Time is continuous. (Or, the length of one elementary connectionist cycle is negligibly small with respect to the macroscopic time scale.)
- The activation level of any agent is bounded between zero and some fixed maximal value M.
- All links in the long-term memory are excitatory.[2]
- There is spontaneous decay that forces each node to loose activation according to an exponential law in the absence of external support.
- There is a threshold θ that clips small activation levels to zero.

[2] Therefore, it could also be said that AMBR uses a modified version of the function proposed by McClelland & Rumelhart (1981). The two functions are equivalent for non-negative inputs. They differ, however, for the hypothesis agents in the constraint satisfaction network.

If we neglect the threshold for the moment, the activation level a of any single node in the AMBR network is governed by the following differential equation:

$$\dot{a} = F(a,n) = -d\,a + E\,n\,(M - a)$$
$$a(t_0) = a_0$$

where $a = a(t)$ is the activation level as a function of time and $\dot{a} = da/dt$ is its first derivative, $n = n(t)$ is the net input to the node, $M = const$ is the maximal activation value, and d and E are parameters that control the rate of decay and excitation, respectively. See (Petrov, 1997) for a mathematical analysis of this equation and for the discrete approximation used in the implementation.

The function described above is the basis of the activation function of concept and instance agents in AMBR. (Hypothesis agents have a more complicated activation function described in Section 5.4.5.) There is one more complication, however—the working memory threshold. Whenever the activation drops below some predefined minimal level θ, it is instantaneously brought to zero (and the agent is forced out of the working memory). Conversely, when the activation level of some node is zero and the magnitude of the net input n is bigger than some critical value n_θ, the activation level of the node jumps instantaneously to the threshold level θ and then proceeds in the usual manner. The critical value n_θ is defined as the minimal support that an agent must receive from outside in order to resist the spontaneous decay and maintain activation equal to the threshold.

5.2.3 Example

This section shows how these abstract formulas apply to the problem of cooling milk in a teapot. The processing starts with the attachment of some agents to the special activation sources in the model—the *goal* and *input nodes* (Section 3.1.5). In this particular case, the human user of the system links the agents `T-of-CM1` and `low-T-CM1` to the goal node. A few of the other agents participating in the description of the problem (e.g., `teapot-CM1` and `metal-CM1`) are attached to the input. These agents rapidly become very active and bring all their coalition partners to the WM. Thus the target problem is presented to the system. (The external context could also be represented on the input node (Kokinov, 1994a). This is not done in the present example.)

Each instance agent from the target sends activation to its respective concept agent in the LTM. These concept agents enter the working memory and in turn activate related concepts and instances.[3] Figure 5.3 illustrates this process. It is an excerpt from the transcript of an actual AMBR run and tracks (roughly) the activation flow originating at `milk-CM1` and `tpot-CM1`.

[3] The example discussed here starts with zero activation of all long-term memory agents. This, however, need not be the case. Kokinov (1994a) has modeled priming effects by starting from some residual activation pattern.

```
T=0.04, adding milk to WM.
T=0.04, adding teapot to WM.
T=0.22, adding cook-vessel to WM.
T=0.22, adding liquid-holder to WM.
T=0.24, adding beverage to WM.
T=0.24, adding dairy-product to WM.
T=0.34, adding cheese to WM.
T=0.40, adding container to WM.
T=0.76, adding food-holder to WM.
T=0.98, adding plate to WM.
T=1.04, adding drinkable-liquid to WM.
T=1.06, adding liquid to WM.
T=1.08, adding food to WM.
T=1.42, adding cup to WM.
T=1.64, adding cow to WM.
T=1.84, adding baking-dish to WM.
T=2.20, adding saucepan to WM.
T=2.50, adding bottle to WM.
T=2.68, adding non-drinkable-liquid to WM.
T=4.70, adding glass to WM.
T=5.80, adding soft-drink to WM.
T=5.80, adding alcoholic-drink to WM.
T=6.15, adding pan to WM.
T=7.85, adding water to WM.
T=8.25, adding ice to WM.
...
```

Fig. 5.3 Excerpt of an AMBR transcript showing the process of bringing concept-agents to the working memory. See text for details.

As evident from the transcript, activation propagates "upward" in the class hierarchy, e.g. `milk-CM1` → `milk` → `dairy-product` → `food`. It also spreads "horizontally" to concepts at the same level of abstraction, e.g., `milk` → `cheese` (directly or via `dairy-product`). Some concepts that are associatively related to the active ones are also brought to the WM, e.g., `cow`. Sooner or later, however, the spread of activation is limited by the decay factor and new concept agents cannot pass the threshold. The number of active concept agents stabilizes, though individual agents occasionally get in or out the WM.

Recall from Section 4.3 that there are top-down `instance` links from the concept agents to some of their instances in the LTM. These links transmit activation from the semantic to the episodic memory and thereby initiate the access of source analogs. Figure 5.4 shows the instances that happened to be activated by the concepts of the previous transcript.

Note that initially there are isolated instances from disparate situations. The reason for their early inclusion in the working memory is their semantic similarity to the elements of the driver. As these instances participate in coalitions, however, they bring their partners to the WM too. Thus, retrieval of episodes is a bottom-up pro-

```
T=0.34, adding milk-MTF to WM.
T=0.42, adding tpot-WTP to WM.
T=1.28, adding food-SFF to WM.
T=1.40, adding cup-IHC to WM.
T=1.80, adding dish-FDO to WM.
T=2.34, adding tpot-ERW to WM.
T=2.86, adding food-FDO to WM.
T=7.80, adding water-WTP to WM.
...
```

Fig. 5.4 Excerpt of an AMBR transcript showing the process of bringing instance-agents to the working memory. Compare with Figure 5.3.

cess in AMBR. Note that there is no need for any centralized data structure. This is in contrast to other models (e.g., Thagard et al., 1990; Forbus et al., 1994a) which treat analog retrieval as an explicit competition at the level of entire situations.

The spreading activation mechanism is influenced by the other mechanisms in the model. These influences are mediated by changes in the topology of the network. Many new agents and new links are added by various mechanisms. This greatly affects the flow of activation and contributes to the dynamic and emergent nature of AMBR computation. To illustrate, consider the activation history of one particular instance agent—`food-FDO`—shown in Figure 5.5.

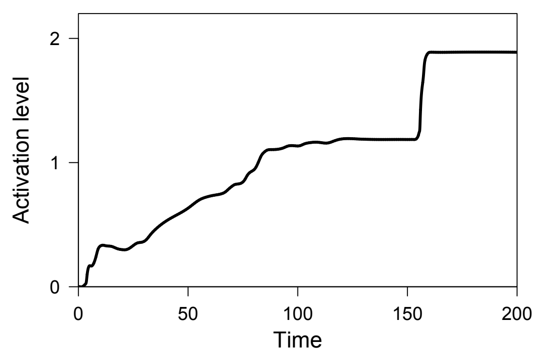


Fig. 5.5 Activation history of the instance agent `food-FDO`.

As we shall see later, `food-FDO` maps to the target instance of milk in our example. Thus the plot shows the gradual increase of the activation of a "successful" agent. Note in particular the sharp bend at time 160. It is due to an influence by the rating mechanism (Section 5.6). At that moment, the rating mechanism makes a commitment that `milk-CM1` corresponds to `food-FDO` and creates a temporary link between the two. In this way the highly active target element (attached to the input node) gives additional strong support to its counterpart.

5.2.4 A Prediction of the Model

In the example from the previous section all target elements were attached to the input and goal nodes simultaneously. This, however, does not have to be the case. On the contrary, it is more reasonable to expect that the elements are attached sequentially, in the order they become available to the system.

For example, suppose a student reads the verbal description of some problem in a textbook. The text is read sequentially and the internalized representation of this text would tend to be constructed sequentially too. In the AMBR model, this process could be crudely approximated by attaching the temporary agents that represent the target sequentially to the input node. In a more elaborate model, these elements would be constructed by the perceptual mechanisms. Similar considerations hold for the order of attachment to the goal node.

When some target elements are attached earlier than others, they will activate their respective concept agents earlier. This entails that the pattern of activation in the whole network shifts towards the association field of the early target elements. They have advantage over the elements that are attached to the source nodes later. Moreover, earlier elements tend to establish hypotheses earlier, which in turn reinforces their advantage. The net result of this process is that the order of presentation will affect the processes of analog access and mapping. This is one prediction of AMBR that could be tested experimentally.

The order effect predicted here is similar to the one demonstrated by Keane, Ledgeway, and Duff (1994). There is a difference, however. The attribute-matching task used in their experiment did not involve access of a source analog from memory. The subjects were given two lists of propositions and asked to find a correspondence between them. The experimental results suggested that the order of propositions in the list affected the time needed to find the correct mapping. We predict that similar order effects could be demonstrated with respect to the process of analog access as well. Specifically, the order will affect the frequency of accessing episodes from memory. Episodes containing elements that are semantically similar to a given target element will be retrieved more frequently when this target element is presented earlier to the participants. Section 6.4 presents a simulation experiment with AMBR that demonstrates such effect.

5.3 Marker Passing

As introduced in Section 3.2.3.2, the marker passing (MP) mechanism is a tool for answering the question, "Given two nodes in the network, is there a path between them?" It is the symbolic counterpart of the spreading activation. Markers are emitted by certain *nodes of origin* and then propagate the network looking for a *marker intersection*. Figure 5.6 illustrates the variant of this general mechanism that is used in AMBR.

Each instance-agent in AMBR emits a marker when entering the WM. It sends it to its parent concept via the `inst-of` link. The concept agent stores the marker in its local buffer and also passes it further to its superordinate concept(s) via the `subc` link(s). Thus markers propagate "upward" in the class hierarchy. Therefore, the presence of a marker in some concept indicates that the instance of origin belongs (directly or by inheritance) to that concept. For example, `drinkable-liquid` in Figure 5.6 can collect markers from three instance-agents: `milk-CM1`, `milk-MTF`, and `water-WTP`. This information can then be used for inheritance of properties, for skolemization purposes, etc.

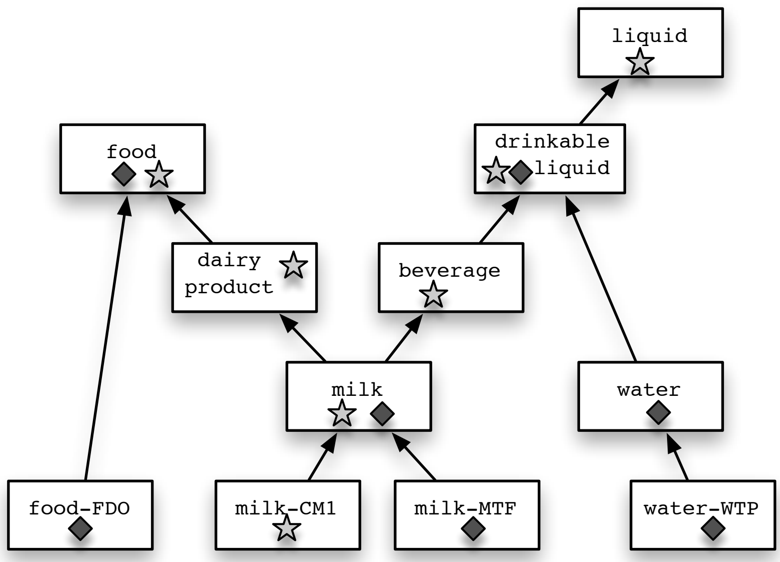


Fig. 5.6 Illustration of the marker passing mechanism. Each box represents an agent. Markers originate from the instance-agents in the bottom row and propagate upwards through the network of concept-agents. Instance-agents belonging to the target coalition (CM1 in this example) emit markers of one *color* (depicted by light stars in the figure), whereas instance-agents retrieved from long-term memory emit markers of a different color (depicted by dark diamonds). An intersection of two markers of complementary colors indicates that the agents that emitted them are instances of the same concept.

The culmination of the marker passing mechanism, however, happens when two complementary[4] markers meet at some concept agent. This intersection indicates that the two instances are *semantically similar* as they belong to the same (super)class. The activation level of the intersection node can be used as a numerical estimate of the degree of similarity in the current context (Kokinov, 1992b, 1994c).

Marker intersections serve another important role in AMBR. They trigger the construction of semantically grounded hypotheses and thus initiate the constraint

[4] Complementary markers have different origins and complementary *colors*.

satisfaction mechanism. More concretely, when a concept agent detects an intersection it formulates a *node construction request* describing the new hypothesis-agent that is to be made. It then sends the request to one of the specialized node constructors which are the only agents in the architecture capable of making a new agent. The node constructor carries out the request and constructs a temporary agent of the prescribed kind. In the particular case, it will be an embryo hypothesis (cf. Section 5.4) involving the two marker origins. The concept agent that has detected the intersection becomes the justification of the new hypothesis. In the example above, three such hypotheses are created: `milk-CM1<->milk-MTF` justified by `milk`, `milk-CM1<->water-WTP` justified by `drinkable-liquid`, and `milk-CM1<->food-FDO` justified by `food`.

One of the biggest issues in marker-passing systems is the *attenuation* of the marking. Without such attenuation there would be too many marker intersections, most of which are useless and drown out the few useful ones. Different systems use different attenuation strategies (see Hendler, 1988, for an overview). Thus Quillian (1966) limits the number of links that a marker can traverse. Charniak (1983) checks the outbranching factor and prevents "promiscuous" nodes from sending markers. Hendler (1988, 1989) uses an energy-like quantity called *zorch*, etc. In AMBR there is no need for a specialized mechanism for attenuation of markers because it follows naturally from the architectural principles of DUAL and the design of AMBR. Specifically, the spread of markers in the model is restricted by the following factors:

- Markers originate only from instance-agents. The concepts do not create new markers; they only pass the existing ones.
- Makers propagate only along links with certain labels (`inst-of`, `subc`, and `c-coref`).
- When there is a marker intersection, the markers stop and do not propagate further. In Figure 5.6, for example, the (diamond-shaped) marker emitted by `milk-MTF` does not propagate past the concept `milk` because it intersects there with the (star-shaped) marker from `milk-CM1`.
- Only active agents can receive and send markers. Thus the spread of markers is limited by the boundaries of the working memory as determined by the spreading activation mechanism.
- Marker passing, as any other symbolic activity in the architecture, takes time and thus depends on the speed of the symbolic processor of the agent receiving and handling the marker. As a consequence, markers move very slowly in the "peripheral" regions of the working memory where activation levels are low.
- Reporting marker intersection depends on a limited resource. There are only a few node constructors in the architecture and each concept agent must recruit one in order to create new hypotheses. When all constructors are busy the agent must wait until some of them becomes available. Thus there is an implicit competition, in which the more active agents have the advantage.

The net result of all these factors is that marker intersections are reported in a temporal order reflecting their potential usefulness for the particular task in the particular context. It is important to stress that this *global* marker passing is a dynamic emergent process. A whole coalition of DUAL agents is needed to cooperatively produce the final result. Each individual agent can do *local* MP only—instances know to create markers and concepts know to store and compare them. The overall result, however, is determined by a multitude of factors each of which has relatively minor impact on its own. Moreover, the relative strength of these factors vary dynamically in response to various external or internal events. Therefore, it is hard to predict a priori what marker intersections will happen in any given case, when they will construct hypotheses, etc. Yet the process exhibits certain emergent regularities: more active (i.e., more relevant) areas of the network report more and faster marker intersections.

5.4 Constraint Satisfaction

5.4.1 Main Points

The multiconstraint theory (Holyoak & Thagard ,1989, 1995) treats analogy-making in the light of three constraints: structural, semantic, and pragmatic. AMBR adopts this general idea. Like ACME, it uses a parallel connectionist algorithm for solving the constraint-satisfaction problem. This does not mean, however, that AMBR is a simple replication of ACME. There are a number of differences:

1. The constraint satisfaction network (CSN) is constructed incrementally by an emergent process. Hypotheses are created locally and are incorporated dynamically and asynchronously.
2. The CSN is integrated with the main network. This eliminates the need for special nodes mediating the semantic and pragmatic influences. Instead, this is accomplished by the relevant instance and concept agents themselves. Moreover, the hypotheses in the CSN send activation back to the agents in the main network. This is crucial for the integration of analogical access and mapping.
3. Instead of covering all possible element pairs, AMBR builds only justified hypotheses. In addition to being much more economical, this eliminates the need for centralized representation of situations.
4. A situation may be only partially accessed and thus participate only with its active elements.
5. Several source situations compete in the CSN simultaneously and thereby allow the emergence of complex analogies and blends when appropriate.
6. It is possible to map relations with different number of arguments, as well as map two arguments from one side to a single one from the other.
7. The system does not wait for the CSN to settle in order to read out the "solution" from the activation pattern. Instead, the CSN is in constant relaxation as the topology of the network changes. There is a rating mechanism that promotes winners and eliminates losers dynamically.

8. Each hypothesis agent undergoes an elaborate life cycle. The CSN involves hypotheses of different kinds.
9. There are hypotheses involving general propositions from the semantic memory (Section 4.4).

The first two points are by far the most important. Previous constraint satisfaction models, and in particular ACME (Holyoak & Thagard, 1989) and ARCS (Thagard et al., 1990), work in successive stages. First, a source analog is retrieved from long-term memory or supplied manually by the experimenter. Second, the constraint satisfaction network is constructed by a sequential symbolic process. Finally, the CSN is waited to settle, thus identifying a coherent set of correspondences. This three-step process is illustrated in Figure 5.7. The stages are carried out by different and independent mechanisms.

Fig. 5.7 Constraint satisfaction as a three-stage process. The stages come one after the other and cannot interact. Compare with Figure 3.2.

In contrast, AMBR (Kokinov, 1994a) views constraint satisfaction as a single integrated process that has three interacting subprocesses. They all run together, each one influencing the rest (Figure 5.8). This is very much in agreement with the overall spirit of the model—compare with Figure 3.4. The whole computation is performed in an integrated fashion: The same representational structures and computational mechanisms are used for all three subprocesses.

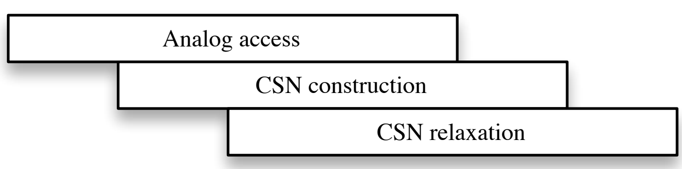


Fig. 5.8 Constraint satisfaction as a set of interacting subprocesses. Compare with Figure 3.4.

This computational scheme has several important advantages:

- It allows for integration of the more global processes of access and mapping in analogy-making.
- The subprocess that builds the CSN can be guided by the associative mechanism to avoid blind construction of implausible hypotheses. In this way, AMBR builds only a small fraction of the hypotheses generated by ACME. This decreases the working-memory demands—a weakness of ACME that has been criticized by

many researchers including its authors themselves (Keane et al., 1994; Kokinov, 1994a; French, 1995; Hummel & Holyoak, 1997).

- At the same time, AMBR retains the flexibility implied by the all-encompassing network used in ACME. AMBR does not construct *all* hypotheses, it constructs only *relevant* ones. And because relevance is determined dynamically, no possibilities are ignored a priori. This benefit is a direct consequence of the dynamic emergent computation that underlies AMBR's constraint satisfaction.

5.4.2 Hypothesis Agents

This section is devoted to the main actors in the constraint satisfaction network. From a declarative point of view, hypothesis-agents carry four main pieces of information, each stored in a specific slot. The first two slots contain the two entities being mapped. They are called *hypothesis elements*. The hypothesis agent as a whole represents the hypothesis that the first element (from the *driver* situation) corresponds to the second element (from a *recipient* situation). There are also hypotheses involving concept agents.

The third slot of a hypothesis-agent contains its *justification(s)*. The justification of a hypothesis is the reason for which it has been created and is being maintained by the system. For example, one possible justification of the hypothesis that `milk-CM1` corresponds to `water-WTP` is that both are drinkable liquids.

There are two kinds of justifications: semantic and structural. A hypothesis has semantic justification when its two elements are semantically similar. Such justifications are established by the marker-passing mechanism. In most cases the two elements belong to close or even identical classes. On some occasions, however, AMBR can construct hypotheses between almost any two entities. This happens when the domains of the two situations being mapped are very remote and hence the markers can intersect only at some very abstract node such as `object`, `relation`, etc. In this way, for example, `tumor` could be mapped to `fortress`. Such occasions are rare because usually the markers intersect earlier.

The second kind of justifications are the structural ones. A given hypothesis is structurally justified when there is another hypothesis that interlocks with the first. For example, the hypothesis that two relations correspond justifies the hypotheses that the arguments of these relations also correspond. Structural justifications are established by the structure correspondence mechanism (Section 5.5).

Semantic justifications are always represented by concept-agents; structural justifications by hypothesis-agents. It is possible (and frequent) that a hypothesis has several justifications. For instance, the hypothesis `milk-CM1<->water-WTP` could be justified by `drinkable-liquid` (semantic) and by `temperature-of-CM1<->temperature-of-WTP` (structural). In AMBR this particular hypothesis will be represented as schematized in Figure 5.9.

The fourth piece of information maintained by each hypothesis agent is a reference to the situation agent of the *driver situation*. In AMBR it is theoretically possi-

```
milk-CM1<-->water-WTP:
  :type (:mature :hypothesis :temporary)
  :t-link ((milk-CM1<-->food-FDO -0.3)
           (milk<==>water +0.3)
           (sit-CM1<==>sit-WTP +0.3) )
  :slot1
    :c-coref milk-CM1
  :slot2
    :c-coref water-WTP
  :slot3
    :c-coref (drinkable-liquid
              temperature-of-CM1<-->T-of-WTP )
  :slot4
    :c-coref sit-CM1
```

Fig. 5.9 Example of a hypothesis-agent. It represents the hypothesis that `milk-CM1` corresponds to `water-WTP`. There are two justifications for this correspondence. Its driver situation is `sit-CM1`. Not all link weights are shown.

ble that two or more target problems are attached simultaneously to the goal node. Each of them initiates its own set of hypotheses. The fourth slot prevents mixing hypotheses from different sets. It is also useful for the other model mechanisms.

Figure 5.9 shows only the symbolic aspect of the hypothesis. In addition, there is a connectionist aspect (as always in DUAL). All references to other agents are also links via which the hypothesis participates in the process of spreading activation. It supports its elements and in turn is supported by them. There are also excitatory links to the justification(s) and the driver situation.

Finally, there are temporary links (t-links) that connect the hypothesis with other hypotheses. These links may be excitatory (for coherent hypotheses) or inhibitory (for conflicting hypotheses). They are invisible to the symbolic aspect of the architecture but are very important for the relaxation of the constraint satisfaction network.

Temporary links with negative weights deserve special comment. They embody the *one-to-one constraint* in analogical mapping. This constraint pushes the CSN towards a solution in which an element `X` from situation 1 is mapped to at most one element from situation 2. There is a strong pressure that the same element `X` should not be mapped to two or more elements, e.g., `Y` and `Z`. Thus, the hypotheses `X<->Y` and `X<->Z` are contradictory and should be connected with inhibitory links.

A problem arises at this point. The constraint-satisfaction network in AMBR is constructed piecemeal by an emergent process. There is no central executive that goes through all hypotheses, identifies conflicting ones and puts inhibitory links between them. Rather, hypotheses are constructed one by one and the creator of each hypothesis has local information only. Under such circumstances, how does the agent `X<->Y` "know" that there is a rival (e.g., `X<->Z`) to compete with?

The answer to this question is: The hypothesis will "ask" the *secretary* of `X`.

5.4.3 Secretaries

Each entity-agent has a secretary associated with it. The secretary is *not* a separate agent; it is part of the entity-agent itself. The term *secretary* is used conventionally to refer to that particular part of a concept or instance agent that keeps track of the correspondences involving the agent.

The job of a secretary is twofold: it handles *hypothesis-registration requests* and carries out the rating mechanism. To that end, each secretary (i.e., instance or concept agent) is equipped with a slot and a few symbolic routines. The slot is labeled `hypoth` and is filled with references to all hypothesis-agents having the entity-agent as element. The same references are used as links that transmit activation from the agent (e.g., `milk-CM1`) to its hypotheses (e.g., `milk-CM1<->water-WTP` and `milk-CM1<->food-FDO`).

One of the first things that a hypothesis agent does after its creation is to send *hypothesis-registration requests* to the respective secretaries. Hypothesis-registration requests (or HR-requests for short) are symbolic structures notifying the secretary about the new hypothesis. Each of the two secretaries receives a request and sends a *secretary answer* back to the hypothesis. There are several kinds of answers but basically all of them could be aggregated into the following two major types:

- **Resign** — This answer means that the new hypothesis agent represents a tentative correspondence that already is represented by another hypothesis-agent. In other words, the new hypothesis is a duplicate of an older one. Such duplicate hypotheses are created because there usually are several justifications for a given correspondence. For example, the marker-passing mechanism could construct the hypothesis `milk-CM1<->water-WTP` on the grounds that both are drinkable liquids. Later on, the structure correspondence mechanism could independently construct the same hypothesis on the grounds that `milk-CM1` and `water-WTP` are corresponding arguments in corresponding relations. This second hypothesis is conceptually identical with the first but will be represented by a different agent. Let us suppose (as is actually implemented in the program) that the name of the second hypothesis agent is `milk-CM1<-1->water-WTP`. When it tries to register at the secretary of milk-CM1, the latter will reply with an answer of type Resign.
- **Establish** — This answer means that the hypothesis agent represents a novel hypothesis that does not coincide with any existing one. In the example above, the first hypothesis (`milk-CM1<->water-WTP`) would receive such answer to its HR-request.

Secretary answers carry more information than the simple resign/establish distinction. Answers of type Resign carry a reference to the *favorite*—the hypothesis in favor of whom to resign. Answers of type Establish carry a (possibly empty) list of references to rival hypotheses.

5.4.4 Life Cycle of Hypothesis-Agents

Hypothesis-agents analyze the answers from the secretaries and act according to their directives. Due to the possibility of answers of type Resign, a new hypothesis is not guaranteed from the beginning that it has raison d'être. It may be a duplicate of an existing hypothesis. If it manages to establish itself, there comes another struggle—it tries to win the competition with rival hypotheses.

AMBR distinguishes three main types of hypothesis-agents: *embryos, mature*, and *winner* hypotheses. They are marked by a tag in the `type` slot. More importantly, they differ in their activation functions and the repertoire available to their symbolic processors.

Each hypothesis-agent starts its life cycle as an embryo. Later on, it either resigns in favor of some other hypothesis or establishes itself and becomes mature. Mature hypotheses have a chance to be *promoted* to winner status (or demoted to loser status). In more detail, the life cycle is the following:

The main rule for hypothesis construction in AMBR is that each hypothesis must have a justification. As stated earlier, there are two ways to construct a hypothesis-agent: by the marker passing or by the structure correspondence mechanism. Either way, the new embryo hypothesis is created by a node constructor and begins its life cycle. It sends hypothesis registration requests to the secretaries of its two elements and waits for the answers. Usually, the two answers are the same—either both are Establish or both are Resign. The embryo takes corresponding actions respectively. Sometimes the secretaries disagree in their answers. This is possible due to the asynchronous and parallel nature of DUAL interactions. Embryo hypotheses are equipped with procedural knowledge to resolve the ambiguities.

When it turns out that the new embryo hypothesis is a duplicate of an existing hypothesis (called a *favorite*), the former resigns in favor of the latter. The resigning hypothesis hands over to the favorite all its declarative knowledge and in particular its justification. Having done that, it fizzles out. In the end, there is one hypothesis agent with two justifications instead of two separate hypotheses with one justification each. This is the mechanism that allows for multiple justifications of AMBR hypotheses despite that each agent is born with just one.

If the analysis of secretary information reveals that the embryo hypothesis represents a novel correspondence between two elements, the embryo establishes itself and becomes a mature hypothesis. From now on, its main goals are to win the competition with alternative hypotheses and to produce children.

The first goal is pursued by creating inhibitory links with the rivals. (The hypothesis receives a list of its rivals as an "enclosure" to the secretary answers.) For fair play, the new agent sends its reference to all competing hypothesis, prompting them to establish symmetric inhibitory links.

The third phase of the life cycle of hypothesis agents begins when (and if) the hypothesis receives a *promotion incentive* from an *authorized secretary*. This topic is discussed in Section 5.6.

5.4.5 The Constraint-Satisfaction Network

The mechanisms described so far gradually build many hypothesis-agents and establish connections between them. In this way, a constraint satisfaction network emerges. The CSN is a formation of agents that cooperatively solve a constraint satisfaction problem. It is integrated with the main network. The hypothesis agents participate in the big population of agents that comprise the model as a whole. The CSN involves the following kinds of links:

1. LTM → CSN: Links from instance and concept agents to the respective hypothesis agents. These links are excitatory and are stored in `hypoth` slots of these entity agents.
2. LTM → CSN: Links from concept agents (e.g., `drinkable-liquid`) to the hypotheses justified by them (if any). These links are excitatory and are stored in `t-link` slots of these concept agents.
3. CSN → LTM: Links from hypotheses to their elements, semantic justifications, and driver situations. These links are excitatory and are stored in S-slots of hypothesis agents.
4. CSN → CSN: Links from a hypothesis to its structure correspondence children (if any). These links are excitatory and are stored in `t-link` slots.
5. CSN → CSN: Links from a hypothesis to its structural justifications (if any). These links are excitatory and are stored in S-slots.
6. CSN → CSN: Links between competing hypotheses. These links are symmetric, have negative weights, and are stored in `t-link` slots of hypothesis agents.

The constraint satisfaction network thus embodies the three constraints posited by the multiconstraint theory (Holyoak & Thagard, 1989). The structural constraint is manifested in categories 4, 5, and 6 above. The semantic constraint appears in category 2, and the pragmatic one—in categories 1 and 2. Note that besides the links discussed here, AMBR has additional mechanisms for enforcing the constraints.

The links from the CSN to the rest of the network (category 3) deserve special attention. Through these links, the constraint satisfaction mechanism influences the pattern of activation in the main network and hence everything in the architecture. This fact has important implications for the integration of analogical access and mapping.

Hypothesis activation function. Hypothesis-agents are special in that they receive not only excitatory but also inhibitory input from their neighbors. They have two separate input zones—*enet* and *inet*. The two connectionist inputs are combined with the current activation level of the agent to determine the change of activation. The change of activation is governed by a modification of Grossberg's activation rule. (Compare with the equation from Section 5.2.2.)

In the original version of Grossberg's (1978) function, the activation can take both positive and negative values. This is inconsistent with the DUAL design principle that all agents in the architecture must have non-negative activation functions. Therefore, AMBR hypotheses use a linear transformation of Grossberg's activation function. The neutral point of the function (i.e., the resting level for zero input) is placed above the working memory threshold. The most negative branch of the function is truncated by the threshold. In this way the most pronounced losers are eliminated automatically—they simply fall out the WM and die. (Recall that all hypotheses are temporary agents). The upper part of the negative branch, however, is situated above the threshold. Thus the hypotheses that are judged implausible but not absolutely outlandish have a chance to survive.

Hypothesis output function. Hypothesis-agents are also characterized by a specific output function. Moreover, it is different for embryo hypotheses and mature hypotheses. Embryo hypotheses do not influence their neighbors at all. (In other words, their output function is the constant zero.) The reason for this decision is that the embryos are do not really participate in the CSN yet. Their output function changes when (and if) they mature. Mature hypotheses have a threshold output function so that only hypotheses whose activation is above the neutral level can influence their neighbors.

5.4.6 Example

As an example of the mechanisms discussed so far, and in preparation for the structure correspondence mechanism that comes next, this section provides a transcript showing the construction of one particular embryo hypothesis: `milk-CM1<-> milk-MTF`. Figure 5.10 illustrates the construction of a hypothesis by the marker-passing mechanism, followed by secretary inquiries. The concept `milk` detects a marker intersection at time 1.72 and sends a node construction request to the special agent `*NC6*`. It constructs an embryo hypothesis at time 2.84. After registering at its two secretaries the hypothesis matures at time 4.98.

```
T=0.16, #<MRK MILK-CM1> received in MILK.
T=1.64, #<MRK MILK-MTF> received in MILK.
T=1.72, #<MRK MILK-CM1> and #<MRK MILK-MTF> intersected at MILK.
T=2.12, #<NCR MILK> received in *NC6*.
T=2.84, creating a new agent: MILK-CM1<-->MILK-MTF
T=3.72, #<HR MILK-CM1<-->MILK-MTF> received in MILK-CM1
T=3.82, #<SA+ nil> received in MILK-CM1<-->MILK-MTF.
T=3.84, #<HR MILK-CM1<-->MILK-MTF> received in MILK-MTF
T=4.96, #<SA+ nil> received in MILK-CM1<-->MILK-MTF.
T=4.98, establishing hypothesis MILK-CM1<-->MILK-MTF.
```

Fig. 5.10 Excerpt of an AMBR transcript showing the construction and establishment of a hypothesis agent. `#<MRK xxx>` is a marker that originated at the instance agent with the given name, `#<NCR xxx>` is a node construction request, `#<HR xxx>` is a hypothesis registration request, and `#<SA+ nil>` is a secretary answer of type Establish.

5.5 Structure Correspondence

The structure-correspondence (SC) mechanism generates new hypotheses on the basis of existing hypotheses. It is carried out by mature hypothesis agents. Their symbolic processors are equipped with routines specialized for the task.

There are two major types of SC, conventionally termed *bottom-up SC* and *top-down SC*. Both come in *strong* and *weak* variants.

5.5.1 Bottom-up Structure Correspondence

Bottom-up SC takes place when there is a hypothesis involving two instance agents. More precisely, it occurs when there is a mature hypothesis whose elements have the tag `:instance` in their `type` slots. Under these circumstances, the symbolic processor of the hypothesis tracks the `inst-of` links of the two instances and retrieves their respective concepts. For example, if the two instances are `milk-CM1` and `water-WTP`, the concept agents will be `milk` and `water`. Then, the original hypothesis initiates the process that will construct a supplementary hypothesis stating a parallel correspondence between the two concepts. The new embryo is constructed in the usual way—by formulating and sending a node construction request. The original hypothesis becomes the justification of the new one.

It frequently happens that the new hypothesis is not really new—the same concepts have been already put into correspondence by an earlier invocation of the structure-correspondence mechanism. For example, the hypothesis `milk-CM1<->water-WTP` generates the concept-level hypothesis `milk<->water`. After a while, another hypothesis, e.g., `milk-CM1<->water-FDO` constructs another instantiation of the same concept-level hypothesis. In such cases, the duplication is detected by the secretaries and the second hypothesis resigns in favor of the first. Eventually, `milk<->water` will have two justifications and there will be appropriate excitatory links. This process enhances the overall degree of connectivity in the CSN.

The mechanism of bottom-up SC creates a pressure that correspondences at the instance level should be coherent with correspondences at the concept level. Stated differently, the mapping of two instances facilitates mapping of the classes to which they belong and vice versa.

The bottom-up SC also creates hypotheses involving the *situation-agents* to which the instances are affiliated. Recall that AMBR maps the target situation to several different bases simultaneously. The bottom-up SC creates hypotheses of the form `sit-CM1<->sit-WTP` and `sit-CM1<->sit-FDO`. The existence of such hypotheses in the CSN creates a pressure that situations are mapped as units. Blends are possible but they happen only when truly warranted (Grinberg & Kokinov, 2003). Normally the model tries to confine the mapping within the scope of two situations only: the target and a single base.

5.5.2 Top-down Structure Correspondence

Top-down SC is present in one form or another in all models of analogy-making. It captures an important aspect of the structural constraint as posited by Gentner (1983) and Holyoak & Thagard (1989): When two propositions correspond, it is highly desirable that their respective arguments also correspond.

The difficulties begin with the disambiguation of the phrase "respective arguments" above. Some models (e.g., Falkenhainer et al., 1986) walk around this difficulty by assuming that the enumeration of the arguments in a proposition can be meaningfully transferred to another proposition. From our point of view, this approach seems too conservative and psychologically implausible. In contrast, Holyoak & Thagard (1989) follow an approach that seems too liberal—they consider all possible argument pairs.

Thanks to the elaborated knowledge representation scheme adopted in DUAL (Kokinov, 1988, 1992), AMBR does not have great difficulties with this problem. Each argument is represented by a separate S-slot with many facets. One of these facets points to the respective slot in the parent concept as discussed in Section 4.4. This greatly facilitates the structure correspondence mechanism and relieves the model of implausible assumptions. Moreover, it supports mapping propositions with different number of arguments (Kokinov, 1994a; Hummel & Holyoak, 1997).

