BO MORGAN

A SUBSTRATE FOR ACCOUNTABLE LAYERED SYSTEMS

Advisor _____

Joseph A. Paradiso
Professor of Media Arts and Sciences
Massachusetts Institute of Technology

Committee Member _____

Marvin Minsky
Professor of Media Arts and Sciences
Professor of Electrical Engineering and Computer Science
Massachusetts Institute of Technology

Committee Member _____

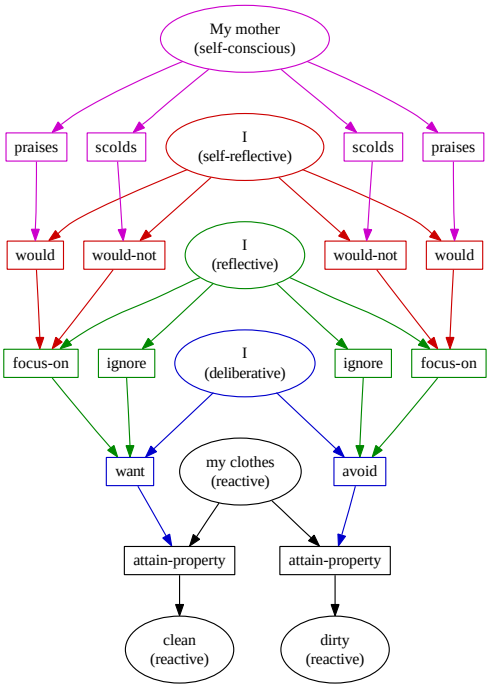
Gerald J. Sussman
Panasonic Professor of Electrical Engineering
Massachusetts Institute of Technology

Committee Member _____

Michael T. Cox
Visiting Associate Research Scientist
University of Maryland

A SUBSTRATE FOR ACCOUNTABLE LAYERED SYSTEMS

BO MORGAN



Ph.D. in the Media Arts and Sciences

Media Laboratory
Massachusetts Institute of Technology

August 2011

Though there be no such thing as Chance in the world;
our ignorance of the real cause of any event
has the same influence on the understanding,
and begets a like species of belief or opinion.

— David Hume

Dedicated to the loving memory of Pushpinder Singh.

1972 – 2006

ABSTRACT

A system built on a layered reflective cognitive architecture presents many novel and difficult software engineering problems. Some of these problems can be ameliorated by erecting the system on a substrate that implicitly supports tracing of results and behavior of the system to the data and through the procedures that produced those results and that behavior. Good traces make the system accountable; it enables the analysis of success and failure, and thus enhances the ability to learn from mistakes.

I have constructed just such a substrate. It provides for general parallelism and concurrency, while supporting the automatic collection of audit trails for all processes, including the processes that analyze audit trails. My system natively supports a Lisp-like language. In such a language, as in machine language, a program is data that can be easily manipulated by a program. This makes it easier for a user or an automatic procedure to read, edit, and write programs as they are debugged.

Here, I build and demonstrate an example of reflective problem solving in a block building toy problem domain. I then apply my approach to a simulation of life in a rigidbody physical environment in order to show scalability to non-trivial problem domains. In my demonstration multiple agents can learn from experience of success or failure or by being explicitly taught by other agents, including the user. In my demonstration I show how procedurally traced memory can be used to assign credit to those deliberative processes that are responsible for the failure, facilitating learning how to better plan for these types of problems in the future.

PUBLICATIONS

Some ideas and figures have appeared previously in the following publications:

Morgan, B.; “Moral Compass: Commonsense Social Reasoning Cognitive Architecture”; http://em-two.net/about; Commonsense Tech Note; MIT Media Lab; 2011 January

Smith, D. and Morgan, B.; "IsisWorld: An open source commonsense simulator for AI researchers"; AAAI 2010 Workshop on Metacognition; 2010 April

Morgan, B.; “A Computational Theory of the Communication of Problem Solving Knowledge between Parents and Children”; PhD Proposal; MIT Media Lab 2010 January

Morgan, B.; “Funk2: A Distributed Processing Language for Reflective Tracing of a Large Critic-Selector Cognitive Architecture”; Proceedings of the Metacognition Workshop at the Third IEEE International Conference on Self-Adaptive and Self-Organizing Systems; San Francisco, California, USA; 2009 September

Morgan, B.; “Learning Commonsense Human-language Descriptions from Temporal and Spatial Sensor-network Data”; Masters Thesis; Massachusetts Institute of Technology; 2006 August

Morgan, B.; “Learning perception lattices to compare generative explanations of human-language stories”; Published Online; Commonsense Tech Note; MIT Media Lab; 2006 July

Morgan, B. and Singh, P.; “Elaborating Sensor Data using Temporal and Spatial Commonsense Reasoning”; International Workshop on Wearable and Implantable Body Sensor Networks (BSN-2006); 2005 November

Morgan, B.; “Experts think together to solve hard problems”; Published Online; Commonsense Tech Note; MIT Media Lab 2005 August

Morgan, B.; “LifeNet Belief Propagation”; Published Online; Commonsense Tech Note; MIT Media Lab; 2004 January

Don't do anything that isn't play.

— Joseph Campbell

ACKNOWLEDGMENTS

I would first like to thank my memory of Push Singh for being a strong memory of a heroic friend and advisor.

I would like to thank my committee: Joe Paradiso for his belief in my ability. Marvin Minsky for consistently providing critical perspective and important problems. Gerry Sussman for loving to program. Mike Cox for leading in the space of meta-cognitive science.

I would like to thank my immediate family: Greg Morgan and Carolyn Spinner for being willing to help me think about anything at any time. Paul Bergman for teaching me how to program. Virginia Barasch for talking about social things. Leaf Morgan for playing Archon.

I'd like to thank those who worked on the project: Dustin Smith for the physical social commonsense kitchen simulation. Radu Raduta for the PID control loops for robot balance and movement. Jon Spaulding for the EEG amplifiers and pasteless electrodes. Gleb Kuznetsov for the first version of a 3D commonsense kitchen critic-selector layout planner. Jing Jian for being so good at finding critical bugs in Funk2 and for writing all of the learned-reactive resources for following recipes in the kitchen. Panupong Pasupat for the error-correcting subgraph isomorphism algorithm and the Winston difference-of-differrence analogy algorithm. Mika Braginsky for the visualization of mental resources, agencies, and layers.

I'd like to thank the following professionals for their encouragement, advice, and inspiration: Walter Bender, Henry Lieberman, Patrick Winston, Whitman Richards, Ed Boyden, Sebastian Seung, Hugh Herr, Ted Selker, and Rebecca Saxe.

I'd like to thank my friends in the Common Sense Computing group: Catherine Havasi, Rob Speer, Hugo Liu, Barbara Barry, Ian Eslick, Jason Alonso, and Dustin Smith.

I would like to thank my popes of GOAM: Scotty Vercoe, Dane Scalise, Kemp Harris, and Dustin Smith.

I would like to thank the researchers of the Mind Machine Project for being an inspiring shooting gallery of high-energy particle AI: Newton Howard, Forrest Green, Kenneth Arnold, Dustin Smith,

Catherine Havasi, Scott Greenwald, Rob Speer, Peter Schmidt-Nielsen, Sue Felshin, David Dalrymple, and Jason Alonso.

I would be remiss if I were not to thank, again in some cases, the following individuals, whom have kept me laughing through the PhD process: Dustin Smith, Grant Kristofek, Mako Hill, Josh Lifton, Mat Laibowitz, Mark Feldmeier, Matt Aldrich, Nan-Wei Gong, Mike Lapinski, Clio Andris, Hannah Perner-Wilson, Jeff Lieberman, David Cranor, and Emöke-Ágnes Horvát.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTINGS

ACRONYMS

GPS General Problem Solver

RMDP Relational Markov Decision Process

FSM Finite State Machine

Part I

INTRODUCTION

1

LEARNING TO ACCOMPLISH GOALS

> Problem-solvers must find relevant data. How does the human mind retrieve what it needs from among so many millions of knowledge items? Different AI systems have attempted to use a variety of different methods for this. Some assign keywords, attributes, or descriptors to each item and then locate data by feature-matching or by using more sophisticated associative data-base methods. Others use graph-matching or analogical case-based adaptation. Yet others try to find relevant information by threading their ways through systematic, usually hierarchical classifications of knowledge—sometimes called "ontologies". But, to me, all such ideas seem deficient because it is not enough to classify items of information simply in terms of the features or structures of those items themselves. This is because we rarely use a representation in an intentional vacuum, but we always have goals—and two objects may seem similar for one purpose but different for another purpose.
>
> — Marvin Minsky

The story of how I have gotten to this point along my current trajectory of AI research is mostly idiosyncratic and not especially interesting. However, only in this story is there reason for my current position. As a young student, I grew up like most, learning that the fundamental basis of my world is a real number domain of space and time. When I spread my arms I thought of real distances and when I remembered the past and projected my future I thought of my life as embedded in a real number line. My first learning algorithms were artificial neural networks, which because of their basis in real numbers were close to my understanding of my physical self and the physical world. I used derivatives and integrals to predict events within space and time. I followed my intuitions from these numerical instincts, which lead me to make probabilistic Gaussian mixture models describing distributions of commonsense narratives over space and time. It was difficult to describe complete logics over these relational numerical distribution spaces. This model was an inference model—nothing more. My goal was to understand processes of deliberation and reflective thinking over these narrative spaces, and this inference model was going to be the basis of my next project. What does a process look like in this narrative space? I would need a more complete logic in order to specify transition functions, in order to simulate more general processes

that could be deliberative. I saw my inference system, although complicated, as a reactive system without much actual thought going on. At this point, I read through many of Marvin Minsky's old papers and discussed my project with him. He was very skeptical that I would make much progress at all if I had probabilistic representations as the fundamental representation in my model. Then, my goal was clearly before me. At this point, I reread almost all of Minsky's papers and his student's PhDs, including Gerry Sussman's work on HACKER, a system that reflectively debugs its own planning system. I came to an epiphany at this point. I saw Minsky's Society of Mind and Emotion Machine as a detailed diagram for an intricate new human programming language. Every single part of the history of AI suddenly fell into place given this overarching goal. The representation for a process that I had been so confused about with my work on probabilistic models suddenly slapped me in the face. I had been staring at it for over twenty years. The representation for a process is the programming language itself. As I approached this new realization cautiously, I became more comfortable in my conclusion through conversations with my close friend and theoretical physicist, Julian Avila. Through these discussions, I realized that although much of physical science had turned to a fundamental understanding of the world as probabilistic, I didn't agree. A probabilistic model must be based on something discrete that is counted, in order to be rational. Now, my task was clear, if I was going to build something like a probabilistic model that counted, I would first need something to count. I abandoned my previous work that treated probability as fundamental and worked on understanding and trying to build Minsky's model, which I saw as a new programming language that "closes the loop" not only once but twice—once for perceiving and acting, and again for reflectively perceiving and debugging the program itself. In this PhD, I describe the resulting learning system that does just this: closes the loop twice.

1.1 CHAPTER OVERVIEW

In this introductory chapter I will give an overview of the problem of reflectively learning to accomplish goals. First, I will develop the assumption that any process that I choose under given constraints can be reflectively monitored and useful information, e.g. beginning and ending times, will be temporarily stored. In other words, I assume that my system natively supports procedural reflection for all processes. Given this assumption, I show in this chapter that the problem of building a system that learns to accomplish goals in its environment becomes a simple loop-less causal structure involving at least four different types of learning. These four types of learning complete a circle of causal learning from goals through actions to physical states and back to goals, but because of time, causality is still in an ordered lattice

structure beginning with goals as the causal roots for all other knowledge.

In later chapters I will introduce experiments I have performed in different physical problem simulations. The first is a simple proof-of-concept demonstration of my reflective learning algorithm in a blocks world problem domain. Also, I show my reflective architecture extended to a larger physical reasoning domain, including multiple AIs learning to cook together in a kitchen environment. In this larger problem space, AIs can communicate high-level procedures specified in a simple programming language that is shared by the AIs, and refers to both mental and physical actions of the AIs.

1.2 THE AI-ENVIRONMENT MODEL

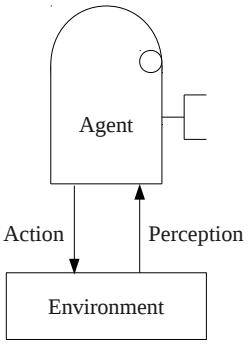


Figure 1: The AI-environment model.

Figure 1 shows the basic AI-environment model. In this model, I make a distinction between the environment and the AI. At any given time, the AI and the environment are each represented as a specific static form of data. Further, these representations change over time, according to a given transfer function. I will treat this system as a deterministic system, although one could imagine adding random variables to the transfer function: the basic theory is the same. It is easier to add randomness to a deterministic theory than the opposite. There are also many benefits to developing a deterministic model with perhaps a pseudorandom aspect because this allows for the repeatability of scientific experiments, for which the model may be used as a metric. The two processes communicate information along two channels: (1) an action channel from the AI to the environment, and (2) a perception channel from the environment to the AI.

1.2.1 *The State-Action Model*

The AI is in an environment, which is in a specific state. My AI performs an action, which can affect the state of the environment. Figure 2 shows a simple Finite State Machine (FSM) state-action model, which has two states for the environment and two actions

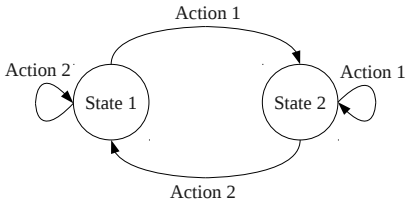


Figure 2: The state action model. Two states are represented by nodes and two actions are represented by edges from each of the two states.

for the AI to perform in each of these states. The state-action model is a simple model for how actions map to changes in the state of the environment.

1.2.2 *A Multiple AI-Environment Model*

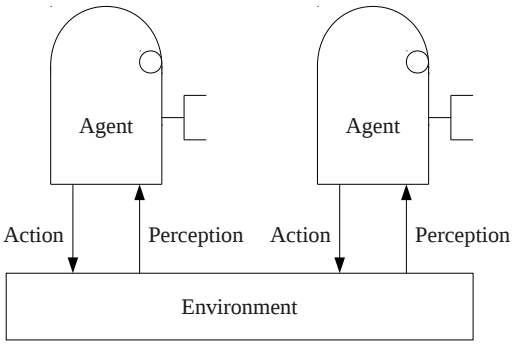


Figure 3: The multiple AI environment model.

In order to model social intelligence, we introduce the multiple AI environment model shown in Figure 3.

1.2.3 *AI Process Communication*

Because AI processes can only directly act on and perceive the environment, all communication between AI processes must occur through the environment process. We assume that AIs can communicate some form of symbolic knowledge structure in the absence of noise. These representations are used to communicate processes from one AI to another. Specific examples of such a process representation that we have used in our experiments are described in more detail in Sections 4.3.2–4.3.3.

1.2.4 *A Relational State Representation*

See Figure 4.

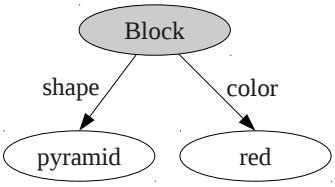


Figure 4: Frame-based relational representation.

1.2.5 *Introducing Reflection Early in the Process*

Now, I have introduced my problem as being divided into at least two processes, at least one AI process and an environment process. Further, I have introduced how a basic process may be thought of as an FSM. At this point, I introduce a model for keeping track of the changes in a computational process: reflective knowledge. I purposefully make this assumption before I define the details of the AI model process because, in my approach, I would like to reflectively reason about potentially any aspect of this AI process.

1.2.6 *Reflective Knowledge*

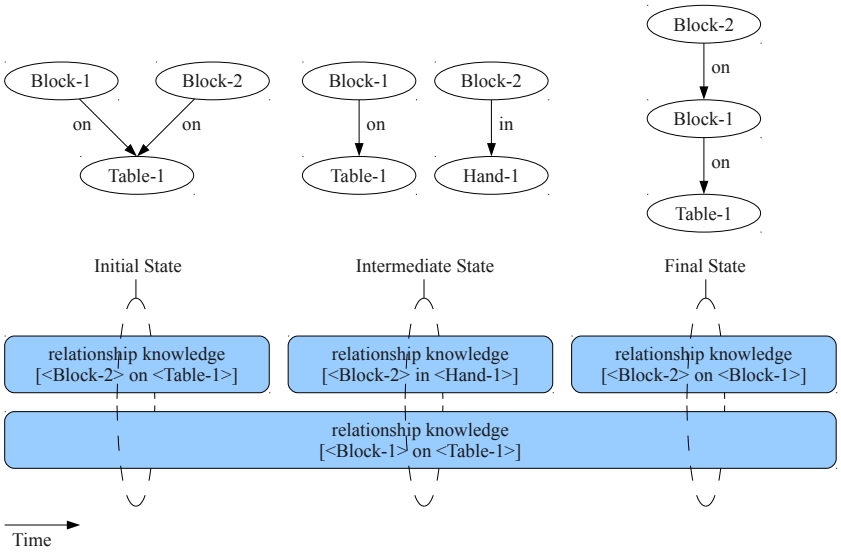


Figure 5: A reflective event representation shows the changes in a labeled graph.

While the term meta-knowledge is used to describe the very general idea of knowledge about knowledge, I use the term reflective knowledge to refer to the specific type of meta-knowledge that refers to knowledge about the changes to a knowledge structure. If I keep track of the changes to a knowledge structure, I can later integrate these changes in order to obtain an equivalent

form of that knowledge structure as it was at any point in the past.[1]

1.2.7 *Frame Perceptions*

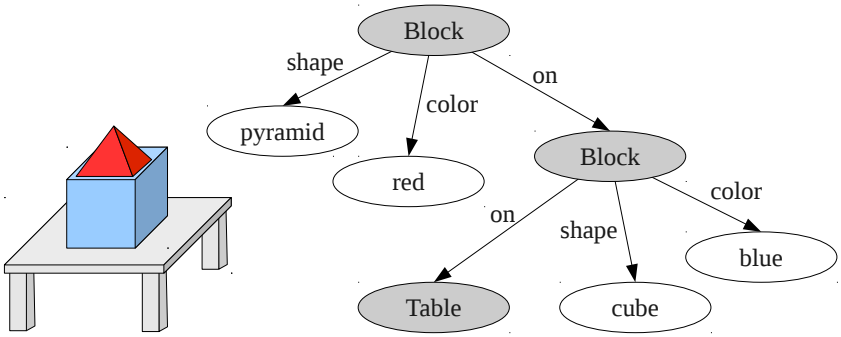


Figure 6: Collections of frames used as perceptual input to AI.

See Figure 6.

1.2.8 *Partially Observable State Model*

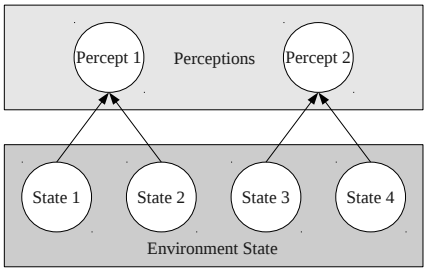


Figure 7: The partially observable model.

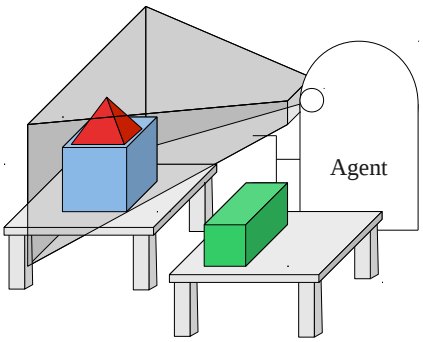


Figure 8: The state of the environment is only partially observable.

The AI process does not have complete access to the state of its environment. The AI's perceptual stream of information

1 See Section A.3 for a discussion of more realistic models of computation, including multiple-core and cluster models.

depends on the state of the environment, but it is a function of the environment and not the actual state of the environment. In other words, the perceptual state that is communicated from the environment to the AI is an injective function mapping the environment to the perception of the AI. See Figures 7 and 8 for two examples of partially observable environments.

1.2.9 *AI Abstract Physical Model*

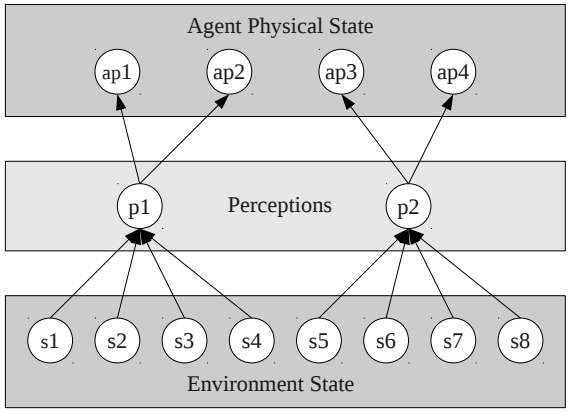


Figure 9: The AI has an abstract physical model of the environment.

See Figure 9.

1.2.10 *A Physical Goal Representation*

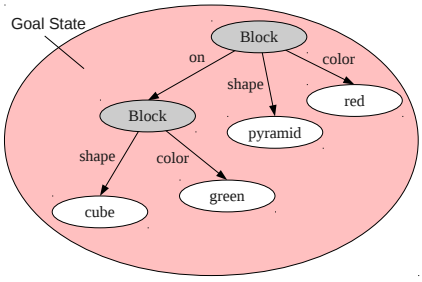


Figure 10: A physical goal representation is a structural relationship that may or may not currently exist within the physical knowledge base.

See Figure 10.

1.3 THE REFLECTIVE LEARNING CYCLE

I have now introduced the definitions for my relational state space, along with how reflective tracing, given certain assumptions, allows the creation of reflective event representations of the action resources within the AI's mind.

In this section, I will describe how these representations and reflective event representations can be used to reflectively learn the different types of knowledge used for planning.

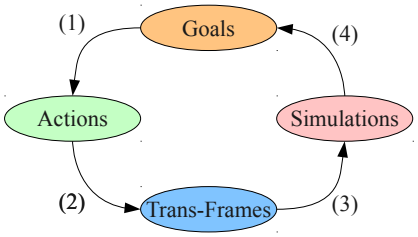


Figure 11: Learning to plan four step cycle.

Figure 11 shows the abstract four-step learning to plan cycle. Planning is an arbitrarily complicated process, but with these four steps I have created basic semantic divisions between the types of knowledge involved in reflectively learning to plan. These basic divisions in the learning process can be summarized as:

1. Changing goal knowledge initiates causal learning.
2. Physical precondition concept for goal occurrence is refined.
3. Physical transition concepts for simulating action resource are refined.
4. Physical concept for goal occurrence is refined.

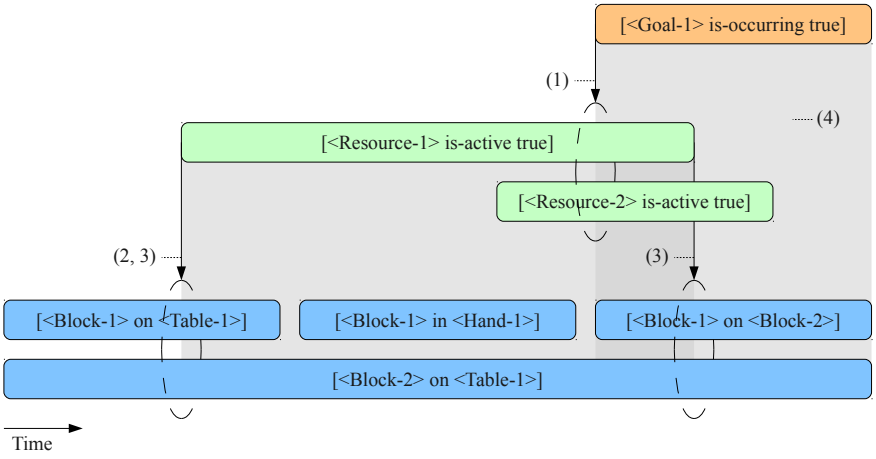


Figure 12: The reflective learning cycle is a way to learn knowledge used to plan toward goals that maintains the causal dependency structure of the knowledge.

A system without a goal has no reason to learn the effects of its actions. Figure 12 shows how this process of learning causal models relating reflective states to physical states is a fundamentally goal-driven process. We show how this goal prediction process leads to at least a four step causal chain of knowledge learning. It is important to point out that although the learning process

is a cycle over time, this ends up creating a causal structure for knowledge dependency without loops—a critical property in general to maintain for any causal representation.

These four goal-oriented cyclical learning steps build four of the major types of knowledge that my goal-oriented inference and planning system uses. Let us now briefly introduce these four types of knowledge before going into a more thorough explanation of the utility of each.

1. Goal-oriented action hypotheses: focuses and constrains the initial action learning search.
2. Action physical precondition goal occurrence hypotheses: allows predicting whether or not an action will accomplish the given goal under a given physical precondition.
3. Action physical precondition trans-frame hypotheses: allows for physically simulating the given action in a given physical state.
4. Physical hypotheses for predicting goal occurrence: allows for predicting if a given physical state implies that the goal is occurring.

Note that there are two different concepts of time that are displayed in Figure 12. It is important to not confuse these. The first kind of time is represented by the sequence of change events in the knowledge structure that was reflectively traced in order to generate the temporal event representation shown in the figure. This first time is represented and labelled as the horizontal axis in the figure. The second kind of time represented in the figure is represented by the enumeration of the four learning steps. This learning process is a reflective learning process because it learns from reflective knowledge gathered from tracing processes.

In the following four sections, we will describe the utility of each of these four different types of learned knowledge in more detail.

1.3.1 *Causal Action Hypotheses*

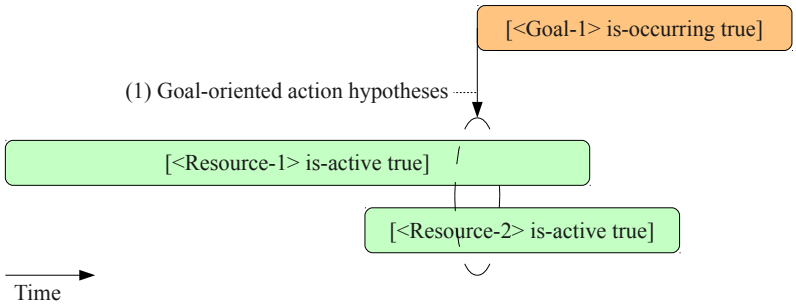


Figure 13: Goal-oriented action hypothesis knowledge.

learning goal: focusing learning

Figure 13 shows the first step of the reflective learning cycle. This first step of the learning cycle is caused reflectively by the goal event beginning to exist. When a goal event comes into existence, a reflective process monitoring this goal knowledge makes a list of potential actions that might be useful for causally predicting this goal condition in the future. In the figure we can see that a simple interval-point intersection operation between the beginning point of the goal event and any action event interval is enough to generate two actions as initial causal hypotheses. See Section 4.3.5 for details on the techniques that I used for generating goal-oriented action hypotheses in my system.