The details of the top-down structure correspondence in AMBR are the following: The symbolic processor of each mature hypothesis checks whether the two elements are propositions. The criterion is whether they contain the tags `:instance` and `:relation` among the fillers of their `type` slots. If this is the case, the symbolic processor attempts to determine the slot-to-slot correspondences. To do this, it needs the so called *pivot concept*.

The pivot concept is a concept which is a common superclass of both relations. For example, if the propositions are instances of the relations `in` and `on`, the pivot concept could be `in-touch-with`, `asymmetric-binary-relation`, or something else depending on the particular problem and context.

The pivot concept is often identified by the marker passing mechanism. When such information is available, the symbolic processor of the "proposition" hypothesis generates the appropriate "argument" hypotheses. When the information is not available, the symbolic processor checks for the obvious (and frequent) case when both propositions are instances of the same relation. In other words, it checks whether the `inst-of` slot of the two instances point to the same concept agent and uses the latter as a pivot concept. Otherwise, it gives up and stops, hoping the MP mechanism will provide the missing information later.

5.5.3 Weak Structure Correspondence

In many cases it is bad to allow the SC mechanism create new hypothesis but it is desirable to make it create additional justification links between existing hypotheses. This is the purpose of the weak SC.

For example, suppose that two situations—CM1 and WTP—are being mapped. As discussed in Section 4.5.3 these situations may involve *states*. Suppose that `initst-CM1` and `initst-WTP` are two such states. The marker passing mechanism detects they are instances of the same concept (namely `init-state`) and creates the hypothesis `initst-CM1<->initst-WTP`. Finally, suppose it establishes and becomes mature. Now the question is, "Should this hypothesis perform top-down structure correspondence?"

Each state has several S-slots pointing to the elements of the respective situation and the initial relations between them. Thus, the two states resemble propositions of type `and` and, therefore, one wishes to generate SC-motivated hypotheses about the arguments of these `and`-like propositions. Applying the usual (i.e. *strong)* structure correspondence mechanism indiscriminately, however, would lead to proliferation of useless hypotheses such as `milk-CM1<->high-temp-WTP`. To avoid this, states (and all agents having the tag `:situation` in general) are exempted from the strong top-down structure correspondence—the hypotheses involving such agents never generate any new hypotheses.

On the other hand, they could establish new justification links. To see why, consider the hypothesis `milk-CM1<->water-WTP`. It has a justification that has nothing to do with the membership of `milk-CM1` in `initst-CM1`. Still this hypothesis is consistent with `initst-CM1<->initst-WTP` and it is desirable to establish excitatory links between the two. Such link would improve the connectivity of the constraint satisfaction network and strengthen the structural constraint on mapping.

The main procedure for weak top-down structure correspondence is the following: Retrieve all S-slots of the two states and construct all possible pairings among them. Do *not* issue node construction requests, however. Instead, check the `hypoth` slot (see Section 5.4.3) of the secretaries of each pair and look for an old hypothesis representing the same correspondence. If such hypothesis is indeed registered at the secretaries, establish excitatory links to it. If there is no such hypothesis, however, then simply ignore the pair.

The weak SC has a bottom-up variant too. It is the converse of the ordinary top-down SC. That is, instead of descending from propositions to arguments, it tries to ascend from arguments to propositions. To illustrate, suppose `milk-CM1` is an argument in the proposition `in-CM1` and `water-WTP` in the proposition `in-WTP`. Suppose further that there is a mature hypothesis `milk-CM1<->water-WTP`. Then this hypothesis will try to establish a link to the hypothesis involving `in-CM1` and `in-WTP` provided such hypothesis is registered at the respective secretaries.

5.6 Rating and Promotion

The mechanisms presented so far are concerned primarily with generating hypotheses and establishing links between them. The final goal of these efforts, however, is to identify a set of correspondences. To that end, the model must make commitments at some point. This is the main objective of the mechanisms discussed in this section.

5.6.1 Rating Mechanism

5.6.1.1 Motivation

Each hypothesis on the secretary list of an entity agent represents a possible correspondence between the entity and some entity "from the other side." The one-to-one constraint on mapping demands that each element from the one domain should map to one element from the other. There is ambiguity, however, because each element typically has several hypotheses registered at its secretary. The relaxation of the constraint satisfaction network resolves these ambiguities using the inhibitory links between the incompatible hypotheses.

A straightforward approach for determining the final set of correspondences is to wait until the CSN settles and then select the hypotheses with maximal activation levels. This strategy is implemented in ACME (Holyoak & Thagard, 1989), which runs the network until all activation levels stabilize. This approach has two drawbacks: (*i*) the decision to stop must be taken centrally and (*ii*) any post-mapping processing can begin only after the mapping stage is over.

The rating mechanism avoids these limitations by *promoting* hypotheses during the run. This allows smooth integration between mapping and post-mapping processing. In particular, the processes of transfer (inference) and evaluation could begin before the CSN has settled completely.

5.6.1.2 Main ideas

Let us introduce the following terminology: a (current) *leader* is the hypothesis with the highest activation level in its set at the moment; a (final) *winner* is the hypothesis that has been explicitly *promoted* and has metamorphosed into a winner-hypothesis agent (cf. Section 5.4.4).

The main purpose of the rating mechanism is to monitor the hypotheses and send *promotion incentives* to those of them that emerge as stable and unambiguous leaders. In addition, it eliminates hypotheses that are obvious *losers* and triggers the skolemization mechanism.

The rating mechanism is carried out by the (secretaries of) instance agents.[5] Not all instances, however, are *authorized* to do so. Promoting winners is an important commitment that should be done carefully and by an "impartial judge." Therefore, only the secretaries of the driver situation are authorized to promote winners. (In the current version of AMBR, the driver situation is always the target. Future versions will switch the source analog as driver for the purposes of the transfer process (Hummel & Holyoak, 1997). They will probably grant some limited rating authority to recipient secretaries as well.)

Whenever an instance agent receives a hypothesis registration request (Section 5.4.3), it checks if it is authorized to do ratings. The criterion is whether its respective situation agent has a `:driver` tag in its `modality` slot. If authorized, the secretary creates a data structure called a *rating table* and stores it in its buffer. (Recall from Section 3.1.3 that each DUAL agent has some limited local memory.) The rating table keeps *individual ratings* for all mature hypotheses on the secretary list. Individual ratings are numerical quantities that characterize the relative success of the respective hypothesis.

The secretary periodically performs *rating surveys* to adjust the ratings. Each survey determines the current leader and increases its individual rating a little. The ratings of all other hypotheses are decreased. The magnitude of the change is proportional to the margin between the activation levels of the leader and its closest competitor. (If there is only one hypothesis, its activation is compared against the neutral level.) Thus, each rating value indicates how long the respective hypothesis has been a leader, how recently, and how strongly so.

Each new hypothesis starts at some intermediate rating level and then goes up or down depending on its relative standing in the total pool of competing hypotheses. If the rating reaches some *critical winner rating*, the hypothesis is considered for promotion. (It is not automatically promoted, however, as discussed below.) On the other hand, if the rating drops below some *critical loser rating*, the hypothesis is considered for elimination.

As a consequence of this computational scheme, hypotheses that are clear and unambiguous leaders rapidly reach promotion. On the other hand, when there are two or more competitors of equal strength or when there is a change in the leadership, the secretary refrains from making premature commitments. The decision is deferred until other secretaries announce their winners and change the balance in the CSN.

5.6.1.3 Promotions and ballotages

When the individual rating of some hypothesis reaches the critical winner level, it becomes a candidate for promotion. As this criterion is not always reliable by itself, the secretary undertakes some additional last-minute checks to determine whether

[5] Concept agents do not rate their hypotheses in the current version of AMBR. The so called *promotion propagation* mechanism will extend this functionality. This mechanism was in experimental stage in 1998 and is not reported here.

the candidate really merits promotion or not. If it does, the secretary sends it a *promotion incentive*. When the candidate is judged inappropriate, the secretary announces a *ballotage*, which means that the rating procedure should be repeated.

The rating mechanism is based on local information only—the activation levels of the hypotheses registered at a single secretary. Hence, it sometimes favors hypotheses that are inconsistent with the global mapping as determined by the CSN as a whole. The relaxation algorithm almost always succeeds to produce a consistent set of leaders for all secretaries. This, however, often takes time, especially when there are strong local anomalies that must be overcome.

Consider the following example. The problem presented in Section 5.1. involves a teapot: `tpot-CM1`. It so happens in one particular run that the target situation maps to a base with a dish (namely `dish-FDO`) instead of a teapot. As other bases also compete in the CSN, there are alternative correspondences for the target teapot. One of them—`tpot-WTP`—proves to be an especially strong competitor. In addition to its greater semantic similarity, it is also supported by the fact that there is an explicit proposition about its material. A similar proposition participates in the target description too. This leads to a triad of mutually supporting hypotheses: `tpot-CM1<->tpot-WTP`, `made-of-CM1<->made-of-WTP`, and `metal-CM1<->metal-WTP`. It is difficult for `dish-FDO`, whose material is not explicated, to beat this cluster singlehandedly.

Still, strong factors elsewhere in the CSN (other propositions, causal structure, etc.) dictate that the target as a whole maps better to situation FDO than to situation WTP. The secretary of `tpot-CM1` does not know this, though. On its local list it sees the hypothesis `tpot-CM1<->tpot-WTP` that has arrived first and remained on top ever since. It consistently dominates the surveys and its individual rating reaches the critical level. If it is promoted, however, it (and its `made-of` entourage) would be an odd-man-out among all other winners from situation FDO.

One way to prevent blendings of this kind is to set a high critical level for promotions. This will give time to the constraint satisfaction network to settle globally and straighten up local inconsistencies. This approach, however, effectively entails that all promotions occur after the mapping process is over. Subsequent processes of transfer, evaluation, etc. must take place when the CSN is frozen. This brings the model back to the pipeline paradigm that is antithetical to the spirit of AMBR (cf. Section 3.2.1).

The current version of the model adopts a different strategy. The authorized secretary takes a step out of the local neighborhood. Before issuing a promotion incentive, it checks whether the candidate hypothesis is consistent with the leader at the level of situations. In the example, the secretary of `tpot-CM1` reads the situation slot of the other element—`tpot-WTP`. Thus it learns that the latter is affiliated to `sit-WTP`. The secretary then contacts its own situation-agent (namely `sit-CM1`) and asks for the leader among the hypotheses at that level. It turns out that the leader there is the hypothesis `sit-CM1<->sit-FDO`, which is incompatible with `sit-WTP`. Therefore the secretary does not promote the candidate hypothesis.

Instead, `tpot-CM1` announces a ballotage and undertakes measures to weaken the unwanted hypothesis. It sends a message to the agent `sit-CM1<->sit-FDO` to create an inhibitory link to `tpot-CM1<->tpot-WTP`. This speeds up the relaxation of the CSN. It also sets the individual rating of the unwanted hypothesis back to the initial level. The rating of the second best hypothesis is also modified if it is below the initial level. Finally, the secretary triggers the skolemization mechanism when appropriate (see Section 5.7). After all these emergency measures, it restarts the rating surveys.

The previous paragraphs may create the impression that AMBR treats blends as something that must be avoided at all cost. This is not the case. Blends do happen in human analogy-making (e.g., Kokinov & Zareva-Toncheva, 2001; Turner & Fauconnier, 1995; Zareva & Kokinov, 2003) and should be accounted by cognitive models. Such blends, however, happen on quite special circumstances and involve bigger and more complex episodes. The ballotage discussed here is designed to prevent blends with few isolated intruders into an otherwise homogeneous mapping. This aspect was improved in subsequent versions of the model (Grinberg & Kokinov, 2003). Even the 1998 version could in principle produce heterogeneous mappings when there is a change of the leading hypothesis at the level of situation-agents.

5.6.1.4 Elimination of losers

In addition to promoting winners, the rating mechanism is also useful for weeding out *loser* hypotheses. Recall that the ratings of all hypotheses except the leader are decreased on each rating survey. When a rating drops below a critical threshold, the respective hypothesis is considered for elimination. If its activation level is also low, the hypothesis receives a *fizzle message*. Those hypothesis whose activation levels are only moderately low are retained as potentially useful.

The elimination of losers adds another dimension to the dynamics of the constraint satisfaction network. It both grows and shrinks. New hypotheses are added by various justifications. At the same time, loser hypotheses die out. As a consequence, the size of the CSN varies dynamically, growing rapidly at first and then shrinking back to retain only the most promising hypotheses. Usually, each promotion is followed by a number of eliminations. At the end of the run, each secretary list contains one winner and one or two (or zero) “reserve” hypotheses.

When the target elements are presented incrementally to the system (e.g., by some perceptual mechanism), the “wavefront” of the CSN follows suite. In this way the model seems to be able to handle situations that are much bigger than the ones used in current simulation experiments. The size of the CSN need never get very big. This has important consequences with respect to working memory limitations (Keane et al., 1994; Hummel & Holyoak, 1997). It is also relevant to the discussion of blending above—the target could match one base in the beginning, form some stable correspondences, and then shift to another base that better fits the target elements that have appeared in the interim.

5.6.2 Promotion Mechanism

This section describes the events triggered by reception of a promotion incentive in a hypothesis agent. This incentive marks the beginning of the third phase of the life cycle of the hypothesis (cf. Section 5.4.4). The mature hypothesis transforms into a winner. In the current version of the model this change involves nothing but removing the `:mature` tag from its type slot and adding `:winner` in its place. More radical restructuring is also possible (e.g., modifying the activation function, decay rate, efficiency coefficient, etc.).

When the due restructuring is complete (and presumably it takes quite a lot of time), the new winner sends *metamorphosis notifications* to its two secretaries to inform them about the change. These notifications make the secretaries even more severe to the losers in their `hypoth` slots. Only a few of the strongest (in terms of activation level and/or ratings) alternatives are spared. These survivors are marked by a `:loser` tag in their `type` slots. This tag is useful for detecting unmapped elements as a prerequisite for transfer.

Moreover, each instance agent creates a temporary excitatory link to its counterpart as designated by the winner hypothesis. For example, the metamorphosis notifications from `tpot-CM1<->dish-FDO` causes each instance to create a link to the other. This creates a direct route for receiving activation from the target and helps to bring more elements of the source situation to the WM (cf. Figure 5.5).

5.7 Skolemization

5.7.1 Motivation

Most analogy models use semantic knowledge for two purposes only—as a source of constraints on mapping and for similarity-based retrieval of episodes from long-term memory (e.g., Falkenhainer, Forbus & Gentner, 1986; Holyoak & Thagard, 1989; Keane & Brayshaw, 1988; Kokinov, 1994a; Hummel & Holyoak, 1997). It is clear, however, that human analogy-making uses semantic knowledge in much more diverse ways than that. We list below two additional ways in which analogy-making can utilize general knowledge about some domain.

First, semantic knowledge is used for reconstruction and elaboration of source analogs. Research on autobiographical memory provides abundant evidence that recollection of past episodes involves much reconstruction in addition to rote retrieval (e.g., Bartlett, 1932; Loftus, Feldman, & Dashiell, 1995; Schacter, 1995; see Kokinov & Petrov, 2001, for more references and an extended discussion). It is reasonable to expect that the same is true for recollection of past problems and their solutions, examples from textbooks, etc. The reconstructive nature of memory, however, is ignored by the current models of analog retrieval.

On the other hand, semantic knowledge is also used for elaboration of the target problem. It can even provide pieces of the solution. For example, the general fact that plates are heat sources and as such are used to heat things is of obvious relevance when asking how to heat water. In the important special case when the participants read (or listen to) a textual description of the problem, there is psycholinguistic evidence that readers make inferences about unstated elements of the situation during the comprehension process. For instance, after listening to a sentence such as “Alice pounded the nail until the board was safely secured,“ listeners have been shown to infer that “Alice used a hammer.” (McKoon and Ratcliff, 1981). Indeed, according to the *constructivist hypothesis* of discourse comprehension (e.g., Bransford, Barclay, & Franks, 1972), the encoding of comprehended text includes substantial information that comes from general knowledge rather than the text itself. There is a debate in the psycholinguistic literature about the degree to which inferences are made during the original encoding of the discourse or during subsequent retrieval from memory (e.g., McKoon & Ratcliff, 1992). However, there is widespread consensus (Whitney, 1998) that *some* inferences are made during encoding (which is our second point here) and *some* are made during memory retrieval (which is our first point). Current analogy models ignore both types of inferences. In sum, a lot of very relevant semantic knowledge goes unused if analogy-making is modeled exclusively in terms of finding correspondences between two episodes.

5.7.2 Main Ideas

AMBR skolemizationconstructs specific propositions on the basis of general propositions. It is a mechanism for elaborating the description of a situation using general knowledge about its elements.

Recall from section 4.4. that a *general proposition* is a proposition involving a general class instead of individual instance. If we ignore the details of the representation scheme (cf. Figure 4.3), general propositions are most easily recognized by the fact that at least one of their arguments is a concept agent. Thus, `made-of(teapot, metal)` is a general proposition as opposed to the specific `made-of(teapot-MTF, metal-MTF)`. Typically only one of the arguments of the general proposition is a concept; the other arguments are *prototype instances.* This creates asymmetry that often better captures the semantics. To illustrate, the proposition `made-of(teapot, metal)` could be read in two ways: “each teapot is made of metal” and “each metal is the material of some teapot.” In contrast, the proposition `made-of(teapot, prototypical-teapot-metal)` allows only the first interpretation.

One way or another, a general proposition represents a fact about some class of objects. The target problem and the episodes in the long-term memory, however, involve specific instances. The purpose of skolemization is to bring the general fact to the level of specific instances. This is done by constructing a new *Skolem* proposition that parallels the general one but in which each concept or prototype argument is replaced by a specific instance.

A question arises at this point, "Where do these specific instances come from?" The AMBR answer is that they are either supplied by the marker passing mechanism or created from scratch. The first choice is preferred whenever possible, falling back to the second only in the absence of appropriate markers.

For example, suppose the skolemization mechanism works on the general proposition `made-of(teapot, metal)`. In order to specialize it, it needs instances of the classes `teapot` and `metal`. Looking for such instances, it checks the buffers of these concept agents for markers. Each marker originates from some instance agent and propagates upward in the class hierarchy (see Section 5.3). Therefore, the origins of all markers arrived at a concept agent are instances of this concept. Suppose the buffer of `teapot` contains a marker from `teapot-MTF`. Thus, `teapot-MTF` could be used as a specialization of the first argument of the general proposition. The same check is done for the second argument. For the sake of the example, suppose that the buffer of `metal` contains no markers. Therefore the skolemization mechanism creates a new instance of this class. Such instances are called *Skolem instances*. By an AMBR convention, their names begin with an asterisk. Thus, the newly created agent may be named `*metal-1`.

After the skolemization mechanism finds an instance argument for each slot of the general proposition, it is ready to construct the *Skolem proposition*. The last ingredient is the *head* of the proposition. It is modeled on the template provided by the head of the general proposition. (Note that the latter is an instance agent belonging to some relational class, see Section 4.4.) In the example above, suppose the new agent is called `*made-of-1`. It is an instance of the relation `made-of` and its two S-slots are filled by `teapot-MTF` and `*metal-1`, respectively.

The final result of the whole process is that there is a proposition explicating the material of `teapot-MTF`. Like all teapots, it is made of metal.

Note that the general proposition may involve a concept higher in the class hierarchy. To extend our example, saucepans, pans, and baking dishes are made of metal too. A single general proposition can cover them all: `made-of(cook-vessel, metal)`.

5.7.3 Triggering Skolemization

Most of the work related to skolemization is carried out of the symbolic processor of a *general hypothesis*. This section describes how such hypotheses are created and prompted to perform skolemization.

A general hypothesis is a hypothesis involving a general proposition. It is created by the marker passing mechanism in the usual way. That is, the head of the general proposition (which is an instance agent) emits a marker when entering the WM. This marker propagates in the usual way and can intersect with other markers. As discussed in Section 5.3, when two complementary markers intersect they give rise to a semantically justified hypothesis. Complementarity rules in this case specify that the other marker must originate from some driver element. Hence, the new hypothesis involves a proposition from the driver situation on one hand, and the general proposition on the other.

Note that even though the semantic memory can potentially have thousands of general propositions, only a small fraction of them (if any) are used in each particular task. These "privileged" propositions are determined by the driver. The elements of the driver transmit the activation necessary for bringing the general propositions (like any agents) to the working memory. Their markers are prerequisite for the creation of general hypotheses (like any hypotheses).

Consider an example: The target problem CM1 (see Section 5.1) contains the proposition `made-of(tpot-CM1, metal-CM1)`. The head of this proposition is the instance agent `made-of-CM1`. In the run that serves as an illustration throughout this chapter, the latter agent happens to form the following general hypotheses: `made-of-CM1<->ckves-made-of-metal` and `made-of-CM1 <->bottle-md-glass`. Each of them involves a general proposition and could be skolemized.

The actual skolemization process begins when the general hypothesis receives a *skolemization incentive* from an authorized secretary. The rating mechanism is responsible for determining which hypotheses receive such incentives, if any. General hypotheses register at the secretaries and participate in rating surveys in the usual way. If such hypothesis is the leader in its set, its rating goes up. When it reaches some critical level, the hypothesis receives a skolemization incentive.

General hypotheses are quite weak compared to hypotheses involving specific propositions. It is, therefore, quite rare that a general hypothesis wins the rating. This is good because affiliated propositions should be preferred to Skolem propositions anyway. In the example above, suppose that `tpot-CM1` maps to some other teapot whose material is explicitly represented too. Under these circumstances the driver proposition `made-of-CM1` naturally maps to the respective recipient proposition and there is no need for skolemization. And so it happens—the specific hypothesis wins the rating and the general hypothesis never receives any skolemization incentive.

Skolemization incentives are also sent during ballotages (see section 5.6.1.2).

5.7.4 Links to Related Research

The term *skolemization* is used in formal logic in honor of the Norwegian mathematician Thoralf Skolem who introduced a method for replacing existentially quantified variables with constants or functions. The simplest type of skolemization is the so-called *existential instantiation*[6] (e.g., Russell & Norvig, 2009). It simplifies formulas of the type $\exists y P(y)$ into $P(c)$, where c is a new constant. Our ex-

[6] Indeed, the term *skolemization* was abandoned in favor of *instantiation* in subsequent AMBR publications (e.g., Kokinov & Petrov, 2000, 2001).

ample above is based on the existential sentence, “There exists an entity `y` such that `is-metal(y)` and `made-of(tpot-MTF,y)`.” The foundational intuition of the instantiation procedure is that the Skolem constant is just giving a name to the entity whose existence is asserted in the original sentence.

The situation is more complicated when the existential quantifier is within the scope of a universal quantifier. Consider the sentence, “Every person has a mother.” It is an abbreviation of the doubly quantified formula, “For any entity `x`, if `is-human(x)` then there exists an entity `y` such that `is-mother-of(x,y)`.” Notice that if we replace the existentially quantified variable `y` with a constant, say `Eve`, we get a completely different statement that asserts (erroneously) that `Eve` is a mother of each and every person. The correct skolemization rule in such cases is to replace `y` not with a constant but with a *Skolem function* of the universally quantified variables with the relevant scope. The substitution template is to replace $\forall x \exists y P(x,y)$ with $P(x, f(x))$, where $f(\cdot)$ is a new function. In our example, a good name for this function would be `the-mother-of`. Again, we are just giving a name to a function whose existence has been asserted by the original sentence.

Substitution rules such as these help convert a given first-order formula into the so-called *Skolem normal form* that has no existential quantifiers and all universal quantifiers appear to the left of everything else.[7] Thoralf Skolem proved that the transformed formula is satisfiable if and only if the original formula is satisfiable. Skolemization is an important step in many automated theorem-proving algorithms and is used in many symbolic AI systems (e.g., Russell & Norvig, 2009).

In the analogy literature, to our knowledge the term *skolemization* appears only in Falkenhainer, Forbus, and Gentner (1989). The Structure Mapping Engine generates *candidate inferences* in the target on the basis of information from the source. These candidate inferences often include entities. Whenever possible, SME replaces all occurrences of base entities with their corresponding target entities. This is analogous to AMBR’s reliance on the marker-passing mechanism to identify and use existing instances. When the source contains an entity that has no corresponding target entity (and is not a constant such as zero that can be transferred verbatim), “SME introduces a new hypothetical entity into the target which is represented as a Skolem function of the original base entity.” (Falkenhainer et al., 1989, p. 22).

It is instructive to compare the skolemization mechanisms in AMBR and SME. Apart for the common terminology, they are similar in that both introduce a new entity into the description of some situation. In both systems, skolemization is guided by the mapping between a target and a source. However, the two mechanisms also differ in important respects. In SME, the Skolem entity is introduced into the target and is not supported by semantic knowledge. Its only support is the presence of an unmapped entity in the source situation. (SME maps only one source at a time.) An additional constraint is that candidate inferences must be *structurally grounded* in an interlocking system of corresponding relations. In short, SME skolemization is a form of analogical transfer. The Skolem functions that SME generates can be read as follows: “a conjectural entity in the target that corresponds to *this* entity in the

[7] This sets the stage for the next simplification step, which is to drop all remaining quantifiers and thereafter treat each free variable as if it is implicitly universally quantified.

source." Falkenhainer et al. (1989, p. 35) point out that scientists sometimes consider such conjectural entities. For eample, *ether* was postulated to provide a medium for the propagation of light waves because other kinds of waves require a medium.

By contrast, AMBR skolemization introduces new entities into the source episodes retrieved from long-term memory. Thus, it is a form of reconstructive memory (Kokinov & Petrov, 2000, 2001). Also, it is guided by semantic knowledge and not just by systematic correspondences to the target. Therefore, at least in some cases it produces deductive inferences that are as valid as the general propositions that support them. The proposition "Every person has a mother." illustrates this point well. With less certain propositions, skolemization supplies default values that are not guaranteed to be correct. This is a form of schematization (e.g., Barclay, 1986). All cases involve an intimate interplay of analogical mapping, memory access, and re-representation, and illustrate the utility of AMBR's interactive approach (Figure 3.4).

The version of AMBR described in this book does not implement analogical transfer, but the skolemization mechanism will undoubtedly prove very useful in this regard. Section 7.1 sketches some ideas about extending AMBR's functionality in this direction. One general term in the literature on analogical inference is *copy with substitution and generation* (Holyoak, Novick, & Melz, 1994). Skolemization is a good mechanism for the generation component.

5.8 Putting It All Together

This section closes the description of AMBR by completing the example introduced in Section 5.1. It shows how the mechanisms of the model work together.

After the target problem CM1 is presented to the system, activation spreads in the network and brings relevant concepts and instances to the working memory (see Section 5.2). Two base situations are activated most and become the major competitors to map to the target. These are the situations FDO and WTP. Of the two, FDO will turn out to be the final winner. Figure 5.11 plots the *retrieval indices* of the two situations. The retrieval index is computed as the mean activation level of all agents affiliated to the respective situation-agent. It is an aggregate numerical measure of the overall accessibility of each episode. Note that these indices are neither computed nor used by the model. They are instruments for monitoring the emergent behavior of the system from the point of view of an external observer.

Figure 5.11 shows that early during the run the two rival coalitions are equally active. Later on, however, FDO continues to grow while WTP levels off and then even goes down. This difference is due to the influence of the mapping process as discussed below.

Note that the winner coalition gets strength *gradually*. In other words, the base episode FDO is not retrieved in an all-or-nothing fashion. Instead, agents enter the working memory one by one. This is characteristic of the decentralized representation of situations discussed in Chapter 4. The transcript in Figure 5.12 lists the exact moments in which individual elements pass the working memory threshold. As ev-

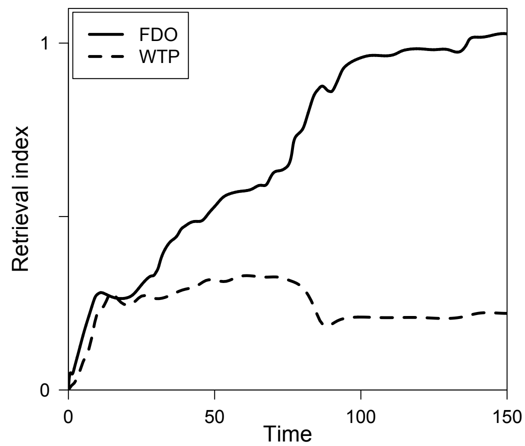


Fig. 5.11 Retrieval indices for two competing coalitions: FDO and WTP. The retrieval index quantifies the overall accessibility of each episode.

```
T=0.40, adding t-of-FDO-o to WM.
T=0.42, adding in-FDO-do to WM.
T=0.78, adding t-of-FDO-f to WM.
T=0.80, adding oven-FDO to WM.
T=0.84, adding high-t-FDO to WM.
T=1.78, adding sit-FDO to WM.
T=1.80, adding dish-FDO to WM.
T=2.68, adding initst-FDO-1 to WM.
T=2.86, adding food-FDO to WM.
T=3.40, adding on-FDO to WM.
T=6.60, adding goalst-FDO to WM.
T=8.20, adding in-FDO-fo to WM.
T=25.30, adding interst-FDO to WM.
T=29.70, adding to-reach-FDO to WM.
T=29.80, adding cause-FDO-t to WM.
T=31.10, adding follows-FDO to WM.
T=31.20, adding endst-FDO to WM.
T=68.00, adding cause-FDO-i to WM.
T=68.10, adding initst-FDO-2 to WM.
```

Fig. 5.12 Transcript showing the moments in which various members of situation FDO enter the working memory.

ident from the transcript (and from the step-like increase of the retrieval index in Figure 5.11), the description of the episode is retrieved from the long-term memory in three parts—roughly at times 8, 30, and 68.

The first group of agents enters the WM by time 8.20. It consists of the elements that are closest to the description of the target problem (cf. Figures 5.1 and 5.2). The causal structure of the base episode is not unfolded yet. It is not present in the target

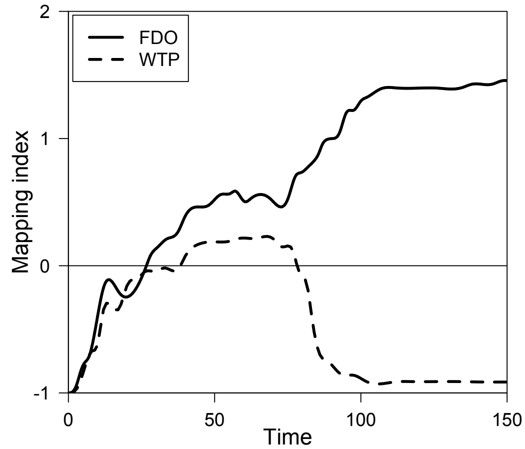


Fig. 5.13 Mapping indices for two competing coalitions: FDO and WTP. The mapping index quantifies the aggregate strength of the hypotheses connecting a given source situation to the target situation.

either. Hence, the working memory now contains two descriptions of comparable complexity. This is favorable for the mapping process (cf. Section 4.5.2).

Meanwhile, the marker passing and structure correspondence mechanisms generate a number of hypotheses. They register at their respective secretaries and are incorporated into the constraint satisfaction network. The competition in the CSN can be monitored with the aid of the *mapping indices* plotted in Figure 5.13. The mapping index is an aggregate numerical measure of the strength of the hypotheses between two situations. Like the retrieval index, it is not used by the model itself.

At time 25 the hypotheses involving FDO elements start to dominate the CSN. The additional activation that they send to the main network allows a second group of agents to enter the working memory. These are the agents that explicate the causal structure of situation FDO. As this episode emerges as the likely winner, it is getting ready for the processes of transfer and evaluation.

There are more obstacles to be overcome, however. The leading set of correspondences includes an unwanted element—`tpot-WTP` and its supporting proposition `made-of-WTP` and `metal-WTP`. It manages to beat `dish-FDO` because the latter lacks an explicit representation of its material. As discussed in Section 5.6.1.2, the rating mechanism detects the blend and announces a ballotage. It also triggers the skolemization mechanism.

The semantic memory contains a general proposition that all cooking vessels are made of metal. As baking dishes are a subclass of cooking vessels, the skolemization mechanism generates a Skolem proposition stating that `dish-FDO` is made of metal too. This occurs between time 80 and 87.60. The transcript in Figure 5.14 lists the *Skolem messages* that are exchanged during this process. The skolemization mechanism adds two new agents to the recipient situation.

Note that thisis a form of re-representation of the base aimed at bringing it in line with the target problem. As the target contains an explicit proposition about the material of the teapot, the source builds a corresponding counterpart. This is also an example of reconstructive memory (Kokinov & Petrov, 2001). On the other hand,

```
T=80.00, #<SkI MADE-OF-CM1> received in MADE-OF-CM1<-->
    CKVES-MD-METAL.
T=80.50, made-of-CM1<-->ckves-md-metal begins skolemization.
T=81.90, #<SM1 MADE-OF-CM1<-->CKVES-MD-METAL> received in
    *MATERIAL-METAL-1.
T=82.10, #<SM1 *MATERIAL-METAL-1> received in
    MADE-OF-CM1<-->CKVES-MD-METAL.
T=82.80, *material-metal-1 affiliates to sit-FDO.
T=84.20, #<SM2 MADE-OF-CM1<-->CKVES-MD-METAL> received in
    *MADE-OF-1.
T=85.00, #<SM2 *MADE-OF-1> received in MADE-OF-CM1<-->
    CKVES-MD-METAL.
T=87.60, *made-of-1 affiliates to sit-FDO.
```

Fig. 5.14 Transcript showing the events related to the skolemization mechanism. `#<SkI xxx>` is a Skolem incentive and `#<SM1 xxx>` and `#<SM2 xxx>` are Skolem messages of different kinds. See text for details.

the proposition `shape-of(dish-FDO, rectang-FDO)` that is included in the original description of that episode never enters the WM. This demonstrates the flexibility of the decentralized representation of AMBR situations.

After the skolemization, the mapping index of WTP (the competitor) drops rapidly (see Figure 5.13). FDO is now clear and unambiguous winner. There are, however, some final rearrangements of the correspondences. In particular, the semantically grounded hypothesis `in-CM1<->in-FDO-do` gives way to `in-CM1<->on-FDO` under the influence of the structural constraint on mapping. The ambiguity between the two `temperature-of` propositions in the base is also resolved. Table 5.1 lists the set of correspondences that lead the ratings at three different times.

Table 5.1 Leading correspondences for each target element at different times during the run. Target elements are listed in the left column. See text for details.

Target element	**T = 50**	**T = 100**	**T = 200**
sit-CM1	sit-FDO	sit-FDO	sit-FDO
milk-CM1	oven-FDO	food-FDO	food-FDO
tpot-CM1	tpot-WTP	oven-FDO	dish-FDO
in-CM1	in-FDO-do	in-FDO-do	on-FDO
T-of-CM1	T-of-FDO-oven	T-of-FDO-food	T-of-FDO-food
low-T-CM1	high-T-FDO	high-T-FDO	high-T-FDO
made-of-CM1	made-of-WTP	*made-of-1	*made-of-1
metal-CM1	metal-WTP	*metal-1	*metal-1
initst-CM1	initst-FDO-1	initst-FDO-1	initst-FDO-1
goalst-CM1	goalst-FDO	goalst-FDO	goalst-FDO
to-reach-CM1	to-reach-FDO	to-reach-FDO	to-reach-FDO

Chapter 6
Simulation Experiments

6.1 Description of the Knowledge Base

This chapter reports the results of several simulation experiments performed with AMBR. The long-term memory of the model is the same for all experiments, with variation of some links as described below.

The LTM consists of 569 permanent agents. 273 of them are concept-agents and encode semantic knowledge about the micro-domain introduced in section 4.1. For example, it is represented that `tea`, `milk`, and `water` are subclasses of `drinkable-liquid`, which in turn is subordinate to `liquid`. The system "knows" that `temperature-of` is a `physprop-relation` and that its first argument must be an `object` while the second one a `temperature-qualifier` such as `high-temp` or `low-temp`. The semantic memory also contains 49 instance-agents. Most of them are general propositions such as `heat-source-is-hot` and `bottle-made-of-glass`.

The remaining agents in the long-term memory represented twelve simple situations. These situations are outlined below. Appendix A contains a full description of one of them as taken directly from the source file fed to the program. Appendix B contains simplified representations in predicate calculus of all situations.

Base situation WTP (Water in a Teapot on a Plate): *There is some water in a teapot. The teapot is made of metal and its color is black. There is also a hot plate. The teapot is on the plate. The temperature of the plate is high.*

The goal is that the temperature of the water is high.

The outcome is that the temperature of the teapot is high because it is on the hot plate. In turn, this causes the temperature of the water to be high, as it is in the teapot.

Base situation BF (Bowl on a Fire burns out): *There is some water in a bowl. The bowl is made of wood. There is also a fire. The bowl is on the fire. The temperature of the fire is high.*

The goal is that the temperature of the water is high.

The outcome is that the bowl burns out because it is made of wood and is on the fire. In turn, this causes the water to dissipate, as it is in the bowl.

Base situation GP (Glass on a hot Plate breaks): *There is some water in a glass. The glass is made of [material] glass. There is also a hot plate. The glass is on the plate. The temperature of the plate is high.*

The goal is that the temperature of the water is high.

The outcome is that the glass breaks because it is made of [material] glass and is on the hot plate. In turn, this causes the water to dissipate, as it is in the glass.

Base situation IHC (Immersion Heater in a Cup):[1] *There is some water in a cup. There is an immersion heater in the water. The immersion heater is hot. The cup is on a saucer. The cup is made of china.*

The goal is that the temperature of the water is high.

The outcome is that the temperature of the water is high due to the hot immersion heater in it.

Base situation FDO (Food on a Dish in an Oven):[2] *There is a baking dish and some food on it. The shape of the dish is rectangular. There is also an oven. The dish is in the oven. The temperature of the oven is high.*

The goal is that the temperature of the food is high.

Since the food is on the dish which in turn is in the oven, the food is in the oven too. This causes the temperature of the food to be high, as the temperature of the oven is high.

Base situation MTF (Milk in a Teapot in a Fridge): *There is some milk in a teapot. The color of the teapot is green. There is also a fridge. The teapot is in the fridge. The temperature of the fridge is low.*

The goal is that the temperature of the milk is low.

Since the milk is in the teapot which in turn is in the fridge, the milk is in the fridge too. This causes the temperature of the milk to be low, as the temperature of the fridge is low.

Base situation ICF (Ice Cube in a Fridge):[3] *There is an ice cube on a glass. The glass is made of [material] glass. There is also a fridge. The glass is in the fridge. The temperature of the fridge is low.*

The goal is that the temperature of the ice cube is low.

Since the ice cube is on the glass which in turn is in the fridge, the ice cube is in the fridge too. This causes the temperature of the ice cube to be low, as the temperature of the fridge is low.

---

[1] See Figure 6.7 for a schematic diagram.

[2] See Figure 5.2 for a schematic diagram.

[3] See Figure 6.8 for a schematic diagram.

Base situation BPF (Butter on a Plate in a Fridge): *There is some butter on a plate. The plate is made of china and its shape is circular. There is also a fridge. The plate is in the fridge. The temperature of the fridge is low.*

The goal is that the temperature of the butter is low.

Since the butter is on the plate which in turn is in the fridge, the butter is in the fridge too. This causes the temperature of the butter to be low, as the temperature of the fridge is low.

Base situation STC (Sugar in Tea in a Cup): *There is some tea in a cup. There is some sugar in the tea. The taste of the sugar is sweet. The cup is on a saucer.*

The goal is that the taste of the tea is sweet.

The outcome is that the taste of the tea is sweet due to sugar in it.

Base situation SFF (Salt in Food in a Fridge): *There is some food on a plate. There is some salt in the food. The taste of the salt is salty. There is also a fridge. The temperature of the fridge is low.*

The goal is that the temperature of the food is low.

The outcome is that the food is both cold and salty. Since the food is on the plate and the plate is in the fridge, the food is in the fridge too. This causes the temperature of the food to be low. In the same time, the salt that is in the food causes its taste to be salty.

Base situation ERW (Egg in Red Water): *There is some water in a teapot. The color of the water is red. The teapot is made of metal. There is also an egg which is in the water.*

The goal is that the color of the egg is red.

The outcome is that the color of the egg is red because it is in the red water.

Base situation GWB (Glass in a Wooden Box): *There is a glass. It is made of [material] glass. The glass is in a box. The box is made of wood.*

The goal is that the box protects the glass.

The outcome is that the box protects the glass.

The verbosity of these (simplified) descriptions reveal how much knowledge is involved even in the seemingly trivial task of heating water. As simple and monotonous as they are, the twelve situations are designed to highlight various subprocesses of analogy-making. The descriptions involve objects and relations in different combinations and at various levels of similarity. Many episodes involve identical objects but are not isomorphic. Others go the other way around. Some episodes fail to achieve their goal and/or have side effects besides the main goal.

6.2 Statistics Over 1000 Runs

6.2.1 Experimental Setting

This section tests the behavior of the model on ten target problems. The goal is to check whether the model can reliably access episodes from long-term memory and map them to the target.

Each target problem is run 100 times, yielding a total of 1000 runs for the ten problems. All parameters of the model are kept constant across all runs (and across all experiments reported in this book in general).

The architecture DUAL is completely deterministic. The behavior of a DUAL-based model such as AMBR depends on five factors (*i*) the target problem, (*ii*) the contents of the long-term memory, (*iii*) the order of presentation of target elements (*order effect*), (*iv)* the residual activation in the long-term memory (*priming effect*), and (*v*) the external environment (*context effect*). The experiments reported in this section vary the first factor as independent variable and use the second one as source of replications. The remaining factors are kept constant. (They are explored in separate experiments. Kokinov (1994a) has demonstrated priming and context effects in an earlier version of AMBR. Order effects are explored in section 6.4 below.)

The knowledge base is replicated 100 times for the purpose of the experiments. Each variant contains the same 569 permanent agents outlined in section 6.1. Most of the links among them are the same too. There are, however, some links that vary randomly across the 100 variants. They are "top-down" links from concepts to instances (i.e. links labeled `instance`). The sampling procedure for picking up links for each KB variant is designed to approximate the (unimplemented) mechanism for dynamic "privileged instances" suggested in section 4.3. A small number of associative links (`a-link`) also differ randomly across KB variants. Thus it could be said that each variant represents a "snapshot" of the long-term memory of the system. The core KB contains approximately 3000 links. Each variant adds about 100 new links (which amounts to less than 4% of the total network connectivity).

Each target problem is run on each KB variant for 200 time units. This period is enough for the model to promote a winner situation in all but one of the 1000 runs. (In this exceptional run the model failed to access any episode from LTM to a sufficient degree.) The dependent variable is the number of times that each source situation is accessed and mapped to the particular target problem.

The activation level of all permanent agents is set to zero at the beginning of each run (i.e. there is no priming). Each target situation is represented by temporary agents. Some of them are attached to the goal and input nodes of the system. All attachments are done simultaneously at the beginning. The input node does not activate any agents apart from the target elements (i.e. the external context is ignored).

6.2.2 Heating Milk

The first pair of problems that are presented to the system involve heating milk (in the micro-domain). There are complementary to each other in the sense that the first has an explicit representation of the goal but the initial conditions are incomplete. In contrast, the second problem specify the initial arrangement in full and asks about the expected outcome of this arrangement. Appendix B contains simplified representations in predicate calculus of all target situations.

Target situation HM1 (Heating Milk, variant 1): *There is a teapot and some milk in it. The teapot is made of metal.*
The goal is that the temperature of the milk is high.

Target situation HM2 (Heating Milk, variant 2): *There is a teapot and some milk in it. The teapot is made of metal. There is also a hot plate. The teapot is on the plate. The temperature of the plate is high.*
The goal, if any, is not represented explicitly.
What is the outcome of this state of affairs?

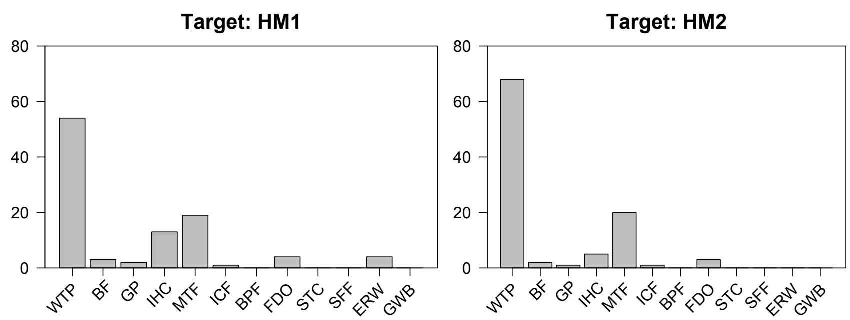


Fig. 6.1 Bar plots showing the frequencies of mapping each long-term memory episode to target problems HM1 and HM2, respectively.

The bar plots in Figure 6.1 demonstrate that in the majority of cases (54% of the runs) the model maps the target HM1 to the prototypical source episode about heating liquids—situation WTP. In these cases AMBR notices the analogy in which `milk-HM1` maps to `water-WTP`.

Two other sources stand out against the rest. Situation MTF is another good match. Its liquid is the same, but it requires the reversal `high-temperature<->low-temperature`. The fact that it is three times less frequent than WTP demonstrates that AMBR is sensitive to pragmatic pressures. The same pressures explain the frequency of situation IHC too—it represents an alternative way to heat liquids (by an immersion heater).

The second variant of the target problem (HM2) generates a similar pattern. The main difference is that situation WTP becomes even stronger (68%) at the expense of IHC. After all, the target problem contains a hot plate, not an immersion heater.

The bar plots reveal also that the model is not confined to the most obvious solutions to a problem. It reaches them most of the time (as it should) but occasionally it chooses more remote analogs. These are the cases with frequencies below 5% in the graphs. Most of them are episodes having some superficial similarity to the target: a teapot, goal related to high temperature, etc. These low-frequency answers are an important attestation of AMBR's flexibility.

6.2.3 Cooling Milk

The second pair of problems is similar to the first except that it deals with low temperatures. It tests whether AMBR is able to respond to a small (yet crucial) change in the target description.

Target situation CM1 (Cooling Milk, variant 1):[4] *There is a teapot and some milk in it. The teapot is made of metal.*

The goal is that the temperature of the milk is low.

Target situation CM2 (Cooling Milk, variant 2): *There is a teapot and some milk in it. The color of the teapot is black. There is also a fridge. The teapot is in the fridge. The temperature of the fridge is low.*

What is the probable goal for this arrangement?

The outcome of this state of affairs is not known.

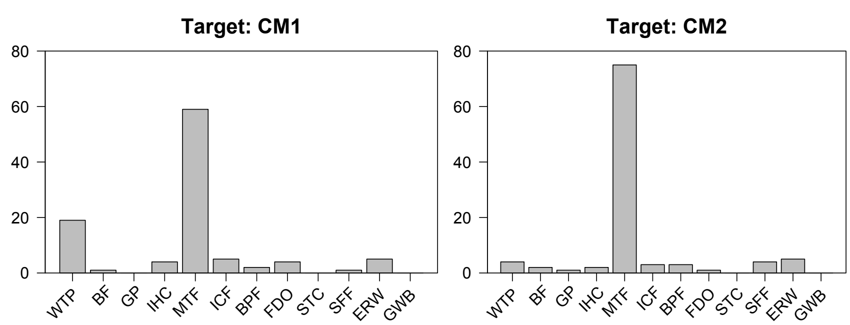


Fig. 6.2 Bar plots showing the frequencies of mapping each long-term memory episode to target problems CM1 and CM2, respectively.

[4] See Figure 5.1 for a schematic diagram.

A brief comparison between the left plots in Figures 6.1 and 6.2 reveals that change in the filler of a single slot in a single target agent can turn the behavior of AMBR to 180 degrees. Specifically, the `inst-of` slot of `low-T-CM1` (the second argument of `temperature-of-CM1`) is filled with a reference to the concept agent `low-temperature` while the respective slot in `high-T-HM1` points to `high-temperature`. (The names of the agents themselves are of course irrelevant.) This change is small but it is in a very important place—the respective agent is attached the goal node and the activation it provides to its parent concept is a major determinant of the overall content of the working memory. As a consequence, CM1 maps to MTF in 59% of the runs versus 19% for WTP. In contrast, the respective percentages for the target problem HM1 are 54% vs. 19%. (Recall that the experiment uses within-subject design as the two targets run over the same set of knowledge bases.) Clearly, the pragmatic constraint plays an important role in AMBR.

Let us now turn to the other problem in the pair: CM2. It is literally similar to the base situation MTF (Gentner, 1983, 1989). The only difference in the two descriptions, apart from the incompleteness of the target, is the color of the teapots. As seen in Figure 6.2, MTF wins in full 75% of the cases. This is the maximal frequency among all 1000 runs. All rival episodes occur with marginally low probabilities. This suggests that AMBR models accurately the empirical finding that analog access is dominated by literal similarities (Gentner & Landers, 1985; Holyoak & Koh, 1987; Ross, 1987).

6.2.4 When the Container is Fragile

The next pair is inspired by the target problem from experimental studies on priming effects (Kokinov, 1990, 1994a). The subjects in these studies were asked how one could heat water in a wooden bowl in a forest. Kokinov (1994a) performed related simulation experiments in the micro-domain.

Target situation WB1 (Water in a Bowl): *There is a bowl and some milk in it. The bowl is made of wood.*
The goal is that the temperature of the water is high.

Target situation WG1 (Water in a Glass): *There is a glass and some water in it. The glass is made of [material] glass.*
The goal is that the temperature of the water is high.

As evident from Figure 6.3, the model is split between two responses to the first problem. WTP is the prototypical case for heating liquids. It could not generate a good solution to the problem, however, as it suggests to put the bowl on the fire where it would burn. Still, it provides a sound match to the target. The other strong episode is BF which is an unsuccessful past attempt to solve this problem.

Note that the source analog that provides the “immersion heater” solution (IHC) works in only 6% of the cases. Incidentally, the subjects of (Kokinov, 1990) had similar difficulties in the absence of priming.

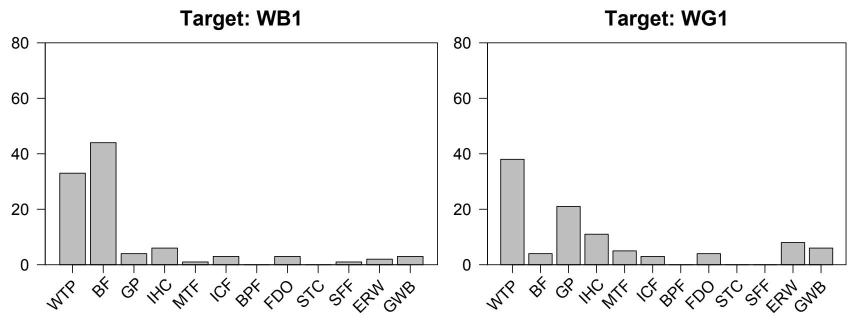


Fig. 6.3 Bar plots showing the frequencies of mapping each long-term memory episode to target problems WB1 and WG1, respectively.

In an attempt to increase the probability of using the immersion heater, target problem WG1 replaces the wooden bowl with a glass. (Situation IHC involves a cup.) The attempt is moderately successful—the frequency of IHC increases to 11%. As a side effect, situation GP takes the place of BF. A look at the descriptions of these two episodes (see Section 6.1) shows that this is to be expected.

6.2.5 Scaling Up: Problems Involving Taste

The problems in this section go away from the temperature-related focus of the current knowledge base. They deal with tastes and are intended to check whether the model is able to switch to this different thematic line. The base episodes are added to the long-term memory for similar reason.

Target situation SF1 (Salty Food, variant 1): *There is a plate and some food on it. The plate is made of china.*
The goal is that the taste of the food is salty.

Target situation SF2 (Salty Food, variant 2): *There is a plate and some food on it. There is some salt in the food.*
The goal, if any, is not represented explicitly.
What is the outcome of this state of affairs?

The bar plots in Figure 6.4 show that indeed the two episodes related to taste are accessed by these targets. Note also that situations STC, SFF, and BPF almost never show up for other target problems. This gives reasons for some optimism about the

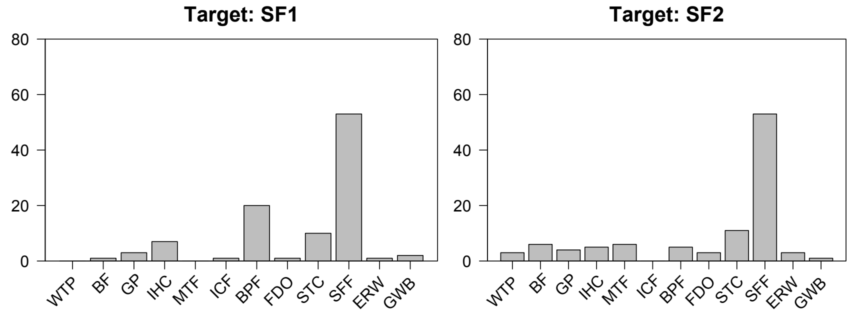


Fig. 6.4 Bar plots showing the frequencies of mapping each long-term memory episode to target problems SF1 and SF2, respectively.

ability of AMBR to scale up to larger memory sizes. It suggests that adding more and more episodes and different "thematic lines" will not lead to diffusion of the answers. Of course, this topic should be explored more rigorously with future (and bigger) versions of the knowledge base. We fully agree that memory of 12 episodes is very insufficient to support any serious claims about the scalability of the model.

6.2.6 Two Final Problems

Target situation EHW (Egg in Hot Water): *There is a teapot and some water in it. There is an egg in the water. The teapot is made of metal. The color of the egg is white. The temperature of the water is high.*

The goal, if any, is not represented explicitly.

What is the outcome of this state of affairs?

Target situation ICC (Ice Cube in Coke): *There is a glass and some coke in it. The glass is made of [material] glass. There is an ice cube in the coke. The temperature of the ice cube is low. There is also a table. The glass is on the table.*

The goal, if any, is not represented explicitly.

What is the outcome of this state of affairs?

The target problem EHW and its long-term memory counterpart ERW are added for similar scaling-up considerations. Problem ICC is used for the experiment discussed in Section 6.3. The respective bar plots are shown in Figure 6.5.

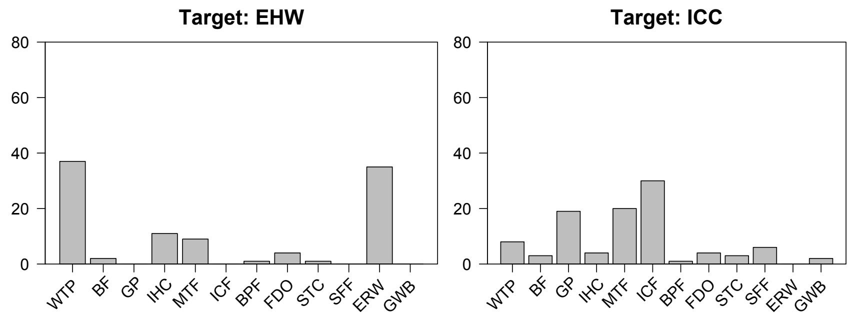


Fig. 6.5 Bar plots showing the frequencies of mapping each long-term memory episode to target problems EHW and ICC, respectively.

Table 6.1 Joint distribution for all 1000 runs. Each cell gives the frequency of accessing and mapping a target problem (row) to a source episode (column).

Target	WTP	BF	GP	IHC	MTF	ICF	BPF	FDO	STC	SFF	ERW	GWB	Total
HM1	54	3	2	13	19	1		4			4		100
HM2	68	2	1	5	20	1		3					100
CM1	19	1		4	59	5	2	4		1	5		100
CM2	4	2	1	2	75	3	3	1		4	5		100
WB1	33	44	4	6	1	3		3		1	2	3	100
WG1	38	4	21	11	5	3		4			8	6	100
SF1		1	3	7		1	20	1	10	53	1	2	99[a]
SF2	3	6	4	5	6		5	3	11	53	3	1	100
EHW	37	2		11	9		1	4	1		35		100
ICC	8	3	19	4	20	30	1	4	3	6		2	100
Total	264	68	55	68	214	47	32	31	25	118	63	14	999

[a] On one run with target problem SF1, no situation agent was promoted as winner.

6.2.7 Variability and Determinism

Table 6.1 summarizes the results of these simulations by presenting the joint distribution produced by all 1000 runs. The fact that few cells are completely empty indicates that the model populates all regions of its problem space. That is, there is some small probability to map any source analog to almost any target. No possibilities are ruled out a priori. On the other hand, AMBR focuses on the episodes that best fit any given problem. It is efficient without being rigid. This is a consequence of the dynamic emergent style of computation that is characteristic of DUAL (Kokinov, Nikolov, & Petrov, 1996).

Note also that although AMBR is completely deterministic, it is still able to demonstrate the variability of behavior evident from the table. As described in Section 6.2.1., the random factor in the experiment amounts to less than 4% of the initial links in the long-term memory. Nevertheless, each target problem generates a whole range of answers. This is again a consequence of the dynamic emergent style of computation. The macroscopic behavior of the system depends on a multitude of interrelated microscopic factors. A small change in the initial conditions can drift the global outcome far away in the problem space. Therefore, the macroscopic behavior of AMBR must be analyzed in probabilistic terms even though all microscopic mechanisms are deterministic.

6.3 Influence of Mapping on Analog Access

6.3.1 Simulation Experiment Method

This section presents a case study exploring the integration of analog access and mapping in AMBR. It contrasts two strategies for combining access and mapping—parallel vs. serial.

6.3.1.1 Design

The experiment consists of two conditions. Both conditions involved running the model on a target problem. In the *parallel condition*, AMBR operates in its normal manner with the mechanisms for access and mapping working in parallel. In the *serial condition*, the program is artificially forced to work serially—first to access and only then to map. The target problem and the content of the long-term memory are identical in all runs. The topics of interest fall into two categories—the final mapping constructed by the program and the dynamics of the underlying computation. The latter is monitored by recording a set of variables describing the internal state of the system at regular time intervals throughout each run.

6.3.1.2 Materials

The experiment uses the knowledge base described in section 6.1. Situation ICC (Ice Cube in Coke) is the target problem. Its verbal description is given in Section 6.2.6. Two of the twelve episodes are most important for the present discussion: situations IHC (Immersion Heater in a Cup with water) and ICF (Ice Cube in a Fridge).

As evident from Figures 6.6, 6.7, and 6.8, both situations IHC and ICF may be considered similar to the target problem. There are some differences, however. Situation ICF involves the same objects and relations as the target but the structure of the two are different. In contrast, situation IHC involves different objects but its system of relations is completely isomorphic to that of the target. According to Gentner (1989), the pair IHC–ICC may be classified as analogy while ICF–ICC as mere appearance. Thus it is expected that situation ICF would be easier to retrieve

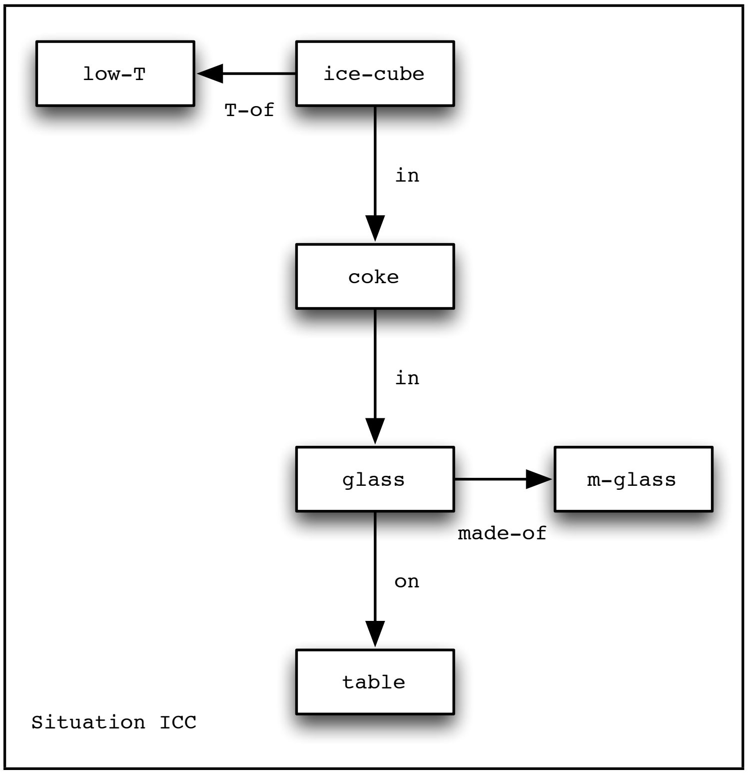


Fig. 6.6 Schematized representation of target situation ICC. Objects are shown as boxes and relations as arrows. The actual AMBR representation is more complex and consists of 15 agents.

from the total pool of episodes stored in LTM. On the other hand, IHC would be more problematic to retrieve but once accessed it would support better mapping.

6.3.1.3 Procedure

The model is run two times on the target problem. The two runs carry out the parallel and serial conditions of the experiment, respectively. The contents of the long-term memory and the parameters of the model are identical in the two conditions.

Recall that situations have decentralized representations in AMBR. The target problem is represented by a coalition of 15 agents standing for the ice-cube, the glass, two instances of the relation in and so on (See Appendix B). 12 of these agents are attached to the special nodes that serve as activation sources in the model. The attachment is the same in the two experimental conditions.

In the parallel condition, the model is allowed to run according to its specification. That is, all AMBR mechanisms run in parallel, interacting with each other. The program iterates until the system reaches a resting state. A number of variables are

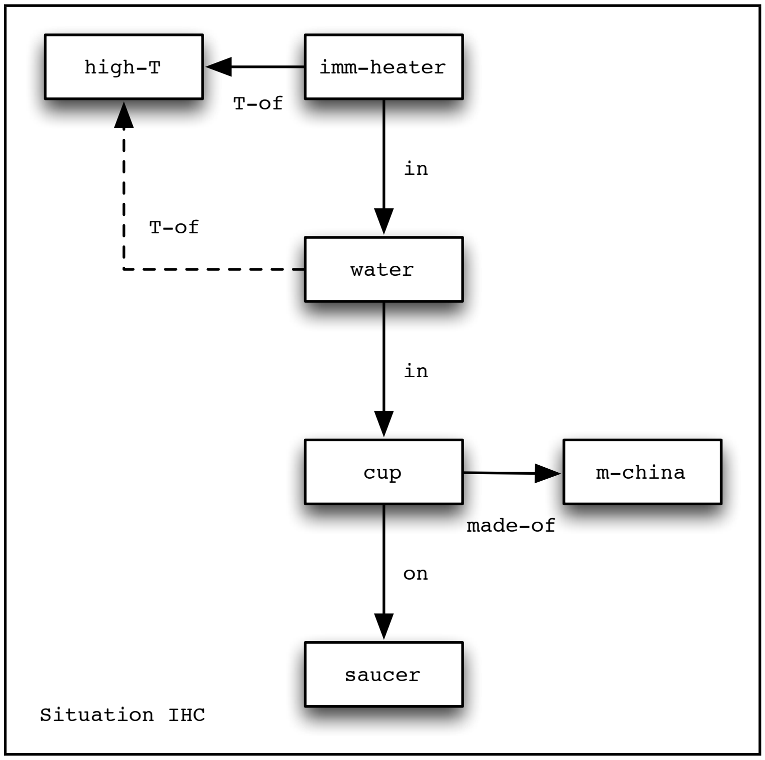


Fig. 6.7 Schematized representation of situation IHC. Dashed arrows stand for relations in the "outcome." The actual AMBR representation is more complex—it consists of 19 agents and explicates the causal structure (not shown in the figure).

recorded at regular intervals throughout the run. Out of these many variables, the *retrieval index* is of special interest. It is computed as the average activation level of the agents involved in each situation.

In short, the data accumulated at the end of the run are the final mapping constructed by the program and a log file of the retrieval indices of all twelve situations from the LTM.

In the serial condition, the target problem is attached to the activation source in the same way and the same data were collected. However, the operation of the program is forcefully modified to separate the processes of access and mapping. To that end, the run is divided in two steps.

During step one, all mapping mechanisms in AMBR are manually switched off. Thus, spreading activation is the only mechanism that remains operational. It is allowed to work until the pattern of activation reaches asymptote. The situation with the highest retrieval index is then identified. If we hypothesize a "retrieval module," this is the situation that it would access from LTM.

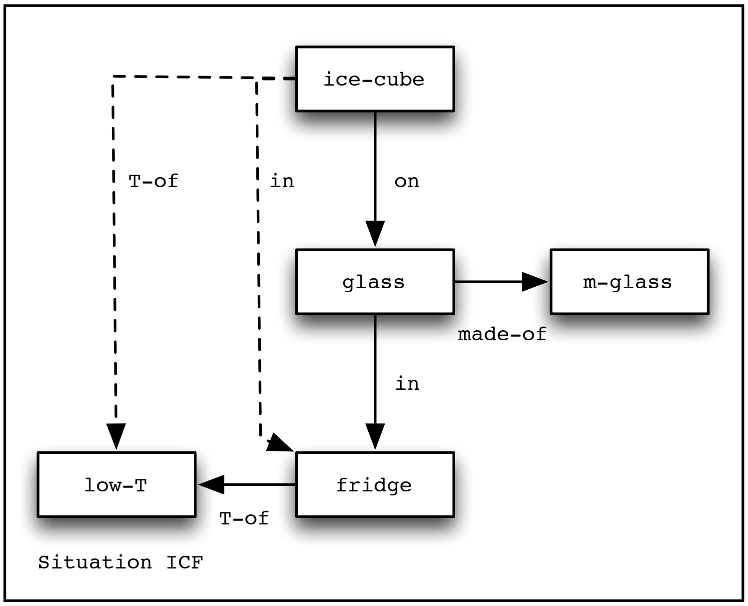


Fig. 6.8 Schematized representation of situation ICF. The actual AMBR representation is more complex—it consists of 21 agents and explicates the causal structure (not shown in the figure).

After the source analog is picked up in this way, the experiment proceeds with step two. The mapping mechanism is switched back on again but it is allowed to work only on the source situation retrieved at step one. This situation is mapped to the target. Thus, the data at the end of the second run are the final mapping constructed at step two and two logs of the retrieval indices.

6.3.2 Results and Discussion

In both experimental conditions the model settles in less than 150 time units and produces consistent mappings. By "consistent" we mean that each element of the target problem is unambiguously mapped to an element from LTM and that all these corresponding elements belong to one and the same base situation. Stated differently, the mappings are one-to-one and there are no blends between situations.

In the parallel condition, the target problem maps to situation IHC, yielding the correspondences `in<->in`, `water<->coke`, `imm-heater<->ice-cube`, `high-T<->low-T`, `made-of<->made-of`, etc. Four elements from the source situation remain unmapped and in particular the agent representing that the water is hot. This proposition is a good candidate for inference by analogy. *Mutatis mutandis*, it could bring the conclusion that the coke is cold.

In the serial condition, situation ICF wins the retrieval stage. This is explained by the high semantic similarity between its elements and those of the target—both deal with ice cubes in glasses, cold temperatures, etc. The asymptotic level of the retrieval index for ICF is about four times greater than that of any other situation. In particular, situation IHC ends up with only 5 out of 19 agents passing the working memory threshold.

According to the experimental procedure, situation ICF is then mapped to the target during the second stage of the run. The correspondences that emerge during the latter stage are shown in Table 6.2. The semantic similarity constraint dominates this run. This is not surprising given the high degree of superficial similarity between the two situations. There is, however, a serious flaw in the set of correspondences. The proposition `T-of(ice-cube-ICC,low-T-ICC)`, which belongs to the *initial* state of the target, is mapped to the proposition `T-of(ice-cube-ICF,low-T-ICF)`, which is a *consequence* in the source. Therefore, the whole analogy between the target problem and situation ICF could hardly generate any useful inference.

Table 6.2 Correspondences constructed by the model in the serial condition.

Base situation ICF	Target situation ICC
`ice-cube`	`ice-cube`
`fridge`	`coke`
`glass`	`glass`
`in (ice-cube, fridge)`	`in (ice-cube, coke)`
`in (glass, fridge)`	`in (coke, glass)`
`on (ice-cube, glass)`	`on (glass, saucer)`
`T-of (fridge, low-T)`	`<unmapped>`
`T-of (ice-cube, low-T)`	`T-of (ice.cube, low-T)`
`low-T`	`low-T`
`made-of (glass, m-glass)`	`made-of (glass, m-glass)`
`m-glass`	`m-glass`
`initstate1`	`initstate`
`initstate2`	`<unmapped>`
`interstate`	`table`
`endstate`	`endstate`
`goalstate`	`<unmapped>`
`follows (initstate1, endst)`	`follows (initstate, endst)`
`to-reach (initstate1, goalst)`	`<unmapped>`
`cause (initstate2, in(ic,fr))`	`<unmapped>`
`cause (interstate, T-of(ic,lT))`	`<unmapped>`

To summarize, when the mechanisms for access and mapping work together, the model constructs an analogy that can potentially solve the problem. On the other hand, when the two mechanisms are separated, the retrieval stage favors a superficially similar but inappropriate base.

The presentation so far concentrated on the final set of correspondences produced by the model. We now turn to the dynamics of the computation as revealed by the retrieval indices. Figure 6.9 plots the retrieval indices for the two critical LTM episodes during the first run of the program (i.e. when access and mapping work in parallel). Figure 6.10 concentrates on the early stage of the first run and compares it with the second run (i.e. when only the access mechanism is allowed to work). Note that the two plots are in different scales.

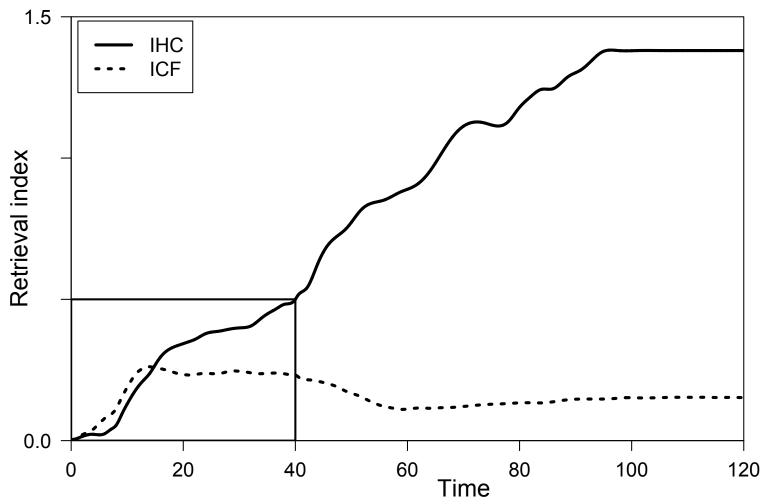


Fig. 6.9 Plot of retrieval indices versus time for the parallel condition. The "south-west" corner of the plot is reproduced in Figure 6.10 with threefold magnification.

These plots tell the following story: At the beginning of the parallel run, several situations are probed tentatively by bringing a few elements from each into the working memory. Of this lot, ICF (with the ice cube) looks more promising than any of its rivals as it has so many objects and relations in common with the target. Therefore, about half of the agents belonging to situation ICF enter the working memory and begin trying to establish correspondences between themselves and the target agents. The active members of the rival situations are doing the same thing, although with lower intensity—their symbolic processor are slower.

At about 15 time units since the beginning of the simulation, however, situation IHC (with the immersion heater) rapidly gains strength and eventually overtakes the original leader. At time 40, it takes the lead and gradually transforms its small advantage into an uncompromising triumph.

The final victory of situation IHC, despite its lower semantic similarity compared to situation ICF, is due to the interaction between the mechanisms of access and mapping in AMBR. More precisely, in this particular case it is the mapping that radically changes the course of access. To illustrate the importance of this influence, Figure 6.10 contrasts the retrieval indices with and without mapping.

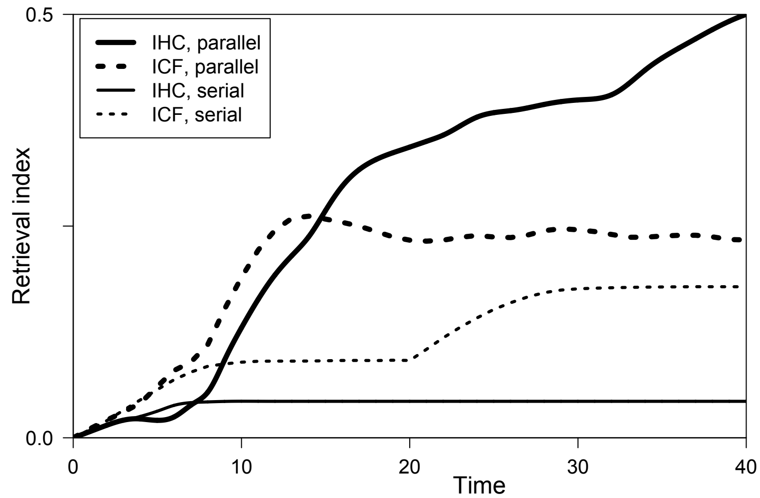


Fig. 6.10 Retrieval indices for situations IHC and ICF with and without mapping influence on access. The thick lines correspond to the parallel condition and replicate (with threefold magnification) the lines from the "south-west" corner of Figure 6.9. The thin lines show "pure" retrieval indices.