1.3.2 *Action Goal Occurrence Hypotheses*

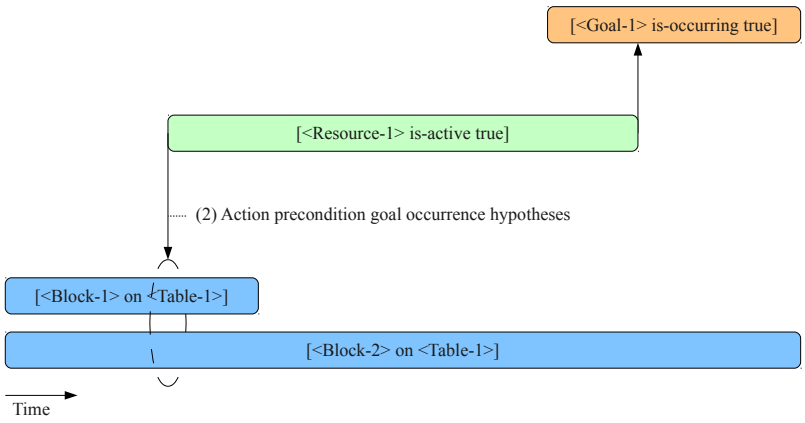


Figure 14: Action precondition goal occurrence hypotheses can be learned from these reflective representations.

Action precondition goal occurrence hypotheses are a form of knowledge that is learned in order to predict whether or not a goal will be occurring if a given action is taken in a given conceptual category of a given abstraction of the state space. Figure 14 shows the reflective event representations that can be used to learn this type of knowledge.

1.3.3 *Action Trans-Frame Hypotheses*

See Figure 15.

See Figure 16.

1.3.4 *Physical Goal Occurrence Hypotheses*

See Figure 17.

1.4 THE PLANNING PROCESS

I have now introduced my definition of reflective event representations,

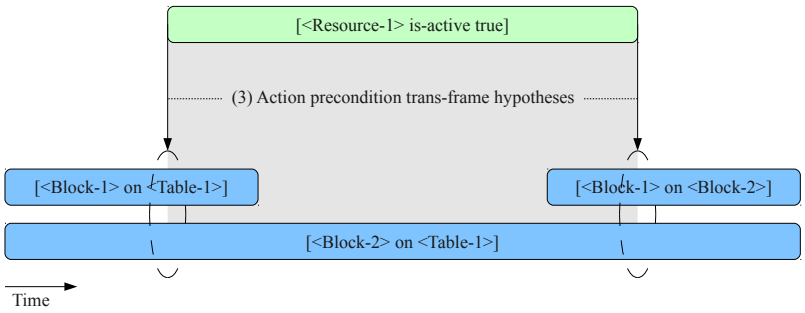


Figure 15: Action precondition trans-frame hypotheses.

Remove Events	Add Events
[<Goal-1> is-occurring false] [<Block-1> on <Table-1>]	[<Goal-1> is-occurring true] [<Block-1> on <Block-2>]

Figure 16: A trans-frame for an event is a list of differences in knowledge between the beginning and ending of the event.

1.4.1 *Partially-Ordered Plan Representation*

The partially-ordered plan representation allows a partially-ordered temporal organization for a set of commands. A plan with a branch-and-join control structure is shown in Figure 18.

1.4.2 *Inferring the Effects of a Plan*

The planning process requires an inference algorithm to infer future and past states based on cause-effect relationships between reflective knowledge and physical knowledge. Figure 19 shows a possible inference for a given plan. There are many ways to plan and the specific planning system that I have implemented for my experiments is discussed further in Section 4.3.4.

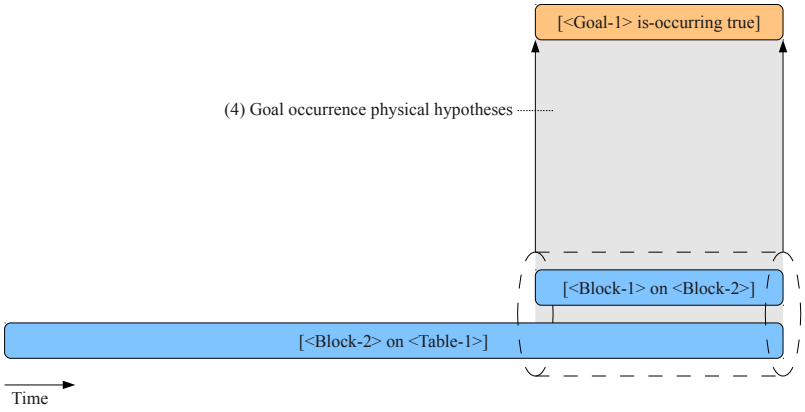


Figure 17: Goal occurrence physical hypotheses.

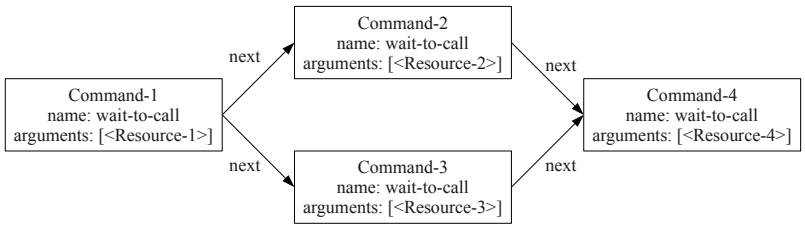


Figure 18: A partially-ordered plan representation containing serial and parallel components.

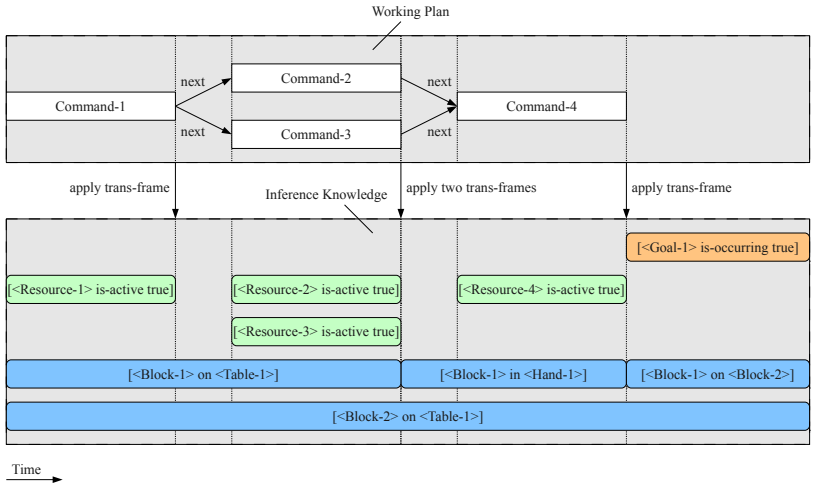


Figure 19: Inferring the effects of a plan.

1.4.3 *Planning Machine Operations*

A few operations for manupulating partially-ordered plans are shown in Figure 20.

1.5 LAYERING REFLECTIVE LEARNING

I have now introduced how reflective representations can be used for learning how to plan. In this section, I will show how using the same reflective approach to learning to plan toward goals can be used to build another layer of reflective control on top of our planner. This next layer of reflective control learns to accomplish reflective goals for the plan domain of the first-level planner.

1.5.1 *Planning Machine Reflective Knowledge*

The same reflective learning to plan cycle can be applied to the actions of the planning process itself, introducing an idea of reflective goals. The basic four steps of the reflective learning to plan cycle are very similar when applied to the deliberate planning process:

1. Changing reflective goal knowledge initiates causal learning.

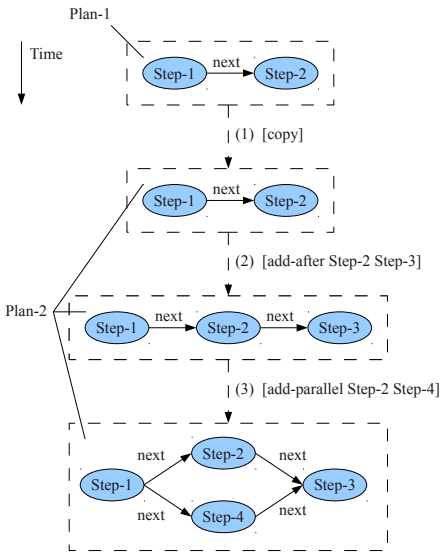


Figure 20: A few planning machine operations.

2. Deliberative precondition concept for reflective goal achievement is refined.

3. Deliberative transition concepts for simulating deliberative action resource are refined.

4. Deliberative concept for reflective goal achievement is refined.

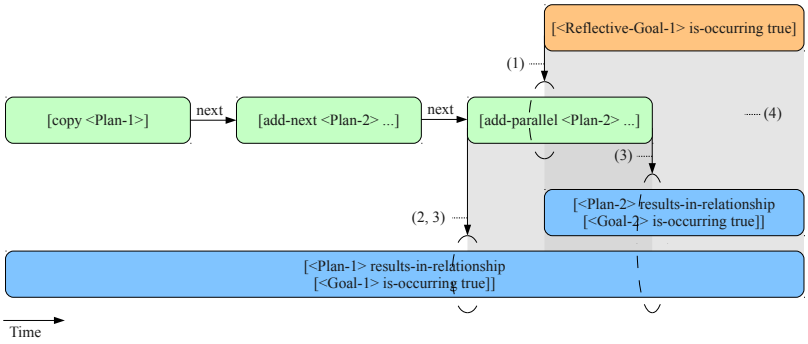


Figure 21: Planning machine reflective knowledge.

See Figure 21.

2 PROBLEMS TO SOLVE

2.1 BUILD A REFLECTIVE KNOWLEDGE SUBSTRATE

The assumption that I introduced in Section 1.2.5, "Introducing Reflection Early in the Process", requires that changes in my knowledge representation can be traced and compiled into other representations, such as the reflective event representations used in learning to accomplish goals.

2.1.1 *Automatic Collection of Audit Trails for All Processes*

Audit trails must be collected so that after the fact the events that any process is resposible for can be used for many types of reflective control purposes, such as learning to plan. Although a system that keeps track of everything that it does would support reflection, the audit trail recorder in a system that does not grow indefinitely in memory consumption would have options for focusing the collection of audit trails, while also allowing for their garbage collection when they are no longer needed by another reflective process.

2.1.2 *General Parallelism and Concurrency*

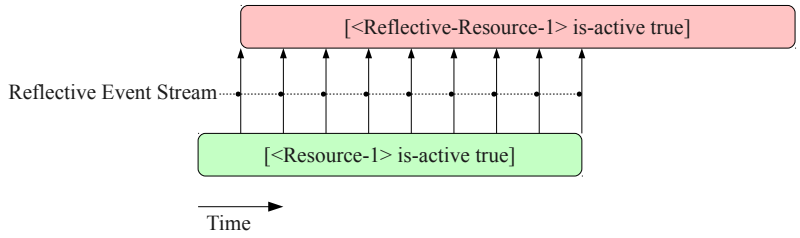


Figure 22: Concurrent parallel reflection efficiency.

Because the procedurally reflective programming paradigm focuses on event streams of changes to memory, there is an inherent ability to express procedurally reflective algorithms in a streaming parallel language.

Figure 22 shows how a simple process can be reflectively monitored without slowing down the fundamental process by more than the constant time factor necessary for generating the event stream. Many different parallel reflective processes can be reflectively processed concurrently without any additional slowdown in the primary problem solving process.

2.1.3 *Program as Data*

The ability for a process to manipulate a program as data makes it easier for that process or a user to read, edit, and write programs as they are debugged. The ability to change the functionality of a program at run-time is a key component to creating an adaptive problem solver. Because of this, a programming language with a run-time compiler is very useful for building these types of adaptive systems that learn to debug and re-program their own subprograms. For purely the reason of developing problem solving learning algorithms, it is critical for a reflective substrate to be able to treat its own programs as data. A compiler is a simple example of a program that must treat a program as data. The planning process is a more general type of compiling problem that also requires learning the effects of primitive actions. So, a system that is planning or learning how to plan, must have some sort of representation of processes that it is interpretting as plans. Compiling simply makes this interpretation step in the planner more efficient, but this is a very convenient efficiency when building practically useful learning systems.

2.2 LAYERED REFLECTIVE PROBLEM SOLVING

2.2.1 *Analogy between Physical Goals and Planning Goals*

2.3 LEARNING BY CREDIT ASSIGNMENT

2.3.1 *Use Reflective Representations for Better Models of Learning*

2.3.2 *Tracing Knowledge Provenance for Credit Assignment of Success or Failure*

3

THEORY AND ALTERNATIVES

3.1 CATEGORY LEARNING

$$h_j \geqslant_g h_k \tag{3.1}$$

$$(\forall x \in X)[(h_k(x) = 1) \rightarrow (h_j(x) = 1)] \tag{3.2}$$

$$\tag{3.3}$$

3.2 TEMPORAL DIFFERENCE LEARNING

$$\overline{V}^{\pi}(s) = E_{\pi}\{R_t | s_t = s\} \tag{3.4}$$

$$= E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s\right\} \tag{3.5}$$

$$= E_{\pi}\left\{R_{t+t} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} | s_t = s\right\} \tag{3.6}$$

$$= E_{\pi}\{R_{t+t} + \gamma V^{\pi}(s_{t+1}) | s_t = s\} \tag{3.7}$$

$$\tag{3.8}$$

3.3 TWO POPULAR APPROACHES TO MODELLING INTELLIGENCE

Recently, there have been two directions of research with the goal of building a machine that explains intelligent human behavior. The first approach is to build a baby-machine that learns from scratch to accomplish goals through interactions with its environment. The second approach is to give the machine an abundance of knowledge that represents correct behavior.

Each of these solutions has benefits and drawbacks. The baby-machine approach is good for dealing with novel problems, but these problems are necessarily simple because complex problems require a lot of background knowledge. The data abundance approach deals well with complicated problems requiring a lot of background knowledge, but fails to adapt to changing environments, for which the algorithm has not already been trained.

3.3.1 *Adaptability in Complex Environments*

We would like to build intelligent machines that are able to perform household tasks, such as cooking, cleaning, and doing the laundry, but these tasks seem insurmountably complex, containing organically unpredictable events. We would like our machines to expertly handle these extremely complicated problems, and we would also like them to adapt to learn in unexpected or novel

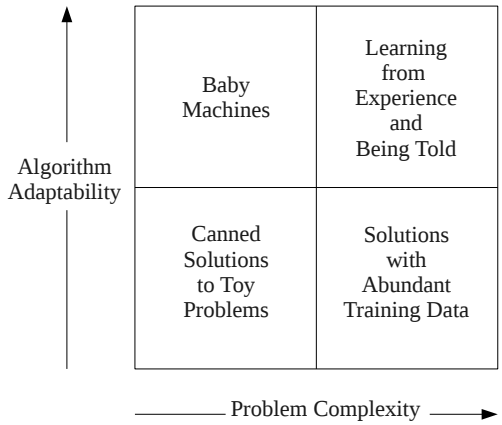


Figure 23: Problem complexity versus algorithm adaptability.

situations. One popular approach to building a machine that performs complicated tasks is to give the machine a large training dataset that details every possible situation that the machine may find itself within, along with the correct action in that situation. This is the so-called "supervised" learning approach. These algorithms do not adapt to novel situations well, and collecting these datasets is often impossible for many problems, such as cooking and cleaning because it is too difficult to enumerate all possible situations, in which the machine may find itself. Also, if the machine is cooking a meal, we would like to be able to explain an idea for a new recipe to the machine, or to perhaps be a partner in discovering new recipes, or we may simply want to explain to the machine that a guest has a specific allergy to walnuts, making that ingredient an exception for this meal but not others. Figure 23 shows how problem complexity and algorithm adaptability can be thought of as a two-dimensional space into which different algorithmic approaches can be used as solutions.

3.3.2 *The Abundant Data Approach*

There have been many approaches to modelling complex forms of reasoning by collecting large amounts of knowledge that describes correct or acceptable behavior in a domain. For example, there are examples of complex multi-AI commonsense simulation environments collects thousands of examples of users interacting in a complicated object-oriented social simulation (Orkin & Roy 2009), (Orkin et al. 2010). These systems have complicated domains, but these projects do not attempt to build AIs that attempt to accomplish goals. Instead, these systems are inference systems that simply try to reproduce typical behavior, rather than goal-directed behavior.

There are many commonsense reasoning systems that do not interact with simulation environments at all, but which attempt to demonstrate commonsense reasoning by being told large amounts of knowledge. The Cyc project is one large such project

that has been told large amounts of logical knowledge (Lenat et al. 1990). There is also effort directed toward populating Cyc with knowledge automatically gathered from the web (Matuszek et al. 2005). The OpenMind project (Singh et al. 2002) is a project that gathers large amounts of approximately correct commonsense knowledge from people online. The OpenMind knowledge has been turned into many inference systems that can compare and generate new commonsense knowledge (Liu & Singh 2004*b*,*a*, Speer et al. 2009).

3.3.3 *The Common Sense Reasoning Problem Domain*

Common sense is the set of common reasoning abilities shared by most people in a given social group. Another way to say this is that common sense is the set of reasoning abilities that one would assume of a typical person that they meet for the first time and know nothing about. For example, most people have a naive theory of physics, so you would expect someone to know that things fall when they are not supported and liquids flow or are absorbed unless they are in a container. Common sense relies on a lot of knowledge that is assumed that most everyone knows.

Building a machine that demonstrates common sense reasoning is a long-standing goal of the field of artificial intelligence. One of the difficulties in developing algorithms for dealing with a common sense reasoning domain is that the algorithm needs a lot of background knowledge about a given domain before it can answer even simple questions about it. However, this knowledge is often only true in very specific situations and has many exceptional cases. For example, the knowledge that most birds can fly is generally true, but we also know that many birds are flightless, such as penguins, ostriches, and road runners. Also, we have knowledge about the typical behavior of objects; for example, we know that refrigerators keep things cold, but we also reason efficiently about exceptional cases, such as when the refrigerator is not plugged in, or when the power goes out.

3.3.4 *Representations for Common Sense Reasoning*

There have been many approaches to artificial intelligence that use first-order logic as a representation for these types of knowledge and their exceptions, but these systems become cumbersome in their inability to express "fuzzy" sorts of relationships, such as when the knowledge is applicable, for example the modifiers, "most of the time", "usually", and "almost never", are difficult to express in first-order logic. When we have a lot of knowledge, we need ways to keep track of in which situations this knowledge is useful. This is a form of "meta-knowledge", or knowledge about knowledge. Meta-knowledge about first-order logic cannot be expressed in first-order logic, so another type of representation

is required for this type of knowledge. Therefore, we need other ways to represent our knowledge in addition to logic.

> "Nonetheless, theorem proving is in the worst case an intractable problem, even with no variables or unification, so no algorithm is going to work on the problem all the time. In this respect, theorem proving, for all its superficial formalism, is a lot like other branches of AI. Where a theorist views a problem as solved if he has found an efficient algorithm or a discouraging lower bound, an AI researcher is often happy to find an algorithm that seems to work on some interesting problems, even though he doesn't really know the bounds of what it can do. Exploration will reveal the extent of its powers-each time it solves one more interesting problem something has been gained." — Drew McDermott

3.4 COMPARABLE COGNITIVE ARCHITECTURES

EM-ONE, Cyc, Icarus, ACT-r, Soar, and Prodigy are comparable cognitive architectures to the one that I have built.

3.4.1 *The EM-ONE Cognitive Architecture*

I worked with Pushpinder Singh from 1999 to 2006 on the first version of the Emotion Machine architecture, EM-ONE (Singh 2005). EM-ONE was an example of a reflective control system that used commonsense stories in order to reason about social problem solving. Morgan (2009) discusses a number of things to learn from the EM-ONE architecture that have informed my current approach.

Push and I have discussed that one weakness in the EM-ONE system is its reliance on tracing only the declarative prolog statements, among other necessary but untraced procedural code. Although EM-ONE contained a large amount of procedural knowledge, none of the effects of this procedural knowledge could be debugged reflectively. Toward solving this problem, I have based my approach on a memory layer that can trace the provenance of select memory events.

3.4.2 *Icarus*

Icarus is a cognitive architecture that supports a form of far transfer learning (Könik et al. 2009). The Icarus system allows for a goal-directed structure mapping learning process. These structure mappings that are learned in this goal-directed way, can be useful forms of analogy that enables "far" transfer in a cognitive system. The Icarus work builds upon a previous model for ana-

logical transfer learning between symbolic relational structures (Gentner 1983).

Because my knowledge representation is relational and symbolic, I see Icarus' approaches to far transfer learning as highly compatible with my architecture's ability to support multiple knowledge domains for a single goal-oriented problem solving AI. I am also interested in a technique of using differences of differences (Winston 1970) as an alternative to the structure mapping form of analogical transfer.

The Icarus system has also been applied to moral reasoning tasks in order to build a theory of moral reasoning in humans (Iba & Langley 2011). I see similar applications of my architecture to these social reasoning domains, but my planned approach to this domain is based on two more layers of reflective control than I have currently implemented (Morgan 2011).

3.4.3 *Computational Models of Cognition about Cognition*

Cox (2005) gives a good overview of the cognitive sciences that deal with the problem of thinking about thinking, or metacognition.

3.4.4 *Shades of Belief as Debugging Meta-Knowledge*

Stein & Barnden (1995) applies a prototype system to perform reflective case-based reasoning over shades of beliefs in knowledge.

3.5 BOUNDED RATIONALITY

There is an approach of economics and game theory that is called bounded rationality (Simon 1972). These models deal with the time constraints of not only acting efficiently in a domain but also in optimizing the planning actions involved in accomplishing goals. This approach assumes that there is an absolute numerical reward value for accomplishing goals. In the sense that absolute values for all of the goals of a system are seldom known in practice, the bounded rationality approach is limited in a general sense similar to the simpler reinforcement learning approach.

3.5.1 *Feedback Control Model for Accomplishing a Single Goal*

Now that we have discussed the basic model of learning from experience what good goal states may be from rewards, let us consider the representations for the state space of the perceptions and actions of my model. Control theory has given us many useful models for AIs that control continuous environments. For example, Figure 24 shows a simple difference feedback control circuit that is used in simple linear control systems. The system

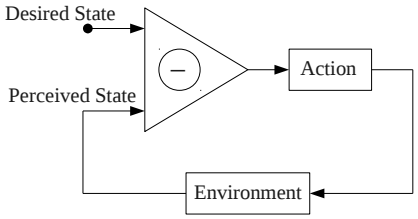


Figure 24: The feedback control model for accomplishing a single goal.

is given a desired state, there is a difference device that calculates the difference between the actual perceived value from the environment, and the control system then executes an action based on that difference, which affects the environment. The result in such a negative feedback loop is that the AI's perception of the environment is closer to the desired state.

3.5.2 *Means-End Analysis*

In 1959, Newell, Shaw, and Simon published a report on a means-end analysis model that was designed to solve any symbolically represented problem (Newell et al. 1959). Their system was called the General Problem Solver (GPS), and worked by being able to work with relational representations of current and desired states. The AI had a catalogue of differences between states that it knew how to minimize. The system worked by finding the largest difference and executing the associated method for reducing this difference. This work has grown into the Soar model (Newell 1990) for better solving symbolic planning problems, and dealing with impasses for when the planning search runs out of options.

3.5.3 *Difference-Engine Model for Accomplishing Multiple Goals*

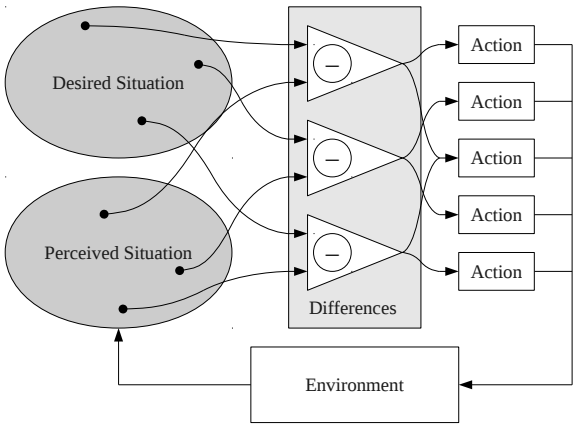


Figure 25: The difference engine model for accomplishing multiple goals.

(Minsky 1988, p. 78)

3.6 PLANNING

The real-time control aspect of my reflective control problem makes it so that most planners and schedulers fail because they do not consider their own run-time as a control problem, ad nauseam, which is my proposed path around the general unconstrained logical reasoning problem. That said, there are a few modern advancements in combining traditional planning techniques into larger learning systems that confront the complexities introduced by acting in a domain. Of these planning or scheduling systems that are incorporated into systems that act in domains, in which they have incomplete domain knowledge, there are few systems that consider the real-time control aspects of not only the domain, but the reasoning processes for acting in that domain as well.

3.6.1 *Planning and Acting in Incomplete Domains*

(**?**)

3.6.2 *Assuming a Correct Model of Environment*

There are many types of processes that make plans for accomplishing goals, which make plans assuming a model of how actions theoretically affect the environment. These processes are called planners when they create a representation for how actions should be performed, potentially including temporal ordering constraints.

Planners are a specific part of a complete learning system, but the primary function of a planner is to find a theoretical solution to a given problem. This theoretical solution, or plan, may be executed and may fail or succeed, in accordance with the initial intentions for executing the plan, the initial intentions for imagining the plan, or any other intentions. If the plan fails, then we may find something to be modified in my knowledge in order to help us in avoiding this failure next time. The planning process is a small part of the complete closed-loop learning algorithm that learns to plan from experience with the environment and other AIs.

If we are thinking about the temporal constraints of the problem solving process itself, then we need to consider a reflective approach to this control problem.