The thin lines in Figure 6.10 show the retrieval indices for the two situations when mapping mechanisms are suppressed. Thus, they indicate the "pure" retrieval index of each situation—the value that is due to the access mechanism alone. The index for situation ICF is much higher than that of IHC and, therefore, ICF is used as source when the mapping is allowed to run only after the access has finished.

The step-like increases of the plots indicate moments in which an agent (or usually a tight sub-coalition of two or three agents) passes the working memory threshold (cf. Figure 5.12). This happens, for instance, with situation ICF between time 20 and 30 of the serial condition (the thin dashed line in Figure 6.10). Thus, accessing a source episode in AMBR is not an all-or-nothing affair. Instead, situations enter the working memory agent by agent and this process extends far after the beginning of the mapping. In this way, not only can the access influence the mapping but also the other way around.

In the interactive condition the mapping mechanism boosts the retrieval index via what we call a *bootstrap cascade*. This cascade operates in AMBR in the following way. First, the access mechanism brings two or three agents of a given situation into the working memory. If the mapping mechanism then detects that these few agents can be plausibly mapped to some target elements, it constructs new correspondence nodes and links in the AMBR network. This creates new paths for the highly active target elements to activate their mates. The latter in turn can then activate their "coalition partners," thus bringing a few more agents into the working memory and so on.

The bootstrap cascade is possible in AMBR due to two important characteristics of this model. First, situations have decentralized representations which may be accessed piece by piece. Second, AMBR is based on a parallel cognitive architecture which provides for concurrent operation of numerous interacting processes. Taken together, these two factors enable seamless integration of the subprocesses of access and mapping in analogy-making.

6.4 Order Effect on Analog Access

6.4.1 Simulation Experiment Method

This section presents an experiment testing the prediction made in Section 5.2.4—the order of presentation of target elements affects the frequency of accessing episodes from memory. More concretely, source analogs containing elements which are semantically similar to a given target element are accessed more frequently when this target element is attached earlier to the input node.

6.4.1.1 Design

The experiment consists of three conditions. The same target problem is presented to the system in all three conditions. In the control condition all target elements are attached simultaneously to the input and goal nodes. In the two experimental conditions the elements are attached in two different (and roughly reverse) orders. The dependent variables are frequencies of accessing and mapping the episodes in the long-term memory.

6.4.1.2 Materials

Target situation EHW presented in section 6.2.6 is used as a target problem. Its verbal description is reproduced below. The 100 variants of the knowledge base described in section 6.2.1 are used as replications.

Target situation EHW (Egg in Hot Water): *There is a teapot and some water in it. There is an egg in the water. The teapot is made of metal. The color of the egg is white. The temperature of the water is high.*

The goal, if any, is not represented explicitly.

What is the outcome of this state of affairs?

Note the following details of this description. On one hand, there is some water whose temperature is high. These elements are similar to the source analogs related to heating water and in particular to situations WTP (Water in a Teapot on a Plate) and IHC (Immersion Heater in a Cup). On the other hand, there is an egg whose color is white. These elements are similar to situation ERW (Egg in Red Water) described in section 6.1.

6.4.1.3 Procedure

The target problem is run three times on the set of 100 knowledge bases, yielding a total of 300 runs. In the control condition, all target elements are attached to the input node at the beginning of the run. The number of times that each of the twelve episodes in the long-term memory are accessed and mapped is recorded.

In the *hot water condition* the agents `water-EHW`, `T-of-EHW`, and `high-T-EHW` are attached to the input node at time zero. The remaining target elements are attached later according to the schedule shown in the left column of Table 6.3.

In the *colored egg condition* the agents `egg-EHW`, `color-of-EHW`, and `white-EHW` are attached to the input node at time zero. The remaining target elements are attached later according to the schedule shown in the right column of Table 6.3.

Table 6.3 Time schedule for attaching different target elements in the two experimental conditions.

Time	*hot water* condition	*colored egg* condition
0	`water`	`egg`
	`high-T`	`white`
	`T-of (water, high-T)`	`color-of (egg, white)`
5	`teapot`	`teapot`
10	`metal`	`metal`
	`made-of (teapot, metal)`	`made-of (teapot, metal)`
15	`in (water, teapot)`	`water`
20	`egg`	`in (egg, water)`
25	`in (egg, water)`	`in (water, teapot)`
30	`white`	`high-T`
	`color-of (egg, white)`	`T-of (water, high-T)`
35	`endst`	`endst`
	`follows (initst, endst)`	`follows(initst, endst)`

6.4.2 Results and Discussion

Figure 6.11 shows bar plots of the frequencies obtained in the two experimental conditions. The bar plot for the control condition is shown in Figure 6.5.

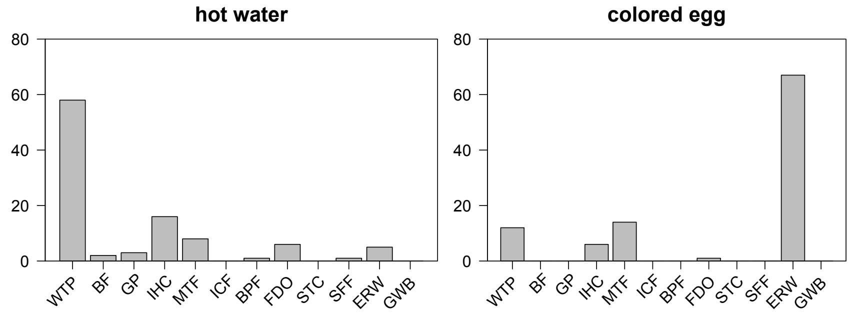


Fig. 6.11 Bar plots showing the frequencies of accessing and mapping each long-term memory episode in the *hot water* and *colored egg* conditions, respectively.

Table 6.4 Observed frequencies of accessing base episodes from memory for the two experimental conditions. The control condition (in parentheses) defines the expected frequencies for the chi-square test. $\chi^2 = 89.5$, df $= 7$, $p < 0.00001$.

Condition	WTP	IHC	ERW	Other	Total
Hot water	58 (37)	16 (11)	5 (35)	21 (17)	100
Color egg	12 (37)	6 (11)	67 (35)	15 (17)	100
Total	70 (74)	22 (22)	72 (70)	36 (34)	100

The data show that each experimental condition differs from the control and from each other. The difference is very significant according to the chi-square test ($\chi^2 = 89.5$, df $= 7$, $p < 0.00001$, Table 6.4). Moreover, the effect is in the predicted direction — the two base situations about heating water appear much more frequently in the hot water condition. The reverse pattern holds for the episode about coloring an egg (ERW).

Thus, order of presentation of the target problem influences the process of accessing source analogs in AMBR. As the mapping process in the model is intimately intertwined with access, it is influenced too. Moreover, the direction of influence is in accord with the well-known primacy effects demonstrated in many studies of short-term memory (e.g., Postman & Phillips, 1965). Elements that appear earlier have greater impact than later elements.

AMBR differs from other models of analog retrieval with respect to the primacy effect (Forbus, Gentner, & Law, 1994; Hummel & Holyoak, 1997). As far as we can judge from the articles, neither MAC/FAC nor LISA predict such order effect on analog access. The first stage of MAC/FAC depends on dot products over feature vectors and, therefore, all target elements necessarily enter simultaneously. Thus the model must wait until all target elements are available and only then can trigger the retrieval process.

LISA do present target (or more precisely *driver*) elements in a temporal order. Indeed, this is the only way of processing available to LISA due to the limitations of the phase set. As argued in section 4.5.1, however, the model uses centralized representation of situations. Therefore, episodes are retrieved as units — either all nodes are flipped from *dormant* to *recipient* mode or none of them. In the current version of LISA this decision is taken probabilistically based on the *Luce retrieval index* computed for each episode in LTM (Hummel, personal communication, January 1998). The important point is that the indices are computed after multiple iterations through the whole driver set. The article does not specify the moment in which the probabilistic decision about bringing an episode to the working memory is taken (Hummel & Holyoak, 1997). If we suppose that this happens after the network has settled, the order of the driver set would have negligible effect on the retrieval indices.

Chapter 7
Possibilities for Future Extensions of AMBR

Throughout this book we have emphasized that analogy-making cannot be decomposed into a sequence of independent components. AMBR advocates an interactionist emergent approach and conceptualizes analogy-making in terms of overlapping subprocesses (Figure 3.4). Still, the current version of the model addresses mainly the subprocesses of access and mapping. Does this mean that AMBR assumes that these two subprocesses can be modeled separately from the rest?

The problem lies in the complexity of analogy-making. As we have argued, it is not an isolated module but emerges out of the general cognitive architecture. We agree with the closing statement of Robert French's (2002) review:

> Analogy-making is so intimately and so deeply part of human cognition that it is probably safe to say that any program capable of doing analogy-making in a manner truly comparable to human beings would stand a very good chance of passing the Turing Test.

For instance, Chalmers, French, and Hofstadter (1992) have argued that analogy is inseparable from high-level perception. Without perception, the mapping between the base and the target is in effect contained in latent form in the representations of the two episodes. As AMBR starts from hand-coded descriptions, it can be criticized for bypassing the really hard problems of analogy-making. The lack of mechanisms for high-level perception definitely is a limitation of the model and we plan to include such mechanisms in a future extension of AMBR. But high-level perception is obviously rooted in low-level perception. Perception at all levels involves attention, which in turn depends on motivation, which is culturally grounded, and so forth.

It follows that *any* model of analogy-making is necessarily incomplete. AMBR makes no exception. We hope, however, that it is open-ended enough to be able to grow. This chapter suggests ways for extending the model in two directions: transfer and perception.

7.1 Possibilities for Transfer

In the current version of AMBR, each run of the model ends in the following way: One by one the *authorized secretaries* (i.e. the agents from the target, see Section 5.6) select one of the hypotheses registered at them and send it a *promotion incentive*. The promoted agents enter the third phase of their life cycle and become *winner hypotheses*. All other hypotheses registered at the respective secretaries become losers. In this way, the model makes commitments about the correspondences between the two episodes. As there are no mechanisms that can advance the process further, the model stops. All symbolic activity comes to an end. The activation in the network reaches a steady state.

In a hypothetical future version of AMBR, the mapping between the base and the target will be used for generating inferences in the target (and possibly the base). We refer to this process as *transfer*. It needs to answer at least three questions:

1. Which members of the two descriptions remain unmapped? These are the potential candidates for transfer. As AMBR uses decentralized representations of situations, this question cannot be answered by going through some list-like structure and crossing out mapped entities.
2. Which unmapped elements really merit transferring? This is a very difficult question. For example, suppose the target problem is to heat some milk in a teapot. The base contains water, a teapot, and a hot-plate (among other things). The color of the teapot in the base is green. The temperature of the plate is high. Neither proposition has an analog in the target and, therefore, both are candidates for transfer. Perhaps the milk will get hot if one paints the teapot green?
3. How to carry elements from the one domain to the other? Objects and propositions in the base cannot be copied literally to the target; they must be "translated." The translation process is sometimes called *copying with substitution and generation* (Holyoak, Novick, & Melz, 1994).

This list looks like a sequence of steps but it should not be understood in this way. According to the overall AMBR philosophy, these "steps" overlap in time. Whenever an element is identified as unmapped (point 1), the evaluation of its relevance and potential usefulness could begin (point 2). There is no need to wait for the other unmapped agents. In addition, the potential usefulness of an element depends on the quality of the inferences that this element could "propose" (point 3). Hence, in our view the whole process should be modeled by a "wave" similar to the one outlined in Section 3.2.1.

How could the transfer process be carried out by AMBR mechanisms? Let us start with the first question above. One possible answer is that the secretaries of the target are authorized to judge which elements are *mapped*, whereas the secretaries of the source are authorized to judge which elements are *unmapped*.

Hummel & Holyoak (1996, 1997) propose two very useful concepts. In addition to the conventional target/base distinction, they introduce a *driver/recipient* distinction. The driver is the one that has the initiative and "makes things happen." In AMBR terminology, it is *authorized*. This could be the target problem or the source episode. Hummel & Holyoak (1997) suggest the following canonical flow of control: First the target is used as driver during the access stage. Once a source is in working memory, mapping can be performed in either direction (including successive switches between the two episodes). After the mapping stage is over, the source is used to drive inferences and schema induction in the target.

We adopt the driver/recipient terminology and agree with the main idea of the previous paragraph. However, we propose a modification—the switches between driver mode and recipient mode should not be done in a way that serializes the process of analogy-making and cut it into separate stages (marked by interventions of the human user). Among other things, this implies that it should be possible that *both situations act as drivers simultaneously*.

For lack of better terminology, we will denote the situation (or, more precisely, the elements thereof) that drives the mapping as *driver-M*. The one that drives the transfer is *driver-T*. The two are authorized for different and complementary activities.

The target problem typically acts as driver-M in AMBR. Its agents access (partially) episodes from the long-term memory, establish correspondences, administer rating surveys, and promote winners. When a source episode emerges as winner, it becomes driver-T and its agents become authorized to identify unmapped elements, judge their potential usefulness for transfer, propose translations in the target, etc. The two coalitions—driver-M and driver-T—work together, each according to its authorization. In this way, the transfer subprocess overlaps in time with mapping, potentially altering the balance in the constraint satisfaction network and affecting the correspondences that remain to be promoted.

More concretely, the driver-T secretaries could identify whether they are unmapped or not by means of a *constraint propagation mechanism* (e.g., Waltz, 1975). Commitment in one place (in the form of a winner promotion) propagates to other places. Consider the example in Figure 7.1.

Figure 7.1 illustrates a fragment of the network at the moment when the driver-M coalition (to the right) has established several hypotheses with the recipient coalition (in the middle). Note that two hypotheses compete for the agent `T-of-w-1`. The secretary of the latter is authorized to promote one of them as winner. Suppose the hypothesis `T-of-w-1<->T-of-w-2` is the winner (due to structural and semantic pressures in the CSN as both propositions involve water). When it is promoted, the rival hypothesis `T-of-w-1<->T-of-p-2` becomes a loser. It notifies the secretary of `T-of-p-2` about this. The latter agent belongs to the recipient situation (in the middle). This same situation, however, is driver-T at the same time. As such, it checks whether it has at least one non-loser hypothesis on its record. When `T-of-w-1<->T-of-p-2` becomes a loser, the secretary detects that `T-of-p-2` is unmapped.

Driver-T secretaries are authorized to trigger the skolemization mechanism (just as driver-M secretaries are). Thus, the general proposition that the temperature of heat-sources is high could be used to augment the driver-M situation. This could be done in the following way: The agent `T-of-p-2`, having missed the chance to map to `T-of-w-1`, now takes the initiative and issues a marker. (Note that it acts as a

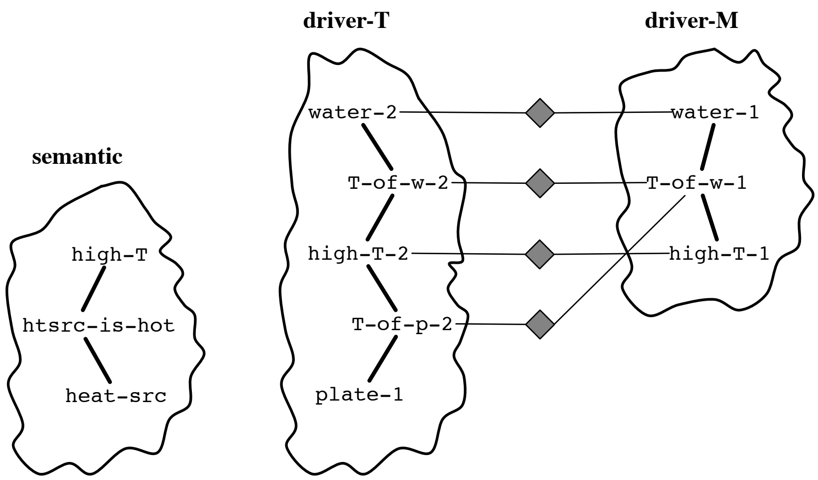


Fig. 7.1 The *driver-M* coalition (to the right) has established four hypotheses (diamonds) with the *driver-T* coalition (in the middle). A general proposition waits in the semantic memory (to the left) and could be used for skolemization. See text for details.

driver at this moment.) As described in Section 5.3, this marker goes to the parent concept `temperature-of`. It intersects there with the marker issued from the general proposition `htsrc-is-hot`. The marker intersection leads to a construction of a new hypothesis agent: `T-of-p-2<->htsrc-is-hot`. The symbolic processor of this new agent can carry out the skolemization protocol (Section 5.7) and augment the description of the "other" situation. As this particular instantiation of the skolemization mechanism has been triggered by the driver-T situation, the new Skolem instances will be added to the driver-M situation. (Note that the latter acts as a recipient with respect to driver-T.)

The skolemization mechanism will re-use the agent `high-T-1` in the recipient. As the concept agent `heat-source` has received no marker from the same coalition, a new Skolem instance will be created and affiliated to the episode shown to the right in Figure 7.1. Suppose the name of this new agent is `sk-htsrc-1`. Finally, a Skolem proposition will be created. It binds `sk-htsrc-1` and `high-T-1` as arguments of a `temperature-of` relation. Let the name of this latter proposition is `sk-T-of-1`. Figure 7.2 depicts the resulting configuration.

The new agents `sk-htsrc-1` and `sk-T-of-1` affiliate to the driver-M coalition. In this way, the description of the target problem is augmented with a heat source. The new agents now take the initiative and issue markers. These markers will create the hypotheses `sk-htsrc-1<->plate-2` and `sk-T-of-1<->T-of-p-2`. Hence, `plate-2` and `T-of-p-2` no longer are unmapped. After the rating mechanism runs its course, the new hypotheses will be promoted as winners. In particular, the hypothesis `sk-T-of-1<->T-of-p-2` will eliminate the general hy-

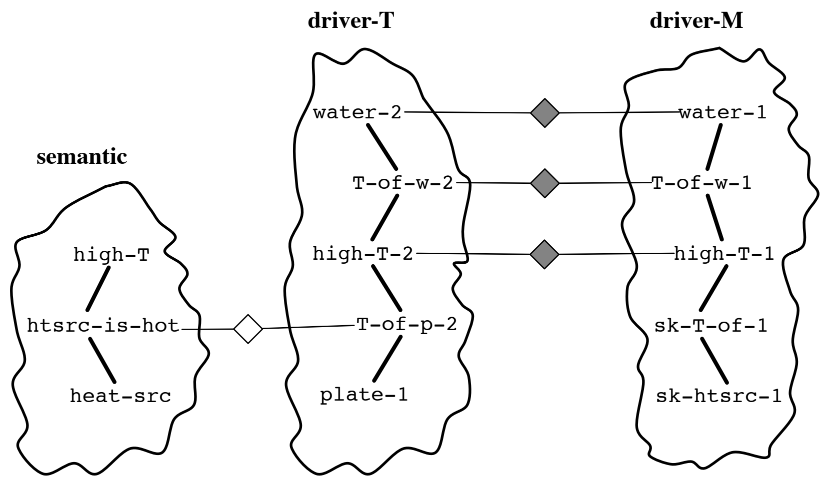


Fig. 7.2 State of the network after the skolemization mechanism has added two new Skolem agents to the *driver-M* coalition introduced in Figure 7.1. The general-hypothesis agent that has carried out the skolemization is depicted by a white diamond. It will be eliminated when the new Skolem instances create hypotheses of their own. See text for details.

pothesis `T-of-p-2<->htsrc-is-hot` that had carried out the skolemization process.

This example suggests that the existing AMBR mechanisms can be useful not only for the processes of analog access and mapping but for the transfer process too. The utility of the mechanisms of rating, marker passing, and skolemization is clear from the example. The other mechanisms are potentially useful too. The spreading activation is a key mechanism for estimating relevance, and such estimates will surely be needed for the selection of candidates for transfer. The constraint satisfaction is also useful when there is a need for selecting one option among alternatives.

Considerations of this kind make us believe that the AMBR model is open-ended enough and its functionality could be extended in the direction of analogical transfer. Moreover, we hope that this could be done without giving up the properties of the current version. Analog transfer could be done in a dynamic emergent way over decentralized representations. It could run in parallel with the subprocesses of mapping and access.

7.2 Possibilities for Perception

This section suggests how AMBR could be extended in the direction of high-level perception (Kokinov et al., 2007; Nestor & Kokinov, 2004; Petkov & Shahbazyan, 2007). As argued by Hofstadter (1984, 1995) and Chalmers, French, & Hofstadter (1992), the process of building representations is a crucial part of analogy-making. The same authors defend the methodological utility of micro-domains for research on high-level perception. (See Forbus et al., 1998, for a critique of this view.) Micro-domains allow the model to focus on building structured representations instead of dealing with low-level details such as filtering noise from images.

One such micro-domain that could be used in the research on DUAL and AMBR is the so-called TEXTSCREEN. It is based on an imaginary text processing program. TEXTSCREEN is deliberately simplified—there is plain text over a limited matrix of screen positions. There are objects like characters, marked areas, etc. The characters can be grouped in words, lines, paragraphs, columns, etc. Most of the objects are directly visible on the screen, where they tend to form regular rectangular patterns.

The objects have attributes such as `long` and `vertical`. There are also a number of relations such as `left-of`, `part-of`, `aligned-with`, etc. Finally, there are various actions (or commands) to navigate through the text, to insert, delete, or move objects, and mark portions of text and thus form aggregate units for subsequent manipulation.

This material is rich enough to allow various configurations on the screen (see Figure 7.3). A model that operates in this environment is presented with a situation which has some defect somewhere on the screen and the task of the system is to locate the defect and correct it. To that end, the model can use previous situations (with solutions) as source analogs.

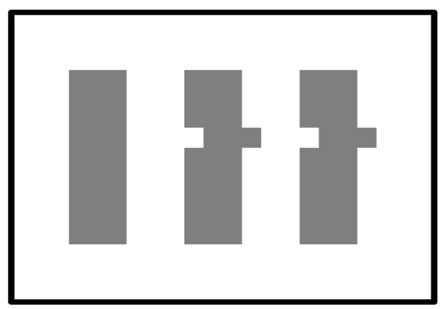


Fig. 7.3 Sample problem in the TEXTSCREEN micro domain.

Obviously, visual perception has much to do with space. Even in a simplified two-dimensional micro-world like TEXTSCREEN, spatial properties, relations, and configurations are all important. This characteristic feature of the environment must be reflected somehow in the cognitive architecture DUAL. The main network cannot meet this requirement because it lacks spatial organization. Therefore, we plan to augment the architecture with a large-scale structure having explicit spatial organization—the *visual array* (VA; Nestor & Kokinov, 2004).

Just like everything else in DUAL, the visual array consists of agents. The defining characteristic of the array is that the agents are arranged in a rectangular matrix. Each agent in the array is associated with a particular position on TEXTSCREEN and can "see" whether the cell is empty or not. Thus the VA is a mediator between two different worlds—the external environment of TEXTSCREEN and the internal representations in the main network. The defining principle of TEXTSCREEN is physical location. On the other hand, the defining principle of the network is interconnectivity. These two principles meet in the visual array—the visual agents have both physical locations and links to other agents. For instance, each agent is linked to the agents in the four neighboring cells. It can interact with them, send them symbols and activation, and so forth.

There are other perceptual agents that are connected to a whole row or column of the agents in the VA. These agents can detect straight lines, lines with defects, etc. When they locate an object in their receptive field, they create a new temporary agent in the main network that represents this external object. Other specialized perceptual agents combine lines in regions and identify various spatial relations between them. They build new agents in the network to represent these regions and relations. Still other agents group things together or parse a complex object into parts. Each perceptual agent works at its own speed depending on its activation level. The activation in turns reflects two kinds of influences: bottom-up from the VA and top-down from the parent concept in the network.

The visual array is a source of activation. It will replace the input node of the current version of DUAL and AMBR. Instead of receiving a hand-coded description of the scene in the form of agents attached manually to the input list, the model should be able to construct its own representations. The representation of each scene would be built agent by agent. Each new agent enters the working memory, sends activation to its respective concept agent, and emits a marker to trigger the mechanisms for finding correspondences. As the simulation experiments on order effects have demonstrated (Section 6.4), AMBR is capable to handle target problems that are presented piecemeal over an extended period of time.

Chapter 8
Conclusion

8.1 Overview of the Book

This book describes AMBR2—a dynamic emergent integrated model of analogical access and mapping based on decentralized representations of situations. It describes in detail the knowledge structures and computational mechanisms used in the model. The behavior of the model is illustrated by many examples, diagrams, and transcripts of actual runs of the computer implementation of the model. The book reports the results of various simulation experiments involving more than 1,200 runs of the program on different target problems. AMBR is compared with a selection of other models and is discussed in the light of the studies of human analogy-making.

AMBR is an emergent and decentralized model. It consists of a population of small entities called *Dual agents*. These agents are the ingredients of the DUAL cognitive architecture that is the foundation of AMBR. They represent all the knowledge and carry out all the processing in the architecture. There is no central executive that controls the operation of the system as a whole. Instead, each agent works locally and performs its simple specific task in close interaction with its immediate neighbors. The global behavior of the model emerges of the coordinated effort of these asynchronous local activities.

AMBR applies the same approach to the phenomena it is intended to model. The subprocesses of analogy-making are explained in terms of coordinated mechanisms. The main intuition behind the research reported here is that there is no "analogy machine" that does analogies according to some fixed centralized algorithm. Instead, analogy is an emergent product of the work of general cognitive mechanisms. The book tries to demonstrate that such approach is feasible. Thus, analog access is based on the mechanism of spreading activation which serves a range of other purposes in the cognitive architecture. The constraint satisfaction mechanism is used for finding correspondences in the model but the same mechanism can apply to various other tasks such as perception and decision making.

AMBR representations of episodes are decentralized. The model does not maintain data structures listing the elements that belong to each situation. Instead, each situation is represented by a *coalition* of agents. This allows for greater flexibility of the representations. New elements can be added when necessary. The skolemization mechanism can augment the description of a given episode based on general semantic information. In the same time, elements that have been needed in the past and potentially belong to the description of the episode stay out of the working memory when they are irrelevant for the problem being solved. Thus the model is capable to re-represent a situation both by addition and omission of elements (Kokinov & Petrov, 2000, 2001). Chapter 5 demonstrates this on a concrete example.

The theme of integration is central to AMBR research. The model conceptualizes the components of analogy-making not as sequential "stages" but as *subprocesses* that run in parallel and interact (Figure 3.4). The version reported in this book integrates the subprocesses of analog access and mapping. A case study reported in Chapter 6 illustrates an interaction of this kind. Other simulation experiments from the same chapter also demonstrate various aspects of these interactions. Chapter 7 suggests possibilities for modeling the subprocesses of transfer and perception. It is argued that they could be added to the current version of the model without forcing radical reconsideration of the existing mechanisms.

Dynamic computation is a characteristic feature of the architecture DUAL and, consequently, of the model built on top of it. Each DUAL agent works at its own speed that varies dynamically as the activation level of the agent vary. Thus, more relevant agents work faster and contribute more to the overall behavior of the system compared to less relevant (and hence less active) ones. In addition, the topology of the AMBR network is constantly changing as new nodes and links are created while others are removed. This *dynamic emergent computation* provides for flexibility and efficiency at the same time (Kokinov, Nikolov, & Petrov, 1996).

8.2 Contributions of This Work

The research reported in this book has made several extensions and improvements of the AMBR model and DUAL architecture with respect to the earlier specification (Kokinov, 1994a). In our estimation, the major contributions are:

8.2.1 From **AMBR1** *(Kokinov, 1994) to* **AMBR2A** *(Petrov, 1997)*

- By far the most important contribution is the transition from centralized to *decentralized representation* of situations in AMBR. In turn, this led to improvements in the marker passing, structure correspondence, and constraint satisfaction mechanisms. It is also an important factor for the integration of the different subprocesses of analogy-making in the model. After 1998, the concept of decentralized representations stimulated some very interesting research on blending of episodes (e.g, Grinberg & Kokinov, 2003; Kokinov & Zareva-Toncheva, 2001;

Zareva & Kokinov, 2003) and on exploring the interplay between memory and reasoning (e.g., Kokinov, 2003, 2006; Kokinov & Petrov, 2001).

- Introduction of the *energetic analogy* and the mechanism of *consumptions* for specifying the exact relationship between the activation level of a DUAL agent and the speed of its symbolic processor (Appendix C; Petrov, 1997; Petrov & Kokinov, 1999).
- Introduction of the notion of *coalitions* and the intermediate level of description of the architecture (the *meso-level*). The conceptual apparatus of coalitions is an important tool for developing and communicating the ideas about emergent computation and decentralized representations.
- Introduction of *secretaries* for the purpose of incremental construction of the constraint satisfaction network. The presence of secretaries also prepares the ground for the rating mechanism in AMBR2B.
- Disclosing the deficiencies of the *activation function* used in AMBR1 and replacing it with a more appropriate one. Detailed mathematical analysis of these functions.
- Developing, testing, and documenting a portable computer implementation of the architecture and the model. The program has been tested under two platforms: Allegro Common Lisp (Windows) and Carnegie Mellon Common Lisp (Unix).
- Enlarging the knowledge base and performing simulation experiments with AMBR2A.

8.2.2 *From* AMBR2A *to* AMBR2B

- Introduction of the mechanisms for *rating* and *promotion*. The authorized secretaries in AMBR2B monitor the activation levels of the hypothesis registered at them. Secretaries promote *winners* and eliminate *losers* when appropriate. Thus the outcome of the mapping process is available within the model itself; there is no need for an external observer to read out the answer from the activation pattern in the constraint satisfaction network. In addition, loser elimination reduces the size of the CSN and opens new possibilities for incremental processing as discussed in Section 5.6.1.3. The rating mechanism also performs *ballotages* to prevent implausible blendings and trigger the skolemization mechanism. The life cycle of hypothesis agents is elaborated.
- Introduction of the *skolemization* mechanism for the purpose of re-representation of past episodes accessed from long-term memory. In this way, general semantic information can be used to augment the descriptions of episodes upon necessity. To our knowledge, this is the first attempt for re-representation of *past* episodes in analogy-making.[1] The skolemization mechanism will undoubtedly be useful for the transfer process too (cf. Chapter 7; Kokinov & Petrov, 2000, 2001).

[1] A different form of skolemization is used in the Structure Mapping Engine for the purpose of positing conjectural entities in the target (Falkenhainer, Forbus, & Gentner, 1989). Section 5.7.4 compares and contrasts the two forms of skolemization.

- Extending the structure correspondence mechanism with abilities for *weak structure correspondence*. It improves the connectivity of the CSN by creating new links (but not new hypothesis agents). Combined with the differential link weighting adopted in AMBR2B, this improves the structural constraint on analogical mapping.
- Elaborating the description of the episodes in the knowledge base and addition of new episodes and concepts. The total number of agents is more than doubled with respect to AMBR2A. There is richer representation of the causal structure of each base situation.
- The simulation experiments with AMBR2B reported in this book involve more than 1,200 runs of the program and show the behavior of the model in detail. The interaction between analog access and mapping is explored. An experiment on order effects shows that AMBR2B is sensitive to the order of presentation of the target elements.
- All new mechanisms are implemented in the computer program. There are also a number of technical improvements of the old implementation. (For example, the routines performed by the symbolic processors of AMBR2A agents were interpreted. In AMBR2B they are compiled.)

8.3 Suggestions for Future Research

Each end is a new beginning.

As stated repeatedly in this book, AMBR2 is but an intermediate stage in a long-term research program. There are many ways in which this research can be continued. Some of them are suggested in this final section.

To begin with, much more experimentation could (and should) be done with the existing version of the model. There are a number of interesting effects that are within its scope but have not been demonstrated in rigorous simulation experiments. For example, AMBR2 could map propositions with different number of arguments, map an object to a relation, etc. The experiments on priming and context effects performed by Kokinov (1994a) could also be replicated and extended. The model should be tested on new kinds of problems in different domains. Of particular interest is whether the model will scale up to larger memory sizes. The sensitivity and robustness of the model for different values of its various parameters is another issue that has not been covered here.

Another possibility for research is to design and implement new computational mechanisms and extend the functionality of AMBR. The subprocess of transfer seems within closest reach. The mechanisms of constraint propagation, switching the base as driver, and skolemization from base to target outlined in Chapter 7 provide a starting point.

A major research direction is to add perceptual capabilities to DUAL and AMBR. This involves the visual array mentioned in Chapter 7 and the TEXTSCREEN micro-domain (Nestor & Kokinov, 2004). The integration of the perceptual mechanisms with the existing computational machinery is a very challenging and intriguing topic. Another research direction of comparable complexity and import is to add learning mechanisms to the architecture.

The research on AMBR involves psychological experimentation too. For instance, the order effect on access presented in sections 5.2.4 and 6.4 is a prediction of the model that could be tested empirically.

The closing statement of Turing's (1950) seminal paper applies here as well: "We can only see a short distance ahead, but we can see plenty there that needs to be done."

Afterword: A Sample of AMBR Research

Georgi Petkov
New Bulgarian University[1]

After 2005, AMBR research continued in two main directions: the JUDGEMAP model for judgment on a scale (Kokinov, Hristova, & Petkov, 2004; Petkov, 2005; Petkov & Kokinov, 2006) and the transfer mechanism (Kiryazov, Petkov, Grinberg, Kokinov, & Balkenius 2007; Kokinov, Grinberg, Petkov, & Kiryazov, 2008; Petkov, Kiryazov, Grinberg, & Kokinov, 2007; Petkov & Shahbazyan, 2007; Shahbazyan & Petkov, 2007). The overall strategy of the research program was motivated by the hypothesis that analogy-making lies at the core of human cognition. As such, we expected that the basic mechanisms for analogy-making would support a wide spectrum of cognitive tasks. We set out to explore the limits of these mechanisms.

The JUDGEMAP model (Petkov, 2005; Petkov & Kokinov, 2006) was the first attempt in this direction. We modeled the process of judgment on a numeric scale, using the basic mechanisms of DUAL and AMBR. JUDGEMAP successfully accounted for various known phenomena of human judgment, including the contrast effect (systematic shift of the judgments in the direction opposite to a certain contextual element) and the assimilation effect (systematic shift of the judgments toward a contextual element). In addition, the model made a novel prediction that was successfully validated with empirical data (Kokinov, Hristova, & Petkov, 2004;

[1] Boicho Kokinov—the author of DUAL and AMBR—meant to write this Afterword but was prevented by his untimely death. Boicho's plan was to summarize concisely some of the developments of the AMBR research program since the 1998 version described in this book. Some key references to Boicho's subsequent work are cited in earlier chapters (e.g., Grinberg & Kokinov, 2003; Kokinov, 2003, 2006; Kokinov, Hristova, & Petkov, 2004; Kokinov & Petrov, 2000, 2001; Kokinov & Zareva-Toncheva, 2001; Nestor & Kokinov, 2004; Petrov & Kokinov, 1999; Zareva & Kokinov, 2003). This is only the tip of the iceberg. A full list of Boicho's publications is available from his web page at http://www.nbu.bg/cogs/personal/kokinov/.
I am very grateful to Georgi Petkov for rising to the challenge and writing this Afterword in Boicho's stead. Georgi is another one of Boicho's doctoral students. At present, he is an Assistant Professor at the Central and Eastern European Center for Cognitive Science at NBU. He has collaborated extensively with Boicho on AMBR and other models, some of which are outlined here. Thank you, Georgi! – *Alex Petrov*

Petkov, 2005). The JUDGEMAP model introduced a method for representing continuum magnitudes with structural representations in DUAL. It also pioneered the conceptualization of decision-making as a process of judgment on a two-point scale.

The other major research direction was to continue the development of the mechanism of transfer of knowledge from the base to the target. The attempt was to merge it with the other mechanisms of AMBR according to the main principles of DUAL and AMBR: Transfer is context-sensitive, emergent, and dynamic. The transfer mechanism is integrated with the other mechanisms and overlaps with them in time (cf. Figure 3.4). Every agent works autonomously, at a speed proportional to its activation, and does not wait for the other agents. The overall behavior of the system emerges from the asynchronous work of the individual agents without any central controller (Kokinov, Nikolov, & Petrov, 1996).