, (1) the model of the cause-effect relationship between actions and the world, (2) the model of cause-effect relationships between planning actions and the creation of successful plans.

3.6.3 *Declarative Programming, Logical Reasoning*

3.7 MACHINE LEARNING

One encompassing goal of the field of machine learning is to develop systems that can accomplish goals.

For example, a Markov Decision Process contains a transfer function, which is basically a combinational device.

Of course, this combinational device would get complicated if I added probability.

3.7.1 *Why Did I Forget to Include Probability in my Theory?*

3.7.2 *The Reinforcement Learning Model*

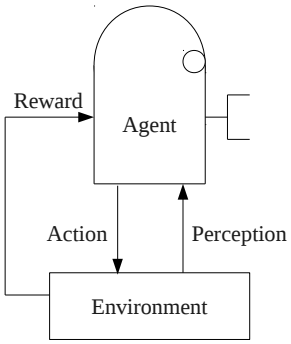


Figure 26: The reinforcement learning model.

Figure 26 shows the basic reinforcement learning model. This model is an AI environment model, but there is an extra information channel from the environment to the AI, which communicates a numerical reward signal. We can now say that the AI has a learning problem. The AI must learn what actions to execute in order to gather the most reward.

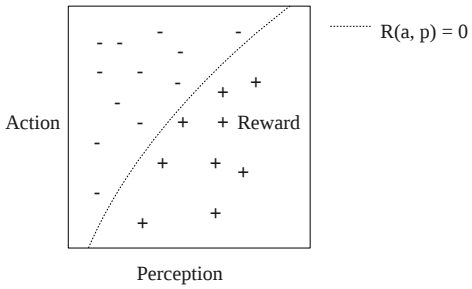


Figure 27: Categorizing perceptions and actions based on goals.

Once we have a basic reinforcement learning algorithm, we can approach this learning problem as a function approximation problem. In other words, we can try to learn what parts of the perception and action space have more or less reward. Figure 27

shows a diagram of this state space with the zero crossing of an approximation of the reward plotted.

3.7.3 *Finding a Good Policy for Gathering Rewards*

Learning an approximation of what parts of a state space are good or bad, based on reward, is not all that is needed to determine what actions the AI should perform. The AI wants to gather the most rewards over time. A simple way to formalize this problem is to learn a policy that determines what action should be executed for every part of the state space, based on some sort of summation of rewards over time. There have a been a number of ways of formalizing this summation process as finite or infinite horizon problems (Sutton & Barto 1998). Dynamic programming can be used for finding an optimal or an approximately optimal policy (Bertsekas 1995).

3.7.4 *Categorizing Perceptions and Actions based on Goals*

One problem with the reinforcement learning approach is that the only representation of success or failure is a single number, the reward. The basic reinforcement learning problem has been defined for finite propositional state spaces.

A representation called Relational Markov Decision Process (RMDP) has been proposed (Guestrin et al. 2003) in order to extend reinforcement learning to larger relational problem domains, but this method only focuses on an object-oriented reward that does not have any global feedback about the overall value of the situation.

3.8 PHILOSOPHY

3.8.1 *The Objective Modelling Assumption*

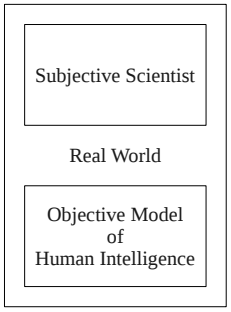


Figure 28: The objective-subjective modelling assumption.

We assume that the phenomenon that we are trying to model, namely human intelligence, is an objective process that we can describe. This is the objective-subjective philosophical assumption that is inherent in any objective scientific hypothesis. We

make this assumption in order to avoid logical problems of circular causality that occur when trying to find a non-objective description of reflective thinking. Figure 28 shows how, given the objective assumption, the subjective scientist is part of the real world, while she is studying an objective phenomenon. Given the objective-subjective assumption, it would be a grave mistake to confuse an objective model for reality itself.

3.8.2 *Being, Time, and the Verb-Gerund Relationship*

3.8.3 *The intensional stance*

3.8.4 *Reflective Representations*

(Perner 1991)

3.9 COGNITIVE SCIENCE

3.9.1 *The Development of Self-Conscious Emotions*

Between the ages of 1-3 years old, children display primary emotions, such as joy, disappointment, and surprise. These emotional processes have been hypothesized to be related to the process of failing or succeeding to accomplish a goal. Around age 4, children begin to display emotions that involve the self, such as guilt and shame. It has been hypothesized that these emotions relate to another person's evaluation of the child's goals as good or bad.

We approach modelling this developmental process by applying Marvin Minsky's theory of the child-imprimer relationship. According to Minsky's theory, at a young age, a human child becomes attached to a person that functions as a teacher. The imprimer could be a parent or a caregiver or another person in the child's life, but the function of the imprimer is to provide feedback to the child in terms of what goals are good or bad for the child to pursue.

3.9.2 *Simulation Theory of Mind versus Theory Theory of Mind*

3.9.3 *Emotion or affect versus goal-oriented cognition*

3.9.4 *Embarrassment, Guilt, and Shame*

3.10 NEUROSCIENCE

3.10.1 *Neural Correlates of Consciousness*

3.10.2 *Learning by Positive and Negative Reinforcement*

3.11 THE LAYERED "MODEL-6" THEORY FOR ORGANIZING REFLECTIVE MODELS

Because the many fields of human cognitive modelling are so disparate it is difficult to see them as a whole. I am inspired by a layered organizational scheme first presented in Minsky (2006). The six layers in this "Model-6" organizational scheme are named as follows:

1. Instinctive Reactions
2. Learned Reactions
3. Deliberative Thinking
4. Reflective Thinking
5. Self-Reflective Thinking
6. Self-Conscious Reflection

I hope that the reader will see that this is a useful way of organizing the massive amount of work that has gone into the computational modelling of abstract human thought processes.

3.11.1 *Instinctive Reactions—Models of Insulation and Interaction*

> *We usually like to think in positive terms about how various parts of systems interact. But to do that, we must first have good ideas about which aspects of a system do* not *interact—since otherwise there would be too many possibilities to consider. In other words, we have to understand* insulations *before we can comprehend interactions. To put this in a stronger form:* No complicated society would actually work if it really depended on interactions among most of its parts*. This is because any such system would be disabled by virtually any distortion, injury, or environmental fluctuation. Nor could any such society evolve in the first place.*

— Minsky (1988)

3.11.1.1 *Models of Failure in Distributed Systems*

Attiya & Welch (2004) discuss algorithms for tolerating a few types of failures in distributed systems, ranging from silent crashes to arbitrary Byzantine failures, where parts of an algorithm perform in completely uncontrolled and unpredictable ways. An algorithm for Byzantine tolerance is presented that is based upon building layers of first abstracting controllers that turn the Byzantine problem into a simpler *identical Byzantine* problem. The identical Byzantine problem is then abstracted to the simpler *omission* problem, where a processor simply omits some messages from being transmitted. The omission problem is then handled in a more abstract layer of the algorithm that when detecting an omission, crashes itself, resulting in a *crash* problem. Crash problems can be dealt in different ways for different algorithms, the simplest being the consensus algorithm. This gives us a basic vocabulary of the most fundamental and basic distributed process problems.

Bertsekas (1982) discuss a basic model for distributed dynamic programming that is organized around a distributed iterative approximation algorithm that computes a function mapping from states to values, $J(x) \in \Re$. His model divides the world into sets of states, $S_1 \cup S_2 \cup \cdots S_n = S$, that depend on neighboring sets of states in order to compute the approximation, J^t, of the optimal value function, J^*. This fits with the idea that commonsense knowledge in an artificial intelligence program should be organized according to what goals it is important to solve. Having a basic optimal control theory of why this is a fundamentally sound organization principle is encouraging: perhaps organizing our knowledge by the values of goal states could also apply in the context of abstract reinforcement learning. Note that the information theoretic organizational approaches such as the inductive logic approach used by relational reinforcement learning do not organize their knowledge according to which goal values are important for calculating which other goal values.

3.11.1.2 *Models of Uncontrolled Distributed Statistical Inference*

Recently, statistical propositional logic has become popular along with graph-based inference techniques for calculating both the most likely propositional state as well as the probability of any single propositional variable. Statistical models are often represented in graphical form that can be easily distributed and computed by sparse matrix multiplication algorithms. Given a statistical model, it is common to use the MAXENT algorithm to approximate the most likely state of such a model. Jaynes (1982) discuss when this may or may not be appropriate. Murphy et al. (1999) discuss Pearl's "loopy" belief propagation algorithm which is an unreliable but efficient inference tool. Yedidia et al. (2005) generalizes this distributed form of statistical inference to hierarchical methods, explaining in the process that the forward-

backward algorithm for hidden markov models, the Viterbi algorithm, Gallager's sum-product algorithm, the "turbo-decoding" algorithm, Pearl's "belief propagation" algorithm for inference on Bayesian networks, the "Kalman filter" for signal processing, and the "transfer matrix" approach in statistical mechanics are all of the same generalized form of statistical inference.

These statistical tools are the basic inference tools that claim nothing about their own applicability, and have no recourse for response for their own failures. These statistical inference tools also know nothing about the goals which they are supposed to help solve. In order for an algorithm to intelligently accomplish goals, it needs to deliberate or plan its usage of these most basic tools.

3.11.1.3 *Models of Distributed Intelligence*

Maes & Brooks (1990) discuss learning to coordinate behavior, an argument for robust robot control using multiple ways of reasoning in a layered architecture.

3.11.1.4 *Models of Computational Reflection*

Maes (1987) gives a great overview of "computational reflection," which is basically keeping track of information about the execution of a program, so that other processes can use that information to better control the program. My approach to computational reflection is what they categorize as a "frame-based" approach, described first in Minsky (1975) and first implemented in Roberts & Goldstein (1977). Maes (1987) describe the primary benefits of the frame-based approach as the following:

- it helps the user cope with the complexity of a large system by providing documentation, history, and explanation facilities,
- it keeps track of relations among representations, such as consistencies, dependencies and constraints,
- it encapsulates the value of the data-item with a default-value, a form to compute it, etc.,
- it guards the status and behavior of the data-item and activates procedures when specific events happen (e.g. the value becomes instantiated or changed).

Sobel & Friedman (1996) discuss an introduction to reflection-oriented programming. Matsuoka et al. (1992) discuss object-oriented concurrent reflective architectures. Oliva et al. (1998) discuss the reflective architecture called "GuaranÃ¡." Watanabe & Yonezawa (1989) discuss reflective computation in object-oriented concurrent systems. Yonezawa (1990) discuss a reflective object oriented concurrent language called "ABCL/R." Ancona et al. (1998) discuss channel reification, which is a reflective model for

distributed computation. Cazzola (1998) discuss evaluation of object-oriented reflective models.

3.11.2 *Learned Reactions—Models of Learning Reactions*

3.11.2.1 *Models of Planning to Perceive*

Pryor & Collins (1995) discuss how planning is used in perceptual processes.

3.11.2.2 *Models of Commonsense Reasoning*

There are many domains of quaitative commonsense reasoning, e.g. physics (Forbus 1994), natural language (Liu & Singh 2004*a*), story narratives (Williams et al. 2005), event planning (Smith 2006), and psychology (Gordon et al. 2008).

3.11.2.3 *Models of Reactive Commonsense Knowledge*

There are also a few large knowledge bses of commonsense knowledge, e.g. ConceptNet (Liu & Singh 2004*b*), FrameNet (Baker et al. 1998), Cyc (Lenat et al. 1990), AnalogySpace (Speer et al. 2009), and WordNet (Fellbaum 1998). Although these knowledge bases have a broad range of reasoning algorithms from linear algebraic semantic analogies in AnalogySpace to intricate microtheoretic logical deduction in Cyc, none of these knowledge bases are able to causal reflectively control the complexity of their algorithms. Because commonsense knowledge is often represented in a declarative form, logical reasoning algorithms are often applied, but undirected logical reasoning methods often cannot handle large amounts of knowledge. I see that reflective tools available in the Funk2 architecture could help with learning to reflectively control these explosive logical reasoning algorithms.

3.11.3 *Deliberative Thinking—Models of Learning to Compile Plans*

Planning and scheduling systems are useful for reasoning about, respectively: (1) what actions should be taken in a domain and (2) how those actions should be ordered in time (Smith et al. 2000). Michalski & Ram (1995) discuss learning as goal-driven inference. Weber et al. (2011) discuss counting failure explanations as a heuristic for guiding planning with knowledge seeking actions. Thagard & Millgram (1995) discuss using a coherence theory of decision in order to perform inference to the best plan.

3.11.3.1 *Goal-Directed Knowledge Acquisition*

Weber et al. (2011)

3.11.3.2 *Reinforcement Learning*

According to Sutton & Barto (1998), the field of reinforcement learning deals with the problem of "learning what to do—how to map situations to actions—so as to maximize a numerical reward signal." Reinforcement learning is an online type of learning problem as opposed to supervised or unsupervised types of learning problems.

Kaelbling et al. (1996) gives a good overview of historical reinforcement learning. The reinforcement learning problem is defined as

- a set of states, $\mathcal{S}$,
- a set of actions, $\mathcal{A}$, and
- a set of scalar reinforcement signals, typically $\{0, 1\}$ or $\mathfrak{R}$.

It is interesting to consider how the well developed idea of "exploration versus exploitation" as discussed in Kaelbling et al. (1996) as a model for deciding when to sequentially meta-reason.

subsubsectionModels of Optimal Behavior

Kaelbling et al. (1996) discuss three primary models of optimal behavior:

1. the *finite horizon* model in which an AI performs receding horizon control, such that total reward is $E(\sum_{t=0}^{h} r_t)$,
2. the *infinite discounted* model in which an AI uses future rewards discounted by a constant factor, γ, such that total reward is $E(\sum_{t=0}^{\infty} \gamma^t r_t)$, and
3. the *average reward* model which is the infinite limit of the finite horizon model, such that total reward is $\lim_{h\to\infty} E(\frac{1}{h} \sum_{t=0}^{h} r_t)$.

These exact optimality criteria are only applicable to relatively small problem domains. Nonetheless, I hypothesize that useful approximations of optimal systems that can be derived from overlapping subsystems with constraining optimality criteria can be derived from these three basic types of optimality. Homeostasis monitors and continual "heartbeat" controllers could be implemented with infinite or average optimality models, while short-term goal-oriented controllers would necessitate the complexities of the full N-step finite horizon model of optimality.

3.11.3.3 *Dynamic Programming and Optimal Control*

Bertsekas (1995) describe techniques for finding optimal and approximate *functional* (*memoizable*) solutions to reinforcement learning control problems, using recursive formulas of flat state spaces that can be optimized very easily by using memoized functions. While this approach does not handle abstract models of the AI's world, there are exciting options of combining a dynamic programming approach with an abstract form of reinforcement learning, such as the use of inductive logic described in Džeroski et al. (2001*a*).

3.11.3.4 *A Theoretical World Model*

A simple formulation of the reinforcement learning problem can be formulated in terms of Markov Decision Process (MDP) environments. An MDP is defined as

- a set of states, $\mathcal{S}$,
- a set of actions, $\mathcal{A}$,
- a reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \mathfrak{R}$, and
- a state transition function, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow \Pi(\mathcal{S})$.

3.11.3.5 *Category Learning to Accomplish Goals*

Category learning in the context of goal-oriented problem-solving is an idea that is discussed by Barsalou (1995).

3.11.3.6 *Models of Learning by Positive or Negative Reward in the context of Relational Knowledge*

Džeroski et al. (2001*a*) describe an interesting combination of inductive logic programming (ILP) (Muggleton 1992) and reinforcement learning. The benefit of using abstract logical representations of propositional states, actions, and state values makes the basic reinforcement learning algorithm able to deal with relational state spaces, such as the blocks world toy problem and the other problem representations that I am focusing on in this PhD. This form of learning abstractions takes a flat mental space and maps it to a hierarchical relational abstraction.

3.11.4 *Reflective Thinking—Models of Learning to Control Deliberation*

3.11.4.1 *Models of Commonsense Psychology*

Understanding human psychology in commonsense terms is an initiative described in Gordon & Hobbs (2004). I see my work as building upon this development of a commonsense language for discussing the types of mental processes that humans have. Also, Gordon et al. (2008) further develops this commonsense psychological language approach to describe the processes of self-reflection, which is relevant to my work. The difference between my approach and the commonsense psychology approach is that my work simulates these mental processes in a computer, using the Funk2 causally reflective procedural programming language. Previous approaches have represented psychological reasoning processes in a declarative logical form, but these logical forms cannot handle the large knowledge bases that are involved in general commonsense reasoning.

Wilson & Nisbett (1977) discuss verbal reports on mental processes. Schank et al. (1972) discuss primitive concepts underlying verbs of thought.

3.11.4.2 *Models of Unsticking Deliberation by Creative Analogical Thinking*

Buchanan & Mitchell (1977) discuss creativity at the metalevel. Mueller (1990) discuss daydreamer.

A storytelling system that is called MINSTREL is discussed in Turner (1994); it is guided by creativity goals in order to develop good narratives. The process of creative storytelling is handled very similarly to mechanical goal-oriented problem solving in this way.

3.11.4.3 *Models of Learning to Plan to Learn*

Ram & Leake (1995*a*) discuss planning to learn. desJardins (1995) discuss a decision-theoretic model for deciding what to learn next for accomplishing goals. Ram et al. (1995) discuss goal-driven learning in multistrategy reasoning and learning systems. Ram & Leake (1995*b*) discuss learning, goals, and learning goals. Cox & Ram (1999*a*) discuss the construction of learning strategies in introspective multistrategy learning.

3.11.4.4 *Models of Learning by Credit Assignment through Temporal State Tracing*

Eligibility traces are a simple reinforcement learning tool for temporal credit assignment for an AI entering positive and negative reward states. For example, the *backward* view of eligibility tracing as applied to the temporal difference, TD(λ), learning method simply augments the basic learning algorithm with an additional eligibitity factor, $e_t(s)$, for each state. Only non-zero values of the eligibility factor are remembered in tractible (approximate) implementations; this amounts to keeping a list of n recently visited states. In the algorithm, the current state's eligibility factor, $e_t(s_t)$, is set to 1 and all other states are discounted by a constant eligibility discount factor, γ. Credit assignment for rewards that are found in the world are assigned not only to the previous state, as in the basic one-step TD(λ) algorithm but also assigned to the value functions in a discounted sense to the previous n states in its eligibility trace. I call this form of credit assignment "Temporal State Tracing" because it simply assigns credit to those states temporally local to the current state, i.e. there is no explicit consideration of causality involved in the assignment.

3.11.4.5 *Models of Learning by Credit Assignment through Causal Tracing of Failure*

I do not know of any reinforcement learning algorithms that reflectively learns how to plan through causal tracing of failures. My method of learning in this way is disussed in Section **??**.

3.11.4.6 *Models of Learning by Explaining Failures*

Hammond (1990) discuss explaining and repairing plans that fail. VanLehn & Jones (1992) discuss a model of the self-explanation effect. Ram & Cox (1995) discuss introspective reasoning using meta-explanations for multistrategy learning. Leake (1995*a*) discuss goal-based explanation evaluation. Leake (1995*b*) discuss toward goal-driven integration of explanation and action. Dominowski (1998) discuss verbalization and problem solving.

3.11.4.7 *Models of Learning by Repairing Failed Plans and Inference Knowledge*

Leake (1996) discuss learning to refine the case-based reasoning process through experience, introspection and expertise. Levin & Beck (2004) discuss how to span the difference between metacognitive failure and success in thinking about seeing. Cox (1996) discuss how to construct a learning strategy under reasoning failure in the terms of introspective multistrategy learning.

3.11.4.8 *The Difference between Metareasoning, Computational Reflection, and Causal Reflection*

Hansen & Zilberstein (2001) discuss a dynamic programming approach to monitoring and control of anytime algorithms. Cox (2007*a*) discuss metareasoning, monitoring, and self-explanation. Cox (2005) present a selected research review of metacognition in computation. Anderson & Oates (2007) present a review of recent research in metareasoning and metalearning. Cox (2007*b*) discuss perpetual self-aware cognitive AIs. Cox & Raja (2008) present a manifesto on metareasoning. Davis (1980) discuss reasoning about control in terms of meta-rules. Fox & Leake (2001) discuss introspective reasoning for index refinement in case-based reasoning. Maes (1988) discuss issues in computational reflection. Minsky (1968) discuss matter, mind, and models. Newell (1982) discuss the knowledge level.

3.11.4.9 *Empirical Models of Reflective Thought*

Baumeister et al. (2007) discuss How emotion shapes behavior: Feedback, anticipation, and reflection, rather than direct causation. Vince (2002) discuss Organizing reflection. Mirvis (2008) discuss Executive Development Through Consciousness-Raising Experiences. Michalsky et al. (2007) discuss Developing Students' Metacognitive Awareness in Asynchronous Learning Networks in Comparison to Face-to-Face Discussion Groups. Bivona et al. (2008) discuss Executive function and metacognitive self-awareness after Severe Traumatic Brain Injury. Iran-Nejad & Gregg (2001) discuss The brain-mind cycle of reflection. Phillips & Silvia (2005) discuss Self-awareness and the emotional consequences of self-discrepancies. Silvia et al. (2006) discuss Emotion concepts and self-focused attention: Exploring parallel effects

of emotional states and emotional knowledge. Lynn (2008) discuss The eq interview: finding employees with high emotional intelligence.

3.11.5 *Self-Reflective Thinking—Models of Control using Self and Other Models*

Anderson & Perlis (2005) discuss the metacognitive loop and the problem of brittleness in terms of logic, self-awareness and self-improvement. Schubert (2005) discuss some knowledge representation and reasoning requirements for self-awareness.

3.11.5.1 *Models Involving Multiple AIs*

Models of multiple intelligent AIs that work together to solve problems have become popular in the field of machine learning called "multi-AI reinforcement learning." Many popular surveys of the field exist (Wei 1995) (Sen & Weiss 1999) (Stone & Veloso 2000) (Shoham et al. 2004) (Yang & Gu 2004) (Panait & Luke 2005).

Boutilier (1996) discuss a specific type of n-person cooperative game theory, in which AIs share common goals, or in other words, a joint value function. Rapoport (2001) discuss n-person game theory in detail.

Bowling & Veloso (2000) describe an analysis of stochastic game theory for multiAI reinforcement learning. Claus & Boutilier (1998) describe the dynamics of reinforcement learning in cooperative multiAI systems. Crites & Barto (1998) describe elevator group control using multiple reinforcement learning AIs. Ghavamzadeh et al. (2006) describe hierarchical multi-AI reinforcement learning. Guestrin et al. (2002) describe coordinated reinforcement learning. Hu & Wellman (1998) describe a theoretical framework and an algorithm for multiAI reinforcement learning. Kapetanakis & Kudenko (2002) describe reinforcement learning of coordination in cooperative multi-AI systems. Matarić (1997*b*) describe reinforcement learning in the multi-robot domain. Park et al. (2001) describe modular Q-learning based multi-AI cooperation for robot soccer. Sandholm & Crites (1996) describe multiAI reinforcement learning in the iterated prisoner's dilemma. Tan (1997) describe independent versus cooperative AIs in multi-AI reinforcement learning. Wang & Sandholm (2003) describe reinforcement learning to play an optimal Nash equilibrium in team Markov games.

3.11.5.2 *MultiAI Reinforcement Learning with Layered Models*

Stone (1997) describe layered Learning in multiAI Systems.

3.11.5.3 *MultiAI Reinforcement Learning with Communication*

Drogoul & Ferber (1994) describe an application to social differentiation in ant colonies using a multi-AI simulation as a tool

for modeling societies. Fischer et al. (2004) describe hierarchical reinforcement learning in communication-mediated multiAI coordination. Price & Boutilier (1999) describe implicit imitation in multiAI reinforcement learning. Sen & Sekaran (1998) describe individual learning of coordination knowledge.

3.11.5.4 *MultiAI Reinforcement Learning with Self- and Other- Models*

Matarić (1997*a*) describe learning social behavior. Nagayuki et al. (2000) describe an approach based on the otherAI's internal model in multi-AI reinforcement learning. Noble & Franks (2004) describe social learning in a multi-AI system. Nowe et al. (2001) describe social AIs playing a periodical policy.

3.11.5.5 *Models of Parent and Child Social Functions*

Castelfranchi (2001) describe the challenges for computational social science and multi-AI learning in a theory of social functions.

3.11.5.6 *Empirical Models of Inferring Person Traits*

Asch (2005) discuss how impressions of personalities are formed. They break personality down into 18 two-part models, or dimensions. See Table 1 for a listing of these dimensions.

Winter & Uleman (2005) discuss the difference between *dispositional* inferences and *situational* inferences. A dispositional statement is a statement about the relationship between person models, such as "kind person" or "she helped the man", whereas a situational statement is a statement that does not refer to such a model of a social relationship between person models, such as "she dropped the groceries". See Table 2 for the list of 18 dispositional properties that they discovered in their research interviewing college students for word associations:

3.11.5.7 *Empirical Models of Self and Other Social Models*

This section discusses much research that often uses words that I often do not see used precisely. For example, the phrase "self-awareness" is often used to describe a sort of mental process that monitors another mental process. In this document, I try to be consistent by using the word "self" to only refer to models of self versus other in terms of AIs thinking about personalities or properties of AIs. Other common uses of the terms "self-awareness" and "self-consciousness" are often less clear and refer to any kind of reflective process perceiving the execution of another process within the mind. There are many forms of reflective thought, and I choose to use the term "self-reflection" to refer to those reflective processes that deal with self and other personality models of inter-AI relationships.

Goverover et al. (2007) discuss treatment to improve self-awareness in persons with acquired brain injury. Baumeister et al. (1993) dis-

1.	generous	ungenerous
2.	wise	shrewd
3.	happy	unhappy
4.	good-natured	irritable
5.	humorous	humorless
6.	sociable	unsociable
7.	popular	unpopular
8.	reliable	unreliable
9.	important	insignificant
10.	humane	ruthless
11.	good-looking	unattractive
12.	persistent	unstable
13.	serious	frivolous
14.	restrained	talkative
15.	altruistic	self-centered
16.	imaginative	hard-headed
17.	strong	weak
18.	honest	dishonest
19.	*warm*	*cold*
20.	*intelligent*	
21.	*envious*	

Table 1: 18 personality dimensions discussed by Asch.