The RecMap model of high-level vision (Shahbazyan & Petkov, 2007; Petkov & Shahbazyan, 2007) uses a specific type of transfer mechanism called *anticipatory mechanism* (Kiryazov et al., 2007; Kokinov et al., 2008; Petkov et al., 2007). It is easiest to illustrate this new mechanism on a concrete example. Suppose that a target situation T is currently being mapped onto a base situation B. The base includes the proposition `Relation-R1-B(arg1-B,arg2-B)`, represented by a coalition of three instance-agents as illustrated in the left "bubble" in Figure 9.1 (see also Figure 4.2). Suppose further that mature hypothesis-agents have been established (cf. Section 5.4.4) that tentatively map *all* arguments of this predicate onto some instances in the target. In our example, `arg1-B` and `arg2-B` have hypothesis-agents (depicted as grey diamonds) mapping them onto `element1-T` and `element2-T`, respectively. This triggers the creation of a new *anticipation-agent* that represents the conjecture that a proposition exists in the target situation that corresponds to `Relation-R1-B` in the base. This conjectural proposition is represented by the agent `ANTICIPATION-Relation-R1-T`, depicted as a grey rectangle in Figure 9.1.

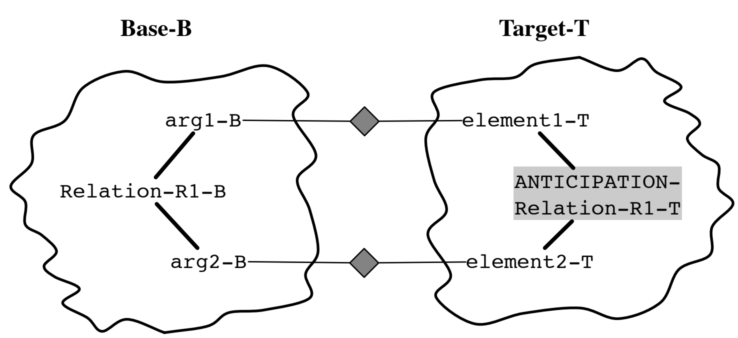


Fig. 9.1 Example of the anticipatory mechanism. Suppose all arguments of a relation in some base situation are mapped to elements from the target situation, but those elements are not interconnected with a relation. Then a copy of this relation is created in the target (depicted as a grey rectangle). This copy is a temporary agent of type `:anticipation-agent`.

The new type of anticipation-agents extends the hierarchy of agent types in AMBR (cf. Figure 3.5). Anticipation-agents are temporary agents, just as hypothesis-agents are temporary. They are discussed in more detail in Section 9.2.1 below.

The anticipatory mechanism is still under development. The transfer should propagate upward in the class hierarchy. More precisely, in many cases it is desirable to create in the target a relational instance-agent that does not belong to literally the same concept class as the corresponding relational instance-agent in the base. Often these two instances are not from the same class but belong to a common superclass. Thus, when a copy of a certain relation is created in the target, copies of the relations above in the class hierarchy should be created as well. This is a topic of further extensions of the model mechanisms.

These first steps of the development of the model were tested with various simulations: A set of simulations tested the RecMap model for high-level vision (Shahbazyan, Petkov, 2007, Petkov, Shahbazyan, 2007). The AMBR model itself, enriched with the anticipatory mechanism, was deployed in robots and tested in a real environment (Kiryazov et al., 2007; Kokinov, Grinberg, Petkov, & Kiryazov, 2008; Petkov, Naydenov, Grinberg, & Kokinov, 2006).

9.1 The JUDGEMAP model

JUDGEMAP is a model of category rating and choice that assumes the structure-mapping ability is fundamental for human cognition (Petkov, 2005; Petkov & Kokinov, 2006). In sharp contrast to all other theories of category rating, JUDGEMAP conceptualizes the process of judgment as a process of mapping a set of stimuli onto the set of scale elements. In other words, the participant always needs a whole *set* of elements to rate—even if he/she has to evaluate only a single stimulus, he/she constructs an interrelated set of items that includes the target stimulus, and then maps the whole set onto the rating scale (Figure 9.2).

JUDGEMAP does not assume the presence of any centralized and static representation of the response categories. This parallels AMBR's avoidance of centralized representations of situations (Section 4.5). This contrasts with other models (e.g., Petrov & Anderson, 2005) that assume that each response category is represented by an anchor, or prototype, or standard, or criterion. In such models one can simply compare the target stimulus with this standard/prototype/anchor and thereby obtain a rating. JUDGEMAP, on the other hand, requires the dynamic construction of a *comparison set*. This peculiar characteristic of the current approach makes it unique in terms of its high context-sensitivity because the formation of the comparison set (and thus the eventual rating that will be produced on its basis) is dynamic and can be influenced in various ways (Hristova & Kokinov, 2006; Petkov et al., 2005).

The process of judgment in JUDGEMAP consists of two overlapping and interdependent subprocesses: formation of the comparison set and mapping of the comparison set onto the set of scale elements.

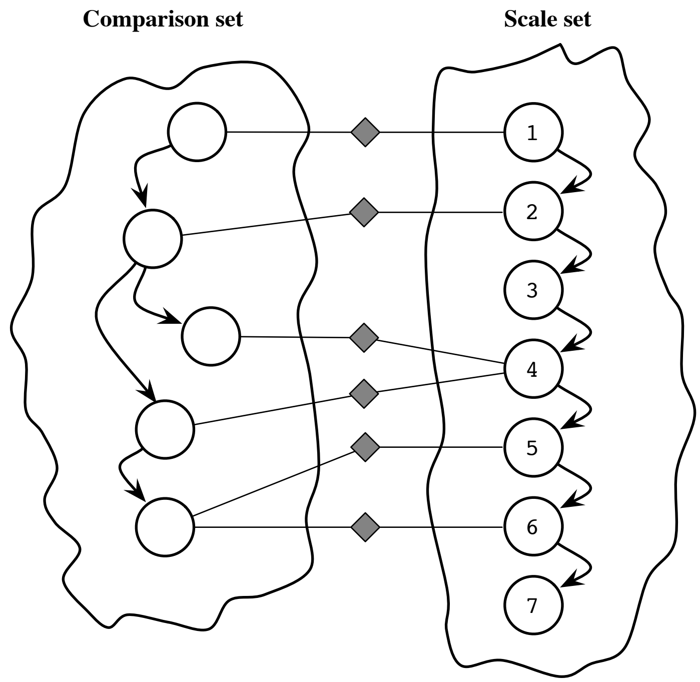


Fig. 9.2 Illustration of the basic idea of the JUDGEMAP model. A dynamically constructed comparison set is mapped onto the set of scale ratings. The mapping preserves the structure of the ordering relations (depicted as arrows). Hypothesis agents are depicted as grey diamonds.

The comparison set in JUDGEMAP consists of those elements whose representations happen to be activated in the judge's working memory on the particular occasion. There are two main sources of comparison set-elements: They may come from perception (if the judge encounters other elements in the environment) and from long-term memory (if the target element reminds the judge of some previously encountered elements). In this respect JUDGEMAP is similar to the EBRW model (Nosofsky & Palmeri, 1997), the Norm Theory (Kahneman & Miller, 1986), and the ANCHOR model (Petrov & Anderson, 2005).

The real difference with all other models, however, is in the mapping subprocess. JUDGEMAP thus builds a bridge between the judgment literature and the analogy-making literature. The mapping subprocess has to preserve the structure of ordering relations in the comparison set when finding their corresponding elements in the scale set (Figure 9.2). This parallels the structure sensitivity of the process of mapping in analogy-making. It is thus natural that JUDGEMAP relies heavily on AMBR's mapping mechanisms developed for analogy-making. This integration with AMBR imposed severe constraints on JUDGEMAP—one cannot simply postulate whatever mechanisms or representations would fit the experimental data on category rating and judgment. JUDGEMAP kept the principles of AMBR and preserved its existing mechanisms. New mechanisms were introduced only when necessary and with great care. In this way, fitting the data was not a simple process. Rather, the fit arose naturally out of AMBR's principles and mechanisms. In sum, the process of judgment was not modeled in isolation, but was integrated with analogy, memory, and in the future with perception and learning. This integrative approach also allowed for modeling and explaining the interactions between various cognitive processes.

9.1.1 Main Principles of JUDGEMAP

JUDGEMAP consists of nothing but DUAL agents of various kinds. Relative to AMBR, it introduces two two novel kinds of agent (*comparison relations* and *correspondence relations*), one novel kind of slot (`:amount`), and a few novel symbolic procedures.

Comparison relations represent classes of specific relations that have two arguments and express a comparison between these arguments. Examples of comparison-relations are concepts like `longer-than`, `cheaper-than`, `better-than`, etc. The class of comparison-relation agents is a subclass of the class of concept-agents (cf. Fig. 3.5). What sets comparison relations apart from ordinary concept-agents is that they are equipped with specialized procedural knowledge that allows them to recognize manifestations of the relation they represent. For example, the agent `longer-than` can compare lengths and `cheaper-than` can compare prices of relevant items. Thus, comparison-relations act as detectors of the respective relation in the environment.

Correspondence relations represent specific judgment tasks. For example, if the task is to judge line lengths on a scale, a correspondence-relation agent represents the proposition that longer lines correspond to higher ratings. The class of correspondence-relation agents is a subclass of the class of hypothesis-agents. Thus, they are temporary and do not participate in long-term memory. They are always active during a given judgment episode because they are attached to the GOAL node, which is a strong source of activation in DUAL (Section 3.1.5). Correspondence-relation agents issue requests for the construction of hypotheses about correspondences.

For example, suppose the comparison-relation agent `longer-than` detects that `line-2` is longer than `line-1` in the environment and creates a new instance-agent representing the proposition `longer-than(line-2,line-1)`. The scale set includes an agent representing the proposition `higher-rating(grade-3, grade-1)`. Now, the correspondence-relation agent `longer=higher-rtg` constructs a hypothesis that these two propositions correspond. Once this hypothesis is established, AMBR's mechanism for top-down structure correspondence (Section 5.5.2) creates hypotheses that map `line-2` onto `grade-3` and `line-1` onto `grade-1`. Gradually, a constraint satisfaction network (Section 5.4.5) emerges that connects the comparison set with the scale set as illustrated in Figure 9.2.

```
cheese-15:
  :type (:instance :object)
  :inst-of cheese
  :c-coref (price-2 quality-12)

price-2:
  :type :instance
  :inst-of price
  :c-coref cheese-15
  :amount 18

quality-12
  :type :instance
  :inst-of quality
  :c-coref cheese-15
  :amount 150
```

Fig. 9.3 Illustration of the `:amount` slot introduced in JUDGEMAP to represent magnitudes. Note that the magnitude along each particular dimension is represented by a separate agent. Thus, a small coalition of agents is needed to represent a stimulus such as a piece of cheese with its price and quality. Each micro-frame has additional slots (not shown in the figure). All connectionist aspects are omitted. Compare with Figures 3.6 and 4.1.

The `:amount` slot is filled with a real number representing the magnitude of a given instance along a given dimension. Figure 9.3 illustrates magnitude representations along two dimensions: price and quality.

Many models of judgment and category rating use real numbers to represent stimulus magnitudes. Care must be taken, however, not to ascribe too many capacities for "mental arithmetic" to the cognitive system. JUDGEMAP does not use the magnitude numbers in any complex calculations. The only purpose of the `:amount` slot is to support comparisons between entities. One of the major innovations of the model is the attempt to represent metric information with a coalition of discrete agents. This coalition represents a hierarchy of ordering relations (see Fig. 9.4). For example, if `line-10` is 10 units long, `line-12` is 12, and `line-20` is 20, then the model can detect that `line-20` is longer than `line-12` and `line-12` is longer than `line-10`. The model also constructs second-order (meta) comparisons between first-order comparisons. An example of such second-order comparison is that the difference between the lengths of `line-20` and `line-12` is larger than the difference between the lengths of `line-12` and `line-10` (Fig. 9.4).

The model does not rely on the exact metric magnitudes of the stimuli. Instead, it maintains a representation of the hierarchical structure of ordering relations and meta-relations. This structure specifies the approximate position of a given stimulus on a continuous scale relative to the other stimuli. The precision of this approximation should increase with the introduction meta-ordering relations of third and higher orders. Second-order relations were sufficient to achieve the precision needed for the judgment tasks in the present simulations.

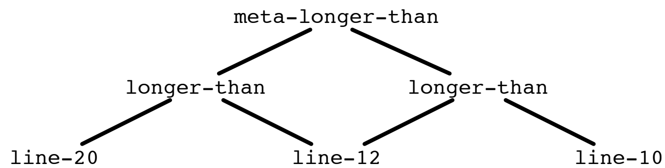


Fig. 9.4 JUDGEMAP constructs first-order comparison relations between instances and, recursively, second-order (meta) comparison relations between first-order comparisons.

Thus, JUDGEMAP performs ordinal-scale and interval-scale judgments using the same set of mechanisms. People make such variegated judgments quite often. For example, not only can people recognize that Mount Everest is higher than Mont Blanc and Mont Blanc is higher than Mount Elbert, but they can also recognize that the difference between Everest and Mont Blanc is larger than that between Mont Blanc and Mount Elbert. This would require a second-order comparison between comparisons. If we also need to be able to say that one difference is much larger than another one, then we would also need third-order comparisons, etc.

The scales for judgment are represented in JUDGEMAP by coalitions of agents (cf. Section 3.1.4). Figure 9.5 illustrates one such coalition. A separate agent equipped with an `:amount` slot stands for each number. Agents representing neighboring numbers are interconnected with associative links stored in `:a-link` slots. The scale values have bottom-up connections to a common *head* agent labeled `7-point-scale` in Figure 9.5. The head may have top-down connections to only a few salient values but not to all values. The head of a 100-point-scale, for example, may link to the agents standing for 10, 25, 50, 75, and 100. This scheme is analogous to the decentralized representation of situations in AMBR (Section 4.5.3).

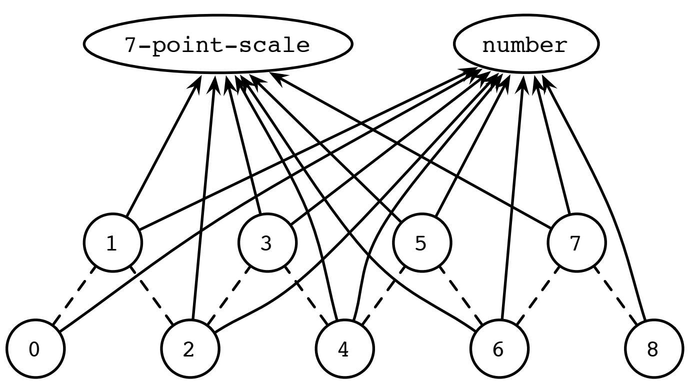


Fig. 9.5 Example of JUDGEMAP's scheme for representing a scale by a coalition of agents. The dashed lines depict bidirectional associative links. The comparison relations are not shown.

It is sometimes necessary for one concept-agent (e.g., `longer-than`) to know all instances of another concept (e.g., `line`) that are currently active. This is accomplished by the exchange of symbolic messages via links of type `:argument`. A novel symbolic procedure was defined for this purpose. When a concept-agent enters WM, it sends *argument-related requests* via its `:argument` links. These requests mean simply, "Send me whatever markers you receive." (See Section 5.3 for a description of the marker passing mechanism.) When a concept-agent receives such request, it stores it in its local memory, and each time it receives a marker, it sends an *argument-related answer* back to the requesting agent. These answers contain copies of the markers, but they do not spread further. They are only used by the symbolic processor of the agent that issued the argument-related request.

9.1.2 Overview of a Typical JUDGEMAP *Run*

This section illustrates how the various mechanisms in JUDGEMAP work together to carry out a dynamic emergent computation. The model is shown a sequence of lines and its task is to rate the length of each line on a seven-point scale.

To designate what task is to be performed, the human operator begins the session by attaching the correspondence-relation agent `longer=higher-rtg` to the GOAL node and the concept-agent `7-point-scale` to the INPUT node. The end points of the scale are designated as prototypical ratings by creating top-down links from the scale head to the end points. These links are temporary and their weights decrease with time. Thus, the initial choice of prototypical ratings matters only for the first several judgments. It does not influence the statistical results derived from long stimulus sequencies.

The presentation of the first stimulus is simulated by attaching the instance-agent `line-200` to both GOAL and INPUT nodes.

The spreading-activation mechanism (Sect. 5.2) begins to spread the activation that emanates from these two special nodes in the architecture. This brings relevant concepts and instances into WM. The transcript in Figure 9.6 lists the moments at which various agents pass the WM threshold. Following the standard AMBR marker passing protocol, each instance-agent emits a marker when it enters the WM. The first such marker is emitted by `line-200` at time 0.22, followed by markers by various scale elements (labeled as "grades" in the transcript). These markers spread to the respective concept-agents and to the superclasses of these concepts (cf. Fig. 5.6). Copies of some of these markers will eventually be sent as argument-related answers to concept-agents outside the `:subc` hierarchy of the originating instances. Whenever a concept-agent receives a marker, it checks whether the originating instance is its direct instance. If so, a temporary `:t-instance` link is created connecting the concept to the instance.[2]

---

[2] This is a natural entry point for a possible learning mechanism. In future versions of the model, we plan to design mechanisms for promoting some of these `:t-links` to permanent status. Prototypes may be formed in this way.

```
T=0.00, adding 7-point-scale to WM
T=0.00, adding longer=higher-rtg to WM
T=0.00, adding line-200 to WM
T=0.10, <ARG-REL-REQ longer=higher-rtg> received in 7-point-scale
T=0.20, adding grade-1 to WM
T=0.21, adding grade-7 to WM
T=0.21, adding length-200 to WM
T=0.22, adding grade-0 to WM
T=0.22, adding grade-2 to WM
T=0.22, adding grade-6 to WM
T=0.22, adding grade-8 to WM
T=0.22, adding relation to WM
T=0.22, adding object to WM
T=0.22, <MRK line-200> received in line
T=0.23, <ARG-REL-REQ longer=higher-rtg> received in longer-than
T=0.24, adding number to WM
T=0.24, adding property to WM
T=0.24, <MRK grade-7> received in 7-point-scale
T=0.24, <MRK grade-1> received in 7-point-scale
T=0.24, adding grade-3 to WM
T=0.24, adding grade-5 to WM
T=0.24, <ARG-REL-REQ longer-than> received in line
T=0.25, adding grade-4 to WM
```

Fig. 9.6 Transcript showing some key events at the beginning of a sample JUDGEMAP run. `T=xxx` denotes the time stamp, `<MRK xxx>` denotes a marker emitted by an instance-agent with the given name, and `<ARG-REL-REQ xxx>` denotes an argument-related request emitted by a concept-agent with the given name. Compare with Figures 5.4 and 5.10.

The processing of the very first stimulus presented to the model needs a special comment. This stimulus must be judged "in a vacuum"—without anything to compare it with. In this special case, no other lines and lengths can be retrieved from LTM into WM, and no comparisons would be formed. Probably, this never happens to people because they have extensive knowledge bases and are always able to retrieve or construct something similar to the target. One possibility to deal with this problem in JUDGEMAP is to use some predefined prototypes of the concept `line`, such as the line spanning the width of the screen. As a side effect of this solution, however, these prototypes would bias subsequent judgments in an artificial and unwanted manner. To avoid this distortion, a different solution was chosen in the model to handle the initial case in which no rating emerges that maps onto the target stimulus for a long time. In this (and only this) case, the system creates one hypothesis without any justifications. This is an exception to the AMBR principle that each hypothesis-agent must have at least one justification (Section 5.4.2). When this condition occurs, JUDGEMAP just takes the currently most active rating and maps it onto the target. In our sample run, this happens to be `grade-4`. At time 249.9 the first stimulus is judged with this rating without any justifications.

9.1.2.1 Judgment of Subsequent Stimuli

Once the first stimulus has been judged, it is removed from the GOAL list but not from the working memory. It still receives activation from its concept via the `:t-instance` link created earlier. However, the removal from the GOAL list causes the erasure of the `:t-driver` tag from the agent's `type` slot. The system is now ready to be presented with the second stimulus. In our example it happens to be 500 units long. The presentation procedure is the same: a temporary instance-agent `line-500` is attached to the GOAL and INPUT nodes, and a `:t-driver` tag is added to its `type` slot. The new agent emits a marker and activates its coalition partners such as `length-500`. The transcript in Figure 9.7 lists the key events of this process.

At time 251.4 the comparison relation `higher-length` receives from the concept-agent `length` an argument-related answer about the marker emitted by `length-500`. Now the comparison-relation concept is ready to apply its specialized procedural knowledge. At time 252.9 a new instance-agent is created to represent the comparison that `length-500` has a higher `:amount` than `length-200` (cf. Fig. 9.3; the latter agent represents the length of the first line stimulus). In other words, the comparison-relation concept `higher-length` has detected a conforming instance in the environment and has represented this fact within the system.

The name of the new agent is `length-500>length-200`. Like every other instance-agent, it emits a marker upon entering WM. This marker, wrapped in an argument-related answer, is received by the comparison-relation `longer-than` at time 257.6. As the arguments of `longer-than` are lines rather than "disembodied" lengths, `longer-than` identifies the relevant lines by following the `:c-coref` links in the agents representing the lengths (cf. Fig. 9.3). These are instance-agents `line-500` and `line-200`. Thus, `longer-than` creates the instance-agent `line-500>line-200` to represent the comparison between the lines. It may seem that having two separate propositions about the same information is redundant. The advantages of such separation, however, are apparent when complex stimuli are judged along multiple dimensions. In the example illustrated in Figure 9.3, it is important to separate the relation `better-cheese` from the relations about the cheese's properties, `better-price` and `better-quality`.

At time 263.0 the correspondence relation `longer=higher-rating` is notified about the newly created instance `line-500>line-200`. The correspondence relation creates a justification agent and interconnects it with the latter instance and with the hypothesis-agent `line-200<==>grade-4` that was created during the processing of the first stimulus. The correspondence relation also sends a message to the new justification agent with instructions what hypotheses to create. Three possible hypotheses are consistent with the fact that the new stimulus (`line-500`) is longer than the old, and that the old stimulus (`line-200`) was rated with `grade-4`. Thus, `line-500` may correspond to `grade-5`, `grade-6` or `grade-7`. The justification agent creates these hypotheses one at a time, in the order that reflects the activation levels of each hypothesis' arguments.

```
T=249.9, adding line-500 to WM
T=250.0, <MRK line-500> received in line
T=250.1, adding length-500 to WM
T=250.4, <MRK length-500> received in length
T=250.7, <ARG-REL-ANS line-500> received in longer-than
T=250.9, <MRK line-500> received in object
T=251.4, <ARG-REL-ANS length-500> received in higher-length
T=251.6, <MRK length-500> received in property
T=252.9, adding length-500>length-200 to WM
T=256.3, <MRK length-500>length-200> received in higher-length
T=257.6, <ARG-REL-ANS length-500>length-200> received in
     longer-than
T=257.9, <ARG-REL-ANS length-500>length-200> received in
     meta-higher-length
T=258.4, <MRK length-500>length-200> received in relation
T=258.7, adding line-500>line-200 to WM
T=261.8, <MRK line-500>line-200> received in longer-than
T=263.0, <ARG-REL-ANS line-500>line-200> received in
     longer=higher-rtg
T=263.1, adding JUSTIFICATION{line-500>line-200} to WM
T=263.2, <ARG-REL-ANS line-500>line-200> received in
     meta-longer-than
T=263.4, <MRK line-500>line-200> received in relation
T=281.8, adding line-500<==>grade-5 to WM
T=286.2, adding line-500<==>grade-6 to WM
T=292.7, adding line-500<==>grade-7 to WM
T=292.9, line-500 was judged with rating grade-5
WM has 32 agents, act 33.641. Justifications: 1
```

Fig. 9.7 Continuation of the transcript from Figure 9.6, listing some key events after the presentation of the second stimulus (`line-500`). `<ARG-REL-ANS xxx>` denotes an argument-related answer carrying a marker emitted by the instance-agent with the given name.

At time 281.8 the first such hypothesis is born: `line-500<==>grade-5`. This event triggers AMBR's rating mechanism (Section 5.6.1). It assigns an initial rating to the new hypothesis and begins to monitor its performance. At time 286.2 a competing hypothesis arrives: `line-500<==>grade-6`. However, it is not strong enough to overtake the first one. The third hypothesis is created at time 292.7, but by then it is too late to make any difference. At time 292.9 the promotion mechanism proclaims `line-500<==>grade-5` as the winner. The winning response is reported, all loser hypotheses fizzle out, and `line-500` is removed from the GOAL list.

Then the system is presented with the third stimulus: `line-1400`. Skipping most details, let us inspect what hypotheses compete with each other at a later moment and what their support is.

The first hypothesis proposes to judge the stimulus with `grade-7`. It receives support from three justifications: `line-1400` is longer than `line-200`, which was judged with 4; it is also longer than `line-500`, which was judged with 5. The third justification is the second-order comparison that `line-1400>line-200` is meta-longer than `line-1400>line-500`. The hypothesis also receives activation from its elements: `line-1400` and `grade-7`. In addition, the hypothesis is inhibited from its competitors according to the principles governing the constraint-satisfaction network (Sect. 5.4.5).

The second hypothesis proposes to judge the stimulus with `grade-6`. It receives activation only from two justifications—the two first-order justifications of the other hypothesis. The first hypothesis thus has an extra source of support from the second-order justification above. This gives it competitive advantage over the second hypothesis. However, the latter receives more activation from its argument `grade-6` than the first hypothesis receives from `grade-7`. This happens because the agent `grade-6`, which won on the previous trial, is very relevant. Such residual activations can cause sequential effects in JUDGEMAP in agreement with the human data. In this particular case, however, the first hypothesis won the competition, although the resolution took a long time because of the interplay of the competing factors. One more hypothesis—about `grade-5`—also emerged, but was too weak to compete with the others. In the end, `line-1400` was judged with `grade-7`.

One might think that, as more and more stimuli are judged, the system will suffer from a combinatorial explosion due to an ever-growing number of possible comparisons, justifications, and correspondences. This does not occur in JUDGEMAP (or indeed in any DUAL model) because of the pruning effect of the working-memory threshold, the limited speed of the symbolic processors (cf. Sect. 3.1.3.3), and the temporal cutoff imposed by the rating mechanism. This robustness is illustrated by the processing of the fourth stimulus on our sample run, which happened to be `line-300` . In fact, at the moment when the hypothesis `line-300<==>grade-4` was promoted, it was supported only by 2 justifications: `line-1400>line-300` and `line-500>line-300`. Far more reasons to judge `line-300` with `grade-4` could in principle be generated by the system given enough time, but their creation was cut short by the rating and promotin mechanisms. As a general rule, these mechanisms do not wait until all moderately relevant hypotheses emerge. Instead, they commit to a response as soon as the current state is satisfactory enough. The competition during the judgment of `line-300` was brief because the competing hypotheses were too weak. During the previous judgments, the system had been "focusing on" the larger ratings, whereas `line-300` was relatively short. Because each hypothesis receives activation not only from its justifications but also from its elements, the small ratings lost the competition. The conflicting information that `line-200` was judged with `grade-4`, had lost much of its activation due to decay.

Later in the judgment process the old instances fizzle out one after another. Note that the order of their fizzling out is not strictly determined by the trial order of their creation. Instead, fizzling out is inversely related to their relevance as operationalized by their activation levels. If a certain line justifies many winner correspondences, it receives feedback activation from them and thereby survives longer in WM. If it is inconsistent with the recent winners, it loses support and dies.

9.1.3 Some Simulations and Comparisons with Behavioral Data

A large set of simulations was performed with the JUDGEMAP model. The results demonstrated that the correlation between the objective stimulus magnitudes and the model's ratings was very close to the respective correlation of human judgments (Petkov, 2005).

The sequential assimilation effect (Petrov & Anderson, 2005) was simulated. It arises in JUDGEMAP because of the residual activation of the previous rating. The overall pattern of activation across the scale values changes dynamically. There are always some ratings that are more active and some that are less active. Furthermore, the hypotheses for correspondences receive positive activation from two sources—their justifications and their elements. The role of the justifications in the competition between hypotheses is intuitively clear. However, the role of the relevance of the elements of the hypotheses does not seem essential for the judgment task. It is a consequence of the basic mechanisms that underlie analogy-making, particularly the mapping mechanism.

JUDGEMAP also accounts for the contrastive context effects found in the human data. These effect arises from two main sources in the model. First, the comparison between the stimuli highlights their differences and hence creates a pressure to differentiate their ratings too. Second, the soft version of the pressure for one-to-one mappings causes a tendency for the scale labels to be used an almost equal number of times (in accordance with the frequency principle). This pressure is inherited from the AMBR model. The importance of one-to-one mapping is obvious in analogy-making, but not in a judgment task. Thus, the assumption that the same mechanism produces the mapping in analogy and judgment explains the emergence of the frequency principle in judgment.

In contrast to many other models of judgment, JUDGEMAP can handle more complex judgment tasks. For example, it has been applied to the task of judging cheeses on the basis of two separate properties: price and quality (Petkov, 2005).

The binary task of judging between two alternatives was modeled as a judgment on a two-point scale There are only two ratings—"accept" and "reject." All alternatives are judged on this scale. The first winner-hypothesis for the rating "accept" is interpreted as the model's choice. Various well-known empirical phenomena of choice were modeled successfully (Petkov & Kokinov, 2006; Petkov, 2006).

Finally, a novel prediction emerged from the model. Because of its context sensitivity, the comparison set is formed on the basis of the recently judged stimuli but also on the basis of the similarity of these stimuli. Thus, stimulus properties irrelevant to the stated judgment task may nevertheless affect the responses. The precise form of this prediction was formulated, simulated with JUDGEMAP, and verified empirically with various tasks and stimuli (Kokinov et al., 2004; Petkov et al., 2005; Hristova et al., 2005).

9.2 The Anticipatory (Transfer) Mechanism

The transfer in AMBR emerges from the work of the *anticipatory mechanism* (Kiryazov et al., 2007; Petkov, Kiryazov, Grinberg, & Kokinov, 2007). This mechanism was used extensively in the RecMap model of top-down influences on high-level perception (Petkov & Shahbazyan, 2007; Shahbazyan & Petkov, 2007). RecMap provided the first testbed for simulation experiments using the new anticipatory mechanism. In RecMap, the anticipatory mechanism participated in an emergent process that transferred the solution of a known problem to a target problem during analogy-making. Subsequent research with robots provided additional simulations and tests of the anticipation mechanism and the AMBR model more generally. AMBR was integrated with some programs for object recognition and was implemented in robots (Kiryazov et al., 2007; Kokinov et al., 2008; Petkov et al., 2006). The full cycle from perception, knowledge representation, retrieval from memory, analogical mapping, formation of anticipations, verification of anticipations, transfer of the solution of the problem, and action executing was tested with the robots as outlined in Section 9.3 below.

9.2.1 Description of the Anticipatory Mechanism

In agreement with the main principles of DUAL (cf. Fig. 3.4), the anticipatory mechanism runs in parallel with all other mechanisms and potentially interacts with them. All mechanisms are carried out locally by the individual agents. The overall behavior of the system is a large-scale emergent product of these local interactions.

The basic idea of the anticipatory mechanism was illustrated in Figure 9.1 above. The creation of an *anticipation-agent* is triggered when all arguments of a given relation from a base episode are mapped to elements from the target, but the relation head in the base has no correspondence in the target. Under these circumstances, the respective relation is transferred from the base to the target. However, the new relation is only considered as a tentative *anticipation*.

In more detail, the anticipation mechanism works as follows: The process is initiated by instance-agents representing relations from a retrieved base. These agents are identified by the presence of the tags `:relation` and `:instance` in their `type` slots. Agents from the target—identified by a `:t-driver` tag—are not eligible. Each eligible agent sends a special symbolic message to all its arguments—that is, to all instance-agents that are fillers of `:c-coref` facets of their S-slots (cf Section 3.1.3.1). In the example in Figure 9.1, `Relation-R1-B` will send such messages to its two arguments `arg1-B` and `arg2-B`. These messages are sent after the agent representing the relation enters the WM and after emitting its usual marker (cf. Section 5.3). The agents representing the arguments receive these messages and store them in their symbolic buffers. Later during the run, as the mapping process unfolds, these arguments will probably be involved in one or more mature hypotheses (cf. Section 5.4.3). Whenever a mature hypothesis is registered with these agents' secretaries, they send back to the agent representing the relation an answer carrying the name of their respective correspondence. For example, `arg1-B` will notify `Relation-R1-B` of its hypothesized correspondence to `element1-T` in the target. The relation head checks whether all its arguments have sent such messages and, if so, issues a node construction request that describes the new anticipation-agent that is to be made. In due time, this results in the creation of a new temporary agent with the tags `:anticipation` and `:relation` in its `type` slot. In our example, the new agent is named `ANTICIPATION-Relation-R1-T` (Fig. 9.1). The `:inst-of` slot of the new agent is inherited from the respective slot of its originator. However, the arguments of the new relation are the target elements that correspond to the respective arguments of the original relation in the base.

Of course, it may happen that one and the same argument is involved in several hypotheses simultaneously. In such cases, several different anticipation-agents will be formed. Actually, a lot of anticipation-agents emerge during a typical simulation run and influence the subsequent behavior of the system.

Once created, anticipation-agents behave as all other instance-agents. They emit markers and may become involved in hypotheses. They participate in the constraint-satisfaction network. Given that they are temporary agents, they die if their activation drops below the threshold. In addition, they may be ordered to fizzle by a special mechanism (see below). The role of the anticipation-agents is to represent relations that are anticipated by analogy to hold in the environment. But they are tentative anticipations and the system is always prepared to reject them.

9.2.2 Modeling High-Level Vision: The RecMap Model

RecMap is an object-recognition model based on analogical mapping (Petkov & Shahbazyan, 2007; Shahbazyan & Petkov, 2007). It relies on four theoretical assumptions (Shahbazyan & Petkov, 2007):

- Objects, scenes, and events are represented by structural descriptions of elements and their configural relations (Biederman, 1987; Hummel & Biederman, 1992).
- Vision is an inferential process in which limited preattentive information is mapped to existing knowledge to generate hypotheses about what is present in the environment
- The role of attention is to bind elements and their relations into integrated representations, as well as to verify whether they are present.
- Attention is biased to the aspects of the visual world that are most consistently supported.

RecMap is an extension of AMBR and introduces three additional mechanisms to the 1998 version. These extensions are the anticipatory mechanism described above, an attentional mechanism outlined here, and a mechanism for the creation and maintenance of *hypotheses for recognition*. The latter mechanism is beyond our present scope.

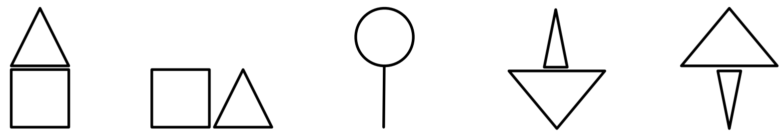


Fig. 9.8 The domain of the RecMap model consists of figures created by simple shapes.

RecMap recognizes objects and scenes in a simple domain consisting of lines, ovals, rectangles, and other perceptual primitives that are organized in different composite figures (Figure 9.8). Some figures are ambiguous—e.g., the fourth stimulus above can be interpreted as either a flowerpot or a candlestick; Some figures differ by a single relation—e.g., the house and truck. Initially, only the primitives—lines and ovals—are attached to the INPUT, assuming them as preattentive information. No relations between the primitives are represented. The model's task is to retrieve some representations from long-term memory, to create initial mappings, to form anticipations about various possible relations between items, to verify some of them, and thus, gradually, to build a structural description of the scene.

We propose that the limited preattentive information is mapped to structural representations in memory and that hypotheses for correspondence and anticipations are formed and as a result of this mapping. The attentional system compares the top-down anticipations with bottom-up perceptual evidence, and the recognition process emerges from these comparisons.

The function of the attentional mechanism is twofold: First, attention binds together the hypotheses for recognition to the respective relation and their arguments. Because the system has no central executive, various hypotheses for recognition for each element emerge locally. The anticipatory mechanism creates anticipation-agents (or "anticipations") positing tentative relations between the elements. When such an anticipation is attended, the elements and the anticipated relation are bound into a single representation. The second function of attention is to verify these anticipations. In the current model version, the verification is accomplished by checking the winning anticipations against a predefined list of relations and their arguments.

The attentional mechanism is the only centralized mechanism in the model. This was done deliberately for higher psychological validity. Because focal attention is needed for the binding process and for the verification of relations, the anticipations are verified serially in order of their activation level, which is an estimate of their relevance (cf Section 3.2.3.1).

The attentional mechanism is nothing but a list of all anticipation-agents, ordered by their activation. Periodically (on every 20 AMBR time cycles) the most active anticipation is verified. There are three possible outcomes: First, the anticipated relation may hold in the environment (modeled just by a predefined list). In this case the respective anticipation-agent is promoted to a regular instance-agent. Second, a different spatial or color relation between the same agents may be present. In this case the respective anticipation-agent fizzles out. Third, the anticipation-agent may be too complex to be recognized perceptually, given that the anticipatory mechanism works on all relations, including causal relations, etc. In this case nothing happens and the anticipation-agent remains in wait for a future resolution of its status.

Once a given anticipation-agent is promoted to a regular instance, it propagates its promotion by a variant of the structure correspondence mechanism (Section 5.5). It sends a message to the higher-order relations in which it participates and they in turn are promoted to regular instances, whereas their competitors fizzle out. Eventually, even the most abstract anticipatory relations either become instances or fizzle out.

Thus, the solution to the object-recognition problem emerges gradually via a process of analogical transfer from relational descriptions stored in long-term memory. This emergent process involves the creation of a large number of anticipation-agents, their interconnection in a constraint satisfaction network, promotion of some of them and fizzling out of others by the attention mechanism, and propagation of the winners. The set of winning anticipation-agents can be conceptualized as knowledge transferred by analogy.

RecMap accounts for various empirical phenomena, including holistic recognition, view-dependency of the recognition process, and recognition by a key element (Petkov & Shahbazyan, 2007; Shahbazyan & Petkov, 2007).

9.3 Simulations with Robots

Last but not least, the AMBR model with the anticipatory mechanism was deployed in AIBO robots and tested in a real environment (Petkov et al., 2006, 2007; Kiryazov et al., 2007; Kokinov et al., 2008).

We used the Sony AIBO robot (ERS-7) platform with implemented software for recognizing simple shapes (cylinders, cubes, etc.), their colors, and the spatial relation among them. The goal of the robot always was to search for its bone-toy, which was hidden behind one of the objects.

The robot must solve several tasks in order to find the toy. First, it creates a preliminary representation of the scene, consisting only of the shapes of the objects. Second, the robot retrieves from memory structural descriptions of various similar situations. Third, according to the RecMap model, the robot generates anticipations about possible spatial relations among the objects. Note, however, that all processes in AMBR run in parallel and thus much more complex anticipation emerge alongside the simple anticipations about spatial relations among objects. For example, the robot may anticipate by analogy with some previous situation that the toy can be found by moving to the cylinder. Note also that although there is a huge number of possible relations between any two items, the model generates anticipations only about relations that had been true in past situations that seem analogical to the current situation. Fourth, the robot sequentially verifies some of the anticipations and reject other ones via the attention mechanism. Fifth, it establishes the most appropriate analogy and removes the inconsistent anticipations. Sixth, it transfers knowledge about what movements would cause the bone to be found. Finally, the robot actually performs the planned movements.

All these steps were successfully performed by the robot in the respective domain (Petkov et al., 2006, 2007; Kiryazov et al., 2007; Kokinov et al., 2008).

Research on DUAL, AMBR, and the various models built on their basis continues at the New Bulgarian University and elsewhere.

Appendix A
Full Representation of a Situation

This appendix presents the unabridged representation of one of the twelve episodes in the current long-term memory. It is taken directly from the Lisp sources that load the knowledge base of AMBR.

The file consists of defagent macros. Each macro defines an agent and fills its slots. The overall syntax is:

```
(defagent agent-name agent-type
    [documentation-string]
    {G-slot-definition}*
    {S-slot-definition}*
 )
```

Most slot fillers are references to other agents. Each reference is also a link and has a label and a weight (see Section 3.1.3.1). When no explicit weight is given, it defaults to 1.0.

```
;;; -*- Mode: Lisp; Syntax: Common-Lisp; Package: AMBR -*-

;;; FILE:       AMBR/kb/episodic/b_WTP.lsp
;;; VERSION:    3.0.0   ; see AMBR/KB/VERSION.LSP
;;; PURPOSE:    Base situation WTP -- 'Water in a Teapot on a hot Plate.'
;;; DEPENDS-ON: AMBR, AMBR/kb/semantic/*.lsp
;;; PROGRAMMER: Alexander Alexandrov Petrov   (apetrov@cogs.nbu.acad.bg)
;;; VARIANTS:   none
;;; CREATED:    30-05-98 [3.0.0]  Elaboration of SIT-WTP from old LTM.LSP.
;;; UPDATED:    18-06-98  Removed IS-GOAL and RESULT propositns. Wght adjstmt
;;;                       CAUSE consequents are propositions now, not states.
;;;                       T-OF-WTP-W1 and T-OF-WTP-W2 coalesced together.
;;; UPDATED:    ...


         ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
         ;;;;;;;;         SITUATION  W T P           ;;;;;;;;
         ;;;;;;;;   Water in a Teapot on a Plate     ;;;;;;;;
         ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(in-package "AMBR")
```

```
;;;;;;;;;   Base situation WTP   ;;;;;;;;;;;
;;;;
;;;; There is some water in a teapot on a hot-plate.
;;;; The plate is hot.  The teapot is made of metal
;;;; and its color is black.
;;;;
;;;; The goal is to heat the water.
;;;;
;;;; The result is that the teapot becomes hot because
;;;; it is on the hot plate.  In turn, this causes the
;;;; water to become hot because it is in the teapot.
;;;;
;;;;;;;;;
;;;; Related situations:
;;;;  + GP  -- Glass on a hot Plate breaks.
;;;;  + IHC -- Imm.Heater in a Cup heats water.
;;;;  + ...
;;;;  * HM1 -- How to Heat Milk in a Teapot?
;;;;  * ...


;;;;;;;  Situation-agent
;;

(defagent   sit-WTP    instance-agent
  "Water in a Teapot on a hot Plate."
  :type      (:instance :situation)
  :inst-of   (situation 0.1)
  :a-link    ((hplate-WTP 0.5)
              (high-T-WTP 0.5)
              (T-of-WTP-w 1.0) )
)

;;;;;;;  Participating objects
;;
;;  water-WTP  :  (inst-of water)
;;  tpot-WTP   :  (inst-of teapot)
;;  hplate-WTP :  (inst-of hot-plate)
;;

(defagent   water-WTP    instance-agent
  :type       (:instance  :object)
  :modality   (:init :goal)
  :situation  (sit-WTP 0.2)
  :inst-of    water
  :c-coref    (((in-WTP . :slot1) 0.3)
               ((T-of-WTP-w  . :slot1) 1.0)
               ((goalst-WTP  . :slot3) 0.2)
               ((interst-WTP . :slot3) 0.1) )
  :a-link     (initst-WTP-1 0.1)
)

(defagent   tpot-WTP    instance-agent
  :type       (:instance  :object)
  :modality   :init
  :situation  (sit-WTP 0.2)
  :inst-of    teapot
  :c-coref    (((in-WTP . :slot2) 0.25)
               ((on-WTP . :slot2) 0.25)
               ((T-of-WTP-t   . :slot1) 0.25)
               ((made-of-WTP  . :slot1) 0.10)
               ((color-of-WTP . :slot1) 0.10)
               ((initst-WTP-2 . :slot3) 0.10) )
  :a-link     (initst-WTP-1 0.1)
  :slot1
    :type      :relation
    :inst-of   (teapot . :slot2)
```

```
    :c-coref   (made-of-WTP 0.2)
    :a-link    (mmetal-WTP  0.2)
  :slot2
    :type      :relation
    :inst-of   (teapot . :slot2)
    :c-coref   (color-of-WTP 0.1)
    :a-link    (black-WTP    0.1)
)

(defagent   hplate-WTP    instance-agent
  :type       (:instance  :object)
  :modality   :init
  :situation  (sit-WTP 0.3)
  :inst-of    hot-plate
  :c-coref    (((T-of-WTP-p . :slot1) 0.75)
               ((on-WTP . :slot1)     0.25) )
  :a-link     ((high-T-WTP   0.5)
               (initst-WTP-1 0.1) )
)


;;;;;;;  Initial relations
;;
;;  in-WTP       : (in water-WTP tpot-WTP)
;;  on-WTP       : (on hplate-WTP tpot-WTP)
;;  T-of-WTP-p   : (temperature-of hplate-WTP high-T-WTP)
;;  made-of-WTP  : (made-of  tpot-WTP mmetal-WTP)
;;  color-of-WTP : (color-of tpot-WTP black-WTP)
;;

(defagent   in-WTP    instance-agent
  "(in water-WTP tpot-WTP)"
  :type       (:instance  :relation)
  :modality   :init
  :situation  (sit-WTP 0.2)
  :inst-of    in
  :c-coref    (((initst-WTP-1 . :slot4) 0.2)
               ((interst-WTP  . :slot2) 0.1) )
  :slot1
    :type       :aspect
    :inst-of    (in . :slot1)
    :c-coref    water-WTP
  :slot2
    :type       :aspect
    :inst-of    (in . :slot2)
    :c-coref    tpot-WTP
)

(defagent   on-WTP    instance-agent
  "(on hplate-WTP tpot-WTP)"
  :type       (:instance  :relation)
  :modality   :init
  :situation  (sit-WTP 0.2)
  :inst-of    on
  :c-coref    (((initst-WTP-1 . :slot3) 0.2)
               ((initst-WTP-2 . :slot2) 0.1) )
  :a-link     (T-of-WTP-t 0.1)
  :slot1
    :type       :aspect
    :inst-of    (on . :slot1)
    :c-coref    hplate-WTP
  :slot2
    :type       :aspect
    :inst-of    (on . :slot2)
    :c-coref    tpot-WTP
)
```

```
(defagent   T-of-WTP-p    instance-agent
  "(temperature-of hplate-WTP high-T-WTP)"
  :type       (:instance  :relation)
  :modality   :init
  :situation  (sit-WTP 0.2)
  :inst-of    temperature-of
  :c-coref    (((initst-WTP-1 . :slot1) 0.3)
               ((initst-WTP-2 . :slot1) 0.1) )
  :a-link     ((T-of-WTP-t  0.1)
               (T-of-WTP-w  0.1)
               (cause-WTP-i 0.1) )
  :slot1
    :type       :aspect
    :inst-of    (temperature-of . :slot1)
    :c-coref    hplate-WTP
  :slot2
    :type       :aspect
    :inst-of    (temperature-of . :slot2)
    :c-coref    high-T-WTP
)

(defagent  high-T-WTP   instance-agent
  :type       (:instance  :object)
  :modality   (:init :goal :result)
  :situation  (sit-WTP 0.3)
  :inst-of    high-temp
  :c-coref    (((T-of-WTP-p   . :slot2) 0.5)
               ((T-of-WTP-t   . :slot2) 0.2)
               ((T-of-WTP-w   . :slot2) 0.5)
               ((initst-WTP-1 . :slot2) 0.1)
               ((goalst-WTP   . :slot2) 0.1) )
  :a-link     (hplate-WTP 0.3)
)

(defagent   made-of-WTP    instance-agent
  "(made-of tpot-WTP mmetal-WTP)"
  :type       (:instance  :relation)
  :modality   :init
  :situation  (sit-WTP 0.2)
  :inst-of    made-of
  :c-coref    (tpot-WTP . :slot1)
  :a-link     (T-of-WTP-t 0.3)
  :slot1
    :type       :aspect
    :inst-of    (made-of . :slot1)
    :c-coref    tpot-WTP
  :slot2
    :type       :aspect
    :inst-of    (made-of . :slot2)
    :c-coref    mmetal-WTP
)

(defagent  mmetal-WTP  instance-agent
  :type       (:instance  :object)
  :modality   :init
  :situation  (sit-WTP 0.2)
  :inst-of    material-metal
  :c-coref    (made-of-WTP  . :slot2)
  :a-link     (tpot-WTP 1.0)
)

(defagent   color-of-WTP    instance-agent
  "(color-of tpot-WTP black-WTP)"
  :type       (:instance  :relation)
  :modality   :init
  :situation  (sit-WTP 0.2)
  :inst-of    color-of
```

```
  :c-coref    (tpot-WTP . :slot2)
  :slot1
    :type     :aspect
    :inst-of  (color-of . :slot1)
    :c-coref  tpot-WTP
  :slot2
    :type     :aspect
    :inst-of  (color-of . :slot2)
    :c-coref  black-WTP
)

(defagent   black-WTP  instance-agent
  :type       (:instance  :object)
  :modality   :init
  :situation  (sit-WTP 0.2)
  :inst-of    black
  :c-coref    (color-of-WTP . :slot2)
  :a-link     (tpot-WTP 1.0)
)


;;;;;;;  Initial states
;;
;;  initst-WTP-1  -to-reach->  goalst-WTP
;;  initst-WTP-1  -follows->   endst-WTP
;;  initst-WTP-2  --cause-->   interst-WTP
;;
;;  initst-WTP-1  :  (init-state T-of-WTP-p high-T-WTP on-WTP in-WTP)
;;  initst-WTP-2  :  (init-state T-of-WTP-p on-WTP tpot-WTP)
;;

(defagent   initst-WTP-1  instance-agent
  "initst-WTP-1  -to-reach->  goalst-WTP"
  :type       (:instance :situation)
  :modality   :init
  :situation  (sit-WTP 0.2)
  :inst-of    init-state
  :c-coref    ((to-reach-WTP . :slot1)
               (follows-WTP  . :slot1) )
  :a-link     ((goalst-WTP   1.0)
               (initst-WTP-2 0.2)
               (water-WTP    0.2)
               (tpot-WTP     0.2)
               (hplate-WTP   0.2) )
  :slot1
    :type       :relation
    :inst-of    (init-state . :slot2)
    :c-coref    T-of-WTP-p
  :slot2
    :type       :aspect
    :inst-of    (init-state . :slot1)
    :c-coref    high-T-WTP
  :slot3
    :type       :relation
    :inst-of    (init-state . :slot2)
    :c-coref    on-WTP
  :slot4
    :type       :relation
    :inst-of    (init-state . :slot2)
    :c-coref    in-WTP
)

(defagent   initst-WTP-2  instance-agent
  "initst-WTP-2  --cause-->  interst-WTP"
  :type       (:instance :situation)
  :modality   :init
  :situation  (sit-WTP 0.2)
```

```
  :inst-of    init-state
  :c-coref    (cause-WTP-i . :slot1)
  :a-link     ((interst-WTP  1.0)
               (initst-WTP-1 0.3)
               (hplate-WTP   0.3) )
  :slot1
    :type       :relation
    :inst-of    (init-state . :slot2)
    :c-coref    T-of-WTP-p
  :slot2
    :type       :relation
    :inst-of    (init-state . :slot2)
    :c-coref    on-WTP
  :slot3
    :type       :aspect
    :inst-of    (init-state . :slot1)
    :c-coref    tpot-WTP
)



;;;;;;;  Goal state
;;
;;  goalst-WTP  <-to-reach-  initst-WTP-1
;;  goalst-WTP   :  (goal-state T-of-WTP-w high-T-WTP water-WTP)
;;
;;  T-of-WTP-w   :  (temperature-of water-WTP high-T-WTP)
;;  to-reach-WTP :  (to-reach initst-WTP-1 goalst-WTP)
;;

(defagent   T-of-WTP-w    instance-agent
  "(temperature-of water-WTP high-T-WTP)"
  :type       (:instance  :relation)
  :modality   (:GOAL   :intend-true
               :RESULT :true )
  :situation  (sit-WTP 0.3)
  :inst-of    temperature-of
  :c-coref    (((goalst-WTP  . :slot1) 0.2)
               ((endst-WTP   . :slot2) 0.2)
               ((cause-WTP-e . :slot2) 0.2) )
  :a-link     ((T-of-WTP-p 0.1)
               (T-of-WTP-t 0.1)
               (hplate-WTP 0.1) )
  :slot1
    :type       :aspect
    :inst-of    (temperature-of . :slot1)
    :c-coref    water-WTP
  :slot2
    :type       :aspect
    :inst-of    (temperature-of . :slot2)
    :c-coref    high-T-WTP
)

(defagent   goalst-WTP    instance-agent
  "goalst-WTP  <-to-reach-  initst-WTP-1"
  :type       (:instance :situation)
  :modality   :goal
  :situation  (sit-WTP 0.2)
  :inst-of    goal-state
  :c-coref    (to-reach-WTP . :slot2)
  :a-link     ((initst-WTP-1 1.0)
               (endst-WTP    0.5) )
  :slot1
    :type       :relation
    :inst-of    (goal-state . :slot2)
    :c-coref    T-of-WTP-w
```

```
  :slot2
    :type       :aspect
    :inst-of    (goal-state . :slot1)
    :c-coref    high-T-WTP
  :slot3
    :type       :aspect
    :inst-of    (goal-state . :slot1)
    :c-coref    water-WTP
)

(defagent   to-reach-WTP    instance-agent
  "(to-reach initst-WTP-1 goalst-WTP)"
  :type       (:instance  :relation)
  :modality   :goal
  :situation  (sit-WTP 0.2)
  :inst-of    to-reach
  :a-link     ((follows-WTP 0.5)
               (cause-WTP-i 0.1)
               (cause-WTP-e 0.2) )
  :slot1
    :type       :aspect
    :inst-of    (to-reach . :slot1)
    :c-coref    initst-WTP-1
  :slot2
    :type       :aspect
    :inst-of    (to-reach . :slot2)
    :c-coref    goalst-WTP
)


;;;;;;;  Intermediary state
;;
;;  interst-WTP  <-cause--  initst-WTP-2
;;  interst-WTP  --cause->  endst-WTP
;;  interst-WTP   :  (inter-state T-of-WTP-t in-WTP water-WTP)
;;
;;  T-of-WTP-t    :  (temperature-of tpot-WTP high-T-WTP)
;;  cause-WTP-i   :  (cause initst-WTP-2 T-of-WTP-t)
;;

(defagent   T-of-WTP-t    instance-agent
  "(temperature-of tpot-WTP high-T-WTP)"
  :type       (:instance  :relation)
  :modality   :result
  :situation  (sit-WTP 0.2)
  :inst-of    temperature-of
  :c-coref    (((cause-WTP-i . :slot2) 0.5)
               ((interst-WTP . :slot1) 0.2)
               ((endst-WTP   . :slot1) 0.2) )
  :a-link     ((T-of-WTP-p 0.3)
               (T-of-WTP-w 0.3)
               (hplate-WTP 0.1) )
  :slot1
    :type      :aspect
    :inst-of   (temperature-of . :slot1)
    :c-coref   tpot-WTP
  :slot2
    :type      :aspect
    :inst-of   (temperature-of . :slot2)
    :c-coref   high-T-WTP
)

(defagent   interst-WTP    instance-agent
  "interst-WTP  <-cause-  initst-WTP-2"
  :type       (:instance :situation)
  :modality   :result
  :situation  (sit-WTP 0.2)
```

```
  :inst-of    inter-state
  :c-coref    (cause-WTP-e . :slot1)
  :a-link     ((endst-WTP    0.6)
               (initst-WTP-2 0.4) )
  :slot1
    :type       :relation
    :inst-of    (inter-state . :slot2)
    :c-coref    T-of-WTP-t
  :slot2
    :type       :relation
    :inst-of    (inter-state . :slot2)
    :c-coref    in-WTP
  :slot3
    :type       :aspect
    :inst-of    (inter-state . :slot1)
    :c-coref    water-WTP
)

(defagent   cause-WTP-i   instance-agent
  "(cause initst-WTP-2 T-of-WTP-t)"
  :type       (:instance  :relation)
  :modality   :result
  :situation  (sit-WTP 0.2)
  :inst-of    cause
  :a-link     ((interst-WTP  1.0)
               (cause-WTP-e  0.5)
               (follows-WTP  0.3)
               (to-reach-WTP 0.2) )
  :slot1
    :type       :aspect
    :inst-of    (cause . :slot1)
    :c-coref    initst-WTP-2
  :slot2
    :type       :aspect
    :inst-of    (cause . :slot2)
    :c-coref    T-of-WTP-t
)


;;;;;;;  End state
;;
;;  endst-WTP  <-follows-  initst-WTP-1
;;  endst-WTP     :  (end-state T-of-WTP-t T-of-WTP-w)
;;
;;  follows-WTP   :  (follows initst-WTP-1 endst-WTP)
;;  cause-WTP-e   :  (cause interst-WTP T-of-WTP-w)
;;

(defagent   endst-WTP    instance-agent
  "endst-WTP  <-follows-  initst-WTP-1"
  :type       (:instance :situation)
  :modality   :result
  :situation  (sit-WTP 0.2)
  :inst-of    end-state
  :c-coref    (follows-WTP . :slot2)
  :a-link     ((interst-WTP 0.1)
               (goalst-WTP  0.2) )
  :slot1
    :type       :relation
    :inst-of    (end-state . :slot2)
    :c-coref    (T-of-WTP-t 0.5)
  :slot2
    :type       :relation
    :inst-of    (end-state . :slot2)
    :c-coref    T-of-WTP-w
)
```

```
(defagent   follows-WTP  instance-agent
  "(follows initst-WTP-1 endst-WTP)"
  :type       (:instance  :relation)
  :modality   :result
  :situation  (sit-WTP 0.2)
  :inst-of    follows
  :a-link     ((to-reach-WTP 0.5)
               (cause-WTP-i  0.2) )
  :slot1
    :type       :aspect
    :inst-of    (follows . :slot1)
    :c-coref    initst-WTP-1
  :slot2
    :type       :aspect
    :inst-of    (follows . :slot2)
    :c-coref    endst-WTP
)

(defagent   cause-WTP-e    instance-agent
  "(cause interst-WTP T-of-WTP-w)"
  :type       (:instance  :relation)
  :modality   :result
  :situation  (sit-WTP 0.2)
  :inst-of    cause
  :a-link     ((endst-WTP    0.5)
               (follows-WTP  0.3)
               (cause-WTP-i  0.2)
               (to-reach-WTP 0.1) )
  :slot1
    :type       :aspect
    :inst-of    (cause . :slot1)
    :c-coref    interst-WTP
  :slot2
    :type       :aspect
    :inst-of    (cause . :slot2)
    :c-coref    T-of-WTP-w
)


;;;;;;  ---- Sanity check ---- ;;;;;;
;;
(check-for-unresolved-references)




;;;;;;;;;;;;  ---  Appendix  ---  ;;;;;;;;;;
;;
;; The information below is not used by the model.
;; It is for interface purposes only.

(defcoalition  sit-WTP
  "Water in a Teapot on a hot Plate."
  :head      sit-WTP                      ; 23 agents
  :members   (sit-WTP
              water-WTP     tpot-WTP      hplate-WTP
              in-WTP        on-WTP
              T-of-WTP-p    high-T-WTP
              T-of-WTP-t    T-of-WTP-w
              made-of-WTP   mmetal-WTP
              color-of-WTP  black-WTP
              initst-WTP-1  initst-WTP-2  interst-WTP
              goalst-WTP    endst-WTP
              to-reach-WTP  follows-WTP
              cause-WTP-i   cause-WTP-e
             ))
```

```
(GENKB-template
  :herald  "Base sit.WTP -- Water in a Teapot on a hot Plate, ver.3.0.0."
  :templates '(
    (water          (:instance (water-WTP    2))
                    (:a-link   (T-of-WTP-w  0.1)) )
    (teapot         (:instance (tpot-WTP     3)) )
    (hot-plate      (:instance (hplate-WTP   5))
                    (:a-link   (T-of-WTP-p  0.2)) )
    (temperature-of (:instance (T-of-WTP-w   3) (T-of-WTP-p 3))
                    (:a-link   (high-T-WTP  0.1)) )
    (high-temp      (:instance (high-T-WTP   5))
                    (:a-link   (hplate-WTP  0.2)) )
    (in             (:instance (in-WTP       1)) )
    (on             (:instance (on-WTP       1)) )
    (made-of        (:instance (made-of-WTP  1)) )
    (material-metal (:instance (mmetal-WTP   1)) )
    (color-of       (:instance (color-of-WTP 1)) )
    (black          (:instance (black-WTP    1)) )
))


;;;;;;;;  Propositional representaion
;;
;;  sit-WTP       :  (inst-of sit-WTP situation)
;;
;;  black-WTP     :  (inst-of black-WTP black)
;;  cause-WTP-i   :  (cause initst-WTP-2 T-of-WTP-t)
;;  cause-WTP-e   :  (cause interst-WTP  T-of-WTP-w)
;;  color-of-WTP  :  (color-of tpot-WTP black-WTP)
;;  endst-WTP     :  (end-state T-of-WTP-t T-of-WTP-w)
;;  follows-WTP   :  (follows initst-WTP-1 endst-WTP)
;;  goalst-WTP    :  (goal-state T-of-WTP-w high-T-WTP water-WTP)
;;  high-T-WTP    :  (inst-of high-T-WTP high-temp)
;;  hplate-WTP    :  (inst-of hplate-WTP hot-plate)
;;  in-WTP        :  (in water-WTP tpot-WTP)
;;  initst-WTP-1  :  (init-state T-of-WTP-p high-T-WTP on-WTP in-WTP)
;;  initst-WTP-2  :  (init-state T-of-WTP-p on-WTP tpot-WTP)
;;  interst-WTP   :  (inter-state T-of-WTP-t in-WTP water-WTP)
;;  made-of-WTP   :  (made-of tpot-WTP mmetal-WTP)
;;  mmetal-WTP    :  (inst-of mmetal-WTP material-metal)
;;  on-WTP        :  (on tpot-WTP hplate-WTP)
;;  T-of-WTP-p    :  (temperature-of hplate-WTP high-T-WTP)
;;  T-of-WTP-t    :  (temperature-of tpot-WTP high-T-WTP)
;;  T-of-WTP-w    :  (temperature-of water-WTP high-T-WTP)
;;  to-reach-WTP  :  (to-reach initst-WTP-1 goalst-WTP)
;;  tpot-WTP      :  (inst-of tpot-WTP teapot)
;;  water-WTP     :  (inst-of water-WTP water)


;;;;;;  End of file  AMBR/KB/EPISODIC/B_WTP.LSP
```

Appendix B
Propositional Descriptions of All Situations

This appendix presents simplified propositional descriptions of all situations involved in the simulation experiments reported in the book. They appear in the order they are introduced in Chapter 6: 12 base episodes + 10 target problems.