1. generous
2. rude
3. good worker
4. helpful
5. ill-mannered
6. thrifty
7. talented
8. bigot
9. friendly
10. concerned citizen
11. absent-minded
12. kindhearted
13. cheap
14. clever
15. inconsiderate
16. willpower
17. considerate
18. clumsy

Table 2: 18 dispositional properties that Winter & Uleman discovered in their research interviewing college students for word associations.

cuss when ego threats lead to self-regulation failure: negative consequences of high self-esteem. Gilbert et al. (2005) discuss perceptions of people. Ross et al. (2005) discuss social roles, social control, and biases in social-perception processes. Hobson et al. (2006) discuss self-awareness in developmental perspective. Hutchinson & Skinner (2007) discuss relationships among adaption-innovation, self-monitoring, and self-consciousness. Keenan & Gorman (2007) discuss the selective causal role of the right hemisphere in self-awareness. Morin (2004) discuss a comparison and integration of various views of the levels of consciousness and self-awareness. Silverman (2008) discuss self-reflection. Smith (2007) discuss self-awareness, perspective-taking, and self-face recognition. Soeiro (2008) discuss state versus trait self-focus: comparing self-awareness to self-consciousness. Uhlmann et al. (2008) discuss varieties of social cognition. Burns & Engdahl (1998) discuss individual selves, self-awareness, and reflectivity in the social construction of consciousness. Duval & Silvia (2002) discuss self-awareness, probability of improvement, and the self-serving bias. Focquaert (2008) discuss an evolutionary cognitive neuroscience perspective on human self-awareness and theory of mind. Whitehead (2001) discuss social mirrors and shared experiential worlds.

3.11.5.8 *Empirical Models of Person Groups*

Menon et al. (2005) discuss the attribution to individual versus group dispositions in culture and the construal of agency.

3.11.6 *Self-Conscious Reflection—Models of Learning by Imprimer Reflection on Self and Other Models*

3.11.6.1 *Empirical Models of Imprimers*

Hinton (2008) discuss the role of leader self-awareness in building trust and improving student learning. Taylor (2007) discuss a conceptual framework and an empirical test for determining leader attunement in order to work toward developing a more general theory of leader self-awareness. Hare (1993) describe how certain types of anti-social personality disorders result in "successful" individuals and a disturbing view of society.

3.11.6.2 *Moral Decisions and Values*

Harsanyi (1980) discuss rule utilitarianism, rights, obligations and the theory of rational behavior. Duval & Lalwani (1999) discuss how changing self or changing standards of correctness also changes objective self-awareness and causal attributions for self-standard discrepancies.

3.12 NATURAL LANGUAGE IN PROBLEM-SOLVING

Bobrow (1968) discuss a system that processes natural language in the context of automated problem-solving.

3.13 MODELS OF STORY AND LANGUAGE UNDERSTANDING

Ram & Moorman (1999) present a theory of reading and understanding. Ram (1999) present a theory of questions and question asking. Rapaport & Shapiro (1999) discuss automated language understanding in fiction. Mahesh et al. (1999) discuss sentence processing while integrating multiple knowledge sources. Domeshek et al. (1999) discuss parsing the semantic content from complex narratives. Peterson & Billman (1999) discuss a theory of semantic correspondence. Moorman & Ram (1999) discuss learning novel concepts requiring creativity while reading. Cox & Ram (1999*b*) discuss the intersection of story understanding and machine learning. Gerrig (1999) discuss natural language text processing in narrative worlds.

3.14 EVALUATING MODELS OF COMPLEX LEARNING

3.14.1 *Evaluating Models of Learning Skill Knowledge*

Anderson & Corbett (1993) describe subjects acquisition of LISP programming skills in an experimental setting. Ohlsson (1993) describe the acquisition of cognitive skills while fosucing on the interaction between knowledge and practice.

3.14.2 *Evaluating Models of Learning by Self-Explanation*

VanLehn & Jones (1993) present a computational model of the process of learning by explaining examples to oneself. Kieras (1993) describe a model of learning schemas from explanations in the domain of practical electronics. Rosenbloom et al. (1993) discuss the problem of learning bias in planning and explanation-based learning.

3.14.3 *Evaluating Models of Learning Declarative Relational Knowledge*

Huffman et al. (1993) analyze the knowledge-level of a model that corrects imperfect domain theories. Hammond & Seifert (1993) present a cognitive science approach to case-based planning. Wilensky (1993) discuss the problem of knowledge acquisition in the domain of natural language processing.

3.14.4 *Visual Programming Languages for Psycological Study of Complex Models*

Cooper (2002) and Cooper & Fox (1998) discuss how to model high-level cognitive processes in an empirical and scientific way. Much of their work is presented in terms of their COGENT cognitive architecture that includes a visual programming language for cognitive scientists to quickly build and evaluate varieties of complex hypotheses.

Cooper (1995) discuss an object-oriented language for cognitive modeling. Cooper et al. (1996) discuss a systematic methodology for cognitive modelling. Cooper et al. (1998) discuss their system, COGENT, an tool for the development of cognitive models. Councill et al. (2003) analyze existing tools and user information requirements with regard to the Soar cognitive architecture. Mulholland & Watt (1998) discuss their Hank system, a friendly cognitive modelling language for psychology students. Mulholland & Watt (1999) discuss how the Hank system can be used for implementing schema theory. Mulholland & Watt (2000) discuss an educational tool for psychology students to learn by building computational cognitive models. Yule & Cooper (2001) discuss a vision of a technology for efficienct experimentation with computational cognitive models.

3.15 MODELS OF THE LARGE-SCALE INTEGRATION OF MULTIPLE MODELS OF COGNITION

Langley et al. (2008) gives a good overview of the current state of the field of cognitive architectural design in terms of what exists and what needs to still be designed and built. Among the features that are not much explored are the types reflective processes that control deliberative processes, which Sloman (2001) refers to as meta-management. I see building layers of these reflective management processes in the top three layers of the Emotion Machine theory as my basic inspiration for higher orders of reflective control beyond meta-management.

3.15.1 *The EM-ONE Model*

Singh (2005) describe the first cognitive architecture based on the Emotion Machine theory of mind, developed in Minsky (2006). This architecture includes the ability to control reasoning using a relational list commonsense narrative representation. In terms of types of representations, the Emotion Machine theory proposes that self models exist in the self-reflective layer and imprimer models exist in the self-conscious layer. My purpose is to elaborate on this example model in a larger and causally reflective programming language. My Funk2 architecture is based on procedural forms of run-time reflection that is built into the underlying programming language, whereas the previous work was based on declarative forms that used an assortment of programming languages, including Lisp and Prolog.

3.15.2 *The Icarus Model*

Langley (2005) describe an architecture that uses meta-knowledge of knowledge. They refer to skill planning as planning toward long-term goals in the abstract meta-knowledge plan space. They refer to short-term planning as planning over instances of the skill plans. I see my approach as complimentary to this plan abstraction approach because I am focused on adaptive learning due to failures of conflicting plans, so I expect my approach to be adaptable to this form of meta-planning as well.

3.15.3 *The ACT-r Model*

Anderson et al. (2004) describe a model that is inspiring to us because it is the largest cognitive architecture that is actively being correlated to human neurological and psychological data. I see my work as heading in this direction, but first I must complete a working model of my theory so that such similar scientific work may proceed in terms of understanding reflective, self-reflective, and self-conscious thought processes.

3.15.4 *The SOAR Model*

Laird et al. (1987) describe the SOAR cognitive architecture, an architecture for general intelligence. Marinier et al. (2009) describe a recent use of SOAR to build a model that unifies cognitive behavior and emotion.

3.15.5 *The PolyScheme Model*

Cassimatis et al. (2004) and Cassimatis (2006) describe a cognitive architecture that combines multiple representations and reasoning methods for implementing human level intelligence. I see the problem of having multiple representations and concurrent reasoning processes as exactly the problem that I am designing the Funk2 causally reflective architecture: in order to control and learn from conflicts in such a mental model.

3.15.6 *The GLAIR Model*

Shapiro & Ismail (2003) describe an architecture that performs a form of belief revision.

3.15.7 *The CIRCA Model*

Musliner (2001) describe an architecture with plan execution processes that operate in real-time concurrently with deliberative planning processes. This architecture is not reflective over it's deliberation, but Funk2 is designed with this concurrency and real-time behavior in mind as well.

3.15.8 *The CogAff Model*

Sloman (2001) describe a three-layer cognitive architecture that is very similar to the bottom three layers of the Emotion Machine theory's bottom three layers, i.e. reactive, deliberative, and meta-management or reflective layers. The CogAff architecture does not make any representational commitments and higher layers of reflection are referred to as complex forms of meta-management.

3.15.9 *The Three Layers Model*

Ng & Bereiter (1995) discuss three levels of goal orientation in learning.

3.15.10 *The AIS Model*

In Hayes-Roth (1995), a reflective architecture is described that operates on cycles. On each cycle, a meta-cognitive controller selects among a set of deliberative controllers.

3.15.11 *The Prodigy Model*

Carbonell et al. (1991) describe an architecture that learns to plan through self-explanation and using what they refer to as causal traces of execution histories. These causal traces are similar to what can be derived from the causal tracing functionality inherent in the Funk2 programming language.

Carbonell et al. (1995) discuss planning and learning in PRODIGY, an integrated architecture.

3.15.12 *The Daydreamer Model*

Mueller (1990) describes a model containing a "society of mind" collection of deliberative thinking resources that perform many different types of planning and analogical transfer between different mental realms.

3.16 PARALLEL-LISP PROGRAMMING LANGUAGES

Halstead (1990) discusses parallel lisp language design, implementations, and programming tools.

Part II

MY APPROACH

4

A SYSTEM

4.1 SYSTEM OVERVIEW

Building a machine that demonstrates the general intelligence required for commonsense reasoning is a longstanding goal of the field of artificial intelligence. There have been many approaches to building a machine that demonstrates general intelligence, some are based on logical representations, others are based on large collections of statistical knowledge, while still others approach the problem by learning everything from scratch from the physical world. I see the problem as requiring a combination of many different types of representations and reasoning processes.

4.1.1 *Reflectively Traced Frame Memory*

Procedural tracing can be thought of as enabling a part of an interpreter that checks for important events as a process runs. This ability can be used to trace the temporal order of events, i.e. generating a trace, or to otherwise organize or summarize events in a more useful way that can be used by other procedures, either concurrently or after the fact. Having this ability built into the memory system, allows keeping track of the provenance of information as it is written and read. This ability is built into all structures in the architecture, so if any tracing of data provenance is required at any time for any object, this ability can be enabled.

At higher levels, these procedural tracing events result in event streams that can be listened to by multiple concurrent processes that each allocate iterators for a stream. As a concurrent process increments its iterator, it reasons about the event that has occurred in the object memory, and the result of this reasoning is the creation of meta knowledge in a causally consistent knowledge base. This is an example of the "glom" theory of cognitive evolution, where cognitive abilities are added on top of previous cognitive abilities without changing the underlying functionality. I've used procedural tracing to maintain multiple consistent representations for the same knowledge.

When a plan fails, I need to correct the knowledge that generated that plan. When multiple processes are adding knowledge to the same knowledge base, it becomes important to keep track of the provenance of this knowledge, when I need to make distinctions between situations that appear identical. If I need to learn a new rule for categorization of situations, or even a new category entirely, I need to go back to the correct features and processes that performed those categorizations of the identical situations that I need to further distinguish.

4.1.2 *Reflecitive Pattern Recognition*

4.1.3 *Efficiency Problems with Reflective Tracing*

There are serious efficiency problems that must be carefully sidestepped when one is dealing with reflective knowledge. For example, infinite reflection loops result in an infinite processor and memory consumption pattern, after only a single change to a knowledge representation. The solution to this problem is to have various means of controlling the reflective tracing focus. I will discuss the various methods for controlling the tracing focus of the reflective memory that I have found useful in the development of my system in Section 4.1.4.

4.1.4 *Methods for Focusing Reflective Tracing*

Section 4.1.4 is referenced from Section 4.1.3.

4.1.5 *Procedural Tracing*

The idea of having procedural tracing at the operating system level is important because it does allow all programs running on the operating system to assign credit to processes when bugs do occur.

Although it is important to have protected memory boundaries between programs for reasons of security, privacy, and stability, having good ways for processes that do share memory to trace the provenance of individual memory events, allows for much tighter and intelligent interaction between all processes in the entire system.

4.1.6 *Trace Only an Appropriate Level of Abstraction*

It does not make sense to trace below a certain depth of processing. If I were to trace all of the lowest level details of execution at all times, there would be no way to take advantage of that amount of detailed information.

There are barriers in my system for only allowing tracing to occur for specific parts of the code. The ability to focus the tracing of object usage allows the possibility of tracing either high or low-level events in a uniform manner.

4.1.7 *Keeping Tracing from Taking Too Much Time and Storage*

When a set of objects are out of the focus of tracing, these objects run operate at full speed and do not increase the memory usage of the tracing component. There is a new proposal by McCarthy to build a new programming language that remembers everything that it does; it is called Elephant 2000 (McCarthy 1994).

4.2 AN OPERATING SYSTEM AND A PROGRAMMING LANGUAGE

I've written a multi-core operating system, including a compiler and a programming language, on top of this traceable and distributed memory layer (Morgan 2009).

Because the system is meant to combine many artificial intelligence techniques, I have tested my platform running thousands of parallel processes concurrently on an 8-core machine. These processes can control and watch one another execute. I feel that the field of AI is not separate from the low-level details of software engineering, and my project embodies that philosophy.

Many people see AI as being a purely theoretical and mathematical field, and I strongly disagree. I need good software engineers in order to solve most of the problems I face in getting these massive software systems to work together.

I have built a layered cognitive architecture on top of my custom operating system, programming language, and compiler.

4.2.1 *Why not use Lisp?*

Lisp is a great programming language. I wrote a custom programming language for the project and didn't use lisp. Lisp simply isn't fast enough, and isn't very well supported; when you find a bug in a lisp compiler, it is difficult to find the support to fix the bug. I wrote the first version of the reflectively traced memory system in lisp and realized that Steele Bourne Common Lisp had memory bugs when the system grew beyond 600 megabytes of RAM. Allegro Lisp is a commercial solution, but it costs many hundreds of dollars for their commercial compiler, and I feel strongly against having that commercial requirement for building academically intentioned open-source software. The main problem with lisp is it's lack of speed and lack of support, so I found myself writing a lot of C extensions even when programming in Lisp. C is good for speeding up inner loops of algorithms as well as necessary for interfacing with the Linux, Mac, or Windows operating systems, which are all written in C.

4.3 A LAYERED COGNITIVE ARCHITECTURE

Further, I have developed a cognitive architecture within my language that provides structures for layering reflective processes, resulting in a hierarchy of control algorithms that respond to failures or other events in the layers below.

4.3.1 *Perceptual Support of Physical Knowledge*

See Figure 29.

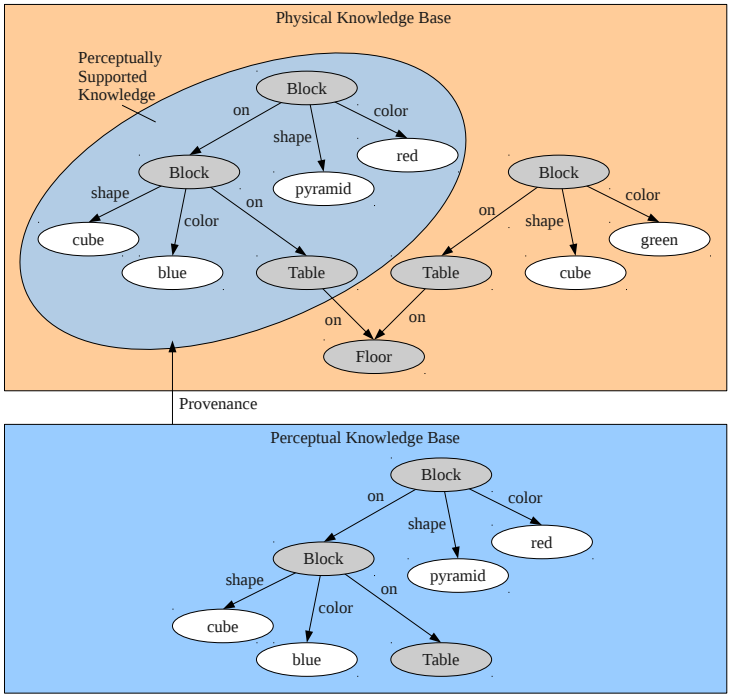


Figure 29: Perceptual provenance provides support for physical knowledge.

[prog	[pick-up	'red	'cube]
	[put-on	'blue	'cube]
	[pick-up	'green	'pyramid]
	[put-on	'red	'cube]]

Table 3: Representation of a serial process.

4.3.2 *A Serial Process Representation*

A representation of a serial process is shown in Table 3. This representation is an ordered tree symbolic representation, which is capable of representing any Lisp-like expression in my language interpretter.

4.3.3 *A Parallel Process Representation*

See Table 4.

[prog	[parog	[use-left-hand-to-pick-up	'red	'cube]
		[use-right-hand-to-pick-up	'green	'pyramid]
	[parog	[use-left-hand-to-put-on	'blue	'cube]]
		[use-right-hand-to-put-on	'red	'cube]]]

Table 4: Representation of two serial parallel processes.

4.3.4 *Details of Inferring the Effects of a Plan*

4.3.5 *Goal-Oriented Action Hypothesis Generation Techniques*

4.3.6 *Debugging Plans by Reflectively Tracing the Provenance of Knowledge*

Section 4.3.4 is referenced from Section 1.4.2.

Section 4.3.5 is referenced from Section 1.3.1.

I trace the procenance of knowledge from perceptual knowledge as it is manipulated into other knowledge representations, driven by a goal-oriented learning algorithm. Normally, when plans fail, rule learning is used to update beginning and ending condition transition function hypotheses for actions relative to various knowledge bases.

In many systems, when these plans fail, it is often unclear which part of the plan was responsible for the failure. In the simplest cases, a specific low-level action may have been executing, which implies that precondition categories were incorrectly mapped to postconditions or the range of postconditions was not broad enough. In the worst cases, it is impossible to tell what part of the plan failed because the plan is such a complicated tangle of compiled numbers that the only recourse is to nudge a few of these numbers using hill-climbing toward an average of a universal goal. Rethinking this problem with my new approach that maintains provenance for knowledge used in creating plans allows debugging the complex web of goal-oriented knowledge, and reflectively, also manipulating the processes involved in creating that knowledge.

4.4 COMPUTER SCIENCE

4.4.1 *Distributed Computing*

4.4.2 *Databases and Knowledge Representation*

5 EXPERIMENTS

5.1 BLOCKS WORLD AS A SIMPLE REAL-TIME SYMBOLIC CONTROL PROBLEM DOMAIN

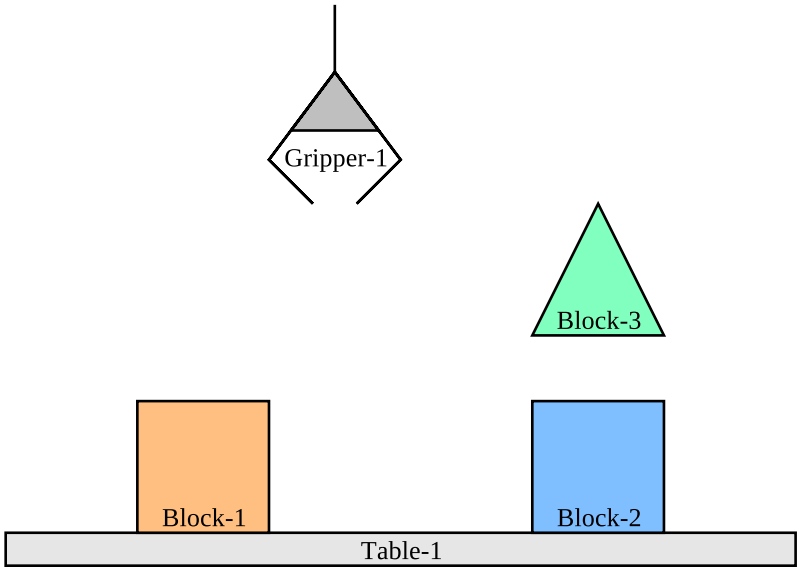


Figure 30: Blocks world is a simple real-time symbolic control problem that I use to demonstrate my reflective control learning theory.

I use blocks world, a canonical toy AI problem, in order to demonstrate my example of reflectively learning to plan. See Figure 30 for a screenshot of my blocks world problem physical simulation.

See Table 5 for an example set of perceptual input that corresponds with the physical situation shown in Figure 30.

5.2 A CONCRETE PLAN

5.2.1 *Credit Assignment Metric*

First, I will develop a metric by which I can measure the performance of my credit assignment algorithm versus other algorithms. At one extreme, we have the brute force approach to learning, in which all possible concepts are re-learned at every opportunity (e.g. at every step in a Markov stepped system, or at every knowledge event in my real-time event driven system). At the other extreme, we re-learn only those concepts whose fundamental training knowledge has changed (instances, hypothesis space, or features). My approach to tracing the provenance of deliberative

[Gripper-1 is me]	[Gripper-1 movement_command []]
[Gripper-1 is-a gripper]	[Gripper-1 color black]
[Gripper-1 is-holding []]	[Block-1 is-a block]
[Block-1 color brown]	[Block-1 shape cube]
[Block-1 on Table-1]	[Block-1 left-of Gripper-1]
[Block-2 is-a block]	[Block-2 color blue]
[Block-2 shape cube]	[Block-2 on Table-1]
[Block-2 right-of Gripper-1]	[Block-3 is-a block]
[Block-3 color green]	[Block-3 shape pyramid]
[Block-3 right-of Gripper-1]	[Table-1 is-a block]
[Table-1 color white]	[Table-1 shape cube]
[Table-1 left-of Gripper-1]	

Table 5: Blocks world AI perceptual input.

knowledge through the reactive plan execution agencies will be somewhere between these two extremes of efficiency.

– Evaluation Metrics

– credit assignment: the process of choosing a subset from the total possible set of concepts that should be focused on as being responsible for the occurrence of a given event.

EXACTLY WHAT IS THE METRIC HERE? HOW WILL IT BE MEASURED? YOU HINT AT SPEED BELOW, AND IF THIS IS TRUE, WILL IT BE MEASURED IN DECISION-CYCLE TIME? DEFINE AND OPERATIONALIZE YOUR METRIC.

The credit assignment process can be more or less efficient in focusing the learning resources toward concepts that can be re-learned.

5.2.2 *Temporal Credit Assignment*

I can measure the performance of my causal credit assignment algorithm versus a typical temporal credit assignment algorithm, such as an adaptation of the Temporal Difference (TD) learning algorithm commonly used in the field of reinforcement learning.

HAVE YOU ALREADY IMPLEMENTED THIS ALGORITHM? WILL YOU USE AN EXISTING IMPLEMENTATION?

5.2.3 *Causal Credit Assignment*

The goal of these tests are to show that learning by provenance speeds the standard temporal credit assignment method, which assigns credit to the sequence of immediately previous states and actions.

ABOVE YOU MENTION YOUR CREDIT ASSIGNMENT *VERSUS* A STANDARD ALGORITHM. HERE YOU STATE THAT YOUR VERSION WILL ACCELERATE THE STANDARD ALGO-

RITHM. DO YOU MEAN THAT YOURS IS FASTER THAT THE STANDARD IN THE LIMIT? OR DO YOU INTEND TO COMBINE THEM TO SPEED UP THE TEMPORAL ALGORITHM?

– I will focus on re-learning category concepts given new training instances that would change those concepts.

– Now, given the above metric for what I will actually be measuring as a performance metric, here are two specific deliberative learning tasks where my causal learning will be superior to a temporal credit assignment method. This is primarily due to the delayed time between deliberation about knowledge and the execution of that knowledge, resulting in a precondition check form of failure. Specifically, the precondition check could simply be what would be Removed by a trans-frame, but doesn't necessarily exist in the physical knowledge at the time of execution. The following are two examples of scenarios:

5.2.4 *The Social Knowledge Trust Task*

Some plans can be learned through a social communication language. Physical plans are added to the deliberative knowledge with the added relationship with the appropriate social provenance markers, e.g. Gripper-1 may represent that a given plan is told to it by The-User (or another social source).

The Decision to use a plan that accomplishes a given goal from one knowledge source or another can be learned, but the learning of this type of action is difficult because of the often indirect temporal connection between this deliberative Decision involved in choosing or creating a plan and the actual execution failure, resulting from executing a plan that has derived from that deliberative decision.

THIS LONG RUN-ON SENTENCE IS TYPICAL OF YOUR WRITING AND PLACES THE BURDEN OF UNDERSTANDING UPON THE READER. WHAT IS LEARNED? A DECISION OR AN ACTION? FURTHERMORE, YOU PREVIOUSLY DISCUSSED LEARNING CONCEPTS. IMPLEMENT A SINGLE LEARNING ALGORITHM AND USE IT TO ILLUSTRATE YOUR THEORY. AGAIN, DECIDE WHAT YOU WILL NOT SAY.

5.2.5 *Working in a World of Building Blocks*

In his PhD thesis, Terry Winograd worked in the world of building blocks (Winograd 1970). This program maintained traces of its goals and subgoals, which enabled it to answer questions about why it performed certain actions. This system worked because it stored goals.

Knowing the goal state of the computation is important, and I do not ignore this aspect in tracing the deliberative layer. My system is able to answer these sorts of questions, as this simply requires climbing the stack of mental resource activations, but when debugging the deliberative process, it is helpful to not only

know the ending point of computation but also the means toward that end.

5.2.6 *Terry Winograd's SHRDLU and Goal Tracing*

I am building upon what was learned from Winograd's thesis (Winograd 1970) in terms of using traces of the deliberative process as well as using a semantic model of the world in order to understand communications between AIs. I have chosen to use a simpler and more direct language interface between AIs that refers more directly to the semantic information and mental processes involved. I have experimented with implementing the original SHRDLU english language parser, although I believe the parsing process can be better controlled as a goal-oriented set of concurrent processes than as the stack-based depth first search that I started writing in my initial experiments.

5.3 A PHYSICAL SIMULATION OF A KITCHEN AS A SOCIAL COMMONSENSE REASONING DOMAIN

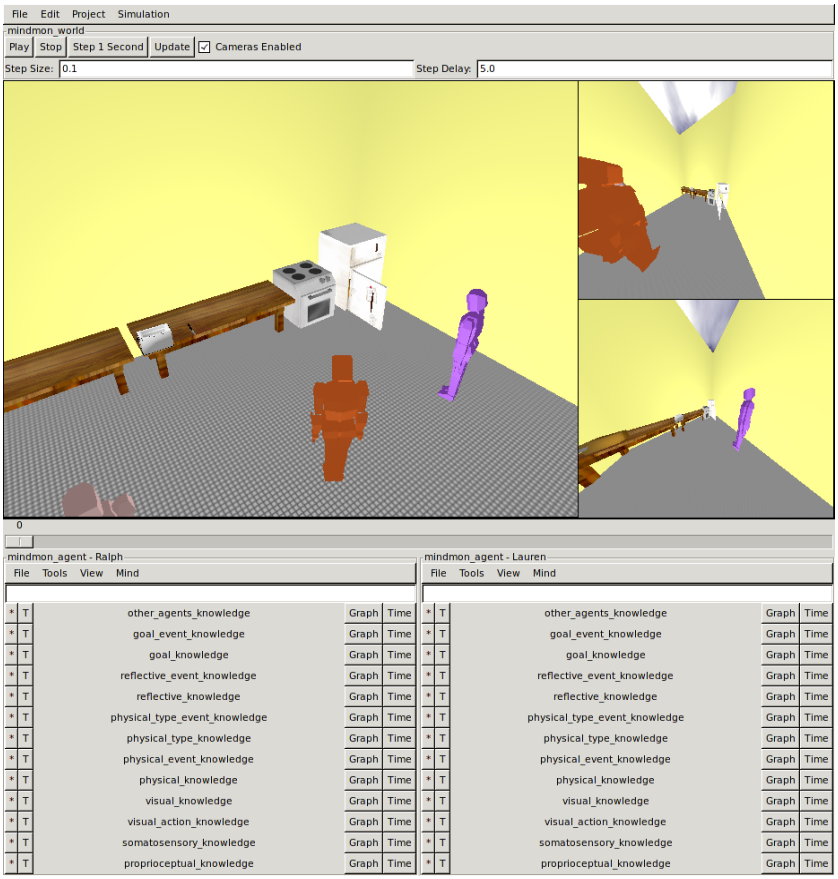


Figure 31: Isis World is a larger real-time symbolic physical simulation, which I use to demonstrate that my reflective goal-oriented learning approach scales to the physical and reflective problem spaces of slightly more complicated learning problems.

I have experimented with applying my cognitive architecture to a larger social commonsense reasoning domain with parents that teach children as they attempt to accomplish cooking tasks in a kitchen. See Morgan (2011) for details about my six-layered reflective theory of social and moral reasoning, which assumes the existence of a procedurally reflective infrastructure with a reflective problem solver written within it.

5.3.1 *Why Not Work Solely Within the Blocks World Domain?*

The building blocks approach is a good precedent. However, there are many problems with only demonstrating a solution on a toy problem. First, an approach demonstrated to solve a small problem, often do not scale to larger problem domains of similar complexity. So, I feel that it is important to show the same reflective approach to learning can also be applied to a domain with a much larger state space than the toy blocks world problem. I then, have shown the theoretical gains of my approach by using the canonical model as a tool for explanation, and now I show that my model does scale to larger problem domains of only slightly more complexity. See Smith & Morgan (2010) for a discussion of the benefits of approaching the social commonsense reasoning problem with a physical simulation of a kitchen.

I have conscripted my domain of object types in the kitchen, such that it is currently comparable to the number of object types that Winograd used in his thesis. My object types do have different ways that they may be used, which is a small addition of complexity. Although I do not introduce many of the complexities of ontological reasoning, a common approach to commonsense reasoning, e.g. Cyc (Lenat et al. 1990), my system demonstrates an important new approach to commonsense reasoning that grounds learning by being told in the domain of goal-oriented reasoning, which allows organizing and debugging knowledge in terms of what goals it is useful for accomplishing.

5.3.2 *Why is Cooking in a Kitchen a Good Problem to Model?*

Smith & Morgan (2010) discuss reasons for why focusing on solving cooking tasks in a kitchen will lead to a better understanding of the reflective control of multiple AIs in a common complicated social environment. Further, Morgan (2010) describes how focusing on this domain from a reflective architectural point of view can lead to future models of self-reflective and self-conscious learning between parents and children.

From an educational perspective, Dewey (1907) has theorized that kitchens are a good example of a rich learning environment for children. Some form of kitchen, or a social place where food is cooked for a family, is ubiquitous across cultures. Kitchens have a clear production goal, namely food. Many mental realms must be used in accomplishing even simple cooking goals, in-

cluding: math, physics, chemistry, thermodynamics, language, society, family, imprimer learning, concurrent planning, etc. See Figure 31 for a screenshot of my graphical user interface to the cognitive architecture, while it interacts with the Isis World physical simulation.

Part III

CONCLUSION

6

RESULTS

6.1 REFLECTIVE KNOWLEDGE SUBSTRATE

6.1.1 *General Parallelism and Concurrency*

My system is built to handle concurrent parallel processes on multi-core operating systems. In this section, I test how efficient my implementation of concurrency performs on a number of standard concurrency timing tasks. In an ideal situation, given N processor core working on independenct problems, I should get a factor of N speedup. Because multiple core processors share memory resources this ideal is never actualized, so we test our implementations performance by developing appropriate metrics.

6.1.2 *Program as Data*

In systems where the program is data, a bottleneck in writing programs that write programs is the efficiency of the run-time compiler. In this section we test the relative performance of our compiler in our-time system.

6.2 LAYERED REFLECTIVE PROBLEM SOLVING

In this section, I evaluate how well different configurations of my learning system accomplishes a range of goals in the blocks world environment.

6.2.1 *Comparison between Learned Reactions and Planning*

I evaluate how adding or removing reflectively learning to plan changes the performance of the blocks world goal accomplishment metric.

6.2.2 *Analogy between Physical Goals and Planning Goals*

I evaluate how adding or removing the second layer of reflectively learning to plan changes the performance of the blocks world goal accomplishment metric.

6.3 LEARNING BY CREDIT ASSIGNMENT

6.3.1 *Tracing Knowledge Provenance for Credit Assignment of Success or Failure*

I evaluate how tracing causality of knowledge provenance can improve learning performance according to the blocks world goal accomplishment metric.

7

DISCUSSION

During our initial experiments, we discovered many organizational benefits of representing parts of our system in a reflectively traced form.

7.1 REFLECTIVE KNOWLEDGE MAINTAINANCE

7.2 SCRIPT DECOMPOSITION AND RECOMPOSITION

8

FUTURE

8.1 A SIMPLE OVERVIEW

My research on reflective thinking is leading toward a larger view of layered human cognition. In the future, I plan to continue to focus on this larger architectural view. Specifically, I am focused on the question of how humans are capable of learning and teaching one another to exhibit virtuous behavior. Humans are able to learn from each other, not just simple actions but entire systems of morality and values. What are the factors that mitigate goal learning? What are the differences between guilt, shame, regret? Are these different or just different words? Is shame fundamentally social? Guilt and shame are common in folk psychology, but they are too poorly understood to scientifically study the details of their specific mechanisms.

Let my working definition of virtuous behavior be the complexities of accomplishing and maintaining social goals, which in my model are initially learned from interactions with the physical world in the context of parents. I am attempting to understand how different forms of cooperative and non-cooperative behavior develop in these situations. I am also interested in how an observant parent can guide the development of these behaviors in children.

So, what is a simple way to begin to talk about what I mean by "social thinking" and "moral thinking"? In the past, these topics have spawned numerous approaches that are more or less representative of my biological understanding. Some approaches have been formalized into computational models that can be used as working scientific theories. Recently, all of these initial approaches have been extended and further researched so that I can calculate probability distributions over these representations. I can now categorize these approaches. A categorization reveals the diversity of human thought covered by different approaches. I am interested in understanding what types of processes are most important to describe when I approach the extremely complicated issue of moral reasoning.

8.2 PHILOSOPHY AND RELATED RESEARCH

8.2.1 *The Social Utility of Moral Reasoning*

When I approach the topic of immorality, there are many complicating factors, so how can I simplify the reality of the situation in order to create a model, which is necessarily not real, but which is still useful for predicting reality? Well, I currently see many

approaches to studying different aspects of moral reasoning, such as how moral judgements are dependent on the causal structure of how a person understands social coersion (Young & Phillips 2011). I also see evidence that engaging emotions such as "pride", "embarrassment", and "regret" has been shown to be correlated to specific fMRI BOLD patterns (Moll et al. 2005). Further, lines have been drawn between mere chauvinism for one's own social group and the more complicating aspects of virtuous behavior that maximizes social utility (Casebeer & Churchland 2003). This gives us initial evidence and philosophical guidance to pursue better models for what I mean by virtuous behavior.

8.2.2 *Current Models*

The field of machine learning has described models of goal-oriented behavior that use a numerical utility function in order to quantify whether or not a given goal has been accomplished. These models are proving useful for finding the beginning of an explanation for how models of the world are learned and what the function of specific neurotransmitters may be in the neocortex (Yu & Dayan 2003). The problem with these simple propositional goal-oriented models is that they cannot handle the large state spaces involved in modelling social relationships and social goals, so I need something slightly more complicated. A popular tool for modelling situations involving people thinking about the goals of other people is the tool of relational representations, e.g., the modal logics of morality (Horty 1994). Augmenting goal-oriented reinforcement learning models with the abilities of relational representations is a being done by a relatively young field (Džeroski et al. 2001*b*) that is proving to be maturing very quickly.

8.2.3 *The Development of Social Goals*

If I am going to develop a theory of virtuous behavior, or social goals, how do I begin to model the cooperative and non-cooperative thought processes that would constitute working toward or against other people? It has been shown that the neural circuits that are involved in how someone thinks about what someone else knows are largely distinct from the circuits involved in perceptual capabilities (Bedny et al. 2009). There have also been distinctions shown between the neural circuits involved in how someone thinks about what someone else knows and executive control of actions (Saxe et al. 2006). There is good reason to believe that the learning of social behavior is bootstrapped from learning causal physical models (Perner 1991).

8.3 MY ONGOING RESEARCH

8.3.0.1 *A Reflective Parallel Programming Language*

I have written a programming language, called Funk2, (Morgan 2009) that allows us to easily build and monitor the execution of my model. My language allows describing systems that reflectively trace changes to collections of knowledge by many different parallel mental processes. Procedural reflection is a computational technique that I have built into this language that is useful for organizing models that monitor and control their own execution (Maes 1987). I have chosen to write my own language because most other programming languages emphasize the idea of abstraction barriers, which allows writing optimized code, but which does not allow a user to access the details of an execution environment. My language allows itself to access many of the details of its own process execution by emphasizing "execution simplicity" over "speed optimization" in many cases. Most programming languages are optimized for speed at the expense of the resulting complexity of the optimized processes and also the lack of traces of reflectively useful execution events. My language is useful for building models of processes that monitor, control, and change other concurrently executing processes. I see this as a very useful tool for modelling the complex types of interactions between learning processes that are described in the cognitive sciences. In other words, the form of procedural reflection enabled by my language gives us a powerful method for organizing models of parallel learning processes.

8.3.0.2 *A Physically Grounded Social Problem Domain*

In order to help us define a grounded problem domain, I have chosen to focus on a rigid-body physical simulation of parents and children in the context of basic cooking tasks in a kitchen environment. I chose parents and children because I feel that major parts of social and moral learning develops during early stages involving these familiar social relationships. I chose the domain of basic kitchen cooking tasks because this is a non-trivial social commonsense reasoning domain that exists in some form in all human cultures.

8.3.0.3 *A Simple Social Reasoning Model*

8.3.0.4 *Physical Reactions*

My cognitive model is organized into layers. The lowest layer represents the simplest processes of thinking, such as physical reflexes, e.g., "I jerk my hand away from the stove if I burn my fingers." The next layer represents processes that have been practiced successfully very often, and which one does not think about, e.g., "My legs go through fixed action patterns if I want to walk across the room."

8.3.0.5 *The Past, Goals, and Deliberative Thinking*

Next, at a slightly higher layer in my model, I begin to model deliberative thinking, which allows one to use memories of the past and imaginative memories of the future in order to make plans for doing things that one has perhaps never done before. For example, if one wants to rescue a cat from being caught in a tree, maybe one has played with cats in the past as well as climbed trees, but one has never done the two at the same time. This would be a form of deliberative thinking, making something like a plan for a goal that one has never exactly done before. If a plan for accomplishing a deliberate goal is successful, perhaps this successful process could be compiled down into a lower level layer, so that it does not need to again be thought of deliberatively.

8.3.0.6 *Causal Abstractions and Reflective Thinking*

Now, I have so far only discussed the types of thought processes that solve problems based on perceptions of problems in the physical world. For example, the processes that reflexively jerk one's hand from a stove, are due to corpuscles in the fingers communicating information to neurons about burning temperatures in the skin of the hand. Further, the learned walking process receives proprioceptive information about the relative angles of the person's limbs, as well as somatosensory information about the pressure on the soles of the feet, as well as vestibular information about the accelleration forces on the head, including gravity. Walking is a very complicated process, but walking is primarily a physical control problem. Now, I am going to introduce a separate perceptual stream of information that does not describe the physical state of the body in the world. This separate perceptual stream, refers to the interactions between the parallel mental processes. I can distinguish between these two types of perceptual streams, by calling one type a Physical perceptual stream and the other I can call a Reflective perceptual stream of information.

Now, once I have a reflective perceptual stream that represents the reflective state of the mind, I can begin to model processes of reflective thinking that solve problems in this reflective state space. For example, a reflective process could remember what deliberative processes are good for making different types of plans to handle different types of physical problems, such as the differences that must be considered between moving physical objects that display agency as opposed to physical objects that do not exhibit any form of agency. Once I have made a distinction in my model between physical perceptions and reflective perceptions, I correlate these two streams, which allows us to infer reflective events from physical events.

8.3.0.7 *The Self/Other Distinction and Self-Reflective Thinking*

If my mental model contains knowledge of the correlations between a physical perceptual stream and a reflective perceptual

stream, then the AI can use these correlations to predict the reflective states of other AIs based purely on their physical actions. For example, if one AI performs a series of physical actions in order to accomplish a goal, such as preparing a slice of buttered toast using a knife and a loaf of bread and some butter in a kitchen, then these actions have been correlated with this AI's reflective states, including the current goals that the AI is pursuing. If this AI then sees the physical world changing, and he knows that he is not causing those changes, then he can attempt to infer the state of mind of another AI. This type of inference process, based on the previous correlations of physical and reflective perceptual streams, is what I refer to as a self-reflective thought process. A self-reflective thought process introduces the social distinction between Self and Other reflective state knowledge that is inferred by using the previously learned correlations between physical perceptions and reflective perceptions. I refer to this type of self-reflective knowledge as "singly recursive" reflective knowledge, or "thinking about what someone else is thinking about". Sometimes this singly recursive reflective knowledge is referred to as "theory of mind" knowledge in the cognitive science literature.

With self-reflective knowlege, an AI can choose to act altruistically by helping another AI accomplish their goals or maliciously by working against their goals. However, there is the possibility that an AI is not always running a self-reflective process that pays attention to another AI's physical actions in order to infer what their goals might be. In many cases, such as in my simulation of parents and children in a kitchen, some AIs might focus single-mindedly on a novel task without thinking about what the goals of surrounding AIs might be. This type of model allows for an AI that works against the goals of another AI without knowing the goals of the other AI. One might argue that this type of situation where one AI clobbers the goals of another AI is due to a form of self-reflective laziness and not due to malicious intentions.

8.3.0.8 *Guilt, Pride, and Self-Conscious Thinking*

Because I am interested in modelling how children learn from parents as well as how parents guide a child's cooperative behavior, I need to extend my model slightly in order to allow for modelling what one AI thinks about what another AI thinks about her. This form of double reflective recursion is what I refer to as a self-conscious knowledge and the problems that are represented in this type of knowledge are handled by what I call self-conscious thought processes. For example, a little boy may be cooking something in the kitchen and he may carelessly work against the goals of his sister; this situation may be recognized by an onlooking parent, and the parent may inform the boy of a self-reflective mistake he has made: "Ralph, your sister was going to use some of that butter that you just finished." The little boy, Ralph in this case, thinks that his mother thinks that he made a self-reflective mistake. I hypothesize that modelling this form of

double reflective recursion in conjunction with respectful social relationships may be useful for understanding emotions such as guilt and pride. I have focused my model of children learning to cooperate in the context of parents on this sort of self-conscious learning feedback.

8.3.1 *A Six Layer Organization of My Theory*

My layered reflective model of mind is inspired by Minsky's Model-6 Emotion Machine theory of how humans perform commonsense human thinking (Minsky 2006).

- Self-Conscious Thinking: processes that manipulate doubly recursive reflective representations.
- Self-Reflective Thinking: processes that manipulate singly recursive reflective representations.
- Reflective Thinking: processes that manipulate representations about one's own mind.
- Deliberative Thinking: processes that manipulate representations of the physical world.
- Learned Reactive Thinking: processes that have been learned and chunked for efficient execution.
- Built-In Reactive Thinking: processes that are given as the simplest and lowest-level aspects of the mental model.

In summary, the control structure in my model can be thought of simply as being roughly organized into six layers of control. The bottom layer represents the physical reactive interface to the physical world. Both perception and action are inputs and outputs from this layer, respectively. Then, at each subsequently higher layer information from below and above are processed as I have described in the previous sections, leading to changes in self-reflective processes caused by self-conscious learning as my highest level example.

8.4 PANALOGY ARCHITECTURE

8.4.1 *Recursive Loops and Infinite Recursive Tracing Descent*

If the focus on the tracing is controlled carefully, these potential loops can be avoided. How to detect and control these potential loops is an interesting area of future automatic debugging research in reflective control.

8.4.2 *Potential Future Uses for Low-Level Tracing*

Lower level objects maybe be interesting to focus on for research in automatic abstraction and simulation of system components.

Optimizing compilers could benefit from this area of future research. E.g. focusing on the CPU object could help to develop better run-time register allocation models.

8.4.3 *Why Should You Use This Radically New Language?*

Because the language is very similar to Lisp, it has proven to be easy for both expert and novice programmers to learn. This has been my experience with the four undergraduates that have worked within the language, who learned it quickly, started writing their own macros to facilitate their style, and one even made additions to the core algorithms.

8.5 AI SPEECH ACTS

At any given point in time, each AI is pursuing a different variety of goals. Each AI executes actions in order to accomplish collections of these goals. I treat language communication as an additional physical action that the AI can choose to perform in order to accomplish its goals.

8.6 MODELLING NOISE IN AI COMMUNICATION

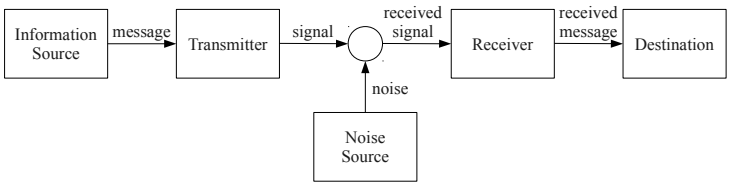


Figure 32: A mathamtical theory of communication (Shannon 1959).

My cognitive theory makes no statements as to the presence of noise in communication channels between AIs and the environment. Communication between AIs and the environment in my implementation occurs across a noiseless communication channel. It is theoretically possible to include a theory of noisy communication channels between AIs and the environment. For example, Shannon's mathmematical theory of communication (Shannon 1959) could be applied as an elaboration to any of the information pathways in my basic theory.

8.7 APPLICATIONS TO EDUCATION AND MENTAL HEALTH

An exciting new field is growing to include applications of cognitive theories of mind to both educational and clinical mental health domains. For example, Mahncke et al. (2006) have reported results of improved cognitive function by using an automated computer cognitive training program targeting age-related cognitive decline.

Part IV

APPENDIX

A

THE CODE

A.1 OPEN-SOURCE DOWNLOAD

All of the code that I developed for this PhD is free and openly developed, can be downloaded from my webpage, and compiled by simply typing "./configure; make". See the on-line Funk2 website, `http://funk2.org/`, for downloads, documentation, user community resources, and latest news. Also, see `http://em-two.net/` for research news on my Moral Compass cognitive architecture that currently is distributed with Funk2.

A.2 THE HACKER PHILOSOPHY OF CODE

The problems of artificial intelligence and greater control of more complicated and interconnected computational systems are not easy domains to approach from a programming perspective. Many of these problems seem to require not only new programming abstractions, methodologies and philosophies, but also bug-free implementations of these potentially paradigm shifting ideas.

In other words, the only people that are going to be able to solve these types of control problems that are on the forefront of science fiction in research are going to be expert computer programmers, or "hackers." In general the ideal hacker is a problem solver that accomplishes their own goals in given complicated systems, not necessarily computer systems.

In terms of computational science, hackers are often confronted by given hardware and software problems that must be overcome in order to implement solutions to their posed problems. Hacking can be adopted as a fun philosophy for accomplishing ones own goals in life, but also, I think that hacking is a required prerequisite for any form of scientific progress. A joy for tinkering that leads to playing, which subsequently leaves the tinkerer as an expert hacker is what is needed for approaching undocumented domains, such as the forefront of artificial intelligence and control theory in general.

A.3 WHAT IS A COMPUTER?

Good question. We've presented a simple model in Section 1.2.1, but this abstract model leaves a lot to be desired for modelling the more realistic computers that most programmers enjoy. For example, hardware platforms with many processing cores per CPU, multiple CPUs per motherboard, and multiple computers per networked cluster of motherboards. In order to reflectively

control computational systems that are organized according to these modern engineering constraints, I have organized my memory infrastructure to span these areas of future reflective control research. For example, basic pointer in the Funk2 programming language includes 17 bits to represent the cluster machine ID, and 9 bits to represent a processor core specific memory allocation pool. We see Funk2 as a platform supporting an open research community of hackers that share computational resources in order to build demonstrations of larger reflective artificial intelligence control systems.

A.4 README

```
funk2: causally reflective programming language

  funk2 [-x <command>] [-p <portnum>] [<source.fu2>]

    <source.fu2>

        A user supplied filename of file from which to read and
            execute source
        code after booting and before exiting.

    -x <command>  [:default [repl]]

        A user supplied command to execute after booting and
            before exiting.

    -p <portnum>  [:default 22222 :try anything from 22222 to
        23221]

        The localhost peer-command-server port number.  Each
            copy of funk2
        sharing a network interface must be able to allocate a
            unique
        peer-command-server port number.



TO PERFORM LOCAL BUILD:

  ./configure
  make


TO RUN LOCAL BUILD:

  ./funk2.sh     (from original compile directory)


TO PERFORM SYSTEM-WIDE INSTALLATION:

  ./configure
  make
  make install  (as root)


TO RUN SYSTEM-WIDE INSTALLATION:

  funk2          (from anywhere on system)



Homepage:

  http://funk2.org/
```

```
Git Access:

  http://git.neuromin.de/


Last Modified: 2010.10.25

Code Mass

-------------------------------------------------------------------------------


    Lines of Funk2 Code.....:   49837 total
    Words of Funk2 Code.....:  246131 total
    Characters of Funk2 Code: 2516032 total

    Lines of C Code.........:  131529 total
    Words of C Code.........:  521743 total
    Characters of C Code....: 6982294 total

    Total Lines of Code.....:  182371 total
    Total Words of Code.....:  770823 total
    Total Characters of Code: 9553461 total

-------------------------------------------------------------------------------


README Last Generated: Mon Aug  1 14:05:18 EDT 2011
```