Note that these are *simplified* representations only! The actual AMBR representations are much more complex. Generally, each line below corresponds to a whole agent with several slots. See Appendix A for an actual representation and compare it with the first group below.

```
;;;;;;  Base sit. WTP (Water in a Teapot on a Plate)

  sit-WTP       :  (inst-of sit-WTP situation)
  black-WTP     :  (inst-of black-WTP black)
  cause-WTP-i   :  (cause initst-WTP-2 T-of-WTP-t)
  cause-WTP-e   :  (cause interst-WTP  T-of-WTP-w)
  color-of-WTP  :  (color-of tpot-WTP black-WTP)
  endst-WTP     :  (end-state T-of-WTP-t T-of-WTP-w)
  follows-WTP   :  (follows initst-WTP-1 endst-WTP)
  goalst-WTP    :  (goal-state T-of-WTP-w high-T-WTP water-WTP)
  high-T-WTP    :  (inst-of high-T-WTP high-temp)
  hplate-WTP    :  (inst-of hplate-WTP hot-plate)
  in-WTP        :  (in water-WTP tpot-WTP)
  initst-WTP-1  :  (init-state T-of-WTP-p high-T-WTP on-WTP in-WTP)
  initst-WTP-2  :  (init-state T-of-WTP-p on-WTP tpot-WTP)
  interst-WTP   :  (inter-state T-of-WTP-t in-WTP water-WTP)
  made-of-WTP   :  (made-of tpot-WTP mmetal-WTP)
  mmetal-WTP    :  (inst-of mmetal-WTP material-metal)
  on-WTP        :  (on tpot-WTP hplate-WTP)
  T-of-WTP-p    :  (temperature-of hplate-WTP high-T-WTP)
  T-of-WTP-t    :  (temperature-of tpot-WTP high-T-WTP)
  T-of-WTP-w    :  (temperature-of water-WTP high-T-WTP)
  to-reach-WTP  :  (to-reach initst-WTP-1 goalst-WTP)
  tpot-WTP      :  (inst-of tpot-WTP teapot)
  water-WTP     :  (inst-of water-WTP water)


;;;;;;  Base sit. BF (Bowl on a Fire burns out)

  sit-BF       :  (inst-of sit-BF situation)
  bowl-BF      :  (inst-of bowl-BF bowl)
  cause-BF-b   :  (cause initst-BF-2 is-burnt-BF)
  cause-BF-d   :  (cause interst-BF  is-dissip-BF)
  endst-BF     :  (end-state is-burnt-BF is-dissip-BF)
```

```
  fire-BF      :  (inst-of fire-BF fire)
  follows-BF   :  (follows initst-BF-1 endst-BF)
  goalst-BF    :  (goal-state T-of-BF-w high-T-BF water-BF)
  high-T-BF    :  (inst-of high-T-BF high-temp)
  in-BF        :  (in water-BF bowl-BF)
  initst-BF-1  :  (init-state T-of-BF-f high-T-BF on-BF in-BF)
  initst-BF-2  :  (init-state T-of-BF-f made-of-BF mwood-BF on-BF)
  interst-BF   :  (inter-state is-burnt-BF in-BF bowl-BF)
  is-burnt-BF  :  (is-burnt-out bowl-BF)
  is-dissip-BF :  (is-dissipated water-BF)
  made-of-BF   :  (made-of bowl-BF mwood-BF)
  mwood-BF     :  (inst-of mwood-BF material-wood)
  on-BF        :  (on fire-BF bowl-BF)
  T-of-BF-f    :  (temperature-of fire-BF high-T-BF)
  T-of-BF-w    :  (temperature-of water-BF high-T-BF)
  to-reach-BF  :  (to-reach initst-BF-1 goalst-BF)
  water-BF     :  (inst-of water-BF water)


;;;;;;  Base sit. GP (Glass on a hot Plate breaks)

  sit-GP       :  (inst-of sit-GP situation)
  cause-GP-b   :  (cause initst-GP-2 is-broken-GP)
  cause-GP-d   :  (cause interst-GP  is-dissip-GP)
  endst-GP     :  (end-state is-broken-GP is-dissip-GP)
  follows-GP   :  (follows initst-GP-1 endst-GP)
  glass-GP     :  (inst-of glass-GP glass)
  goalst-GP    :  (goal-state T-of-GP-w high-T-GP water-GP)
  high-T-GP    :  (inst-of high-T-GP high-temp)
  hplate-GP    :  (inst-of hplate-GP hot-plate)
  in-GP        :  (in water-GP glass-GP)
  initst-GP-1  :  (init-state T-of-GP-p high-T-GP on-GP in-GP)
  initst-GP-2  :  (init-state T-of-GP-p made-of-GP mglass-GP on-GP)
  interst-GP   :  (inter-state is-broken-GP in-GP glass-GP)
  is-broken-GP :  (is-broken glass-GP)
  is-dissip-GP :  (is-dissipated water-GP)
  made-of-GP   :  (made-of glass-GP mglass-GP)
  mglass-GP    :  (inst-of mglass-GP material-glass)
  on-GP        :  (on hplate-GP glass-GP)
  T-of-GP-p    :  (temperature-of hplate-GP high-T-GP)
  T-of-GP-w    :  (temperature-of water-GP high-T-GP)
  to-reach-GP  :  (to-reach initst-GP-1 goalst-GP)
  water-GP     :  (inst-of water-GP water)


;;;;;;  Base sit. IHC (Immersion Heater in a Cup with water)

  sit-IHC       :  (inst-of sit-IHC situation)
  cause-IHC     :  (cause initst-IHC T-of-IHC-w)
  cup-IHC       :  (inst-of cup-IHC cup)
  endst-IHC     :  (end-state T-of-IHC-w)
  follows-IHC   :  (follows initst-IHC endst-IHC)
  goalst-IHC    :  (goal-state T-of-IHC-w high-T-IHC water-IHC)
  high-T-IHC    :  (inst-of high-T-IHC high-temp)
  imm-htr-IHC   :  (inst-of imm-htr-IHC immersion-heater)
  in-IHC-iw     :  (in imm-htr-IHC water-IHC)
  in-IHC-wc     :  (in water-IHC cup-IHC)
  initst-IHC    :  (init-state T-of-IHC-ih high-T-IHC in-IHC-iw imm-htr-IHC)
  made-of-IHC   :  (made-of cup-IHC mchina-IHC)
  mchina-IHC    :  (inst-of mchina-IHC material-china)
  on-IHC        :  (on saucer-IHC cup-IHC)
  saucer-IHC    :  (inst-of saucer-IHC saucer)
  T-of-IHC-ih   :  (temperature-of imm-htr-IHC high-T-IHC)
  T-of-IHC-w    :  (temperature-of water-IHC high-T-IHC)
  to-reach-IHC  :  (to-reach initst-IHC goalst-IHC)
  water-IHC     :  (inst-of water-IHC water)
```

```
;;;;;;  Base sit. FDO (Food on a Dish in an Oven)

  sit-FDO       :  (inst-of sit-FDO situation)
  cause-FDO-i   :  (cause initst-FDO-2 in-FDO-fo)
  cause-FDO-t   :  (cause interst-FDO  T-of-FDO-f)
  dish-FDO      :  (inst-of dish-FDO baking-dish)
  endst-FDO     :  (end-state T-of-FDO-f)
  follows-FDO   :  (follows initst-FDO-1 endst-FDO)
  food-FDO      :  (inst-of food-FDO food)
  goalst-FDO    :  (goal-state T-of-FDO-f high-T-FDO food-FDO)
  high-T-FDO    :  (inst-of high-T-FDO high-temp)
  initst-FDO-1  :  (init-state T-of-FDO-o high-T-FDO on-FDO in-FDO-do)
  initst-FDO-2  :  (init-state on-FDO in-FDO-do)
  interst-FDO   :  (inter-state in-FDO-fo T-of-FDO-o oven-FDO)
  in-FDO-fo     :  (in food-FDO oven-FDO)
  in-FDO-do     :  (in dish-FDO oven-FDO)
  on-FDO        :  (on dish-FDO food-FDO)
  oven-FDO      :  (inst-of oven-FDO oven)
  rectang-FDO   :  (inst-of rectang-FDO rectang-shape)
  shape-of-FDO  :  (shape-of dish-FDO rectang-FDO)
  T-of-FDO-o    :  (temperature-of oven-FDO high-T-FDO)
  T-of-FDO-f    :  (temperature-of food-FDO high-T-FDO)
  to-reach-FDO  :  (to-reach initst-FDO-1 goalst-FDO)


;;;;;;  Base sit. MTF (Milk in a Teapot in a Fridge)

  sit-MTF       :  (inst-of sit-MTF situation)
  cause-MTF-i   :  (cause initst-MTF-2 in-MTF-mf)
  cause-MTF-t   :  (cause interst-MTF  T-of-MTF-m)
  color-of-MTF  :  (color-of tpot-MTF green-MTF)
  endst-MTF     :  (end-state T-of-MTF-m)
  follows-MTF   :  (follows initst-MTF-1 endst-MTF)
  fridge-MTF    :  (inst-of fridge-MTF fridge)
  goalst-MTF    :  (goal-state T-of-MTF-m low-T-MTF milk-MTF)
  green-MTF     :  (inst-of green-MTF green)
  initst-MTF-1  :  (init-state T-of-MTF-f low-T-MTF in-MTF-mt in-MTF-tf)
  initst-MTF-2  :  (init-state in-MTF-mt in-MTF-tf)
  interst-MTF   :  (inter-state in-MTF-mf T-of-MTF-f fridge-MTF)
  in-MTF-mf     :  (in milk-MTF fridge-MTF)
  in-MTF-mt     :  (in milk-MTF tpot-MTF)
  in-MTF-tf     :  (in tpot-MTF fridge-MTF)
  low-T-MTF     :  (inst-of low-T-MTF low-temp)
  milk-MTF      :  (inst-of milk-MTF milk)
  T-of-MTF-f    :  (temperature-of fridge-MTF low-T-MTF)
  T-of-MTF-m    :  (temperature-of milk-MTF low-T-MTF)
  to-reach-MTF  :  (to-reach initst-MTF-1 goalst-MTF)
  tpot-MTF      :  (inst-of tpot-MTF teapot)


;;;;;;  Base sit. ICF (Ice Cube in a Fridge)

  sit-ICF       :  (inst-of sit-ICF situation)
  cause-ICF-i   :  (cause initst-ICF-2 in-ICF-if)
  cause-ICF-t   :  (cause interst-ICF  T-of-ICF-i)
  endst-ICF     :  (end-state T-of-ICF-i)
  follows-ICF   :  (follows initst-ICF-1 endst-ICF)
  fridge-ICF    :  (inst-of fridge-ICF fridge)
  glass-ICF     :  (inst-of glass-ICF glass)
  goalst-ICF    :  (goal-state T-of-ICF-i low-T-ICF ice-cube-ICF)
  ice-cube-ICF  :  (inst-of ice-cube-ICF ice-cube)
  initst-ICF-1  :  (init-state T-of-ICF-f low-T-ICF on-ICF-ig in-ICF-gf)
  initst-ICF-2  :  (init-state on-ICF-ig in-ICF-gf)
  interst-ICF   :  (inter-state in-ICF-if T-of-ICF-f fridge-ICF)
  in-ICF-if     :  (in ice-cube-ICF fridge-ICF)
  on-ICF-ig     :  (on ice-cube-ICF glass-ICF)
  in-ICF-gf     :  (in glass-ICF fridge-ICF)
```

```
  low-T-ICF     :  (inst-of low-T-ICF low-temp)
  made-of-ICF   :  (made-of glass-ICF mglass-ICF)
  mglass-ICF    :  (inst-of mglass-ICF material-glass)
  T-of-ICF-f    :  (temperature-of fridge-ICF low-T-ICF)
  T-of-ICF-i    :  (temperature-of ice-cube-ICF low-T-ICF)
  to-reach-ICF  :  (to-reach initst-ICF-1 goalst-ICF)


;;;;;;  Base sit. BPF (Butter on a Plate in a Fridge)

  sit-BPF       :  (inst-of sit-BPF situation)
  butter-BPF    :  (inst-of butter-BPF butter)
  cause-BPF-i   :  (cause initst-BPF-2 in-BPF-bf)
  cause-BPF-t   :  (cause interst-BPF  T-of-BPF-b)
  circular-BPF  :  (inst-of circular-BPF circular-shape)
  endst-BPF     :  (end-state T-of-BPF-b)
  follows-BPF   :  (follows initst-BPF-1 endst-BPF)
  fridge-BPF    :  (inst-of fridge-BPF fridge)
  goalst-BPF    :  (goal-state T-of-BPF-b low-T-BPF butter-BPF)
  initst-BPF-1  :  (init-state T-of-BPF-f low-T-BPF on-BPF in-BPF-pf)
  initst-BPF-2  :  (init-state on-BPF in-BPF-pf)
  interst-BPF   :  (inter-state in-BPF-bf T-of-BPF-f fridge-BPF)
  in-BPF-bf     :  (in butter-BPF fridge-BPF)
  in-BPF-pf     :  (in plate-BPF fridge-BPF)
  low-T-BPF     :  (inst-of low-T-BPF low-temp)
  made-of-BPF   :  (made-of plate-BPF mchina-BPF)
  mchina-BPF    :  (inst-of mchina-BPF material-china)
  on-BPF        :  (on plate-BPF butter-BPF)
  plate-BPF     :  (inst-of plate-BPF plate)
  shape-of-BPF  :  (shape-of plate-BPF circular-BPF)
  T-of-BPF-f    :  (temperature-of fridge-BPF low-T-BPF)
  T-of-BPF-b    :  (temperature-of butter-BPF low-T-BPF)   ; goal
  to-reach-BPF  :  (to-reach initst-BPF-1 goalst-BPF)


;;;;;;  Base sit STC (Sugar in Tea in a Cup)

  sit-STC         :  (inst-of sit-STC situation)
  cause-STC       :  (cause initst-STC taste-of-STC-t)
  cup-STC         :  (inst-of cup-STC cup)
  endst-STC       :  (end-state taste-of-STC-t)
  follows-STC     :  (follows initst-STC endst-STC)
  goalst-STC      :  (goal-state taste-of-STC-t sweet-STC tea-STC)
  in-STC-st       :  (in sugar-STC tea-STC)
  in-STC-tc       :  (in tea-STC cup-STC)
  initst-STC      :  (init-state taste-of-STC-s sweet-STC in-STC-st sugar-STC)
  on-STC          :  (on saucer-STC cup-STC)
  saucer-STC      :  (inst-of saucer-STC saucer)
  sugar-STC       :  (inst-of sugar-STC sugar)
  sweet-STC       :  (inst-of sweet-STC sweet-taste)
  taste-of-STC-s  :  (taste-of sugar-STC sweet-STC)
  taste-of-STC-t  :  (taste-of tea-STC sweet-STC)
  tea-STC         :  (inst-of tea-STC tea)
  to-reach-STC    :  (to-reach initst-STC goalst-STC)


;;;;;;  Base sit. SFF (Salt in Food in a Fridge)

  sit-SFF         :  (inst-of sit-SFF situation)
  cause-SFF-i     :  (cause initst-SFF-2 in-SFF-ff)
  cause-SFF-tmp   :  (cause interst-SFF T-of-SFF-fd)
  cause-SFF-tst   :  (cause initst-SFF-3 taste-of-SFF-f)
  endst-SFF       :  (end-state T-of-SFF-fd taste-of-SFF-f)
  follows-SFF     :  (follows initst-SFF-1 endst-SFF)
  food-SFF        :  (inst-of food-SFF food)
  fridge-SFF      :  (inst-of fridge-SFF fridge)
  goalst-SFF      :  (goal-state T-of-SFF-fd low-T-SFF food-SFF)
```

```
  in-SFF-ff      :  (in food-SFF fridge-SFF)
  in-SFF-pf      :  (in plate-SFF fridge-SFF)
  in-SFF-sf      :  (in salt-SFF food-SFF)
  initst-SFF-1   :  (init-state in-SFF-ff T-of-SFF-fr fridge
                                          in-SFF-sf T-of-SFF-fr)
  initst-SFF-2   :  (init-state on-SFF in-SFF-pf)
  initst-SFF-3   :  (init-state in-SFF-sf taste-of-SFF-s salty-SFF)
  interst-SFF    :  (inter-state in-SFF-ff T-of-SFF-fr fridge-SFF)
  low-T-SFF      :  (inst-of low-T-SFF low-temp)
  on-SFF         :  (on plate-SFF food-SFF)
  plate-SFF      :  (inst-of plate-SFF plate)
  salt-SFF       :  (inst-of salt-SFF salt)
  salty-SFF      :  (inst-of salty-SFF salt-taste)
  taste-of-SFF-s :  (taste-of salt-SFF salty-SFF)
  taste-of-SFF-f :  (taste-of food-SFF salty-SFF)
  T-of-SFF-fd    :  (temperature-of food-SFF low-T-SFF)
  T-of-SFF-fr    :  (temperature-of fridge-SFF low-T-SFF)
  to-reach-SFF   :  (to-reach initst-SFF-1 goalst-SFF)


;;;;;;  Base sit. ERW (Egg in Red Water)

  sit-ERW        :  (inst-of sit-ERW situation)
  cause-ERW      :  (cause initst-ERW  color-of-ERW-e)
  color-of-ERW-e :  (color-of egg-ERW red-ERW)
  color-of-ERW-w :  (color-of water-ERW red-ERW)
  egg-ERW        :  (inst-of egg-ERW egg)
  endst-ERW      :  (end-state color-of-ERW-e)
  follows-ERW    :  (follows initst-ERW endst-ERW)
  goalst-ERW     :  (goal-state color-of-ERW-e egg-ERW)
  in-ERW-ew      :  (in egg-ERW water-ERW)
  in-ERW-wt      :  (in water-ERW tpot-ERW)
  initst-ERW     :  (init-state color-of-ERW-w red-ERW egg-ERW in-ERW-ew)
  made-of-ERW    :  (made-of tpot-ERW mmetal-ERW)
  mmetal-ERW     :  (inst-of mmetal-ERW material-metal)
  red-ERW        :  (inst-of red-ERW red)
  to-reach-ERW   :  (to-reach initst-ERW goalst-ERW)
  tpot-ERW       :  (inst-of tpot-ERW teapot)
  water-ERW      :  (inst-of water-ERW water)


;;;;;;  Base sit. GWB (Glass in a Wooden Box)

  sit-GWB       :  (inst-of sit-GWB situation)
  box-GWB       :  (inst-of box-GWB box)
  cause-GWB     :  (cause in-GWB protects-GWB)
  endst-GWB     :  (end-state protects-GWB)
  follows-GWB   :  (follows initst-GWB endst-GWB)
  glass-GWB     :  (inst-of glass-GWB glass)
  goalst-GWB    :  (goal-state protects-GWB)
  in-GWB        :  (in glass-GWB box-GWB)
  initst-GWB    :  (init-state glass-GWB box-GWB in-GWB)
  made-of-GWB-b :  (made-of box-GWB mwood-GWB)
  made-of-GWB-g :  (made-of glass-GWB mglass-GWB)
  mglass-GWB    :  (inst-of mglass-GWB material-glass)
  mwood-GWB     :  (inst-of mwood-GWB material-wood)
  protects-GWB  :  (protects box-GWB glass-GWB)
  to-reach-GWB  :  (to-reach initst-GWB goalst-GWB)


;;;;;;  Target problem HM1  (Heating Milk, variant 1)

  sit-HM1       :  (inst-of sit-HM1 situation)
  goalst-HM1    :  (goal-state T-of-HM1 high-T-HM1)
  in-HM1        :  (in milk-HM1 tpot-HM1)
  initst-HM1    :  (init-state milk-HM1 tpot-HM1 in-HM1 made-of-HM1)
  high-T-HM1    :  (inst-of high-T-HM1 high-temp)
```

```
  made-of-HM1   :  (made-of tpot-HM1 mmetal-HM1)
  mmetal-HM1    :  (inst-of mmetal-HM1 material-metal)
  milk-HM1      :  (inst-of milk-HM1 milk)
  T-of-HM1      :  (temperature-of milk-HM1 high-T-HM1)
  to-reach-HM1  :  (to-reach initst-HM1 goalst-HM1)
  tpot-HM1      :  (inst-of tpot-HM1 teapot)


;;;;;;  Target problem HM2 (Heating Milk, variant 2)

  sit-HM2       :  (inst-of sit-HM2 situation)
  endst-HM2     :  (end-state ???)
  follows-HM2   :  (follows initst-HM2 endst-HM2)
  high-T-HM2    :  (inst-of high-T-HM2 high-temp)
  hplate-HM2    :  (inst-of hplate-HM2 hot-plate)
  in-HM2        :  (in milk-HM2 tpot-HM2)
  initst-HM2    :  (init-state hplate-HM2 on-HM2 in-HM2 T-of-HM2)
  milk-HM2      :  (inst-of milk-HM2 milk)
  on-HM2        :  (on hplate-HM2 tpot-HM2)
  T-of-HM2      :  (temperature-of hplate-HM2 high-T-HM2)
  tpot-HM2      :  (inst-of tpot-HM2 teapot)


;;;;;;  Target problem CM1  (Cooling Milk, variant 1)

  sit-CM1       :  (inst-of sit-CM1 situation)
  goalst-CM1    :  (goal-state T-of-CM1 low-T-CM1)
  in-CM1        :  (in milk-CM1 tpot-CM1)
  initst-CM1    :  (init-state milk-CM1 tpot-CM1 in-CM1 made-of-CM1)
  low-T-CM1     :  (inst-of low-T-CM1 low-temp)
  made-of-CM1   :  (made-of tpot-CM1 mmetal-CM1)
  milk-CM1      :  (inst-of milk-CM1 milk)
  mmetal-CM1    :  (inst-of mmetal-CM1 material-metal)
  T-of-CM1      :  (temperature-of milk-CM1 low-T-CM1)
  to-reach-CM1  :  (to-reach initst-CM1 goalst-CM1)
  tpot-CM1      :  (inst-of tpot-CM1 teapot)


;;;;;;  Target problem CM2  (Cooling Milk, variant 2)

  sit-CM2       :  (inst-of sit-CM2 situation)
  black-CM2     :  (inst-of black-CM2 black)
  color-of-CM2  :  (color-of tpot-CM2 black-CM2)
  goalst-CM2    :  (goal-state ???)
  to-reach-CM2  :  (to-reach initst-CM2 goalst-CM2)
  fridge-CM2    :  (inst-of fridge-CM2 fridge)
  in-CM2-mt     :  (in milk-CM2 tpot-CM2)
  in-CM2-tf     :  (in tpot-CM2 fridge-CM2)
  initst-CM2    :  (init-state fridge-CM2 in-CM2-tf in-CM2-mt T-of-CM2)
  low-T-CM2     :  (inst-of low-T-CM2 low-temp)
  milk-CM2      :  (inst-of milk-CM2 milk)
  T-of-CM2      :  (temperature-of fridge-CM2 low-T-CM2)
  tpot-CM2      :  (inst-of tpot-CM2 teapot)

;;;;;;  Target problem WB1  (Water in a wooden Bowl)

  sit-WB1       :  (inst-of sit-WB1 situation)
  bowl-WB1      :  (inst-of bowl-WB1 bowl)
  goalst-WB1    :  (goal-state T-of-WB1 high-T-WB1)
  in-WB1        :  (in water-WB1 bowl-WB1)
  initst-WB1    :  (init-state water-WB1 bowl-WB1 in-WB1 made-of-WB1)
  high-T-WB1    :  (inst-of high-T-WB1 high-temp)
  made-of-WB1   :  (made-of bowl-WB1 mwood-WB1)
  mwood-WB1     :  (inst-of mwood-WB1 material-wood)
  T-of-WB1      :  (temperature-of water-WB1 high-T-WB1)
  to-reach-WB1  :  (to-reach initst-WB1 goalst-WB1)
  water-WB1     :  (inst-of water-WB1 water)
```

```
;;;;;;  Target problem WG1  (Water in a Glass)

  sit-WG1       :  (inst-of sit-WG1 situation)
  color-of-WG1  :  (color-of glass-WG1 white-WG1)
  glass-WG1     :  (inst-of glass-WG1 glass)
  goalst-WG1    :  (goal-state T-of-WG1 high-T-WG1)
  in-WG1        :  (in water-WG1 glass-WG1)
  initst-WG1    :  (init-state water-WG1 glass-WG1 in-WG1 made-of-WG1)
  high-T-WG1    :  (inst-of high-T-WG1 high-temp)
  made-of-WG1   :  (made-of glass-WG1 mglass-WG1)
  mglass-WG1    :  (inst-of mglass-WG1 material-glass)
  T-of-WG1      :  (temperature-of water-WG1 high-T-WG1)
  to-reach-WG1  :  (to-reach initst-WG1 goalst-WG1)
  water-WG1     :  (inst-of water-WG1 water)
  white-WG1     :  (inst-of white-WG1 white)


;;;;;;  Target problem SF1  (Salty Food, variant 1)

  sit-SF1       :  (inst-of sit-SF1 situation)
  food-SF1      :  (inst-of food-SF1 food)
  goalst-SF1    :  (goal-state taste-of-SF1 salty-SF1)
  initst-SF1    :  (init-state food-SF1 plate-SF1 on-SF1 made-of-SF1)
  made-of-SF1   :  (made-of plate-SF1 mchina-SF1)
  mchina-SF1    :  (inst-of mchina-SF1 material-china)
  on-SF1        :  (on plate-SF1 food-SF1)
  plate-SF1     :  (inst-of plate-SF1 plate)
  salty-SF1     :  (inst-of salty-SF1 salt-taste)
  taste-of-SF1  :  (taste-of food-SF1 salty-SF1)
  to-reach-SF1  :  (to-reach initst-SF1 goalst-SF1)


;;;;;;  Target problem SF2  (Salty Food, variant 2)

  sit-SF2       :  (inst-of sit-SF2 situation)
  endst-SF2     :  (end-state ???)
  follows-SF2   :  (follows initst-SF2 endst-SF2)
  food-SF2      :  (inst-of food-SF2 food)
  in-SF2        :  (in salt-SF2 food-SF2)
  initst-SF2    :  (init-state salt-SF2 food-SF2 plate-SF2)
  on-SF2        :  (on plate-SF2 food-SF2)
  plate-SF2     :  (inst-of plate-SF2 plate)
  salt-SF2      :  (inst-of salt-SF2 salt)


;;;;;;  Target problem EHW  (Egg in Hot Water)

  sit-EHW       :  (inst-of sit-EHW situation)
  color-of-EHW  :  (color-of egg-EHW white-EHW)
  egg-EHW       :  (inst-of egg-EHW egg)
  endst-EHW     :  (end-state ???)
  follows-EHW   :  (follows initst-EHW endst-EHW)
  in-EHW-ew     :  (in egg-EHW water-EHW)
  in-EHW-wt     :  (in water-EHW tpot-EHW)
  initst-EHW    :  (init-state egg-EHW in-EHW-ew in-EHW-wt T-of-EHW)
  high-T-EHW    :  (inst-of high-T-EHW high-temp)
  made-of-EHW   :  (made-of tpot-EHW mmetal-EHW)
  mmetal-EHW    :  (inst-of mmetal-EHW material-metal)
  T-of-EHW      :  (temperature-of water-EHW high-T-EHW)
  tpot-EHW      :  (inst-of tpot-EHW teapot)
  water-EHW     :  (inst-of water-EHW water)
  white-EHW     :  (inst-of white-EHW white)
```

```
;;;;;;  Target problem ICC  (Ice Cube in Coke)

  sit-ICC       :  (inst-of sit-ICC situation)
  follows-ICC   :  (follows initst-ICC endst-ICC)
  coke-ICC      :  (inst-of coke-ICC coke)
  endst-ICC     :  (end-state ???)
  glass-ICC     :  (inst-of glass-ICC glass)
  ice-cube-ICC  :  (inst-of ice-cube-ICC ice-cube)
  in-ICC-ic     :  (in ice-cube-ICC coke-ICC)
  in-ICC-cg     :  (in coke-ICC glass-ICC)
  initst-ICC    :  (init-state ice-cube-ICC in-ICC-ic in-ICC-cg T-of-ICC)
  low-T-ICC     :  (inst-of low-T-ICC low-temp)
  made-of-ICC   :  (made-of glass-ICC mglass-ICC)
  mglass-ICC    :  (inst-of mglass-ICC material-glass)
  on-ICC        :  (on table-ICC glass-ICC)
  table-ICC     :  (inst-of table-ICC table)
  T-of-ICC      :  (temperature-of ice-cube-ICC low-T-ICC)
```

Appendix C
The Energetic Analogy: Activation as Power

This appendix describes the exact relationship between symbolic speed and connectionist activation in DUAL (Section 3.1.3.3; Petrov, 1997; Petrov & Kokinov, 1999). This relationship rests on the following energetic analogy: *the manipulation of symbols can be conceptualized as work and the connectionist activation as power*. Doing work requires energy, which is supplied to the symbolic processor by the connectionist aspect of the agent. The energy is calculated by integrating the power over time. The speed of the symbolic computation depends on the power (i.e., the activation level). The same amount of work is completed rapidly when there is plenty of power, slowly when power is scarce, and not at all if it is lacking completely.

The symbolic processing in the architecture can be categorized into segments of increasing complexity (Petrov, 1997). (*i*) A *symbolic operation* is the smallest unit of symbol manipulation. The operations are simple, atomic, and deterministic. They are the elementary instructions of the symbolic processor. (*ii*) A *symbolic step* is a sequence of operations performed by a single agent without intervening symbolic interactions with other agents. (*iii*) A *rigid symbolic process* is a fixed, a priori specified sequence of steps performed by a single agent. There may be intervening interactions. (*iv*) An *emergent symbolic process* is distributed over a *coalition* of interacting agents. It does not have any complete a priori specification. Rather, the course of computation is determined dynamically by the interplay of various pressures (Kokinov et al., 1996).

Each symbolic operation is characterized by some *consumption* C. This is a real number specifying the amount of symbolic work embedded in the operation. Different operations may have different consumptions. They are free parameters of the particular model and may be fixed on theoretical grounds or estimated from empirical data. This scheme offers considerably more freedom than the alternative proposals that typically assume equal consumption for all operations.

If a fine-grained analysis at the level of individual operations is not warranted, consumptions may be specified at the level of symbolic steps. The latter are often more convenient due to their larger grain size. A symbolic step is performed by a single agent and by definition there is no symbolic exchange with other agents during the step. Thus as far as the inter-agent communication is concerned, steps can be treated as units, disregarding the constituent operations. What matters is the final outcome (in the form of a message send to another agent) and the timing of its appearance.

Each symbolic processor acts as a machine that transforms connectionist energy into symbolic work. Not all energy, however, is converted into useful work. There is some overhead for covering the internal needs of the processor itself. The *efficiency coefficient* η is defined as the ratio of the useful work A to the total energy input E: $\eta = A/E$. This coefficient characterizes the symbolic processor. Different processors can have different efficiencies. In a cognitive model, for instance, processors performing highly automated tasks have η close to 1 while processors performing novel tasks have low efficiency. The efficiency can even be adjusted dynamically by some kind of learning — the basic rule is that it increases with practice.

Suppose a symbolic processor starts working on some operation (or step) at time t_0. The amount of energy needed for the operation can be calculated in advance — it is $E = C/\eta$, where C is the consumption of the operation. This energy must be provided by the connectionist aspect of the agent. This takes time, as the rate of supply is limited. The *energy function* that describes the accumulation of energy in time is defined by the integral:

$$E(t) = \int_{t_0}^{t} a(\tau) d\tau \tag{C.1}$$

where $a(\tau)$ is the activation level. Activation levels in DUAL must be above some positive threshold in order for the symbolic processor to work. If $a(\tau)$ drops below the threshold even for a moment, the symbolic processing is aborted and all intermediate results are lost. Because $a(\tau)$ is always positive, $E(t)$ is an increasing function and thus has an inverse E^{-1}. The inverse function expresses the time needed to produce a given amount of energy.

Putting all pieces together, the exact moment in which the symbolic operation is completed is:

$$t = t_0 + E^{-1}(C/\eta) \tag{C.2}$$

The outcome of the operation becomes available at that moment. It may be a message sent to another agent or a modification of the internal micro-frame. Then the processor moves to the next operation as prescribed by the algorithm and the whole cycle repeats.

When the symbolic processor is idle, all energy produced by the connectionist aspect of the agent goes unused. It cannot be accumulated. In other words, it is not allowed to amass energy "on store" and then expend it all at once, thereby attaining very high peak power.

The energetic analogy offers the following advantages: (*i*) It provides for variable-speed symbolic computation and all the associated benefits. Indeed, it is clear that the specification described in this section implies that the more active agents run more rapidly. (*ii*) The activation levels can change dynamically and all changes have instant effect. (*iii*) The architecture DUAL has a well-defined notion of time. It is measured on a continuous scale and frames the occurrence of all symbolic events. (*iv*) The speed of each DUAL agent is defined independently of that of the other agents. Thus the architecture can be run without any modification on parallel hardware. (*v*) The relationship between symbolic speed and connectionist activation is specified without recourse to any particular implementation. (*vi*) Symbolic processes can be finely parameterized and the parameters have straightforward interpretation — consumptions and efficiency coefficients.

C.1 S-Lisp: A Language for Variable-Speed Symbolic Computations

The DUAL architecture has been fully implemented. All programs are written Common Lisp (Steele, 1990) with CLOS (Keene, 1989). An extension of Lisp called *S-Lisp* ("suspendable" Lisp) has been developed (Petrov, 1997) for the purposes of the variable-speed symbolic computations. A rudimentary compiler translates S-Lisp programs into plain Lisp. This section outlines the main features of the language and the principles of its implementation.

S-Lisp is an extension of Common Lisp. Its main difference from plain Lisp is that it supports four additional special operators: `s-progn`, `s-eval`, `s-values`, and `suspended-value-bind`. They are suspendable analogs to the respective Lisp operators. S-Lisp also supports most (but not all) plain Lisp primitives such as `progn`, `if`, `let`, and `setq`. The language also supports function calls and recursion, which in turn allows for loops.

`S-progn` establishes a sequence of symbolic steps to be executed at variable speed by the processor of some DUAL agent called a *host*. The complementary suspension primitive, `s-eval`, signals that a given S-Lisp form is suspendable and announces the amount of energy needed for it. The two suspension primitives go together. `s-eval` may appear only within the lexical scope of an `s-progn`; it is an error elsewhere. Conversely, `s-progn` is like an ordinary `progn` in all respects except the treatment of `s-eval` and the other suspension primitives. A very simple S-Lisp program is illustrated below:

```
(s-progn host
    (s-eval 0.5 (prepare x))
    (when (s-eval 0.2 (check x))
      (s-eval 2.5 (work-on x)))
    (s-eval 0.1 (cleanup)) )
```

The remaining two suspension primitives, `s-values` and `suspended-value-bind`, are used to export and import values from functions defined via `s-progn`.

The implementation of `s-progn` and `s-eval` is based on delayed evaluation. While the theoretical specification of DUAL postulates that symbolic processes run smoothly and at variable speed, the implementation carries them out in instantaneous jumps. Pauses are imposed between the jumps to produce the timing postulated by the theory.

When a processor begins working on some symbolic operation, it does not actually execute it. Instead, it wraps it in a *closure* and stores it on a stack. One such stack is maintained for each processor (i.e. DUAL agent). There is an *energy balance* associated with each stack. The energy balance is equal to the difference between supplied and consumed energy. When the balance is negative, the processor waits until it becomes positive. On each connectionist cycle, the connectionist machinery increases the balance with some small amount depending on the activation level and the efficiency coefficient of the host. After some time, the balance becomes positive and the top closure on the stack is popped and executed.

The S-Lisp compiler analyzes the source code, identifies all occurrences of `s-eval` and replaces them with plain Lisp forms that generate closures and arrange them on the stack. The stack is established by the enclosing `s-sprogn` form. The top-level loop of the implementation checks the stacks of all active DUAL agents and pops the ones with positive energetic balance. This scheme also supports the parallel work of multiple agents.

S-Lisp programs should be written with care because many intuitions from plain Lisp are violated. In particular, `s-progn` does not return any useful value. This is because a call to `s-progn` does not execute the forms in its body; it delays them. Thus the value that the programmer expects will be computed later, long after the original call to `s-progn` is over. `s-values` must be used to export a value out of a suspendable function and this value must be bound via `suspended-value-bind`.

A *mailbox technique* is used to transfer suspended values through destructive operations executed as side effects. A mailbox is a data structure containing a field that can be modified destructively. The job of `s-values` is to make a new empty mailbox and to arrange that the suspended value (or multiple values) will be stored in it when the suspended computation is completed. The job of `s-progn` is to catch the mailbox, open it at the appropriate time, and bind the values to local variables accessible within its lexical scope.

References

1. Anderson, J. (1983). *The architecture of cognition*. Cambridge, MA: Harvard University Press.
2. Anderson, J. & Lebiere, C. (1998). *The atomic components of thought*. Mahwah, NJ: LEA.
3. Anderson, J. & Thompson, R. (1989). Use of analogy in a production system architecture. In S. Vosniadou & A. Ortony (Eds.), *Similarity and analogical reasoning*. New York: Cambridge University Press.
4. Barclay, C.R. (1986). Schematization of autobiographical memory. In D. C. Rubin (Ed.), *Autobiographical memory* (pp. 82–99). Cambridge, UK: Cambridge University Press.
5. Barsalou, L. (1999). Perceptual symbol systems. *Behavioral and Brain Sciences, 22* (4), 577–660.
6. Bartlett, F. (1932). *Remembering*. Cambridge, MA: Cambridge University Press.
7. Biederman, I. (1987). Recognition-by-components: A theory of human image understanding. *Psychological Review, 94*, 115–147.
8. Blank,D. (1997). *Learning to see analogies: A connectionist exploration*. Unpublished Ph.D. Thesis, Departments of Computer Science and Cognitive Science, Indiana University, Bloomington, IN.
9. Bongard, M. (1970). *Pattern recognition*. New York: Spartan Books.
10. Boroojerdi, B., Phipps, M., Kopylev, L., Wharton, C., Cohen, L., & Grafman, J. (2001). Enhancing analogic reasoning with rTMS over the left prefrontal cortex. *Neurology, 56*, 526–528.
11. Bransford, J. D., Barclay, J. R. & Franks, J. J. (1972). Sentence memory: A constructive versus interpretive approach. *Cognitive Psychology, 3*, 193–209.
12. Bunge, S., Wendelken, C., Badre, D., & Wagner, A. (2005). Analogical reasoning and prefrontal cortex: Evidence for separable retrieval and integration mechanisms. *Cerebral Cortex, 15* (3), 239–249.
13. Carbonell, J. (1983). Learning by analogy: Formulating and generalizing plans from past experience. In R. Michalski, J. Carbonell, & T. Mitchell (Eds.), *Machine learning*. Palo Alto, CA: Tioga.
14. Catrambone, R. & Holyoak, K. (1989). Overcoming contextual limitations in problem-solving transfer. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 15*, 1147–1156.
15. Cer, D. M. & O'Reilly, R. C. (2006). Neural mechanisms of binding in the hippocampus and neocortex: Insights from computational models. In H. Zimmer, A. Mecklinger, & U. Lindenberger (Ed.), *Handbook of binding and memory: Perspectives from Cognitive Neuroscience* (pp. 193–220). Oxford, Oxford University Press.
16. Chalmers, D., French, R. & Hofstadter, D. (1992). High-level perception, representation, and analogy: A critique of artificial intelligence methodology. *Journal for Experimental and*

Theoretical Artificial Intelligence, 4(3), 185–211. (Also available in edited form in Hofstadter et al., 1995)
17. Charniak, E. (1983). Passing markers: A theory of contextual influence in language comprehension. *Cognitive Science, 7 (3)*, 171–190.
18. Christie, S. & Gentner, D. (2010). Where hypotheses come from: Learning new relations by structural alignment. *Journal of Cognition and Development, 11* (3), 356–373.
19. Christoff, K., Prabhakaran, V., Dorfman, J., Zhao, Z., Kroger, J., Holyoak, K., & Gabrieli, J. (2001). Rostrolateral prefrontal cortex involvement in relational integration during reasoning. *NeuroImage, 14*, 1136–1149.
20. Clement, C. & Gentner, D. (1991). Systematicity as a selection constraint in analogical mapping. *Cognitive Science, 15*, 89–132.
21. Cowan, N. (2000). The magical number 4 in short-term memory: A reconsideration of mental storage capacity. *Behavioral and Brain Sciences, 24*, 87–185.
22. Decoste, D. (1991). Dynamic across-time measurement interpretation. *Artificial Intelligence, 51*, 273–341.
23. Dehaene, S., Izard, V., Pica, P., & Spelke, E. (2006). Core knowledge of geometry in an Amazonian indigene group. *Science, 311*, 381–384.
24. Devitt, M. (1999) Theories of reference. In R. A. Wilson & F. C. Keil (Eds.), *The MIT Encyclopedia of the Cognitive Sciences* (pp. 714–715). Cambridge, MA: MIT Press.
25. Doumas, L. & Hummel, J. (2005). Approaches to modeling human mental representations: What works, what doesn't, and why. In K. Holyoak & R. Morrison (Eds.), *The Cambridge handbook of thinking and reasoning* (pp. 73–91). Cambridge, UK: Cambridge University Press.
26. Doumas, L., Hummel, J. & Sandhoffer, C. (2008). A Theory of the discovery and predication of relational concepts. *Psychological Review, 115* (1), 1–43.
27. Dunbar, K. & Blanchette, I. (2001). The *in vivo / in vitro* approach to cognition: The case of analogy. *Trends in Cognitive Sciences, 5* (8), 334–339.
28. Dunker, K. (1945). On problem solving. *Psychological Monographs 38* (270).
29. Eliasmith, C. (2013). *How to build a brain: A neural architecture for biological cognition*. Oxford University Press.
30. Eliasmith, C. & Anderson, C. (2013). *Neural engineering: Computation, representation, and dynamics in neurobiological systems*. Cambridge, MA: MIT Press.
31. Eliasmith, C., Stewart, T., Choo, X., Bekolay, T., DeWolf, T., Tang, C., & Rasmussen, D, (2012). A large-scale model of the functioning brain. *Science, 338*, 1202–1205.
32. Eliasmith, C. & Thagard, P. (2001). Integrating structure and meaning: A distributed model of analogical mapping. *Cognitive Science, 25* (2), 245–286.
33. Evans, T.G. (1968). A program for the solution of a class of geometric-analogy intelligence-test questions. In M. Minsky (Ed.), *Semantic information processing*. Cambridge, MA: MIT Press.
34. Fahlman, S. (1979). *NETL: A system for representing and using real world knowledge.* Cambridge, MA: MIT Press.
35. Falkenhainer, B. (1988). *Learning from physical analogies: A study in analogy and the explanation process.* Unpublished PhD Thesis, University of Illinois at Urbana-Champaign.
36. Falkenhainer, B. (1990a). Analogical interpretation in context. In *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society* (pp. 69–76). Hillsdale, NJ: Erlbaum.
37. Falkenhainer, B. (1990b). A unified approach to explanation and theory formation. In J. Shrager & P. Langley (Eds.), *Computational models of scientific discovery and theory formation* (pp. 157–196). Los Altos, CA: Morgan Kaufmann.
38. Falkenhainer, B., Forbus, K., & Gentner, D. (1986). The structure-mapping engine. *Proceedings of the Fifth Annual Conference on Artificial Intelligence* (pp. 272–277). Los Altos, CA: Morgan Kaufman.
39. Falkenhainer, B., Forbus, K., & Gentner, D. (1989). The structure-mapping engine: Algorithm and examples. *Artificial Intelligence, 41*, 1–63.
40. Feldman, J. (2013). The neural binding problem(s). *Cognitive Neurodynamics, 7*, 1–11.

41. Feldman, J. & Ballard, D. (1982). Connectionist models and their properties. *Cognitive Science, 6* (3), 205–254.
42. Fodor, J. (1993). *The modularity of mind*. Cambridge, MA: MIT Press.
43. Forbus, K. (1984). Qualitative Process Theory. *Artificial Intelligence, 24*, 85–168.
44. Forbus, K. (2001). Exploring analogy in the large. In D. Gentner, K. Holyoak, & B. Kokinov (Eds.) *The analogical mind: Perspectives from cognitive science* (pp. 23–58). Cambridge, MA: MIT Press.
45. Forbus, K. (2010). AI and Cognitive Science: The past and next 30 years. *Topics in Cognitive Science, 2*, 345–356.
46. Forbus, K., Ferguson, R., & Gentner, D. (1994). Incremental structure mapping. *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society* (pp. 313–318). Georgia, Hillsdale, NJ: LEA.
47. Forbus, K., Gentner, D., & Law, K. (1994). MAC/FAC: A model of similarity-based retrieval. *Cognitive Science, 19*, 141–205.
48. Forbus, K., Gentner, D., Markman, A. & Ferguson, R. (1998). Analogy just looks like high level perception: Why a domain-general approach to analogical mapping is right. *Journal of Experimental and Theoretical Artificial Intelligence, 10*, 231–257.
49. Forbus, K. & Hinrichs, T. (2006). Companion cognitive systems: A step towards human-level AI. *AI Magazine, 27* (2), 83–95.
50. Forbus, K., Klenk, M., & Hinrichs, T. (2009). Companion cognitive systems: Design goals and lessons learned so far. *IEEE Intelligent Systems, 24*, 36–46.
51. Forbus, K. & Oblinger, D. (1990). Making SME greedy and pragmatic. *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society* (pp. 61–68). Hillsdale, NJ: Erlbaum.
52. Forbus, K., Usher, J., Lovett, A., Lockwood, K., & Wetzel, J. (2011). CogSketch: Sketch understanding for cognitive science research and for education. *Topics in Cognitive Science, 3* (4), 648–666.
53. Foundalis, H. (2006). *Phaeco: A cognitive architecture inspired by Bongard's problems*. Unpublished Ph.D. Thesis, Department of Computer Science and the Cognitive Science Program. Indiana University, Bloomington, IN.
54. French, R. (1995). *The subtlety of sameness: A theory and computer model of analogy-making*. Cambridge, MA: MIT Press.
55. French, R. (2002). The computational modeling of analogy-making. *Trends in Cognitive Sciences, 6* (5), 200–205.
56. Gayler, R. (2003). Vector symbolic architectures answer Jackendoff's challenges for cognitive neuroscience. In P. Slezak (Ed.) *Proceedings of the ICCS/ASCS International Conference on Cognitive Science* (pp. 133-138).
57. Gayler, R. & Levy, S. (2011). Compositional connectionism in cognitive science II: The localist/distributed dimension. *Connection Science, 23* (2), 85–89.
58. Gentner, D. (1983). Structure-mapping: A theoretical framework for analogy. *Cognitive Science, 7* (2), 155–170.
59. Gentner, D. (1989). The mechanisms of analogical learning. In S. Vosniadou & A. Ortony (Eds.), *Similarity and analogical reasoning* (pp. 199–241). New York: Cambridge University Press.
60. Gentner, D. (2010). Bootstrapping the mind: Analogical processes and symbol systems. *Cognitive Science, 34*, 752–775.
61. Gentner D. & Forbus K. (2011). Computational models of analogy. *WIREs Cognitive Science, 2* (3), 266–276. doi: 10.1002/wcs.105
62. Gentner, D., Holyoak, K. & Kokinov, B. (Eds.) *The analogical mind: Perspectives from cognitive science.* Cambridge, MA: MIT Press.
63. Gentner, D. & Landers, R. (1985). Analogical reminding: A good match is hard to find. In *Proceedings of the International Conference on Systems, Man and Cybernetics*. New York: IEEE.
64. Gentner, D., Loewenstein, J., & Hung, B. (2007). Comparison facilitates children's learning of names for parts. *Journal of Cognition and Development, 8*, 285–307.