A.5 FILE LISTING

```
start_emacs.sh
README
configure.ac
Makefile.am
funk-mode.el
c/configurator.c
c/debugbreak.c
c/f2_agent.c
c/f2_agent.h
c/f2_ansi.c
c/f2_ansi.h
c/f2_apropos.c
c/f2_apropos.h
c/f2_archconfig.h
c/f2_array.c
c/f2_array.h
c/f2_atomic.h
c/f2_buffered_file.c
c/f2_buffered_file.h
c/f2_buffered_socket.c
c/f2_buffered_socket.h
c/f2_bug.c
c/f2_bug.h
c/f2_bytecodes.c
c/f2_bytecodes.h
c/f2_cause.c
c/f2_cause.h
c/f2_chunk.c
c/f2_chunk.h
c/f2_circular_buffer.c
c/f2_circular_buffer.h
c/f2_command_line.c
c/f2_command_line.h
c/f2_compile.c
c/f2_compile.h
```

```
c/f2_compile_x86.c
c/f2_compile_x86.h
c/f2_core_extension.c
c/f2_core_extension_funk.c
c/f2_core_extension_funk.h
c/f2_core_extension.h
c/f2_cpu.c
c/f2_cpu.h
c/f2_debug_macros.h
c/f2_dlfcn.c
c/f2_dlfcn.h
c/f2_dptr.c
c/f2_dptr.h
c/f2_dynamic_memory.c
c/f2_dynamic_memory.h
c/f2_f2ptr.c
c/f2_f2ptr.h
c/f2_fiber.c
c/f2_fiber.h
c/f2_fileio.c
c/f2_fileio.h
c/f2_frame_objects.c
c/f2_frame_objects.h
c/f2_funk2_node.c
c/f2_funk2_node.h
c/f2_funktional.c
c/f2_funktional.h
c/f2_garbage_collector.c
c/f2_garbage_collector.h
c/f2_garbage_collector_pool.c
c/f2_garbage_collector_pool.h
c/f2_globalenv.c
c/f2_globalenv.h
c/f2_global.h
c/f2_glwindow.c
c/f2_glwindow.h
c/f2_gmodule.c
c/f2_gmodule.h
c/f2_graph.c
c/f2_graph_cluster.c
c/f2_graph_cluster.h
c/f2_graph.h
c/f2_graph_match_error_correcting.c
c/f2_graph_match_error_correcting.h
c/f2_graphviz.c
c/f2_graphviz.h
c/f2_gtk.c
c/f2_gtk.h
c/f2_hash.c
c/f2_hash.h
c/f2_html.c
c/f2_html.h
c/f2_knowledge.c
c/f2_knowledge.h
c/f2_larva.c
c/f2_larva.h
c/f2_load.c
c/f2_load.h
c/f2_malloc.c
c/f2_malloc.h
c/f2_management_thread.c
c/f2_management_thread.h
c/f2_matlab.c
c/f2_matlab.h
c/f2_memblock.c
c/f2_memblock.h
c/f2_memory.c
c/f2_memory.h
c/f2_memorypool.c
c/f2_memorypool.h
c/f2_module_registration.c
c/f2_module_registration.h
```

```
c/f2_natural_language.c
c/f2_natural_language.h
c/f2_never_delete_list.c
c/f2_never_delete_list.h
c/f2_nil.c
c/f2_nil.h
c/f2_number.c
c/f2_number.h
c/f2_object.c
c/f2_object.h
c/f2_opengl.c
c/f2_opengl.h
c/f2_optimize.c
c/f2_optimize.h
c/f2_package.c
c/f2_package.h
c/f2_package_handler.c
c/f2_package_handler.h
c/f2_packet.c
c/f2_packet.h
c/f2_partial_order.c
c/f2_partial_order.h
c/f2_peer_command_server.c
c/f2_peer_command_server.h
c/f2_perception_lattice.c
c/f2_perception_lattice.h
c/f2_primes.c
c/f2_primes.h
c/f2_primfunks.c
c/f2_primfunks__errno.c
c/f2_primfunks__errno.h
c/f2_primfunks__fcntl.c
c/f2_primfunks__fcntl.h
c/f2_primfunks.h
c/f2_primfunks__ioctl.c
c/f2_primfunks__ioctl.h
c/f2_primfunks__locale.c
c/f2_primfunks__locale.h
c/f2_primfunks__string.c
c/f2_primfunks__string.h
c/f2_primobject__boolean.c
c/f2_primobject__boolean.h
c/f2_primobject__char_pointer.c
c/f2_primobject__char_pointer.h
c/f2_primobject__circular_buffer.c
c/f2_primobject__circular_buffer.h
c/f2_primobject__doublelinklist.c
c/f2_primobject__doublelinklist.h
c/f2_primobject__dynamic_library.c
c/f2_primobject__dynamic_library.h
c/f2_primobject__environment.c
c/f2_primobject__environment.h
c/f2_primobject__fiber_trigger.c
c/f2_primobject__fiber_trigger.h
c/f2_primobject__file_handle.c
c/f2_primobject__file_handle.h
c/f2_primobject__frame.c
c/f2_primobject__frame.h
c/f2_primobject__hash.c
c/f2_primobject__hash.h
c/f2_primobject__interval_tree-bak.c
c/f2_primobject__interval_tree-bak.h
c/f2_primobject__largeinteger.c
c/f2_primobject__largeinteger.h
c/f2_primobject__list.c
c/f2_primobject__list.h
c/f2_primobject__matrix.c
c/f2_primobject__matrix.h
c/f2_primobject__object.c
c/f2_primobject__object.h
c/f2_primobject__object_type.c
c/f2_primobject__object_type.h
```

```
c/f2_primobject__ptypehash.c
c/f2_primobject__ptypehash.h
c/f2_primobject__redblacktree.c
c/f2_primobject__redblacktree.h
c/f2_primobjects.c
c/f2_primobject__set.c
c/f2_primobject__set.h
c/f2_primobjects.h
c/f2_primobject__stream.c
c/f2_primobject__stream.h
c/f2_primobject__tensor.c
c/f2_primobject__tensor.h
c/f2_primobject__traced_cmutex.c
c/f2_primobject__traced_cmutex.h
c/f2_primobject_type.c
c/f2_primobject_type.h
c/f2_primobject_type_handler.c
c/f2_primobject_type_handler.h
c/f2_print.c
c/f2_print.h
c/f2_processor.c
c/f2_processor.h
c/f2_processor_mutex.c
c/f2_processor_mutex.h
c/f2_processor_thread.c
c/f2_processor_thread.h
c/f2_processor_thread_handler.c
c/f2_processor_thread_handler.h
c/f2_protected_alloc_array.c
c/f2_protected_alloc_array.h
c/f2_ptype.c
c/f2_ptype.h
c/f2_ptypes.c
c/f2_ptypes.h
c/f2_ptypes_memory.h
c/f2_reader.c
c/f2_reader.h
c/f2_redblacktree.c
c/f2_redblacktree.h
c/f2_reflective_memory.c
c/f2_scheduler.c
c/f2_scheduler.h
c/f2_scheduler_thread_controller.c
c/f2_scheduler_thread_controller.h
c/f2_search.c
c/f2_search.h
c/f2_set.c
c/f2_set.h
c/f2_signal.c
c/f2_signal.h
c/f2_simple_repl.c
c/f2_simple_repl.h
c/f2_socket.c
c/f2_socket_client.c
c/f2_socket_client.h
c/f2_socket.h
c/f2_socket_server.c
c/f2_socket_server.h
c/f2_sort.c
c/f2_sort.h
c/f2_staticmemory.c
c/f2_staticmemory.h
c/f2_status.c
c/f2_status.h
c/f2_string.c
c/f2_string.h
c/f2_surrogate_parent.c
c/f2_surrogate_parent.h
c/f2_system_file_handler.c
c/f2_system_file_handler.h
c/f2_system_headers.h
c/f2_system_processor.c
```

```
c/f2_system_processor.h
c/f2_terminal_print.c
c/f2_terminal_print.h
c/f2_termios.c
c/f2_termios.h
c/f2_time.c
c/f2_time.h
c/f2_trace.c
c/f2_trace.h
c/f2_tricolor_set.c
c/f2_tricolor_set.h
c/f2_user_thread_controller.c
c/f2_user_thread_controller.h
c/f2_virtual_processor.c
c/f2_virtual_processor.h
c/f2_virtual_processor_handler.c
c/f2_virtual_processor_handler.h
c/f2_virtual_processor_thread.c
c/f2_virtual_processor_thread.h
c/f2_xmlrpc.c
c/f2_xmlrpc.h
c/f2_zlib.c
c/f2_zlib.h
c/funk2.c
c/funk2.h
c/funk2_main.c
c/test.c
fu2/action.fu2
fu2/actor.fu2
fu2/actortest.fu2
fu2/assembler.fu2
fu2/bootstrap-apropos.fu2
fu2/bootstrap-array.fu2
fu2/bootstrap-boot.fu2
fu2/bootstrap-bug.fu2
fu2/bootstrap-bugs.fu2
fu2/bootstrap-cause.fu2
fu2/bootstrap-compound_object.fu2
fu2/bootstrap-cons.fu2
fu2/bootstrap-core_extension.fu2
fu2/bootstrap-core_extension_funk.fu2
fu2/bootstrap-critic.fu2
fu2/bootstrap-critics-reactive.fu2
fu2/bootstrap-default_critics.fu2
fu2/bootstrap-define_bug_responses.fu2
fu2/bootstrap-dynamic_library.fu2
fu2/bootstrap-fiber.fu2
fu2/bootstrap-frame.fu2
fu2/bootstrap.fu2
fu2/bootstrap-garbage_collector.fu2
fu2/bootstrap-graph.fu2
fu2/bootstrap-graph-old.fu2
fu2/bootstrap-grid.fu2
fu2/bootstrap-hash.fu2
fu2/bootstrap-largeinteger.fu2
fu2/bootstrap-list.fu2
fu2/bootstrap-object.fu2
fu2/bootstrap-package.fu2
fu2/bootstrap-primobject.fu2
fu2/bootstrap-ptypes.fu2
fu2/bootstrap-reader.fu2
fu2/bootstrap-redblacktree.fu2
fu2/bootstrap-repl.fu2
fu2/bootstrap-set_theory.fu2
fu2/bootstrap-sort.fu2
fu2/bootstrap-string.fu2
fu2/bootstrap-surrogate_parent.fu2
fu2/bootstrap-terminal_print.fu2
fu2/bootstrap-type_conversions.fu2
fu2/bootstrap-zlib.fu2
fu2/brainviz.fu2
fu2/cardgame-ai.fu2
```

```
fu2/cardgame.fu2
fu2/cause.fu2
fu2/characters.fu2
fu2/compile.fu2
fu2/emailcharacters.fu2
fu2/emotionmachine.fu2
fu2/english_eval.fu2
fu2/graph.fu2
fu2/graph_match_test.fu2
fu2/graphviz.fu2
fu2/internet.fu2
fu2/link_grammar_wrapper.fu2
fu2/miscfunks.fu2
fu2/neuralmom_brain_area.fu2
fu2/neuralmom_demo.fu2
fu2/neuralmom_nervous_system.fu2
fu2/neuralmom_occipital_cortex.fu2
fu2/opengl.fu2
fu2/pattern.fu2
fu2/planner.fu2
fu2/primfunks_apropos.fu2
fu2/primfunks_arithmetic.fu2
fu2/primfunks_array.fu2
fu2/primfunks_bug.fu2
fu2/primfunks_cause.fu2
fu2/primfunks_chunk.fu2
fu2/primfunks_compile.fu2
fu2/primfunks_core_extension.fu2
fu2/primfunks_core_extension_funk.fu2
fu2/primfunks_cpu.fu2
fu2/primfunks_dlfcn.fu2
fu2/primfunks_errno.fu2
fu2/primfunks_fcntl.fu2
fu2/primfunks_fiber.fu2
fu2/primfunks_fiber_trigger.fu2
fu2/primfunks_frame.fu2
fu2/primfunks.fu2
fu2/primfunks_garbage_collector.fu2
fu2/primfunks_gmodule.fu2
fu2/primfunks_graph.fu2
fu2/primfunks_hash.fu2
fu2/primfunks_ioctl.fu2
fu2/primfunks_largeinteger.fu2
fu2/primfunks_locale.fu2
fu2/primfunks_management_thread.fu2
fu2/primfunks_object.fu2
fu2/primfunks_optimize.fu2
fu2/primfunks_package.fu2
fu2/primfunks_package_handler.fu2
fu2/primfunks_primes.fu2
fu2/primfunks_primobjects.fu2
fu2/primfunks_primobject_type.fu2
fu2/primfunks_primobject_type_handler.fu2
fu2/primfunks_print.fu2
fu2/primfunks_ptypes.fu2
fu2/primfunks_reader.fu2
fu2/primfunks_redblacktree.fu2
fu2/primfunks_scheduler.fu2
fu2/primfunks_search.fu2
fu2/primfunks_set.fu2
fu2/primfunks_socket.fu2
fu2/primfunks_sort.fu2
fu2/primfunks_string.fu2
fu2/primfunks_surrogate_parent.fu2
fu2/primfunks_terminal_print.fu2
fu2/primfunks_termios.fu2
fu2/primfunks_time.fu2
fu2/primfunks_trace.fu2
fu2/primfunks_virtual_processor_handler.fu2
fu2/primfunks_zlib.fu2
fu2/reactive.fu2
fu2/readline_wrapper.fu2
```

```
fu2/repl.fu2
fu2/rlglue-wrapper.fu2
fu2/serialize.fu2
fu2/story.fu2
fu2/thought_process.fu2
fu2/trace.fu2
fu2/x86-compile.fu2
fu2/x86-compile-machine_code.fu2
fu2/x86-compile-mov.fu2
misc/fables.fu2
misc/frog_and_toad.fu2
misc/frog-and-toad.fu2
misc/officer_joke.fu2
misc/roboverse-blocks_world.fu2
misc/roboverse-demo.fu2
misc/simple_game.fu2
extension/cairo/cairo.c
extension/cairo/cairo.h
extension/causality/causality.c
extension/conceptnet/conceptnet.c
extension/conceptnet/conceptnet.h
extension/concept_version_space/concept_version_space.c
extension/concept_version_space/concept_version_space.h
extension/equals_hash/equals_hash.c
extension/equals_hash/equals_hash.h
extension/event_stream/event_stream.c
extension/event_stream/event_stream.h
extension/fibermon/fibermon.c
extension/forgetful_event_stream/forgetful_event_stream.c
extension/forgetful_event_stream/forgetful_event_stream.h
extension/forward_planner/forward_planner.c
extension/frame_ball/frame_ball.c
extension/graph_isomorphism/graph_isomorphism.c
extension/graph_isomorphism/graph_isomorphism.h
extension/image/image.c
extension/image/image.h
extension/image_sequence/image_sequence.c
extension/image_sequence/image_sequence.h
extension/interval_tree/interval_tree.c
extension/interval_tree/interval_tree.h
extension/lick/lick.c
extension/lick/lick.h
extension/mentality/mentality.c
extension/mentality/mentality.h
extension/meta_semantic_knowledge_base/meta_semantic_knowledge_
    base.c
extension/meta_semantic_knowledge_base/meta_semantic_knowledge_
    base.h
extension/movie/movie.c
extension/movie/movie.h
extension/propogator/propogator.c
extension/propogator/propogator.h
extension/semantic_action_event/semantic_action_event.c
extension/semantic_action_event/semantic_action_event.h
extension/semantic_agent/semantic_agent.c
extension/semantic_agent/semantic_agent.h
extension/semantic_category/semantic_category.c
extension/semantic_category/semantic_category.h
extension/semantic_causal_event/semantic_causal_event.c
extension/semantic_causal_event/semantic_causal_event.h
extension/semantic_causal_object/semantic_causal_object.c
extension/semantic_causal_object/semantic_causal_object.h
extension/semantic_containment_object/semantic_containment_
    object.c
extension/semantic_containment_object/semantic_containment_
    object.h
extension/semantic_directed_action_event/semantic_directed_
    action_event.c
extension/semantic_directed_action_event/semantic_directed_
    action_event.h
extension/semantic_event_knowledge_base/semantic_event_knowledge
    _base.c
```

```
extension/semantic_event_knowledge_base/semantic_event_knowledge
    _base.h
extension/semantic_event/semantic_event.c
extension/semantic_event/semantic_event.h
extension/semantic_event_sequence/semantic_event_sequence.c
extension/semantic_event_sequence/semantic_event_sequence.h
extension/semantic_event_tree/semantic_event_tree.c
extension/semantic_event_tree/semantic_event_tree.h
extension/semantic_frame/semantic_frame.c
extension/semantic_frame/semantic_frame.h
extension/semantic_goal_action_causal_hypothesis/semantic_goal_
    action_causal_hypothesis.c
extension/semantic_goal_action_causal_hypothesis/semantic_goal_
    action_causal_hypothesis.h
extension/semantic_goal/semantic_goal.c
extension/semantic_goal/semantic_goal.h
extension/semantic_knowledge_base/semantic_knowledge_base.c
extension/semantic_knowledge_base/semantic_knowledge_base.h
extension/semantic_know_of_existence_event/semantic_know_of_
    existence_event.c
extension/semantic_know_of_existence_event/semantic_know_of_
    existence_event.h
extension/semantic_know_of_relationship_event/semantic_know_of_
    relationship_event.c
extension/semantic_know_of_relationship_event/semantic_know_of_
    relationship_event.h
extension/semantic_object/semantic_object.c
extension/semantic_object/semantic_object.h
extension/semantic_object_type/semantic_object_type.c
extension/semantic_object_type/semantic_object_type.h
extension/semantic_ordered_object/semantic_ordered_object.c
extension/semantic_ordered_object/semantic_ordered_object.h
extension/semantic_packable_object/semantic_packable_object.c
extension/semantic_packable_object/semantic_packable_object.h
extension/semantic_physical_object/semantic_physical_object.c
extension/semantic_physical_object/semantic_physical_object.h
extension/semantic_physical_object_type_relation/semantic_
    physical_object_type_relation.c
extension/semantic_physical_object_type_relation/semantic_
    physical_object_type_relation.h
extension/semantic_physical_object_type/semantic_physical_object
    _type.c
extension/semantic_physical_object_type/semantic_physical_object
    _type.h
extension/semantic_proprioception/semantic_proprioception.c
extension/semantic_proprioception/semantic_proprioception.h
extension/semantic_proprioceptual_object/semantic_proprioceptual
    _object.c
extension/semantic_proprioceptual_object/semantic_proprioceptual
    _object.h
extension/semantic_proprioceptual_orientation/semantic_
    proprioceptual_orientation.c
extension/semantic_proprioceptual_orientation/semantic_
    proprioceptual_orientation.h
extension/semantic_proprioceptual_position/semantic_
    proprioceptual_position.c
extension/semantic_proprioceptual_position/semantic_
    proprioceptual_position.h
extension/semantic_realm/semantic_realm.c
extension/semantic_realm/semantic_realm.h
extension/semantic_relationship_key/semantic_relationship_key.c
extension/semantic_relationship_key/semantic_relationship_key.h
extension/semantic_resource_action_event/semantic_resource_
    action_event.c
extension/semantic_resource_action_event/semantic_resource_
    action_event.h
extension/semantic_resource_action_sequence/semantic_resource_
    action_sequence.c
extension/semantic_resource_action_sequence/semantic_resource_
    action_sequence.h
extension/semantic_resource_event_knowledge_base/semantic_
    resource_event_knowledge_base.c
```

```
extension/semantic_resource_event_knowledge_base/semantic_
    resource_event_knowledge_base.h
extension/semantic_resource/semantic_resource.c
extension/semantic_resource/semantic_resource.h
extension/semantic_self/semantic_self.c
extension/semantic_self/semantic_self.h
extension/semantic_situation_category/semantic_situation_
    category.c
extension/semantic_situation_category/semantic_situation_
    category.h
extension/semantic_situation/semantic_situation.c
extension/semantic_situation/semantic_situation.h
extension/semantic_situation_transition/semantic_situation_
    transition.c
extension/semantic_situation_transition/semantic_situation_
    transition.h
extension/semantic_somatosensation/semantic_somatosensation.c
extension/semantic_somatosensation/semantic_somatosensation.h
extension/semantic_somatosensory_object/semantic_somatosensory_
    object.c
extension/semantic_somatosensory_object/semantic_somatosensory_
    object.h
extension/semantic_temporal_object/semantic_temporal_object.c
extension/semantic_temporal_object/semantic_temporal_object.h
extension/semantic_time/semantic_time.c
extension/semantic_time/semantic_time.h
extension/semantic_visual_object/semantic_visual_object.c
extension/semantic_visual_object/semantic_visual_object.h
extension/timeline/timeline.c
extension/timeline/timeline.h
built-in/alien/alien.fu2
built-in/ansi/primfunks-ansi.fu2
built-in/basic_bug_responses/basic_bug_responses.fu2
built-in/graph_cluster/bootstrap-graph_cluster.fu2
built-in/graph_cluster/primfunks-graph_cluster.fu2
built-in/graph_match_error_correcting/graph_match_error_
    correcting.fu2
built-in/graph_match_error_correcting/graph_match_error_
    correcting-primfunks.fu2
built-in/graphviz/graphviz.fu2
built-in/graphviz/graphviz-primfunks.fu2
built-in/gtk/bootstrap-gtk.fu2
built-in/gtk/primfunks-gtk.fu2
built-in/math/math.fu2
built-in/mutex/mutex.fu2
built-in/natural_language/dictionary_frame.fu2
built-in/natural_language/natural_language_command.fu2
built-in/natural_language/natural_language-primfunks.fu2
built-in/natural_language/parse_tree.fu2
built-in/natural_language/skb-test.fu2
built-in/number/bootstrap-number.fu2
built-in/number/primfunks-number.fu2
built-in/object_lattice/bootstrap-object_lattice.fu2
built-in/object_lattice/primfunks-object_lattice.fu2
built-in/perception_lattice/bootstrap-perception_lattice.fu2
built-in/perception_lattice/primfunks-perception_lattice.fu2
built-in/utilities/errno.fu2
built-in/utilities/fcntl.fu2
built-in/utilities/ioctl.fu2
built-in/utilities/socket.fu2
built-in/xmlrpc/bootstrap-xmlrpc.fu2
built-in/xmlrpc/primfunks-xmlrpc.fu2
example/blocks_world/blocks_world_block.fu2
example/blocks_world/blocks_world.fu2
example/blocks_world/blocks_world_gripper_controller.fu2
example/blocks_world/blocks_world_gripper.fu2
example/blocks_world/blocks_world_physics.fu2
example/blocks_world/blocks_world_sprite.fu2
example/blocks_world/blocks_world_window.fu2
example/divisi2/divisi2.fu2
example/em_two_webpage/em_two_webpage.fu2
example/english_language/english_dictionary.fu2
```

```
example/english_language/english_dictionary_parse.fu2
example/funk2-htmldoc/funk2-htmldoc.fu2
example/funk2-webpage/funk2-webpage.fu2
example/graph_match/graph_match.fu2
example/graph_match/graph_match-test.fu2
example/gtk_timeline/gtk_timeline.fu2
example/isismon/isismon_agent.fu2
example/isismon/isismon_builtin_reactive_physical_activator.fu2
example/isismon/isismon_deliberative_forward_planner_activator.
    fu2
example/isismon/isismon_deliberative_goal_activator.fu2
example/isismon/isismon.fu2
example/isismon/isismon_knowledge.fu2
example/isismon/isismon_learned_reactive_physical_activator.fu2
example/isis_world_client/isis_world_client.fu2
example/isis_world_demo/isis_agent_body.fu2
example/isis_world_demo/isis_visual_agent.fu2
example/isis_world_demo/isis_visual_object.fu2
example/isis_world_demo/isis_world_demo.fu2
example/isis_world_demo/isis_world.fu2
example/isis_world_demo/jj.fu2
example/little_carol_world/little_carol_world.fu2
example/macbeth/macbeth.fu2
example/mind/agency.fu2
example/mind/agent_body.fu2
example/mind/character.fu2
example/mind/mental_layer.fu2
example/mind/mind.fu2
example/mindmon-1.0/mindmon-1.0.fu2
example/mindmon-blocks_world/mindmon-blocks_world.fu2
example/mindmon-isis_world/mindmon-isis_world-builtin_reactive_
    physical_activator.fu2
example/mindmon-isis_world/mindmon-isis_world-deliberative_goal_
    activator.fu2
example/mindmon-isis_world/mindmon-isis_world.fu2
example/mindmon-isis_world/mindmon-isis_world-learned_reactive_
    physical_activator.fu2
example/mindmon/mindmon_agent.fu2
example/mindmon/mindmon_agent_tool.fu2
example/mindmon/mindmon_agent_tool_widget.fu2
example/mindmon/mindmon_agent_widget.fu2
example/mindmon/mindmon.fu2
example/mindmon/mindmon_knowledge.fu2
example/mindmon/mindmon_world.fu2
example/mind/physical_world.fu2
example/mind/resource.fu2
example/mind/self_model.fu2
example/mind/story.fu2
example/mind/story-graph.fu2
example/moral_compass-isis_world/moral_compass-isis_world-
    builtin_reactive_physical_agency_resources.fu2
example/moral_compass-isis_world/moral_compass-isis_world.fu2
example/moral_compass-isis_world/moral_compass-isis_world-
    learned_reactive_physical_agency_resources.fu2
example/moral_compass-isis_world/moral_compass-isis_world-
    learned_reactive_physical_agency_resources-functions.fu2
example/moral_compass/moral_agent_body.fu2
example/moral_compass/moral_compass.fu2
example/moral_compass/old_reflective_event_knowledge_agency-bak.
    fu2
example/moral_compass/self_conscious_imprimer_agency.fu2
example/moral_compass/self_conscious_layer.fu2
example/moral_compass/self_reflective_layer.fu2
example/moral_compass/self_reflective_other_agents_knowledge_
    agency.fu2
example/muddy_carol/muddy_carol.fu2
example/rct_webpage/rct_webpage.fu2
example/reflective_mind-blocks_world/reflective_mind-blocks_
    world.fu2
example/reflective_mind/builtin_reactive_layer.fu2
example/reflective_mind/builtin_reactive_neural_plug_agency.fu2
example/reflective_mind/builtin_reactive_physical_agency.fu2
```

```
example/reflective_mind/builtin_reactive_sensory_agency.fu2
example/reflective_mind/deliberative_execution_agency.fu2
example/reflective_mind/deliberative_forward_planner_agency.fu2
example/reflective_mind/deliberative_forward_planner_agency-test
    .fu2
example/reflective_mind/deliberative_goal_creation_agency.fu2
example/reflective_mind/deliberative_goal_event_knowledge_agency
    .fu2
example/reflective_mind/deliberative_goal_knowledge_agency.fu2
example/reflective_mind/deliberative_layer.fu2
example/reflective_mind/deliberative_physical_event_knowledge_
    agency.fu2
example/reflective_mind/learned_reactive_language_agency.fu2
example/reflective_mind/learned_reactive_layer.fu2
example/reflective_mind/learned_reactive_physical_agency.fu2
example/reflective_mind/learned_reactive_physical_knowledge_
    agency.fu2
example/reflective_mind/learned_reactive_sensory_agency.fu2
example/reflective_mind/reflective_credit_assignment_agency.fu2
example/reflective_mind/reflective_event_knowledge_agency.fu2
example/reflective_mind/reflective_layer.fu2
example/reflective_mind/reflective_mind.fu2
example/reflective_mind/reflective_mind_perception.fu2
example/reflective_mind/reflective_mind_proprioceptual_object.fu
    2
example/reflective_mind/reflective_mind_visual_agent.fu2
example/reflective_mind/reflective_mind_visual_object.fu2
example/roboverse/roboverse.fu2
example/socket-client/socket-client.fu2
example/socket-server/socket-server.fu2
example/traced_mind/traced_mind.fu2
example/traced_mind/traced_resource.fu2
example/visualize/isismon_agent_visualization.fu2
example/visualize/visualize_test.fu2
extension/cairo/cairo-core.fu2
extension/conceptnet/conceptnet-core.fu2
extension/concept_version_space/concept_version_space-core.fu2
extension/equals_hash/equals_hash-core.fu2
extension/event_stream/event_stream-core.fu2
extension/fibermon/fibermon1.fu2
extension/fibermon/fibermon-core.fu2
extension/fibermon/fibermon.fu2
extension/forgetful_event_stream/forgetful_event_stream-core.fu2
extension/forward_planner/forward_planner-core.fu2
extension/frame_ball/frame_ball-core.fu2
extension/graph_isomorphism/graph_isomorphism-core.fu2
extension/image/image-core.fu2
extension/image_sequence/image_sequence-core.fu2
extension/interval_tree/interval_tree-core.fu2
extension/lick/lick-core.fu2
extension/mentality/mentality-core.fu2
extension/meta_semantic_knowledge_base/meta_semantic_knowledge_
    base-core.fu2
extension/movie/movie-core.fu2
extension/propogator/propogator-core.fu2
extension/semantic_action_event/semantic_action_event-core.fu2
extension/semantic_agent/semantic_agent-core.fu2
extension/semantic_category/semantic_category-core.fu2
extension/semantic_causal_event/semantic_causal_event-core.fu2
extension/semantic_causal_object/semantic_causal_object-core.fu2
extension/semantic_containment_object/semantic_containment_
    object-core.fu2
extension/semantic_directed_action_event/semantic_directed_
    action_event-core.fu2
extension/semantic_event_knowledge_base/semantic_event_knowledge
    _base-core.fu2
extension/semantic_event/semantic_event-core.fu2
extension/semantic_event_sequence/semantic_event_sequence-core.
    fu2
extension/semantic_event_tree/semantic_event_tree-core.fu2
extension/semantic_frame/semantic_frame-core.fu2
```

```
extension/semantic_goal_action_causal_hypothesis/semantic_goal_
    action_causal_hypothesis-core.fu2
extension/semantic_goal/semantic_goal-core.fu2
extension/semantic_knowledge_base/semantic_knowledge_base-core.
    fu2
extension/semantic_know_of_existence_event/semantic_know_of_
    existence_event-core.fu2
extension/semantic_know_of_relationship_event/semantic_know_of_
    relationship_event-core.fu2
extension/semantic_object/semantic_object-core.fu2
extension/semantic_object_type/semantic_object_type-core.fu2
extension/semantic_ordered_object/semantic_ordered_object-core.
    fu2
extension/semantic_packable_object/semantic_packable_object-core
    .fu2
extension/semantic_physical_object/semantic_physical_object-core
    .fu2
extension/semantic_physical_object_type_relation/semantic_
    physical_object_type_relation-core.fu2
extension/semantic_physical_object_type/semantic_physical_object
    _type-core.fu2
extension/semantic_proprioception/semantic_proprioception-core.
    fu2
extension/semantic_proprioceptual_object/semantic_proprioceptual
    _object-core.fu2
extension/semantic_proprioceptual_orientation/semantic_
    proprioceptual_orientation-core.fu2
extension/semantic_proprioceptual_position/semantic_
    proprioceptual_position-core.fu2
extension/semantic_realm/semantic_realm-core.fu2
extension/semantic_relationship_key/semantic_relationship_key-
    core.fu2
extension/semantic_resource_action_event/semantic_resource_
    action_event-core.fu2
extension/semantic_resource_action_sequence/semantic_resource_
    action_sequence-core.fu2
extension/semantic_resource_event_knowledge_base/semantic_
    resource_event_knowledge_base-core.fu2
extension/semantic_resource/semantic_resource-core.fu2
extension/semantic_self/semantic_self-core.fu2
extension/semantic_situation_category/semantic_situation_
    category-core.fu2
extension/semantic_situation/semantic_situation-core.fu2
extension/semantic_situation_transition/semantic_situation_
    transition-core.fu2
extension/semantic_somatosensation/semantic_somatosensation-core
    .fu2
extension/semantic_somatosensory_object/semantic_somatosensory_
    object-core.fu2
extension/semantic_temporal_object/semantic_temporal_object-core
    .fu2
extension/semantic_time/semantic_time-core.fu2
extension/semantic_visual_object/semantic_visual_object-core.fu2
extension/timeline/timeline-core.fu2
test/cairo-test/cairo-test.fu2
test/concept_version_space-test/concept_version_space-test.fu2
test/gtk-test/gtk-test.fu2
test/interval_tree-test/interval_tree-test.fu2
test/optimize-test/optimize-test.fu2
test/propogator-test/propogator-test.fu2
test/timeline-test/timeline-test.fu2
test/xmlrpc-test/xmlrpc-test.fu2
built-in/alien/alien.fpkg
built-in/ansi/ansi.fpkg
built-in/basic_bug_responses/basic_bug_responses.fpkg
built-in/graph_cluster/graph_cluster.fpkg
built-in/graph_match_error_correcting/graph_match_error_
    correcting.fpkg
built-in/graphviz/graphviz.fpkg
built-in/gtk/gtk.fpkg
built-in/math/math.fpkg
built-in/mutex/mutex.fpkg
```

```
built-in/natural_language/natural_language.fpkg
built-in/number/number.fpkg
built-in/object_lattice/object_lattice.fpkg
built-in/perception_lattice/perception_lattice.fpkg
built-in/utilities/utilities.fpkg
built-in/xmlrpc/xmlrpc.fpkg
example/blocks_world/blocks_world.fpkg
example/divisi2/divisi2.fpkg
example/em_two_webpage/em_two_webpage.fpkg
example/english_language/english_language.fpkg
example/funk2-htmldoc/funk2-htmldoc.fpkg
example/funk2-webpage/funk2-webpage.fpkg
example/graph_match/graph_match.fpkg
example/graph_match/graph_match-test.fpkg
example/gtk_timeline/gtk_timeline.fpkg
example/isismon/isismon.fpkg
example/isis_world_client/isis_world_client.fpkg
example/isis_world_demo/isis_world_demo.fpkg
example/little_carol_world/little_carol_world.fpkg
example/macbeth/macbeth.fpkg
example/mind/mind.fpkg
example/mindmon-1.0/mindmon-1.0.fpkg
example/mindmon-blocks_world/mindmon-blocks_world.fpkg
example/mindmon-isis_world/mindmon-isis_world.fpkg
example/mindmon/mindmon.fpkg
example/moral_compass-isis_world/moral_compass-isis_world.fpkg
example/moral_compass/moral_compass.fpkg
example/muddy_carol/muddy_carol.fpkg
example/rct_webpage/rct_webpage.fpkg
example/reflective_mind-blocks_world/reflective_mind-blocks_
    world.fpkg
example/reflective_mind/reflective_mind.fpkg
example/roboverse/roboverse.fpkg
example/socket-client/socket-client.fpkg
example/socket-server/socket-server.fpkg
example/traced_mind/traced_mind.fpkg
example/visualize/isismon_agent_visualization.fpkg
example/visualize/visualize_test.fpkg
extension/cairo/cairo.fpkg
extension/conceptnet/conceptnet.fpkg
extension/concept_version_space/concept_version_space.fpkg
extension/equals_hash/equals_hash.fpkg
extension/event_stream/event_stream.fpkg
extension/fibermon/fibermon.fpkg
extension/forgetful_event_stream/forgetful_event_stream.fpkg
extension/forward_planner/forward_planner.fpkg
extension/frame_ball/frame_ball.fpkg
extension/graph_isomorphism/graph_isomorphism.fpkg
extension/image/image.fpkg
extension/image_sequence/image_sequence.fpkg
extension/interval_tree/interval_tree.fpkg
extension/lick/lick.fpkg
extension/mentality/mentality.fpkg
extension/meta_semantic_knowledge_base/meta_semantic_knowledge_
    base.fpkg
extension/movie/movie.fpkg
extension/propogator/propogator.fpkg
extension/semantic_action_event/semantic_action_event.fpkg
extension/semantic_agent/semantic_agent.fpkg
extension/semantic_category/semantic_category.fpkg
extension/semantic_causal_event/semantic_causal_event.fpkg
extension/semantic_causal_object/semantic_causal_object.fpkg
extension/semantic_containment_object/semantic_containment_
    object.fpkg
extension/semantic_directed_action_event/semantic_directed_
    action_event.fpkg
extension/semantic_event_knowledge_base/semantic_event_knowledge
    _base.fpkg
extension/semantic_event/semantic_event.fpkg
extension/semantic_event_sequence/semantic_event_sequence.fpkg
extension/semantic_event_tree/semantic_event_tree.fpkg
extension/semantic_frame/semantic_frame.fpkg
```

```
extension/semantic_goal_action_causal_hypothesis/semantic_goal_
    action_causal_hypothesis.fpkg
extension/semantic_goal/semantic_goal.fpkg
extension/semantic_knowledge_base/semantic_knowledge_base.fpkg
extension/semantic_know_of_existence_event/semantic_know_of_
    existence_event.fpkg
extension/semantic_know_of_relationship_event/semantic_know_of_
    relationship_event.fpkg
extension/semantic_object/semantic_object.fpkg
extension/semantic_object_type/semantic_object_type.fpkg
extension/semantic_ordered_object/semantic_ordered_object.fpkg
extension/semantic_packable_object/semantic_packable_object.fpkg
extension/semantic_physical_object/semantic_physical_object.fpkg
extension/semantic_physical_object_type_relation/semantic_
    physical_object_type_relation.fpkg
extension/semantic_physical_object_type/semantic_physical_object
    _type.fpkg
extension/semantic_proprioception/semantic_proprioception.fpkg
extension/semantic_proprioceptual_object/semantic_proprioceptual
    _object.fpkg
extension/semantic_proprioceptual_orientation/semantic_
    proprioceptual_orientation.fpkg
extension/semantic_proprioceptual_position/semantic_
    proprioceptual_position.fpkg
extension/semantic_realm/semantic_realm.fpkg
extension/semantic_relationship_key/semantic_relationship_key.
    fpkg
extension/semantic_resource_action_event/semantic_resource_
    action_event.fpkg
extension/semantic_resource_action_sequence/semantic_resource_
    action_sequence.fpkg
extension/semantic_resource_event_knowledge_base/semantic_
    resource_event_knowledge_base.fpkg
extension/semantic_resource/semantic_resource.fpkg
extension/semantic_self/semantic_self.fpkg
extension/semantic_situation_category/semantic_situation_
    category.fpkg
extension/semantic_situation/semantic_situation.fpkg
extension/semantic_situation_transition/semantic_situation_
    transition.fpkg
extension/semantic_somatosensation/semantic_somatosensation.fpkg
extension/semantic_somatosensory_object/semantic_somatosensory_
    object.fpkg
extension/semantic_temporal_object/semantic_temporal_object.fpkg
extension/semantic_time/semantic_time.fpkg
extension/semantic_visual_object/semantic_visual_object.fpkg
extension/timeline/timeline.fpkg
test/cairo-test/cairo-test.fpkg
test/concept_version_space-test/concept_version_space-test.fpkg
test/gtk-test/gtk-test.fpkg
test/interval_tree-test/interval_tree-test.fpkg
test/optimize-test/optimize-test.fpkg
test/propogator-test/propogator-test.fpkg
test/timeline-test/timeline-test.fpkg
test/xmlrpc-test/xmlrpc-test.fpkg
example/visualize/framework.py
example/visualize/globalvars.py
example/visualize/graphics.py
example/visualize/graphics_testing.py
example/visualize/physics.py
example/visualize/vector.py
example/visualize/visualization.py
example/visualize/xmlclient.py
example/visualize/xmlserver.py
python/funk2module/setup.py
```

BIBLIOGRAPHY

Ancona, M., Cazzola, W., Dodero, G. & Gianuzzi, V. (1998), Channel Reification: A Reflective Model for Distributed Computation, pp. 32–36.

Anderson, J., Bothell, D., Byrne, M., Douglass, S., Lebiere, C. & Qin, Y. (2004), 'An integrated theory of the mind', *Psychological Review* **111**(4), 1036–1060.

Anderson, J. R. & Corbett, A. T. (1993), Acquisition of LISP Programming Skill, *in* S. Chipman & A. L. Meyrowitz, eds, 'Foundations of Knowledge Acquisition: Cognitive Models of Complex Learning', Kluwer Academic Publishers, Boston, Massachusetts, chapter 1, pp. 1–24.

Anderson, M. & Oates, T. (2007), 'A review of recent research in metareasoning and metalearning', *AI MAGAZINE* **28**(1), 12.

Anderson, M. & Perlis, D. (2005), 'Logic, self-awareness and self-improvement: The metacognitive loop and the problem of brittleness'.

Asch, S. E. (2005), Forming Impressions of Personality, *in* D. L. Hamilton, ed., 'Social Cognition: Key Readings in Social Psychology', Psychology Press, New York, New York, chapter 22, pp. 362–371.

Attiya, H. & Welch, J. (2004), *Distributed computing: fundamentals, simulations, and advanced topics*, Wiley-Interscience.

Baker, C. F., Fillmore, C. J. & Lowe, J. B. (1998), The berkeley framenet project, *in* 'Proceedings of the 17th international conference on Computational linguistics', Association for Computational Linguistics, Morristown, NJ, USA, pp. 86–90.

Barsalou, L. W. (1995), Deriving Categories to Achieve Goals, *in* A. Ram & D. Leake, eds, 'Goal-driven learning', MIT Press, Cambridge, Massachusetts, chapter 5, pp. 121–176.

Baumeister, R., Heatherton, T. & Tice, D. (1993), 'When ego threats lead to self-regulation failure: negative consequences of high self-esteem.', *Journal of Personality and Social Psychology* **64**(1), 141.

Baumeister, R., Vohs, K. & Nathan DeWall, C. (2007), 'How emotion shapes behavior: Feedback, anticipation, and reflection, rather than direct causation', *Personality and Social Psychology Review* **11**(2), 167.

Bedny, M., Pascual-Leone, A. & Saxe, R. (2009), 'Growing up blind does not change the neural bases of theory of mind', *Proceedings of the National Academy of Sciences* **106**(27), 11312.

Bertsekas, D. (1982), 'Distributed dynamic programming', *Automatic Control, IEEE Transactions on* **27**(3), 610–616.

Bertsekas, D. (1995), 'Dynamic programming and optimal control'.

Bivona, U., Ciurli, P., Barba, C., Onder, G., Azicnuda, E., Silvestro, D., Mangano, R., Rigon, J. & Formisano, R. (2008), 'Executive function and metacognitive self-awareness after Severe Traumatic Brain Injury', *Journal of the International Neuropsychological Society* **14**(05), 862–868.

Bobrow, D. G. (1968), Natural Language Input for a Computer Problem-Solving System, *in* M. Minsky, ed., 'Semantic Information Processing', MIT Press, Cambridge, Massachusetts, chapter 3, pp. 135–215.

Boutilier, C. (1996), Planning, learning and coordination in multiagent decision processes, *in* 'Proceedings of the 6th conference on Theoretical aspects of rationality and knowledge', Morgan Kaufmann Publishers Inc. San Francisco, CA, USA, pp. 195–210.

Bowling, M. & Veloso, M. (2000), 'An Analysis of Stochastic Game Theory for Multiagent Reinforcement Learning', *Computer Science Department, Carnegie Mellon University* .

Buchanan, B. & Mitchell, T. (1977), 'Model-directed learning of production rules', *ACM SIGART Bulletin* pp. 44–44.

Burns, T. & Engdahl, E. (1998), 'The social construction of consciousness. Part 2: individual selves, self-awareness, and reflectivity', *Journal of Consciousness Studies* **5**(2), 166–184.

Carbonell, J., Etzioni, O., Gil, Y., Joseph, R., Knoblock, C., Minton, S. & Veloso, M. (1991), 'Prodigy: An integrated architecture for planning and learning', *ACM SIGART Bulletin* **2**(4), 51–55.

Carbonell, J., Etzioni, O., Gil, Y., Joseph, R., Knoblock, C., Minton, S. & Veloso, M. (1995), Planning and Learning in PRODIGY: Overview of an Integrated Architecture, *in* A. Ram & D. Leake, eds, 'Goal-driven learning', MIT Press, Cambridge, Massachusetts, chapter 11, pp. 297–306.

Casebeer, W. & Churchland, P. (2003), 'The neural mechanisms of moral cognition: a multiple-aspect approach to moral judgment and decision-making', *Biology and Philosophy* **18**(1), 169–194.

Cassimatis, N. (2006), 'A cognitive substrate for achieving human-level intelligence', *AI Magazine* **27**(2), 45–56.

Cassimatis, N., Trafton, J., Bugajska, M. & Schultz, A. (2004), 'Integrating cognition, perception and action through mental simulation in robots', *Robotics and Autonomous Systems* **49**(1-2), 13–23.

Castelfranchi, C. (2001), 'The theory of social functions: challenges for computational social science and multi-agent learning', *Cognitive Systems Research* **2**(1), 5–38.

Cazzola, W. (1998), Evaluation of Object-Oriented Reflective Models, *in* 'Object-oriented technology: ECOOP'98 Workshop Reader: ECOOP'98 Workshops, Demos, and Posters: Brussels, Belgium, July 20-24, 1998: Proceedings', Springer, p. 386.

Claus, C. & Boutilier, C. (1998), The dynamics of reinforcement learning in cooperative multiagent systems, *in* 'Proceedings of the National Conference on Artificial Intelligence', JOHN WILEY & SONS LTD, pp. 746–752.

Cooper, R. (1995), Towards an object-oriented language for cognitive modeling, *in* 'Proceedings of the 17 th Annual Conference of the Cognitive Science Society', pp. 556–561.

Cooper, R. (2002), *Modelling high-level cognitive processes*, Lawrence Erlbaum.

Cooper, R. & Fox, J. (1998), 'COGENT: A visual design environment for cognitive modeling', *Behavior Research Methods Instruments and Computers* **30**(4), 553–564.

Cooper, R., Fox, J., Farringdon, J. & Shallice, T. (1996), 'A systematic methodology for cognitive modelling', *Artificial Intelligence* **85**(1-2), 3–44.

Cooper, R., Yule, P., Fox, J. & Sutton, D. (1998), 'COGENT: An environment for the development of cognitive models', *Mind Modelling: A Cognitive Science Approach to Reasoning, Learning and Discovery* pp. 55–82.

Councill, I., Haynes, S. & Ritter, F. (2003), Explaining Soar: Analysis of existing tools and user information requirements, *in* 'Proceedings of the Fifth International Conference on Cognitive Modeling', Citeseer, pp. 63–68.

Cox, M. (1996), 'Introspective multistrategy learning: Constructing a learning strategy under reasoning failure'.

Cox, M. (2005), 'Metacognition in computation: A selected research review', *Artificial intelligence* **169**(2), 104–141.

Cox, M. (2007*a*), Metareasoning, monitoring, and self-explanation, *in* 'Proceedings of the First International Workshop on Metareasoning in Agent-based Systems, AAMAS-07', pp. 46–60.

Cox, M. (2007*b*), 'Perpetual self-aware cognitive agents', *AI MAGAZINE* **28**(1), 32.

Cox, M. & Raja, A. (2008), Metareasoning: A manifesto, *in* 'Proceedings of AAAI 2008 Workshop on Metareasoning: Thinking about Thinking', pp. 1–4.

Cox, M. & Ram, A. (1999*a*), 'Introspective multistrategy learning: On the construction of learning strategies', *Artificial Intelligence* **112**(1-2), 1–56.

Cox, M. T. & Ram, A. (1999*b*), On the Intersection of Story Understanding and Learning, *in* A. Ram & K. Moorman, eds, 'Understanding Language Understanding: Computational Models of Reading', MIT Press, Cambridge, Massachusetts, chapter 11, pp. 397–434.

Crites, R. & Barto, A. (1998), 'Elevator group control using multiple reinforcement learning agents', *Machine Learning* **33**(2), 235–262.

Davis, R. (1980), *Meta-Rules: Reasoning about Control*, Massachusetts Institute of Technology, Artificial Intelligence Laboratory.

desJardins, M. (1995), Goal-Directed Learning: A Decision-Theoretic Model for Deciding What to Learn Next, *in* A. Ram & D. Leake, eds, 'Goal-driven learning', MIT Press, Cambridge, Massachusetts, chapter 8, pp. 241–250.

Dewey, J. (1907), *The School and Society: The School and the Life of the Child*, University of Chicago Press, Chicago, chapter 2, pp. 47–73.

Domeshek, E., Jones, E. & Ram, A. (1999), Capturing the Contents of Complex Narratives, *in* A. Ram & K. Moorman, eds, 'Understanding Language Understanding: Computational Models of Reading', MIT Press, Cambridge, Massachusetts, chapter 4, pp. 73–106.

Dominowski, R. (1998), 'Verbalization and problem solving', *Metacognition in educational theory and practice* pp. 25–45.

Drogoul, A. & Ferber, J. (1994), 'Multi-agent simulation as a tool for modeling societies: Application to social differentiation in ant colonies', *Lecture Notes in Computer Science* **830**, 3–23.

Duval, T. & Lalwani, N. (1999), 'Objective self-awareness and causal attributions for self-standard discrepancies: Changing self or changing standards of correctness', *Personality and Social Psychology Bulletin* **25**(10), 1220.

Duval, T. & Silvia, P. (2002), 'Self-awareness, probability of improvement, and the self-serving bias', *Journal of Personality and Social Psychology* **82**(1), 49–61.

Džeroski, S., De Raedt, L. & Driessens, K. (2001*a*), 'Relational reinforcement learning', *Machine Learning* **43**(1), 7–52.

Džeroski, S., De Raedt, L. & Driessens, K. (2001*b*), 'Relational reinforcement learning', *Machine Learning* **43**(1), 7–52.

Fellbaum, C. (1998), *WordNet: An electronic lexical database*, MIT press.

Fischer, F., Rovatsos, M. & Weiss, G. (2004), Hierarchical reinforcement learning in communication-mediated multiagent coordination, *in* 'Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 3', IEEE Computer Society Washington, DC, USA, pp. 1334–1335.

Focquaert, F. (2008), 'An Evolutionary Cognitive Neuroscience Perspective on Human Self-awareness and Theory of Mind', *Philosophical Psychology* **21**(1), 47–68.

Forbus, K. (1994), 'Qualitative process theory: Twelve years after', *Artificial Intelligence in Perspective* **59**(1), 115.

Fox, S. & Leake, D. B. (2001), 'Introspective reasoning for index refinement in case-based reasoning', *J. Exp. Theor. Artif. Intell.* **13**(1), 63–88.

Gentner, D. (1983), 'Structure-mapping: A theoretical framework for analogy*', *Cognitive science* **7**(2), 155–170.

Gerrig, R. J. (1999), Text Processing and Narrative Worlds, *in* A. Ram & K. Moorman, eds, 'Understanding Language Understanding: Computational Models of Reading', MIT Press, Cambridge, Massachusetts, chapter 13, pp. 461–482.

Ghavamzadeh, M., Mahadevan, S. & Makar, R. (2006), 'Hierarchical multi-agent reinforcement learning', *Autonomous Agents and Multi-Agent Systems* **13**(2), 197–229.

Gilbert, D. T., Pelham, B. W. & Krull, D. S. (2005), On Cognitive Busyness: When Person Perceives Meet Persons Perceived, *in* D. L. Hamilton, ed., 'Social Cognition: Key Readings in Social Psychology', Psychology Press, New York, New York, chapter 19, pp. 314–323.

Gordon, A. & Hobbs, J. (2004), 'Formalizations of commonsense psychology', *AI Magazine* **25**(4), 49–62.

Gordon, A., Hobbs, J. & Cox, M. (2008), Anthropomorphic Self-Models for Metareasoning Agents, *in* 'Aaai-08 workshop: metareasoning: Thinking about thinking'.

Goverover, Y., Johnston, M., Toglia, J. & Deluca, J. (2007), 'Treatment to improve self-awareness in persons with acquired brain injury.', *Brain Inj* **21**(9), 913–23.

Guestrin, C., Koller, D., Gearhart, C. & Kanodia, N. (2003), Generalizing plans to new environments in relational mdps, *in* 'In International Joint Conference on Artificial Intelligence (IJCAI-03', Citeseer.

Guestrin, C., Lagoudakis, M. & Parr, R. (2002), Coordinated reinforcement learning, *in* 'MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-', pp. 227–234.

Halstead, R. (1990), 'New ideas in parallel lisp: Language design, implementation, and programming tools', *Parallel Lisp: Languages and Systems* pp. 1–57.

Hammond, K. (1990), 'Explaining and repairing plans that fail', *Artificial Intelligence* **45**(1-2), 173–228.

Hammond, K. J. & Seifert, C. M. (1993), A Cognitive Science Approach to Case-Based Planning, *in* S. Chipman & A. L. Meyrowitz, eds, 'Foundations of Knowledge Acquisition: Cognitive Models of Complex Learning', Kluwer Academic Publishers, Boston, Massachusetts, chapter 7, pp. 245–268.

Hansen, E. & Zilberstein, S. (2001), 'Monitoring and control of anytime algorithms: A dynamic programming approach', *Artificial Intelligence* **126**(1), 139–157.

Hare, R. D. (1993), *Without Conscience: The Disturbing World of the Psychopaths Among Us*, The Guilford Press.

Harsanyi, J. (1980), 'Rule utilitarianism, rights, obligations and the theory of rational behavior', *Theory and Decision* **12**(2).

Hayes-Roth, B. (1995), 'An architecture for adaptive intelligent systems', *Artificial Intelligence* **72**(1-2), 329–365.

Hinton, L. (2008), The Role of Leader Self-Awareness in Building Trust and Improving Student Learning, PhD thesis, Loyola University Chicago.

Hobson, P., Chidambi, G., Lee, A. & Meyer, J. (2006), 'Self awareness in developmental perspective', *Monographs of the Society for Research in Child Development* **71**(2), 1–28.

Horty, J. (1994), 'Moral dilemmas and nonmonotonic logic', *Journal of philosophical logic* **23**(1), 35–65.

Hu, J. & Wellman, M. (1998), Multiagent reinforcement learning: Theoretical framework and an algorithm, *in* 'Proceedings of the Fifteenth International Conference on Machine Learning', Vol. 242, Citeseer, p. 250.

Huffman, S. B., Pearson, D. J. & Laird, J. E. (1993), Correcting Imperfect Domain Theories: A Knowledge-Level Analysis, *in* S. Chipman & A. L. Meyrowitz, eds, 'Foundations of Knowledge Acquisition: Cognitive Models of Complex Learning', Kluwer Academic Publishers, Boston, Massachusetts, chapter 6, pp. 209–244.

Hume, D. (1902), *Enquiries concerning the human understanding: and concerning the principles of morals*, Vol. 921, Clarendon Press.

Hutchinson, L. & Skinner, N. (2007), 'Self-awareness and cognitive style: Relationships among adaption-innovation, self-monitoring, and self-consciousness', *Social Behavior and Personality* **35**(4), 551–560.

Iba, W. & Langley, P. (2011), 'Exploring moral reasoning in a cognitive architecture'.

Iran-Nejad, A. & Gregg, M. (2001), 'The brain-mind cycle of reflection', *The Teachers College Record* **103**(5), 868–895.

Jaynes, E. (1982), 'On the rationale of maximum-entropy methods', *Proceedings of the IEEE* **70**(9), 939–952.

Kaelbling, L., Littman, M. & Moore, A. (1996), 'Reinforcement learning: A survey', *Arxiv preprint cs.AI/9605103* .

Kapetanakis, S. & Kudenko, D. (2002), Reinforcement learning of coordination in cooperative multi-agent systems, *in* 'Proceedings of the National Conference on Artificial Intelligence', Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, pp. 326–331.

Keenan, J. & Gorman, J. (2007), 'The Causal Role of the Right Hemisphere in Self-Awareness: It is the Brain that is Selective', *Cortex* **43**(8), 1074–1082.

Kieras, D. E. (1993), Learning Schemas from Explanations in Practical Electronics, *in* S. Chipman & A. L. Meyrowitz, eds, 'Foundations of Knowledge Acquisition: Cognitive Models of Complex Learning', Kluwer Academic Publishers, Boston, Massachusetts, chapter 3, pp. 83–118.

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.

Laird, J., Newell, A. & Rosenbloom, P. (1987), 'Soar: An architecture for general intelligence', *Artificial intelligence* **33**(1), 1–64.

Langley, P. (2005), An adaptive architecture for physical agents, *in* 'Web Intelligence, 2005. Proceedings. The 2005 IEEE/WIC/ACM International Conference on', pp. 18–25.

Langley, P., Laird, J. & Rogers, S. (2008), 'Cognitive architectures: Research issues and challenges', *Cognitive Systems Research* .

Leake, D. B. (1995*a*), Goal-Based Explanation Evaluation, *in* A. Ram & D. Leake, eds, 'Goal-driven learning', MIT Press, Cambridge, Massachusetts, chapter 9, pp. 251–286.

Leake, D. B. (1995*b*), Toward Goal-Driven Integration of Explanation and Action, *in* A. Ram & D. Leake, eds, 'Goal-driven learning', MIT Press, Cambridge, Massachusetts, chapter 20, pp. 455–478.

Leake, D. B. (1996), 'Experience, introspection and expertise: Learning to refine the case-based reasoning process', *J. Exp. Theor. Artif. Intell.* **8**(3-4), 319–339.

Lenat, D. B., Guha, R. V., Pittman, K., Pratt, D. & Shepherd, M. (1990), 'Cyc: toward programs with common sense', *Commun. ACM* **33**(8), 30–49.

Levin, D. & Beck, M. (2004), 'Thinking about seeing: Spanning the difference between metacognitive failure and success', *Thinking and seeing: Visual metacognition in adults and children* pp. 121–144.

Liu, H. & Singh, P. (2004*a*), Commonsense reasoning in and over natural language, *in* 'Knowledge-Based Intelligent Information and Engineering Systems', Springer, pp. 293–306.

Liu, H. & Singh, P. (2004*b*), 'Conceptnet—a practical commonsense reasoning tool-kit', *BT technology journal* **22**(4), 211–226.

Lynn, A. (2008), *The eq interview: finding employees with high emotional intelligence*, Amacom Books, chapter 3: finding employees with high emotional intelligence.

Maes, P. (1987), 'Concepts and experiments in computational reflection', *SIGPLAN Not.* **22**(12), 147–155.

Maes, P. (1988), Issues in computational reflection, *in* 'Meta-level architectures and reflection', North-Holland, pp. 21–35.

Maes, P. & Brooks, R. (1990), Learning to coordinate behaviors, *in* 'Proceedings of the Eighth National Conference on Artificial Intelligence', Vol. 7902.

Mahesh, K., Eiselt, K. P. & Holbrook, J. K. (1999), Sentence Processing in Understanding: Interaction and Integration of Knowledge Sources, *in* A. Ram & K. Moorman, eds, 'Understanding Language Understanding: Computational Models of Reading', MIT Press, Cambridge, Massachusetts, chapter 3, pp. 27–72.

Mahncke, H., Connor, B., Appelman, J., Ahsanuddin, O., Hardy, J., Wood, R., Joyce, N., Boniske, T., Atkins, S. & Merzenich, M. (2006), 'Memory enhancement in healthy older adults using a brain plasticity-based training program: a randomized, controlled study', *Proceedings of the National Academy of Sciences* **103**(33).

Marinier, R., Laird, J. & Lewis, R. (2009), 'A computational unification of cognitive behavior and emotion', *Cognitive Systems Research* **10**(1), 48–69.

Matarić, M. (1997*a*), 'Learning social behavior', *Robotics and Autonomous Systems* **20**, 191–204.

Matarić, M. (1997*b*), 'Reinforcement learning in the multi-robot domain', *Autonomous Robots* **4**(1), 73–83.

Matsuoka, S., Watanabe, T., Ichisugi, Y. & Yonezawa, A. (1992), 'Object-Oriented Concurrent Reflective Architectures', *Object-Based Concurrent Computing* pp. 211–226.

Matuszek, C., Witbrock, M., Kahlert, R., Cabral, J., Schneider, D., Shah, P. & Lenat, D. (2005), Searching for common sense: Populating cyc from the web, *in* 'Proceedings of the National Conference on Artificial Intelligence', Vol. 20, Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, p. 1430.

McCarthy, J. (1994), 'Elephant 2000: A programming language based on speech acts', *Unpublished Manuscript, Stanford University* .

McDermott, D. (1987), 'Logic, problem solving, and deduction', *Annual Review of Computer Science* **2**(1), 187–229.

Menon, T., Morris, M. W., yue Chiu, C. & yi Hong, Y. (2005), Culture and the Construal of Agency: Attribution to Individual Versus Group Dispositions, *in* D. L. Hamilton, ed., 'Social Cognition: Key Readings in Social Psychology', Psychology Press, New York, New York, chapter 21, pp. 333–353.

Michalski, R. & Ram, A. (1995), Learning as Goal-Driven Inference, *in* A. Ram & D. Leake, eds, 'Goal-driven learning', MIT Press, Cambridge, Massachusetts, chapter 21, pp. 479–490.

Michalsky, T., Zion, M. & Mevarech, Z. (2007), 'Developing Students' Metacognitive Awareness in Asynchronous Learning Networks in Comparison to Face-to-Face Discussion Groups', *Journal of Educational Computing Research* **36**(4), 395–424.

Minsky, M. (1968), Matter, Mind, and Models, *in* M. Minsky, ed., 'Semantic Information Processing', MIT Press, Cambridge, Massachusetts, chapter 9, pp. 425–432.

Minsky, M. (1975), 'A Framework for Representing Knowledge', *The Psychology of Computer Vision* pp. 211–279.

Minsky, M. (1988), *The society of mind*, Simon and Schuster.

Minsky, M. (1991), Logical vs. analogical or symbolic vs. connectionist or neat vs. scruffy, *in* 'Artificial intelligence at MIT expanding frontiers', MIT press, pp. 218–243.

Minsky, M. (2006), *The Emotion Machine: Commonsense Thinking, Artificial Intelligence, and the Future of the Human Mind*, Simon & Schuster, New York, New York.

Mirvis, P. (2008), 'Executive Development Through Consciousness-Raising Experiences', *The Academy of Management Learning and Education (AMLE)* **7**(2), 173–188.

Moll, J., Zahn, R., de Oliveira-Souza, R., Krueger, F. & Grafman, J. (2005), 'The neural basis of human moral cognition', *Nature Reviews Neuroscience* **6**(10), 799–809.

Moorman, K. & Ram, A. (1999), Creativity in Reading: Understanding Novel Concepts, *in* A. Ram & K. Moorman, eds, 'Understanding Language Understanding: Computational Models of Reading', MIT Press, Cambridge, Massachusetts, chapter 10, pp. 359–396.

Morgan, B. (2009), 'Funk2: A distributed processing language for reflective tracing of a large critic-selector cognitive architecture', *Proceedings of the Metacognition Workshop at the Third IEEE International Conference on Self-Adaptive and Self-Organizing Systems* .

Morgan, B. (2010), 'A computational theory of the communication of problem solving knowledge between parents and children', *PhD Proposal* .

Morgan, B. (2011), 'Moral compass: Commonsense social reasoning cognitive architecture', *Commonsense Tech Note* .
URL: *http://em-two.net/about/*

Morin, A. (2004), 'Levels of consciousness and self-awareness: A comparison and integration of various views'.

Mueller, E. (1990), *Daydreaming in humans and machines: a computer model of the stream of thought*, Intellect Books.

Muggleton, S. (1992), *Inductive logic programming*, Morgan Kaufmann.

Mulholland, P. & Watt, S. (1998), Hank: A friendly cognitive modelling language for psychology students, *in* 'Proceedings of the IEEE Symposium on Visual Languages', IEEE Computer Society Washington, DC, USA, p. 210.

Mulholland, P. & Watt, S. (1999), Programming with a purpose: Hank, gardening and schema theory, *in* 'Proceedings 1 Ith Annual Workshop of the Psychology of Programming Interest Group', Citeseer.

Mulholland, P. & Watt, S. (2000), 'Learning by building: A visual modelling language for psychology students', *Journal of Visual Languages and Computing* **11**(5), 481–504.

Murphy, K., Weiss, Y. & Jordan, M. (1999), Loopy belief propagation for approximate inference: An empirical study, *in* 'Proceedings of Uncertainty in AI', pp. 467–475.

Musliner, D. (2001), 'Imposing real-time constraints on self-adaptive controller synthesis', *Lecture Notes in Computer Science* **1936**, 143–160.

Nagayuki, Y., Ishii, S. & Doya, K. (2000), Multi-agent reinforcement learning: an approach based on the otheragent's internal model, *in* 'MultiAgent Systems, 2000. Proceedings. Fourth International Conference on', pp. 215–221.

Newell, A. (1982), 'The knowledge level', *Artif. Intell.* **18**(1), 87–127.

Newell, A. (1990), 'Unified theories of cognition'.

Newell, A., Shaw, C. & Simon, H. (1959), Report on a general problem-solving program, *in* 'Proceedings of the International Conference on Information Processing', pp. 256–264.

Ng, E. & Bereiter, C. (1995), Three Levels of Goal Orientation in Learning, *in* A. Ram & D. Leake, eds, 'Goal-driven learning', MIT Press, Cambridge, Massachusetts, chapter 14, pp. 355–380.

Noble, J. & Franks, D. (2004), 'Social learning in a multi-agent system', *Computing and Informatics* **22**(6), 561–574.

Nowe, A., Parent, J. & Verbeeck, K. (2001), 'Social agents playing a periodical policy', *Lecture notes in computer science* pp. 382–393.

Ohlsson, S. (1993), The Interaction Between Knowledge and Practice in the Acquisition of Cognitive Skills, *in* S. Chipman & A. L. Meyrowitz, eds, 'Foundations of Knowledge Acquisition: Cognitive Models of Complex Learning', Kluwer Academic Publishers, Boston, Massachusetts, chapter 5, pp. 147–208.

Oliva, A., Garcia, I. C. & Buzato, L. E. (1998), 'The Reflective Architecture of GuaranÃ¡'.

Orkin, J. & Roy, D. (2009), Automatic learning and generation of social behavior from collective human gameplay, *in* 'Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1', International Foundation for Autonomous Agents and Multiagent Systems, pp. 385–392.

Orkin, J., Smith, T., Reckman, H. & Roy, D. (2010), Semi-automatic task recognition for interactive narratives with eat & run, *in* 'Proceedings of the Intelligent Narrative Technologies III Workshop', ACM, pp. 1–8.

Panait, L. & Luke, S. (2005), 'Cooperative multi-agent learning: The state of the art', *Autonomous Agents and Multi-Agent Systems* **11**(3), 387–434.

Park, K., Kim, Y. & Kim, J. (2001), 'Modular Q-learning based multi-agent cooperation for robot soccer', *Robotics and Autonomous Systems* **35**(2), 109–122.

Perner, J. (1991), *Understanding the representational mind*, MIT Press.

Peterson, J. & Billman, D. (1999), Semantic Correspondence Theory, *in* A. Ram & K. Moorman, eds, 'Understanding Language Understanding: Computational Models of Reading', MIT Press, Cambridge, Massachusetts, chapter 9, pp. 299–358.

Phillips, A. & Silvia, P. (2005), 'Self-awareness and the emotional consequences of self-discrepancies', *Personality and Social Psychology Bulletin* **31**(5), 703.

Price, B. & Boutilier, C. (1999), Implicit imitation in multiagent reinforcement learning, *in* 'MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-', MORGAN KAUFMANN PUBLISHERS, INC., pp. 325–334.

Pryor, L. & Collins, G. (1995), Planning to Perceive, *in* A. Ram & D. Leake, eds, 'Goal-driven learning', MIT Press, Cambridge, Massachusetts, chapter 10, pp. 287–296.

Ram, A. (1999), A Theory of Questions and Question Asking, *in* A. Ram & K. Moorman, eds, 'Understanding Language Understanding: Computational Models of Reading', MIT Press, Cambridge, Massachusetts, chapter 8, pp. 253–298.

Ram, A. & Cox, M. T. (1995), Introspective Reasoning Using Meta-Explanations for Multistrategy Learning, *in* A. Ram & D. Leake, eds, 'Goal-driven learning', MIT Press, Cambridge, Massachusetts, chapter 7, pp. 211–240.

Ram, A., Cox, M. T. & Narayanan, S. (1995), Goal-Driven Learning in Multistrategy Reasoning and Learning Systems, *in* A. Ram & D. Leake, eds, 'Goal-driven learning', MIT Press, Cambridge, Massachusetts, chapter 18, pp. 421–438.

Ram, A. & Leake, D. (1995*a*), *Goal-driven learning*, MIT Press.

Ram, A. & Leake, D. (1995*b*), 'Learning, goals, and learning goals', *Goal-driven learning* pp. 1–37.

Ram, A. & Moorman, K. (1999), Toward a Theory of Reading and Understanding, *in* A. Ram & K. Moorman, eds, 'Understanding Language Understanding: Computational Models of Reading', MIT Press, Cambridge, Massachusetts, chapter 1, pp. 1–10.

Rapaport, W. J. & Shapiro, S. C. (1999), Cognition and Fiction, *in* A. Ram & K. Moorman, eds, 'Understanding Language Understanding: Computational Models of Reading', MIT Press, Cambridge, Massachusetts, chapter 2, pp. 11–26.

Rapoport, A. (2001), *N-person game theory: Concepts and applications*, Courier Dover Publications.

Roberts, R. & Goldstein, I. (1977), 'The FRL primer. Memo 408', *Artificial Intelligence Laboratory, Massachusetts Institute of Technology* .

Rosenbloom, P. S., Lee, S. & Unruh, A. (1993), Bias in Planning and Explanation-Based Learning, *in* S. Chipman & A. L. Meyrowitz, eds, 'Foundations of Knowledge Acquisition: Cognitive Models of Complex Learning', Kluwer Academic Publishers, Boston, Massachusetts, chapter 8, pp. 269–308.

Ross, L. D., Amabile, T. M. & Steinmetz, J. L. (2005), Social Roles, Social Control, and Biases in Social-Perception Processes, *in* D. L. Hamilton, ed., 'Social Cognition: Key Readings in Social Psychology', Psychology Press, New York, New York, chapter 20, pp. 324–332.

Sandholm, T. & Crites, R. (1996), 'Multiagent reinforcement learning in the iterated prisoner's dilemma', *Biosystems* **37**(1-2), 147–166.

Saxe, R., Schulz, L. & Jiang, Y. (2006), 'Reading minds versus following rules: Dissociating theory of mind and executive control in the brain', *Social Neuroscience* **1**(3-4), 284.

Schank, R., Goldman, N., Rieger, C. & Riesbeck, C. (1972), 'Primitive concepts underlying verbs of thought.'.

Schubert, L. (2005), Some KR&R Requirements for Self-Awareness, *in* 'Metacognition in Computation: Papers from 2005 AAAI Spring Symposium', pp. 106–113.

Sen, S. & Sekaran, M. (1998), 'Individual learning of coordination knowledge', *Journal of Experimental & Theoretical Artificial Intelligence* **10**(3), 333–356.

Sen, S. & Weiss, G. (1999), '6 Learning in Multiagent Systems', *Multiagent systems: A modern approach to distributed artificial intelligence* p. 259.

Shannon, C. (1959), 'A mathematical theory of communication', *The Bell System Technical Journal* **27**, 379–423, 623–656.

Shapiro, S. & Ismail, H. (2003), 'Anchoring in a grounded layered architecture with integrated reasoning', *Robotics and Autonomous Systems* **43**(2-3), 97–108.

Shoham, Y., Powers, R. & Grenager, T. (2004), Multi-agent reinforcement learning: a critical survey, *in* 'AAAI Fall Symposium on Artificial Multi-Agent Learning'.

Silverman, E. (2008), 'Ongoing Self-Reflection', *American Journal of Speech-Language Pathology* **17**(1), 92.

Silvia, P., Phillips, A., Baumgaertner, M. & Maschauer, E. (2006), 'Emotion concepts and self-focused attention: Exploring parallel effects of emotional states and emotional knowledge', *Motivation and Emotion* **30**(3), 225–231.

Simon, H. (1972), 'Theories of bounded rationality', *Decision and organization* **1**, 161–176.

Singh, P. (2005), EM-ONE: An Architecture for Reflective Commonsense Thinking, PhD thesis, Massachusetts Institute of Technology.

Singh, P., Lin, T., Mueller, E., Lim, G., Perkins, T. & Li Zhu, W. (2002), 'Open mind common sense: Knowledge acquisition from the general public', *On the Move to Meaningful Internet Systems 2002: CoopIS, DOA, and ODBASE* pp. 1223–1237.

Sloman, A. (2001), Varieties of affect and the CogAff architecture schema, *in* 'Proceedings Symposium on Emotion, Cognition, and Affective Computing AISB', Vol. 1, pp. 39–48.

Smith, D. (2006), Modeling and Reasoning about Everyday Events, Master's thesis, Massachusetts Institute of Technology.

Smith, D., Frank, J. & Jónsson, A. (2000), 'Bridging the gap between planning and scheduling', *The Knowledge Engineering Review* **15**(1), 47–83.

Smith, D. & Morgan, B. (2010), Isisworld: An open source commonsense simulator for ai researchers, *in* 'Workshops at the Twenty-Fourth AAAI Conference on Artificial Intelligence'.

Smith, E. (2007), Self-awareness, perspective-taking, and self-face recognition, PhD thesis, NORTHWESTERN UNIVERSITY.

Sobel, J. M. & Friedman, D. P. (1996), An Introduction to Reflection-Oriented Programming, *in* 'Proceedings of Reflection 1996'.

Soeiro, L. G. (2008), State Versus Trait Self-Focus: Comparing Self-Awareness to Self-Consciousness, PhD thesis, Richard L. Conolly College, Long Island University.

Speer, R., Krishnamurthy, J., Havasi, C., Smith, D., Lieberman, H. & Arnold, K. (2009), An interface for targeted collection of common sense knowledge using a mixture model, *in* 'IUI '09: Proceedings of the 13th international conference on Intelligent user interfaces', ACM, New York, NY, USA, pp. 137–146.

Stein, G. & Barnden, J. (1995), Towards more flexible and commonsensical reasoning about beliefs, *in* 'Proceedings of the 1995 AAAI Spring Symposium on Representing Mental States and Mechanisms', pp. 127–135.

Stone, P. (1997), Layered Learning in Multiagent Systems, *in* 'Proceedings of the Fourteenth National Conference on Artificial Intelligence and the Ninth Innovative Applications of Artificial Intelligence Conference', Amer Assn for Artificial, p. 819.

Stone, P. & Veloso, M. (2000), 'Multiagent systems: A survey from a machine learning perspective', *Autonomous Robots* **8**(3), 345–383.

Sutton, R. & Barto, A. (1998), *Reinforcement learning: An introduction*, Vol. 9, MIT Press.

Tan, M. (1997), 'Multi-agent reinforcement learning: Independent vs. cooperative agents', *Readings in agents* pp. 487–494.

Taylor, S. N. (2007), A Conceptual Framework and Empirical Test for Leader Attunement: Toward a Theory of Leader Self-Awareness, PhD thesis, Case Western Reserve University.

Thagard, P. & Millgram, E. (1995), Inference to the Best Plan: A Coherence Theory of Decision, *in* A. Ram & D. Leake, eds, 'Goal-driven learning', MIT Press, Cambridge, Massachusetts, chapter 19, pp. 439–454.

Turner, S. (1994), *The creative process: a computer model of storytelling and creativity*, Lawrence Erlbaum.

Uhlmann, E., Pizarro, D. & Bloom, P. (2008), 'Varieties of Social Cognition', *Journal for the Theory of Social Behaviour* **38**(3), 293–322.

VanLehn, K. & Jones, R. (1992), 'A model of the self-explanation effect', *Journal of the Learning Sciences* pp. 1–59.

VanLehn, K. & Jones, R. M. (1993), Learning by Explaining Examples to Oneself: A Computational Model, *in* S. Chipman & A. L. Meyrowitz, eds, 'Foundations of Knowledge Acquisition: Cognitive Models of Complex Learning', Kluwer Academic Publishers, Boston, Massachusetts, chapter 2, pp. 25–82.

Vince, R. (2002), 'Organizing reflection', *Management Learning* **33**(1), 63.

Wang, X. & Sandholm, T. (2003), 'Reinforcement learning to play an optimal Nash equilibrium in team Markov games', *Advances in neural information processing systems* pp. 1603–1610.

Watanabe, T. & Yonezawa, A. (1989), Reflective Computation in Object-Oriented Concurrent Systems and its Applications, *in* 'IWSSD '89: Proceedings of the 5th International Workshop on Software Specification and Design', ACM, New York, NY, USA, pp. 56–58.

Weber, C., Morwood, D. & Bryce, D. (2011), 'Goal-directed knowledge acquisition', *Proceedings of ICML 2011 Workshop: Planning and Acting with Uncertain Models* .

Wei, G. (1995), 'Adaptation and learning in multi-agent systems: Some remarks and a bibliography', *Adaptation and Learning in Multi-Agent Systems* pp. 1–21.

Whitehead, C. (2001), 'Social mirrors and shared experiential worlds', *Journal of Consciousness Studies* **8**(4), 3–36.

Wilensky, R. (1993), Knowledge Acquisition and Natural Language Processing, *in* S. Chipman & A. L. Meyrowitz, eds, 'Foundations of Knowledge Acquisition: Cognitive Models of Complex Learning', Kluwer Academic Publishers, Boston, Massachusetts, chapter 9, pp. 209–334.

Williams, R., Barry, B. & Singh, P. (2005), ComicKit: acquiring story scripts using common sense feedback, *in* 'Proceedings of the 10th international conference on Intelligent user interfaces', ACM New York, NY, USA, pp. 302–304.

Wilson, T. & Nisbett, R. (1977), 'Telling more than we can know: Verbal reports on mental processes', *Psychological Review* **84**, 231–259.

Winograd, T. (1970), Procedures as a Representation for Data in a Computer Program for Understanding Natural Language, PhD thesis, Massachusetts Institute of Technology.

Winston, P. (1970), Learning structural descriptions from examples, PhD thesis, Massachusetts Institute of Technology.

Winter, L. & Uleman, J. S. (2005), When Are Social Judgments Made? Evidence for the Spontaneuousness of Trait Inferences, *in* D. L. Hamilton, ed., 'Social Cognition: Key Readings in Social Psychology', Psychology Press, New York, New York, chapter 18, pp. 299–313.

Yang, E. & Gu, D. (2004), 'Multiagent reinforcement learning for multi-robot systems: A survey', *University of Essex Technical Report CSM-404, Department of Computer Science* .

Yedidia, J., Freeman, W. & Weiss, Y. (2005), 'Constructing free-energy approximations and generalized belief propagation algorithms', *Information Theory, IEEE Transactions on* **51**(7), 2282–2312.

Yonezawa, A. (1990), A reflective object oriented concurrent language ABCL/R, *in* 'Proceedings of the US/Japan workshop on Parallel Lisp on Parallel Lisp: languages and systems table of contents', Springer, pp. 254–256.

Young, L. & Phillips, J. (2011), 'The paradox of moral focus', *Cognition* **119**(2), 166–178.

Yu, A. & Dayan, P. (2003), Expected and unexpected uncertainty: Ach and ne in the neocortex, *in* 'Advances in Neural Information Processing Systems 15: Proceedings of the 2002 Conference', Vol. 15, The MIT Press, p. 173.

Yule, P. & Cooper, R. (2001), Towards a technology for computational experimentation, *in* 'Proceedings of the 4 th International Conference on Cognitive Modelling', pp. 223–228.

COLOPHON

This thesis was typeset with LaTeX 2ε using Hermann Zapf's *Palatino* and *Euler* type faces (Type 1 PostScript fonts *URW Palladio L* and *FPL* were used). The listings are typeset in *Bera Mono*, originally developed by Bitstream, Inc. as "Bitstream Vera". (Type 1 PostScript fonts were made available by Malte Rosenau and Ulrich Dirr.)

The typographic style was inspired by Bringhurst's genius as presented in *The Elements of Typographic Style* (Bringhurst 2002). It is available for LaTeX via CTAN as "`classicthesis`".

NOTE: The custom size of the textblock was calculated using the directions given by Mr. Bringhurst (pages 26–29 and 175/176). 10 pt Palatino needs 133.21 pt for the string "abcdefghijklmnopqrstuvwxyz". This yields a good line length between 24–26 pc (288–312 pt). Using a "*double square textblock*" with a 1:2 ratio this results in a textblock of 312:624 pt (which includes the headline in this design). A good alternative would be the "*golden section textblock*" with a ratio of 1:1.62, here 312:505.44 pt. For comparison, `DIV9` of the `typearea` package results in a line length of 389 pt (32.4 pc), which is by far too long. However, this information will only be of interest for hardcore pseudo-typographers like me.

To make your own calculations, use the following commands and look up the corresponding lengths in the book:

```
\settowidth{\abcd}{abcdefghijklmnopqrstuvwxyz}
\the\abcd\ % prints the value of the length
```

Please see the file `classicthesis.sty` for some precalculated values for Palatino and Minion.

145.86469pt

DECLARATION

I, Bo Morgan, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Cambridge, August 2011

Bo Morgan