65. Gentner, D. & Toupin, C. (1986). Systematicity and surface similarity in the development of analogy. *Cognitive Science, 10*, 277–300.
66. Gick, M & Holyoak, K. (1980). Analogical problem solving. *Cognitive Psychology, 12*, 306–355.
67. Gick, M. & Holyoak, K. (1983). Schema induction and analogical transfer. *Cognitive Psychology, 15*, 1–38.
68. Godden, D. & Baddeley, A. (1975). Context-dependent memory in two natural environments: On land and under water. *British Journal of Psychology, 66*, 325–331.
69. Goshwami, U. (1992). *Analogical reasoning in children*. Hillsdale, NJ: Erlbaum.
70. Grinberg, M. & Kokinov, B. (2003). Analogy-based episode blending in AMBR. In B. Kokinov & W. Hirst (Eds.) *Constructive Memory*. Sofia: NBU Press.
71. Halford, G., Wilson, W. & Phillips, S. (1998). Processing capacity defined by relational complexity: Implications for comparative, developmental, and cognitive psychology. *Behavioral and Brain Sciences, 21*, 803–865.
72. Halford, G., Wilson, W. & Phillips, S. (2010). Relational knowledge: The foundation of higher cognition. *Trends in Cognitive Sciences, 14* (11), 497–505.
73. Hall, R. (1989). Computational approaches to analogical reasoning: A comparative analysis. *Artificial Intelligence, 39*, 39–120.
74. Harnad, S. (1990). The symbol grounding problem. *Physica D, 42*, 335–346.
75. Hendler, J. (1988). *Integrating marker-passing and problem-solving: A spreading-activation approach to improved choice in planning*. Hillsdale, NJ: Erlbaum.
76. Hendler, J. (1989). Marker-passing over microfeatures: Towards a hybrid symbolic/connectionist model. *Cognitive Science, 13*, 79–106.
77. Hofstadter, D. (1983). The architecture of Jumbo. *Proceedings of the International Machine Learning Workshop*. Monticello, IL.
78. Hofstadter, D. (1984). *The Copycat project: An experiment in nondeterminism and creative analogies*. AI Memo No. 755, Massachusetts Institute of Technology, Cambridge, MA.
79. Hofstadter, D. & Mitchell, M. (1991). The Copycat project: A model of mental fluidity and analogy-making. *CRCC Technical Report No. 58*, Center for Research on Concepts and Cognition, Indiana University, Bloomington, IN. (Also available in K. Holyoak & J. Barnden (Eds.), (1994). *Advances in connectionist and neural computation theory. Vol. 2: Analogical connections*. Norwood, NJ: Ablex.)
80. Hofstadter, D. & the Fluid Analogies Research Croup (1995). *Fluid concepts and creative analogies: Computer models of the fundamental mechanisms of thoughts*. New York: Basic Books.
81. Hofstadter, D. & Sander, E. (2013). *Surfaces and essences: Analogy as the fuel and fire of thinking*. New York: Basic Books.
82. Holland, J., Holyoak, K., Nisbett, R., & Thagard, P. (1986). *Induction: Processes of inference, learning, and discovery*. Cambridge, MA: MIT Press.
83. Holyoak, K. (2005). Analogy. In K. Holyoak & R. Morrison (Eds.), *The Cambridge handbook of thinking and reasoning* (pp. 117–142). Cambridge, UK: Cambridge University Press.
84. Holyoak, K. (2012). Analogy and relational reasoning. In K. Holyoak & R. Morrison (Eds.), *The Oxford handbook of thinking and reasoning* (pp. 234–259). New York: Oxford University Press.
85. Holyoak, K. & Koh, K. (1987). Surface and structural similarity in analogical transfer. *Memory & Cognition, 15*(4), 332–340.
86. Holyoak, K., Novick, & Melz (1994a). Component processes in analogical transfer: Mapping, pattern completion, and adaptation. In K. Holyoak & J. Barnden (Eds.), *Advances in connectionist and neural computation theory, Vol. 2: Analogical connections* (pp. 113–180). Norwood, NJ: Ablex.
87. Holyoak, K. & Thagard, P. (1989). Analogical mapping by constraint satisfaction. *Cognitive Science, 13*, 295–355.
88. Holyoak, K. & Thagard, P. (1995). *Mental leaps: Analogy in creative thought*. Cambridge, MA: MIT Press.

89. Hristova, P., Petkov, G., & Kokinov, B. (2005). The influence of irrelevant information on price judgments. In: *Proceedings of the International Conference on Cognitive Economics*. Sofia: New Bulgarian University Press
90. Hummel, J. (2011). Getting symbols out of a neural architecture. *Connection Science, 23* (2), 109–118.
91. Hummel, J. & Biederman, I. (1992). Dynamic binding in a neural network for shape recognition. *Psychological Review, 99* (3), 480–517.
92. Hummel, J. & Holyoak, K. (1996). LISA: A computational model of analogical inference and schema induction. *Proceedings of the Eighteenth Annual Conference of the Cognitive Science Society.* Hillsdale, NJ: Erlbaum.
93. Hummel, J. & Holyoak, K. (1997). Distributed representations of structure: A theory of analogical access and mapping. *Psychological Review, 104* (3), 427–466.
94. Hummel, J. & Holyoak, K. (2003). A symbolic-connectionist theory of relational inference and generalization. *Psychological Review, 110* (2), 220–264.
95. Hummel, J. & Holyoak, K. (2005). Relational reasoning in a neurally plausible cognitive architecture: An overview of the LISA project. *Current Directions in Psychological Science, 14* (3), 153–157.
96. Hummel, J. & Stankiewicz, B. (1998). Two roles for attention in shape perception: A structural description model of visual scritiny. *Visual Cognition, 5*, 49–79.
97. Jani, N. & Levine, D. (2000). A neural network theory of proportional analogy-making. *Neural Networks, 13* (2), 149–183.
98. Jilk, D., Lebiere, C., O'Reilly, R., & Anderson, J. (2008). SAL: An explicitly pluralistic cognitive architecture. *Journal of Experimental and Theoretical Artificial Intelligence, 20* (3), 197–218.
99. Kahneman, D. & Miller, D. (1986). Norm theory: Comparing reality to its alternatives. *Psychological Review, 93* (2), 136–153.
100. Kandel, E., Schwartz, J., & Jessell, T. (2000). *Principles of Neural Science* (4th ed.). New York: McGraw Hill.
101. Kanerva, P. (1988). *Sparse distributed memory*. Cambridge, MA: MIT Press.
102. Kanerva, P. (1998). Dual role of analogy in the design of a cognitive computer. In K. Holyoak, D. Gentner, & B. Kokinov (Eds.), *Advances in analogy research: Integration of theory and data from the cognitive, computational, and neural sciences* (pp. 164–170). Sofia, Bulgaria: New Bulgarian University Press.
103. Kant, I. (1997). *Critique of pure reason* (P. Guyer & A. Wood, Trans. & Eds.). Cambridge: Cambridge University Press. (Original work published 1781)
104. Keane, M. (1987). On retrieving analogues when solving problems. *Quaterly Journal of Experimental Psychology, 39A*, 29–41.
105. Keane, M. (1988). *Analogical problem solving*. Chichester: Ellis Horwood.
106. Keane, M. & Brayshaw, M. (1988). The incremental analogy machine: A computational model of analogy. In D. Sleeman (Ed.), *Third European Working Session on Machine Learning*. London: Pitman.
107. Keane, M., Ledgeway, K. & Duff, S. (1994). Constraints on analogical mapping: A comparison of three models. *Cognitive Science, 18*, 387–438.
108. Kedar-Cabelli, S. (1988). Towards a computational model of purpose-directed analogy. In A. Prieditic (Ed.), *Analogica*. Los Altos, CA: Morgan Kaufmann.
109. Keene, S. (1989). *Object-Oriented Programming in Common Lisp: A Programmer's Guide to CLOS*. Boston: Addison-Wesley.
110. Kiryazov, K., Petkov, G., Grinberg, M., Kokinov, B., & Balkenius, C. (2007). The interplay of analogy-making with active vision and motor control in anticipatory robots. In M. Butz et al. (Eds.), *Anticipatory Behavior in Adaptive Learning Systems: From Brains to Individual and Social Behavior* (pp. 233-253). Lecture Notes in AI, vol. 4520. Berlin: Springer-Verlag.
111. Klenk, M. & Forbus, K. (2009a). Domain transfer via cross-domain analogy. *Cognitive Systems Research, 10*, 240–250.

112. Klenk, M. & Forbus, K. (2009b). Persistent mappings in cross-domain analogical learning of physical domains. In B. Kokinov, K. Holyoak, & D. Gentner (Eds.), *New frontiers in analogy research: Proceedings of the Second International Conference on Analogy*. Sofia, Bulgaria: New Bulgarian University Press.
113. Knowlton, B., Morrison, R., Hummel, J. & Holyoak, K. (2012). A neurocomputational system for relational reasoning. *Trends in Cognitive Sciences, 16* (7), 373–381.
114. Koch, C. & Segev, I. (2000). The role of single neurons in information processing. *Nature Neuroscience, 3*, 1171–1177.
115. Kokinov, B. (1988). Associative memory-based reasoning: How to represent and retrieve cases. In T. O'Shea & V. Sgurev (Eds.), *Artificial Intelligence III: Methodology, systems, applications*. Amsterdam: Elsevier.
116. Kokinov, B. (1989). About modeling some aspects of human memory. In F. Klix, N.Streitz, Y.Waern, & H.Wandke (Eds.), *MACINTER II*. Amsterdam: North-Holland.
117. Kokinov, B. (1990). Associative memory-based reasoning: Some experimental results. *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum.
118. Kokinov, B. (1992a). Inference evaluation in deductive, inductive, and analogical reasoning. *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum.
119. Kokinov, B. (1992b). Similarity in analogical reasoning. In B. du Boulay & V. Sgurev (Eds.), *Artificial Intelligence V: Methodology, systems, applications*. Amsterdam, Elsevier.
120. Kokinov, B. (1994a). A hybrid model of reasoning by analogy. In K. Holyoak & J. Barnden (Eds.), *Advances in connectionist and neural computation theory, Vol. 2: Analogical connections* (pp. 247–318). Norwood, NJ: Ablex.
121. Kokinov, B. (1994b). The DUAL cognitive architecture: A hybrid multi-agent approach. In A. Cohn (Ed.), *Proceedings of the Eleventh European Conference of Artificial Intelligence* (pp. 203–207). London: John Wiley & Sons, Ltd.
122. Kokinov, B. (1994c). The context-sensitive cognitive architecture DUAL. *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum.
123. Kokinov, B. (1995). A dynamic approach to context modeling. In P. Brézillon & S. Abu-Hakima (Eds.), *Working notes of the IJCAI-95 Workshop on "Modeling context in knowledge representation and reasoning."* Paris: Institute B. Pascal. LAFORIA 95/11.
124. Kokinov, B. (1997). AMBR view on analogy. Paper presented at the *Conference on Thinking*. London. March, 1997.
125. Kokinov, B. (1998). Analogy is like cognition: Dynamic, emergent, and context-sensitive. In K. Holyoak, D. Gentner, & B. Kokinov (Eds.), *Advances in analogy research: Integration of theory and data from the cognitive, computational, and neural sciences*. Sofia, Bulgaria: New Bulgarian University Press.
126. Kokinov, B. (2003). The mechanisms of episode construction and blending in DUAL and AMBR: Interaction between memory and analogy. In B. Kokinov & W. Hirst (Eds.), *Constructive Memory* (pp. 117–121). Sofia, Bulgaria: New Bulgarian University Press.
127. Kokinov, B. (2006). Analogy and constructive memory. In D. Fum, F. del Missier, & A. Stocco (Eds.), *Proceedings of the Seventh International Conference on Cognitive Modeling*. Trieste, Italy: Edizioni Goliardiche.
128. Kokinov, B., Bliznashki, S., Kosev, S., & Hristova, P. (2007). Analogical mapping and perception: Can mapping cause a re-representation of the target stimulus? In D. McNamara & G. Trafton (Eds.) *Proceedings of the Twenty-Ninth Annual Conference of the Cognitive Science Society* (pp. 592–597). Mahwah, NJ: Erlbaum.
129. Kokinov, B. & French, R. (2002). Computational models of analogy-making. In L. Nadel (Ed.), *Encyclopedia of Cognitive Science* (pp. 113–118). London: Macmillan.
130. Kokinov, B., Grinberg, M., Petkov, G., & Kiryazov, K. (2008). Anticipation by analogy. In G. Pezzulo, M. Butz, C. Castelfranchi, & R. Falcone (Eds.), *The Challenge of Anticipation* (pp. 189–218). Lecture Notes on AI, vol. 5225. Bremen, Germany: Springer-Verlag.
131. Kokinov, B., Hadjiilieva, K., & Yoveva, M. (1997). Is a hint always useful? *Proceedings of the Nineteenth Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum.

132. Kokinov, B., Holyoak, K., & Gentner, D. (Eds.) (2009). *New frontiers in analogy research: Proceedings of the Second International Conference on Analogy*. Sofia, Bulgaria: New Bulgarian University Press.
133. Kokinov, B., Hristova, P. & Petkov, G. (2004). Does irrelevant information play a role in judgment? In K. Forbus, D. Gentner, & T. Regier (Eds.) *Proceedings of the Twenty-Sixth Annual Conference of the Cognitive Science Society* (pp. 720–725). Mahwah, NJ: Erlbaum.
134. Kokinov, B., Nikolov, V. & Petrov, A. (1996). Dynamics of emergent computation in DUAL. In A. Ramsay (Ed.), *Artificial Intelligence: Methodology, systems, applications* (pp. 303–311). Amsterdam: IOS Press.
135. Kokinov, B. & Petrov, A. (2000). Dynamic extension of episode representation in analogy-making in AMBR. *Proceedings of the Twenty-Second Annual Conference of the Cognitive Science Society* (pp. 274–279). Hillsdale, NJ: Erlbaum.
136. Kokinov, B. & Petrov, A. (2001). Integrating memory and reasoning in analogy-making: The AMBR model. In D. Gentner, K. Holyoak, & B. Kokinov (Eds.) *The analogical mind: Perspectives from cognitive science* (pp. 59–124). Cambridge, MA: MIT Press.
137. Kokinov, B. & Yoveva, M. (1996). Context effects on problem solving. In *Proceedings of the Eighteenth Annual Conference of the Cognitive Science Society* (pp. 586–590). La Jolla, CA: Erlbaum.
138. Kokinov, B. & Zareva-Toncheva, N. (2001). Episode blending as a result of analogical problem solving. *Proceedings of the Twenty-Third Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum.
139. Kolodner, J. (1993). *Case-based reasoning*. San Mateo, CA: Morgan Kaufmann.
140. Könik, T., O'Rorke, P., Shapiro, D., Choi, D., Nejati, N., & Langley, P. (2009). Skill transfer through goal-driven representation mapping. *Cognitive Systems Research, 10* (3), 270–285.
141. Krawczyk, D. (2012). The cognition and neuroscience of relational reasoning. *Brain Research, 1428*, 13–23.
142. Krawczyk, D., Holyoak, K. & Hummel, J. (2005). The one-to-one constraint in analogical mapping and inference. *Cognitive Science, 29* (5), 797–806.
143. Kriete, T. & Noelle, D. (2011). Generalization benefits of output gating in a model of prefrontal cortex. *Connection Science, 23* (2), 119–129.
144. Kriete, T., Noelle, D., Cohen, J., & O'Reilly, R. (2013). Indirection and symbol-like processing in the prefrontal cortex and basal ganglia. *Proceedings of the National Academy of Sciences, USA*. doi: 10.1073/pnas.1303547110.
145. Kripke, S. A. (1980). *Naming and necessity*. Cambridge, MA: Harvard University Press.
146. Kuehne, S., Forbus, K., Gentner, D., & Quinn, B. (2000) SEQL: Category learning as progressive abstraction using structure mapping. In L. R. Gleitman & A. K. Joshi (Eds.), *Proceedings of the Twenty-Second Annual Conference of the Cognitive Science Society* (pp. 770–775). Philadelphia, PA.
147. Lakoff, G. (1987). *Women, fire, and dangerous things*. Chicago: University of Chicago Press.
148. Lakoff, G. & Johnson, M. (1980). *Metaphors we live by*. Chicago: University of Chicago Press.
149. Lakoff, G. & Johnson, M. (1999). *Philosophy in the flesh: The embodied mind and its challenge to Western thought*. New York: Basic Books.
150. Larkey, L. & Love, B. (2003). CAB: Connectionist Analogy Builder. *Cognitive Science, 27* (5), 781–794.
151. Leech, R., Mareschal, D. & Cooper, R. (2008). Analogy as relational priming: A developmental and computational perspective on the origins of a complex cognitive skill. *Behavioral and Brain Sciences, 31*, 357–414.
152. Lenat, D. (1995). CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM, 38* (11), 33–38.
153. Linhares, A. (2000). A glimpse at the metaphysics of Bongard problems. *Artificial Intelligence, 121* (1-2), 251–270.
154. Loewenstein, J., Thompson, L., & Gentner, D. (1999). Analogical encoding facilitates knowledge transfer in negotiation. *Psychonomic Bulletin & Review, 6*, 586–597.

155. Loftus, E., Feldman, J., & Dashiell, R. (1995). The reality of illusory memories. In D. Schacter (Ed.), *Memory distortions: How minds, brains, and societies reconstruct the past* (pp. 47–68). Cambridge, MA: Harvard University Press.
156. Lovett, A. (2012). *Spatial Routines: A framework for modeling spatial problem-solving*. Unpublished Ph.D. Thesis, Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL.
157. Lovett, A. & Forbus, K. (2011). Cultural commonalities and differences in spatial problem-solving: A computational analysis. *Cognition, 121*, 281–287.
158. Lovett, A., Forbus, K. & Usher, J. (2010). A structure-mapping model of Raven's Progressive Matrices. In S. Ohlsson & R. Catrambone (Eds.), *Proceedings of the Thirty-Second Annual Conference of the Cognitive Science Society* (pp. 2761–2766). Austin, TX: Cognitive Science Society.
159. Lovett, A., Tomai, E., Forbus, K. & Usher, J. (2009). Solving geometric analogy problems through two-stage analogical mapping. *Cognitive Science, 33* (7), 1192–1231.
160. Lu, H., Chen, D. & Holyoak, K. (2012). Bayesian analogy with relational transformations. *Psychological Review, 119* (3), 617–648.
161. Marshall, J. (1999). *Metacat: A self-watching cognitive architecture for analogy-making and perception*. Unpublished Ph.D. Thesis, Department of Computer Science and the Cognitive Science Program. Indiana University, Bloomington, IN.
162. Marshall, J. (2006). A self-watching model of analogy-making and perception. *Journal of Experimental and Theoretical Artificial Intelligence, 18* (3), 267–307.
163. McCarthy, J. & Hayes, P. (1969). Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer & D. Michie (Eds.), *Machine Intelligence*, Vol. 4. Edinburgh: Edinburgh University Press.
164. McClelland, J. & Rumelhart, D. (1981). An interactive activation model of context effects in letter perception: Part 1. An account of basic findings. *Psychological Review, 88*, 375–407.
165. McGraw, G. (1995). *Letter Spirit (part one): Emergent high level perception of letters using fluid concepts*. Unpublished Ph.D. Thesis, Department of Computer Science and the Cognitive Science Program. Indiana University, Bloomington, IN.
166. McKoon, G. & Ratcliff, R. (1981). The comprehension process and memory structures involved in instrumental inference. *Journal of Verbal Learning and Verbal Behavior, 20*, 671–682.
167. McKoon, G. & Ratcliff, R. (1992). Inference during reading. *Psychological Review, 99* (3), 440–466.
168. McLure, M. D., Friedman, S. E., & Forbus, K.(2010). Learning concepts from sketches via analogical generalization and near-misses. In S. Ohlsson & R. Catrambone (Eds.), *Proceedings of the Thirty-Second Annual Conference of the Cognitive Science Society* (pp. 1726–1731). Austin, TX: Cognitive Science Society.
169. Medin, D., Goldstone, R., & Gentner, D. (1993). Respects for similarity. *Psychological Review, 100* (2), 254–278.
170. Michalski, R. (1989). Two-tiered concept meaning, inferential matching, and conceptual cohesiveness. In S. Vosniadou & A. Ortony (Eds.), *Similarity and analogical reasoning*. New York: Cambridge University Press.
171. Minsky, M. (1986). *The society of mind*. New York: Simon & Schuster.
172. Mitchell, M. (1993). *Analogy-making as perception: A computer model*. Cambridge, MA: MIT Press.
173. Morrison, R., Krawczyk, D., Holyoak, K., Hummel, J., Chow, T., Miller, B., & Knowlton, B. (2004). A neurocomputational model of analogical reasoning and its breakdown in frontotemporal lobar degeneration. *Journal of Cognitive Neuroscience, 16* (2), 260–271.
174. Müller, H. J., Elliott, M. A., Herrmann, C. S., & Mecklinger, A. (2001). Neural binding of space and time: An introduction. *Visual Cognition, 8* (3/4/5), 273–285.
175. Nairne, J. (1990). A feature model of immediate memory. *Memory & Cognition, 18*, 251–269.
176. Nairne, J. (2002). Remembering over the short-term: The case against the standard model. *Annual Review of Psychology, 53*, 53–81.

177. Neath, I. & Nairne, J. (1995). Word-length effects in immediate memory: Overwriting trace-decay theory. *Psychological Bulletin & Review, 2*, 429–441.
178. Nestor, A. & Kokinov, B. (2004). Towards active vision in the DUAL cognitive architecture. *International Journal on Information Theories & Applications, 11* (1), 9–15.
179. Newell, A. (1990). *Unified Theories of Cognition*. Cambridge, MA: Harvard University Press.
180. Nosofsky, R. & Palmeri, T. (1997). An exemplar based random walk model of speeded categorization and absolute judgment. *Psychological Review, 104* (2), 266–300.
181. O'Donoghue, D. P., Bohan, A., & Keane, M. (2006). Seeing things — Inventive reasoning with geometric analogies and topographic maps. *New Generation Computing, 24*, 267–288.
182. O'Reilly, R. (1998). Six principles for biologically based computational models of cortical function. *Trends in Cognitive Sciences, 2* (11), 455–462.
183. O'Reilly, R. (2006). Biologically based computational models of high-level cognition. *Science, 314*, 91–94.
184. O'Reilly, R., Braver. T., & Cohen, J. (1999). A biologically-based neural network model of working memory. In P. Shah & A. Miyake (Eds.), *Models of working memory: Mechanisms of active maintenance and executive control* (pp. 375–411). New York: Cambridge University Press.
185. O'Reilly, R., Busby, R., & Soto, R. (2003). Three forms of binding and their neural substrates: Alternatives to temporal synchrony. In A. Cleermans (Ed.), *The unity of consciousness: Binding, integration, and dissociation* (pp. 168–192). Oxford: Oxford University Press.
186. O'Reilly, R., Hazy, T. & Herd, S. (in press). The Leabra cognitive architecture: How to pPlay 20 principles with nature and win! In S. Chipman (Ed.), *Oxford Handbook of Cognitive Science*. Oxford: Oxford University Press.
187. O'Reilly, R. & Munakata, Y. (2000). *Computational explorations in cognitive neuroscience: Understanding the mind by simulating the brain*. Cambridge, MA: MIT Press.
188. O'Reilly, R., Petrov, A. A., Cohen, J., Lebiere, C., Herd, S., & Kriete (in press). How limited systematicity emerges: A computational cognitive neuroscience approach. In P. Calvo & J. Symons (Eds.), *Systematicity and cognitive architecture: Conceptual and empirical issues 25 years after Fodor & Pylyshyn's challenge to connectionism*. Cambridge, MA: MIT Press.
189. Petkov, G. (2005). Judgment as mapping (JUDGEMAP2). In: C. Chidini, B.-L. Doan, & P. Brézillion (Eds.), *Proceedings of the Fifth International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-05)* (pp. 95–104). Paris. http://www.lip6.fr/lip6/reports/2005/lip6-2005-006.pdf
190. Petkov, G. (2006). Modeling analogy-making, judgment, and choice with the same basic mechanisms. In D. Fum, F. del Missier, & A. Stocco (Eds.), *Proceedings of the Seventh International Conference on Cognitive Modeling* (pp. 220–225). Trieste, Italy: Edizioni Goliardiche.
191. Petkov, G., Hristova, P., & Kokinov, B. (2005). How irrelevant information influences judgment. In: C. Chidini, B.-L. Doan, & P. Brézillion (Eds.), *Proceedings of the Fifth International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-05)* (pp. 75–78). Paris. http://www.lip6.fr/lip6/ reports/2005/lip6-2005-007.pdf
192. Petkov, G., Kiryazov, K., Grinberg, M., & Kokinov, B. (2007). Modeling top-down perception and analogical transfer with single anticipatory mechanism. In S. Vosniadou et al. (Eds.), *Proceedings of the European Conference on Cognitive Science* (pp. 395–400). Hillsdale, NJ: Earlbaum.
193. Petkov, G. & Kokinov, B. (2006). JUDGEMAP—Integration of analogy-making, judgment, and choice. In R. Sun & N. Miyake (Eds.), *Proceedings of the Twenty-Eighth Annual Conference of the Cognitive Science Society* (pp. 1950–1955). Mahwah, NJ: Erlbaum.
194. Petkov, G., Naydenov, Ch., Grinberg, M., & Kokinov, B. (2006). Building robots with analogy-based anticipation. In C. Freksa, M. Kohlhase, & K. Schill. (Eds.), *Proceedings of the 29th German Conference on Artificial Intelligence (KI-2006)* (pp. 72–86). Lecture Notes on AI, vol. 4314. Bremen, Germany: Springer-Verlag.

195. Petkov, G. & Shahbazyan, L. (2007). The RecMap model of active recognition based on analogical mapping. In R. Lewis, T. Polk & J. Laird (Eds.), *Proceedings of the Eighth International Conference on Cognitive Modeling* (pp. 293–298). Ann Arbor, MI.
196. Petrov, A. A. (1997). *Extensions of DUAL and AMBR.* Unpublished M.Sc. Thesis, Central and East European Center for Cognitive Science, New Bulgarian University, Sofia, Bulgaria.
197. Petrov, A. A. (1998). *A Dynamic Emergent Computational Model of Analogy-Making Based on Decentralized Representations.* Unpublished Ph.D. Thesis, Central and East European Center for Cognitive Science, New Bulgarian University, Sofia, Bulgaria.
198. Petrov, A. A. (2008). Relational priming plays a supportive but not leading role in adult analogy-making. *Behavioral and Brain Sciences, 31* (4), 392–393.
199. Petrov, A. A. & Anderson, J. (2005). The dynamics of scaling: A memory-based anchor model of category rating. *Psychological Review, 112* (2), 383–416.
200. Petrov, A. A. & Kokinov, B. (1999). Processing symbols at variable speed in DUAL: Connectionist activation as power supply. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence* (vol. 2, pp. 846–851).
201. Petrov, A. A., Huang, T., & O'Reilly, R. (2013). Analogical mapping with nonstationary inputs: A neural model with learned coarse-coded conjunctive distributed representations [Abstract]. In J.-P. Thibaut, R. French, & Y. Glady (Eds.), *Proceedings of the Third International Conference on Analogy*. Université de Bourgogne, Dijon, France.
202. Petrov, A. A., Van Horn, N., & Todd, J. (2011). The visual identification of relational categories. *Journal of Vision, 11* (12):11, 1–11.
203. Plate, T. (2003). *Holographic reduced representation: Distributed representation for cognitive structures*. Stanford, CA: CSLI Publications.
204. Postman, L. & Phillips, L.W. (1965). Short-term temporal changes in free recall. *Quaterly Journal of Experimental Psychology, 17*, 132–138.
205. Quillian, M. (1969). The teachable language comprehender: A simulation program and theory of language. *Communication of the ACM, 12*, 459–476.
206. Quine, W. V. (1968). Ontological relativity. *The Journal of Philosophy, 65* (7), 185–212.
207. Rachkovskij, D. (2004). Some approaches to analogical mapping with structure-sensitive distributed representations. *Journal of Experimental and Theoretical Artificial Intelligence, 16* (3), 125–144.
208. Ramscar, M. & Yarlett, D. (2003). Semantic grounding in models of analogy: An environmental approach. *Cognitive Science, 27* (1), 41–71.
209. Rasmussen, D. & Eliasmith, C. (2011). A neural model of rule generation and inductive reasoning. *Topics in Cognitive Science, 3*, 140–153.
210. Raven, J., Raven, J. C., & Court, J. H. (2000). *Manual for Raven's Progressive Matrices and Vocabulary Scales*. Oxford: Oxford Psychologists Press.
211. Rehling, J. (2001). *Letter Spirit (part two): Modeling creativity in a visual domain*. Unpublished Ph.D. Thesis, Department of Computer Science and the Cognitive Science Program. Indiana University, Bloomington, IN.
212. Robbins, R. & Aydede, M. (Eds.) (2008). *The Cambridge Handbook of Situated Cognition*. Cambridge, MA: Cambridge University Press.
213. Roskies, A. (1999). The binding problem: Review introduction. *Neuron, 24* (1), 7—9.
214. Ross, B. (1984). Remindings and their effects in learning a cognitive skill. *Cognitive Psychology, 16*, 371–416.
215. Ross, B. (1987). This is like that: The use of earlier problems and the separation of similarity effects. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 13*, 629–639.
216. Rumelhart, D., McClelland, J., & the PDP Research Group (1986). *Parallel distributed processing, Vol. 1: Foundations*. Cambridge, MA: MIT Press.
217. Russell, S. & Norvig, P. (2009). *Artificial Intelligence: A Modern Approach* (3rd Ed.). Upper Saddle River, NJ: Prentice Hall.
218. Salvucci, D. & Anderson, J. (2001). Integrating analogical mapping and general problem solving: The path-mapping theory. *Cognitive Science, 25* (1), 67–110.

219. Schacter, D. (Ed.) (1995). *Memory distortions: How minds, brains, and societies reconstruct the past*. Cambridge, MA: Harvard University Press.
220. Schwering, A., Krumnack, U., Kühnberger, K.-U., & Gust, H. (Eds.) (2007). *Analogies: Integrating multiple cognitive abilities*. Publication of the Institute of Cognitive Science, University of Osnabrück, Vol. 5-2007. Osnabrück, Germany.
221. Schwering, A., Krumnack, U., Kühnberger, K.-U., & Gust, H. (2009). Syntactic principles of heuristic-driven theory projection. *Cognitive Systems Research, 10* (3), 251–269.
222. Schwering, A., Kühnberger, K.-U., & Kokinov, B. (2009). Analogies – Integrating cognitive abilities. *Cognitive Systems Research, 10* (3), 175–177. [Editorial to a special issue.]
223. Searle, J. (1980). Minds, brains, and programs. *Behavioral and Brain Sciences, 3*, 417–457.
224. Seifert, C., McKoon, G., Abelson, R., & Ratcliff, R. (1986). Memory connections between thematically similar episodes. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 12*, 220–231.
225. Shahbazyan, L. & Petkov, G. (2007). Towards modeling recognition at multiple levels. In A. Schwering, U. Krumnack, K.-U. Kühnberger, & H. Gust (Eds.), *Analogies: Integrating multiple cognitive abilities* (pp. 77–82). Publication of the Institute of Cognitive Science, University of Osnabrück, Vol. 5-2007. Osnabrück, Germany.
226. Shadlen, M. & Movshon, J. A. (1999). Synchrony unbound: A critical evaluation of the temporal binding hypothesis. *Neuron, 24* (1), 67–77.
227. Shastri, L. & Ajjanagadde, V. (1993). From simple associations to systematic reasoning: A connectionist representation of rules, variables and dynamic bindings using temporal synchrony. *Behavioral and Brain Sciences, 16*, 417–494.
228. Singer, W. (2007). Binding by synchrony. *Scholaripedia, 2* (12):1657.
229. Singer, W. & Gray, C. M. (1995). Visual feature integration and the temporal correlation hypothesis. *Annual Review of Neuroscience, 18*, 555–586.
230. Smolensky, P. (1990). Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence, 46*, 159–216.
231. Smolensky, P. & Legendre, G. (2006). *The harmonic mind: From neural computation to optimality-theoretic grammar*. Cambridge, MA: MIT Press.
232. Steele, G. L., Jr. (1990). *Common Lisp: The Language* (2nd Ed.). Bedford, MA: Digital Press.
233. Stewart, T., Bekolay, T., & Eliasmith, C. (2011). Neural representations of compositional structures: Representing and manipulating vector spaces with spiking neurons. *Connection Science, 23* (2), 145–153.
234. Swadlow, H. & Waxman, S. (2012). Axonal conduction delays. *Scholaripedia, 7* (6):1451.
235. Taylor, E. & Hummel, J. (2009). Finding similarity in a model of relational reasoning. *Cognitive Systems Research, 10*, 229–239.
236. Thagard, P., Holyoak, K., Nelson, G., & Gochfeld, D. (1990). Analog retrieval by constraint satisfaction . *Aftificial Intelligence, 46*, 259-310.
237. Treisman, A. (1999). Solutions to the binding problem: Progress through controversy and convergence. *Neuron, 24* (1), 105–110.
238. Turner, M. & Fauconnier, G. (1995). Conceptual integration and formal expression. *Metaphor and Symbolic Activity, 10* (3), 183–204.
239. Tversky, A. (1977). Features of similarity. *Psychological Review, 84*, 327–352.
240. Turing, A. (1950). Computing machinery and intelligence. *Mind, 59*, 433–460.
241. Ullman, S. (1984). Visual routines. *Cognition, 18*, 97–159.
242. Veloso, M. (1994). *Planning and learning by analogical reasoning*. Berlin: Springer-Verlag.
243. Viskontas, I., Morrison, R., Holyoak, K., Hummel, J., & Knowlton, B. (2004). Relational integration, inhibition, and analogical reasoning in older adults. *Psychology and Aging, 19* (4), 581–591.
244. von der Malsburg, C. (1995). Binding in models of perception and brain function. *Current Opinion in Neurobiology, 5* (4), 520–526.
245. Waltz, D.I. (1975). Understanding line drawings of scenes with shadows. In P.Winston (Ed.), *The psychology of computer vision*. McGraw-Hill.

246. Waltz, J., Knowlton, B., Holyoak, K., Boone, K., Mishkin, F., de Menezes Santos, M., Thomas, C., & Miller, B. (1999). A system for relational reasoning in human prefrontal cortex. *Psychological Science, 10* (2), 119–125
247. Wharton, C., Holyoak, K., Downing, P., Lange, T., & Wickens, T. (1991). Retrieval competition in memory for analogies. *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society* (pp. 528–533). Hillsdale, NJ: Erlbaum.
248. Whitney, P. (1998). *The psychology of language*. Boston, MA: Houghton Mifflin.
249. Wilson, W., Halford, G., Gray, B., & Phillips, S. (2001). The STAR-2 model for mapping hierarchically structured analogs. In D. Gentner, K. Holyoak, & B. Kokinov (Eds.) *The analogical mind: Perspectives from cognitive science* (pp. 125–159). Cambridge, MA: MIT Press.
250. Wright, S. Matlen, B., Baym, C. Ferrer, E., & Bunge, S. (2008). Neural Correlates of Fluid Reasoning in Children and Adults. *Frontiers in Human Neuroscience, 1* (8), 1–8.
251. Zareva, N. & Kokinov, B. (2003). Blending of non-similar episodes as a result of analogical mapping with a third one. *Proceedings of the Twenty-Fifth Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum.