Chapter 1

Introduction

The twentieth century is full of science fiction dreams of robots. We have Asimov's Robot series, Star Trek's Data, and HAL from "2001: A Space Odyssey". Recent Artificial Intelligence research on autonomous agents made the dream of robots that interact with humans in a human environment a goal. The current level of autonomous agent research is not near the sophistication of science fiction robots, but any autonomous agent shares with these robots the ability to perceive and interact with the environment. This direction of research has at its core the agent's self sufficiency and ability to perceive the environment and to communicate with (or manipulate) the environment.

Machine learning researchers argue that a self sufficient agent in a human environment must learn and adapt. The ability to learn is vital both because real environments are constantly changing and because no programmer can account for every possible situation when building an agent. The ultimate goal of learning research is machines that learn to interact with their environments. This thesis in concerned with machines that learn the effects of their actions on the environment — namely, a world model.

Before diving into the specifics of the problem addressed in this thesis, let us imagine the future of learning machines. In the following scenario, a robot training technician is qualified to supervise learning robots. She describes working with a secretary robot that has no world knowledge initially and learns to communicate and perform the tasks of a secretary.

September 22, 2xxx

Dear Mom,

I just finished training a desk-top secretary — one of the high-tech secretaries that practically run a whole office by themselves. It's hard to believe that in just a week, a pile of metal can learn to be such a useful and resourceful tool. I think you'll find the training process of this robot interesting.

The secretary robots are interesting because they learn *how to do their job. Unlike the simple communicator that you and I have, you don't have to input all of the information the secretary robot needs (like addresses and telephone numbers). Rather it learns the database as you use it. For example, if you wanted to view someone whose location wasn't in your database, the secretary will find it and get him on view for you. It can also learn new procedures as opposed to our hardwired communicators which means it improves and changes with your needs. I can't wait until these things become cheap enough for home use.*

The training begins by letting the machine experiment and try things almost at random. I spent a whole day just making sure it doesn't cause any major disasters by sending a bad message to an important computer. After a day of mindless playing around the secretary understood the effects of its actions well enough to move to the next training phase. It had to learn to speak first, which it did adequately after connecting to the Language Acquisition Center for a couple of hours.

At this point my work began. I had to train the secretary's office skills. I gave it tasks to complete and reinforced good performance. If it was unsuccessful I showed it how to do the task. At this stage in training the robot is also allowed to ask for explanations, which I had to answer. This process is tedious because you have to repeat tasks many times until the training is sufficiently ingrained. It still doesn't perform perfectly, but the owners understand that she will continue to improve.

The last phase of training is at the secretary's future work place. The secretary's owner trains it in specific office procedures, and it accumulates the database for its office.

I am anxiously awaiting my next assignment — a mobile robot...

Ruti

The secretary robot in the above scenario is different from current machines (robots or software applications) because it leaves the factory with a learning program but without world knowledge or task oriented knowledge. Current technology does not rely on learning. Rather, machines leave the factory with complete, hardwired knowledge of their task and necessary aspects of their work environment. Any information specific to the work-place must be given to the machine manually. For example, the user of a fax program must enter fax numbers explicitly. Unlike the secretary robot, the program cannot learn additional numbers by accessing a directory independently. The remainder of this section discusses the novel aspects of the machine learning research in this thesis.

The agents in this research, like the robot in the futuristic letter, are autonomous agents. An autonomous agent perceives its environment directly and can take actions, such as move an object or go to a place, which affect its environment directly. It acts autonomously based on its own perceptions and goals, and makes use of what it knows or has learned about the world. Although people may give the agent a high level goal, the agent possesses internal goals and motivations, such as survival and avoiding negative reinforcement.

Any autonomous agent repeats the following basic behavior cycle:

Algorithm 1 Agent

repeat forever

  save old perceptions

  select the next action

  perceive

  learn

  predict and/or plan

Steps in the cycle may be emphasized or de-emphasized in different situations. For example, if a robot is about to fall off a cliff, a long goal oriented planning step is superfluous. The action selection, therefore, uses a planning and decision making algorithm which relies heavily on world knowledge. The agent can learn world knowledge from the saved and current perceptions, as well as from predictions. The learning step increases or improves the agent's world knowledge thus improving all of the action selection, prediction, and planning procedures.

An autonomous agent must clearly have a great deal of knowledge about its environment. It must be able to use this knowledge to reason about its environment, predict the effects of its actions, select appropriate actions, and plan ahead to achieve goals. All the above problems — prediction, action selection, planning, and learning — are open problems and important areas of research. This thesis is concerned with how the agent learns world knowledge, which I see as a first step to solving all the remaining problems.

As we see in the secretary robot training scenario there are several stages in learning. In the initial stage, the agent has little or no knowledge about the environment and it learns a general world model. In later stages the agent already has some understanding of the environment and it learns specific domain information, and goal-oriented knowledge. This thesis deals with the initial stages of learning, where the agent has no domain knowledge. It uses the a perceptual interface with its environment and a learning algorithm to learn a world model of its environment.

The robot training scenario also indicates that there are several learning paradigms. The agent learns by experimenting initially. Subsequent learning stages include learning from examples, reinforcement learning, explanation-based learning, and apprenticeship learning. This thesis addresses autonomous learning from experiments as in the early stages of learning. In this learning paradigm, the agent cannot use the help of a teacher. For example, in the early training of the secretary robot, the trainer plays the role of a babysitter more than that of a teacher. The trainer is available in case of emergency which is especially important for mobile robots that can damage themselves. Rather than learn from a teacher, the agent learns through the perceived effects of its own actions. It selects its actions independently; the goal of building an accurate world model is its only motivation.

There is a large body of research on autonomously learning world models. Such research is successful when the agent learns a simple environment, or begins with some knowledge of the environment, but the learning techniques are too slow to be practical for complex domains. This thesis explores complex environments with *manifest causal structure*— a term I use to indicate that the causes for the effects sensed in the environment are generally perceptible. In more common terms, there is little or no "hidden state" in the environment - or rather in the sensory interface of the agent with the environment. This class of environments in unexplored in machine learning literature and this thesis shows that environments with manifest causal structures allow the agent to use simple learning techniques to create a causal model of its environment. Thus the agent can learn a valid world model in a reasonable length of time even for realistic problems.

To summarize, in this thesis an autonomous agent "lives" in a complex environment with manifest causal structure. The agent begins learning with no prior knowledge about the environment and learns a causal model of its environment from direct interaction with the environment. The goal of the work in this thesis is to successfully and efficiently learn a world model.

The agent learns the world model in two phases. First it learns a set of rules that describe the environment in the lowest possible terms of the agent's perceptions. Once the perception based model is adequate, the agent learns higher level concepts using the previously learned rules. Both the rule learning algorithm and the concept learning algorithms are novel. The concept learning is especially exciting since previous learning research has not been successful in learning general concepts in human readable form.

This thesis demonstrates the learning algorithms in the Macintosh environment. The Macintosh environment is a realistic environment, not the deterministic finite state machine common in machine learning research. While it is a complex environment it has manifest causal structure and therefore is a suitable environment for this thesis. In the Macintosh environment, like the secretary robot scenario, the agent must first learn how the operating system responds to its actions. Later this knowledge can be the basis for achieving task in the environment.

The remainder of this chapter has two parts. The first part discusses the motivation for this research. Section 1.1 discusses the manifest causal structure property in detail and illustrates its usefulness and relation with human and animal environments. Section 1.2 gives a brief overview of work on learning world models and contrasts previous approaches to learning causal world models with the approach this thesis. Section 1.3 presents some of the large body of previous work in Artificial

Intelligence that uses causal world models to plan, predict, and reason.

The second part of this chapter is more technical and presents without details the salient ideas of the thesis. Section 1.4 overviews the Macintosh environment in which the agent experiments and learns. Section 1.5 describes the structure of the world model the agent learns, the methodology for learning the causal rules that make up the world model, and the two concept learning paradigms: collapsing correlated perceptions and generalizing rules. For a complete discussion of the algorithms mentioned in this chapter refer to the respective chapter for each topic.

1.1 Manifest Causal Structure

> Definition **manifest**: readily perceived by the senses and esp. by the sight. *synonyms: obvious, evident* [Webster's dictionary]

The category of problems in this thesis are environments with manifest causal structure. As the name indicates, in such environments the agent can in general directly sense the causes for any changes in the environment (that it can perceive). In particular, the agent can sense (almost all) the relevant information to learning the effects of its actions in the environment.

The restriction of environment types to environments with manifest causal structure contrasts research on learning in environments with hidden information (Drescher 1989, Rivest & Schapire 1989, Rivest & Schapire 1990). The manifest causal structure of the environment eliminates the need to search beyond the perceptions for causes to changes in the environment. Thus strategies for learning the world model can be fast and simple (compared with other techniques, such as the schema mechanism (Drescher 1989)).

While the agent may need a great deal of sensory information to achieve the manifest causal structure property, the sensory interface does not necessarily capture the complete state of the environment. This direction of research is in contrast with much of the work on autonomous agent learning, such as Q-learning (Sutton 1990, Sutton 1991, Watkins 1989), where states of the world are enumerated and the agent is given the complete state of the environment. For the manifest causal structure property, locally complete sensory information usually suffices since changes to the local environment can in most cases be explained by the local information.

This thesis draws an important distinction between the true environment in which the agent lives and the environment as the agent perceives it. The true environment is the environment the agent lives in as it truly is, and it may be deterministic or non-deterministic. Notice that a non-deterministic environment is not completely manifest, i.e. its causal structure cannot be captured in all cases. For generality, environments with manifest causal structure can exhibit some unpredictable events as long as the predictable events occur almost all the time.

The environment, as the agent perceives it, is a product of the underlying environment and the agent's perceptual interface. The perceptual interface can make the underlying environments manifest or partially hidden. Typically the perceptual interface will collapse several underlying world states to one perceptual state, thereby hiding some aspects of the environment. Such environments have manifest causal structure if effects are predictable almost all the time.

In simple environments very little sensory data is sufficient to achieve manifest causal structure. For example, consider a room with a single light and a light switch that can be in either on or off position. One binary sensor is sufficient to perceive every aspect of the environment – light on/off. In more complex environments the sensory interface must be much more complicated. For example, in the real world people and other animals have developed very effective sensory organs that perceive the environment (such that they achieve the manifest causal structure property), and they are able to understand the causal structure and react effectively.

I believe the restriction of the problem to environments with manifest causal structure is a natural one. People, as well as other animals, only know how to cope with environments of this type. In fact it is so important to people that their environment be manifest that they go a step beyond the perceptual abilities with which nature endowed them. People built microscopes to perceive cellular level environments, night vision goggles for dark environments, telescopes for very large environments, and particle accelerators for sub-atomic environments. Many agent-environment systems with appropriate sensory interfaces have a manifest causal structure, such as animals in the real world. In an artificial environment it is straightforward to give the agent sufficient sensory data to achieve the manifest causal structure property.

While it is probably believable that software environments with a single agent can have manifest causal structure, it is not clear that the notion generalizes to more complex environments and real-world settings. One complication is the presence of multiple actors in the environment. Very few environments are completely private. Even in one's office the phone may ring or someone could knock on the door. In some cases, such as a private office, occurrences due to other actors may be rare enough that the environment is still predictable almost all the time. Often other actors are continually present and affecting one's environment. An environment with multiple actors can have manifest causal structure if the agent can perceive the actions of other actors and predict the results of their actions. The problem of determining the necessary perceptions in any environment is difficult and remains up to the agent designer.

A complication in real-world environments is the abundance of perceptual stimuli. An agent in an environment with manifest causal structure likewise has many perceptions. The advantage of the large number of perceptions is that the causes of situations are found in the perceptions; the disadvantage is that the search space for the causes is large. In a complex environment it is also possible for the agent to perceive too much. That is, the agent may perceive irrelevant information that makes the environment appear probabilistic or even random, when a more focused set of perceptions would show a predictable relevant subset of the environment.

People are very good at attenuating to relevant aspects of their environment. For example, when I work alone in my office I would immediately respond to a beep on my computer (indicating new mail), but when I am in a meeting I do not seem to hear these beeps at all. People are similarly good at recognizing when they do not perceive enough of the environment, and extend their perceptions by turning on a light or using a magnifying glass for example. Such smart perceptual interfaces may be one way to achieve manifest causal structure in complex environments.

Currently, endowing machines in the real world (i.e., autonomous robots) with perception is very problematic. We do not have the technology to give robots the quality of information that is necessary to achieve an environment with manifest causal structure. Most of the sensors used in robotics are very low level and give only limited information. The more complex sensors such as cameras give a large amount of information, but we do not have efficient ways to interpret this information. As a result, much of the environment remains hidden. Thus, although I believe that if we were able to create a sensory interface for robots that achieves the manifest causal structure property then the techniques for learning and performing in the environment would be useful in robots, I do not expect these techniques to be practical for any robots currently in use.

The following discussion summarizes the possible environment types and under what conditions environments have manifest causal structure. In the remainder of this thesis *environment* refers to the agent's perceived environment and *underlying environment* to the true environment.

There are four types of underlying environment/perceptual interface combinations:

1. deterministic/manifest
2. deterministic/hidden
3. non-deterministic/manifest
4. non-deterministic/hidden

The first environment type is deterministic from the agent's perspective. The environment is essentially a finite automaton (assuming a finite number of perceptions) and therefore it has completely manifest causal structure. The three remaining environment types appear probabilistic to the agent. In the second environment type there are probabilistic transitions when two underlying states that collapse to one perceptual state have different effects following an action. The effect of this action in the perceptual state appears to probabilistically choose one of the two effects from the underlying states. In the third environment type the probabilistic effects come from the underlying environment, and the fourth environment type is probabilistic for both of the above reasons. Such probabilistic environments have Θ-manifest causal structure if for every perceptual state, following any action one effect has probability at least Θ. (Θ is typically a high probability, such as 0.9.) Environments with manifest causal structure are defined precisely in section 3.2.

This thesis shows that in environments with manifest causal structure the agent learns efficiently using straightforward strategies. The learning techniques are simple to implement and efficient in practice, and the techniques should extend to environments which are more complex than the kinds of environments dealt with in past research.

Although it seems intuitive that environments with manifest causal structure should be easy to learn, since all the relevant information is available, the idea has not been explored by researchers. Much learning research deals with learning with "hidden state", such as (Rivest & Schapire 1989, Rivest & Schapire 1990) and (Drescher 1989). Other research Sutton (1990) and (Watkins 1989) assumes that the agent perceives the complete state of the environment. The agent, then, has all the relevant information, but it cannot extract the causal structure of the environment, i.e., the specific causes in a state that lead to any effect. The restriction of the learning problem to environment with manifest causal structure does not trivialize the problem. The inherent difficulties of learning (such as the need for many trials, the huge search space, the problem of representing and using the learned information) remain, but the learning strategies do not have to be smart about inventing causes, only about grasping what is perceived.

1.2 Learning World Models

Autonomous agents typically learn one of two types of world models. The first is a mapping from states of the world (or sets of sensations) to actions (formally $S \rightarrow A$.)

The second is a mapping from states and actions to states (formally $S \times A \rightarrow S$.) I call the first mapping a *goal directed world model*, since it prescribes what action to take with respect to a goal, and the second a *causal world model*, since it indicates the resulting state when taking an action in a given state.

There are several known techniques for learning a goal directed world model. Among these are the reinforcement learning algorithms such as genetic algorithms and the bucket brigade algorithm (Holland 1985, Wilson 1986), temporal differencing techniques (Sutton & Barto 1987, Sutton & Barto 1989), Q-learning (Watkins 1989, Sutton 1990), and variants of Q-learning (Sutton 1991, Mataric 1994, **?**) . These techniques are useful for some applications but do not scale well and suffer from the following common limitation. Since the agent learns a goal directed world model it throws away a lot of the information it perceives and keeps only information that is relevant to its goal. If the agent's goal changes it has to throw away all its knowledge and re-learn its environment with this new perspective.

For example, suppose a secretary robot needs to contact a client. It quickly learns a goal directed model which prescribes the proper sequence of digits to dial on the telephone. Six months later the client moves to a new location with a new telephone number. The secretary is unable to contact the client using its current model. It now has a new goal (dialing the new number sequence) and it must re-learn the entire procedure for contacting the client. If the secretary learns a causal world model, then it spends some time making the right plan each time it calls the client. When the client's number changes, however, it still knows that the proper tool for communication is the telephone and it learns only the new number. Thus following a change in the environment, the secretary can patch a causal world model, but must learn a new goal directed world model.

The advantage of a causal world model is that it stores more information about how the environment behaves. Therefore a local change in the environment forces small adjustments in the model, but not learning a completely new model. In addition the causal knowledge can be used to reason about the environment, and specifically to predict the outcome of actions.

This thesis concentrates on learning a causal world model because in the initial stages of learning the agent learns general domain knowledge that is relevant to many tasks. A goal directed world model is more appropriate for learning to perform specific tasks.

The disadvantage of causal world model is that there is a great deal of information to manage and it is difficult to learn and use the model. Some of these problems have known solutions. For example, for the secretary above, contacting the client using the causal world model requires explicit planning, but the inefficiency of regenerating plans can be avoided by *chunking* plans (Laird, Newell & Rosenbloom 1978).

Causal world models have been efficiently learned for very restricted environments such as finite automata (**?**, Rivest & Schapire 1989, Rivest & Schapire 1990) or with some prior information as in learning behavior networks (Maes 1991). Shen (1993) implements an algorithm for learning a causal model with no prior information. The assumptions that the environment is deterministic and contains a single agent are restrictive, and experiments with the Hanoi Tower environment do not indicate if this approach will scale. Attempts to learn a causal model of more complex environments with no prior information, such as the schema mechanism (Drescher 1989) have not resulted in efficient learning.

1.3 Using Causal World Models

Although this thesis concentrates on the problem of learning causal models, it is important to note that there is a large body of work in AI using causal models for planning, predicting, and causal reasoning.

Planning research is concerned with using a causal world model to find a sequence of actions that will reach a goal state (see STRIPS (Fikes & Nilsson 1971) and GPS (Newell, Shaw & Simon 1957)). The main issues in planning are the efficiency of the search, and robustness of the plans to failing actions, noise, or unexpected environmental conditions (see **?**),and Georgeff & Lansky (1987) for discussion on reactive planning). With the exception of unexpected environmental conditions, which are rare in environments with manifest causal structure, these planning issues are important for an agent using the world model learned in this thesis.

Causal reasoning paradigms solve prediction and backward projection problems. Prediction problems are: "given a causal model and an initial state, what will be the final state following a given sequence of actions?" Backward projection problem are: "given a causal model, an initial state, and a final state, what actions and intermediate states occurred?".

Early work on causal reasoning uncovered the *frame problem*: knowing the starting state and action does not necessarily mean that we know everything that is true in the resulting state. A simple solution to the frame problem is to assume that any condition that is not explicitly changed by the action remains the same. This simple solution is inadequate when there is incomplete information about the state or actions. For example, Hanks and McDermott (Hanks & McDermott 1985, Hanks & McDermott 1986, Hanks & McDermott 1987) propose the Yale shooting problem where a person is alive and holds a loaded gun at time 1, he shoots the gun at time 2, and the question is if he is alive following the shooting. Two solutions exist for this problem. The natural solution where the person shot is not alive, and a solution where the gun is unloaded prior to shooting and the person remains alive. There are several approaches to solving this problem in the nonmonotonic reasoning literature (for example, Stein & Morgenstern (1989), Hanks & McDermott (1985), and Baker & Ginsberg (1989)).

In an environment with manifest causal structure an agent is typically concerned with prediction problems not with backward projection problems, since it perceives relevant past conditions. The agent also perceives all the actions that take place in the environment, so prediction is straightforward given an accurate world model. For example, in the Yale shooting problem it is not possible for the *unload* action to take place without observing this action, so the only fissible solution is the correct one — that the person shot is not alive. Thus, due to the restriction of the environment type, the learning and prediction algorithms in this thesis use the assumption that conditions remain unchanged unless a change is explicit in some rule.

1.4 The Macintosh Environment

This thesis uses the Macintosh environment to explore principles of efficient autonomous learning in an environment with manifest causal structure. In the Macintosh environment the agent "observes" the screen of an Apple Macintosh computer and learns the Macintosh user interface and how to manipulate objects on the screen. This learning problem is realistic; many people have learned the Macintosh interface, which makes this task an interesting machine learning problem. The Macintosh environment is a complex environment with a manifest causal structure (that is why it sells so well), thus fulfilling the requirements for this thesis.

Learning the Macintosh screen interface is more challenging for an agent with no prior knowledge than for people. People find the structure of windows natural because it simulates papers on a desk. Bringing a window to the front has the same effect as moving a paper from a pile to the top of the pile and so on. The learning agent has such foreknowledge. Furthermore, when people learn to work on a computer they typically have a user manual or tutor to tell them the meaning of specific symbols and tricks of the trade. The agent learns the meaning (and function) of the symbols and boxes on the screen, as well as how windows interact, strictly through experimentation.

Learning the Macintosh environment suggests the possibility of machines learning the operation of complex computer systems. Although a very general application of the learning algorithm, such as the secretary robot, is overly ambitious at this time, some applications seem realistic. For example, there has been considerable interest in interface agents recently (?). Research on interface agents to date concentrates on agents that assist the user of computer software or networking software. The interface agents learn procedures that the user follows frequently, and repeats these procedures automatically or on demand. In this way the agent takes over some tedious tasks, such as finding an interesting node on the network.

The learning agent in this thesis can be part of a "smarter" interface agent. The smarter agent can learn about the software environment, and using this knowledge act as a tutor or advisor to a user. The agent can spend time learning about the environment in "screensaver" mode, where the agent uses the environment at those intervals when a screen-saver would run. It then uses the learned model to answer the user's question about the software environment. The implementation of such an application is outside the scope of this thesis, but it is an interesting direction for future research.

The agent's perceptions can be simulated in several ways. People view the screen of a Macintosh as a continuous area where objects (lines, windows, text) can be in any position. Of course, the screen is not continuous: it is made up of a finite number of pixels. The agent can perceive the value of each pixel as a primitive sensation, but this scheme is not a practical representation for learning high level concepts. I represent the screen as a set of rectangular objects with properties and relationships among them. (The perceptual representation is presented in full in section 2.2.)

The agent can manipulate the objects on the screen with *click-in object* actions (other natural actions for this environment *double click*, and *drag* will not be implemented in this thesis.) The actions affect the screen in the usual way. Notice that although time is continuous in this environment it can be discretized based on when actions are completed.

Although the Macintosh environment has manifest causal structure, to the agent it does not appear deterministic. The Macintosh operating system is deterministic, of course, but the agent relies on its perceptions of the environment with no memory which introduces some non-determinism. Consider the situation where there is one large window visible on the screen, when the agent closes this window we expect that only the background will be visible. But if a smaller window may be completely obscured by the top window, then the hidden window will be visible after closing the top window. The Macintosh windowing interface does not randomly pop up a window when the top window is closed, but the agent uses its perceptions of the current state alone to predict. With these perceptions the events of the lower window becoming visible as opposed to only the background being visible appear probabilistic.

Other characteristics of the Macintosh environment are discrete time and space, and that the learner is the only actor affecting the environment. These characteristics will affect some of the strategies of the learning algorithm, although for generality the learning algorithm will not take advantage of some of these assumptions.

1.5 The Agent's World Model

This thesis develops an algorithm for learning a causal world model efficiently for environments with manifest causal structure. The structure of the world model is based on schemas from the schema mechanism (Drescher 1989), although I refer to them as rules not schemas. As in the schema mechanism, the world model is a collection of rules which describe the effects of actions on perceptual conditions. I will write rules as follows:

$$\text{precondition} \rightarrow \text{action} \rightarrow \text{postcondition}$$

where the precondition and postcondition are some logical combination of the perceptual conditions of the environment, and higher level concepts that the agent learns.

A rule describes the effects of the action on the environment. It indicates that if the precondition is currently true in the environment, then if the action is taken, the postcondition will be true. Notice that the rules in this model are not production rules, which suggest taking the action, or STRIPS operators (Fikes & Nilsson 1971), which add or remove conditions in the environment. Rather they remember a causal relationship that is true for the environment, and taken as a set they form a causal world model that is goal independent.

Once the agent learns a reliable set of rules it can use the world model to predict and plan. Known algorithms such as GPS (Newell et al. 1957) and STRIPS (Fikes & Nilsson 1971) can be adapted to plan and predict using these causal rules.

1.5.1 Learning Rules

Before discussing the methodology for learning rules consider what the agent should learn. Let us consider an example from the Macintosh environment. Figure 1-1 shows two screen situations from the Macintosh environment. In the first one *window 2* is active and overlaps *window 1*, and the second situation shows that following a click in *window 2's zoom-box window 2* is active and covers the complete screen, and *window 1* is not visible.

Several observations are important in this example:

***window 2* is still the active window** Recall that the agent assumes that perceptions are persistent unless a rule explicitly indicates a change. Since there was no change, no rule is needed for describing this observation.

***window 1* is not visible** A rule that describes this important change has the format

$$\textit{window 2 is active} \wedge \textit{window 1 is visible}$$
$$\rightarrow \textit{click-in} \text{ window 2's zoom-box} \rightarrow \textit{window 1 is not visible}$$

Similar rules describe the disappearance of all the parts of *window 1*.

The description of the rule above is high level, and uses concepts such as *active* and *visible* which are unknown to the agent initially. The rule that the agent learns will be expressed in terms of its perceptions of the screen, and higher level concepts, such as *active*, when the agent learns such concepts. A discussion on learning concepts follows in section 1.5.2. For the time being I will continue to discuss the Macintosh environment with high level descriptions, and we can assume that some set of perceptual conditions captures the description. A complete breakdown of the perceptual interface is given in chapter 2.

The rule learning algorithm for the Macintosh environment begins with an empty set of rules (no prior knowledge.) After every action the agent takes, it proposes new rules for all the unexpected events due to this action. Unexpected events are perceptions whose value changed inexplicably following the action. At each time-step the learning algorithm also evaluates the current set of rules, updates rule reliabilities, and removes "bad" rules, i.e., rule that do not predict reliably.

The main points of the rule learning algorithm are discussed below.

Creating new rules The procedure for creating new rules has as input the following: (1) the postcondition, i.e. a relation on some objects to explain, (2) the

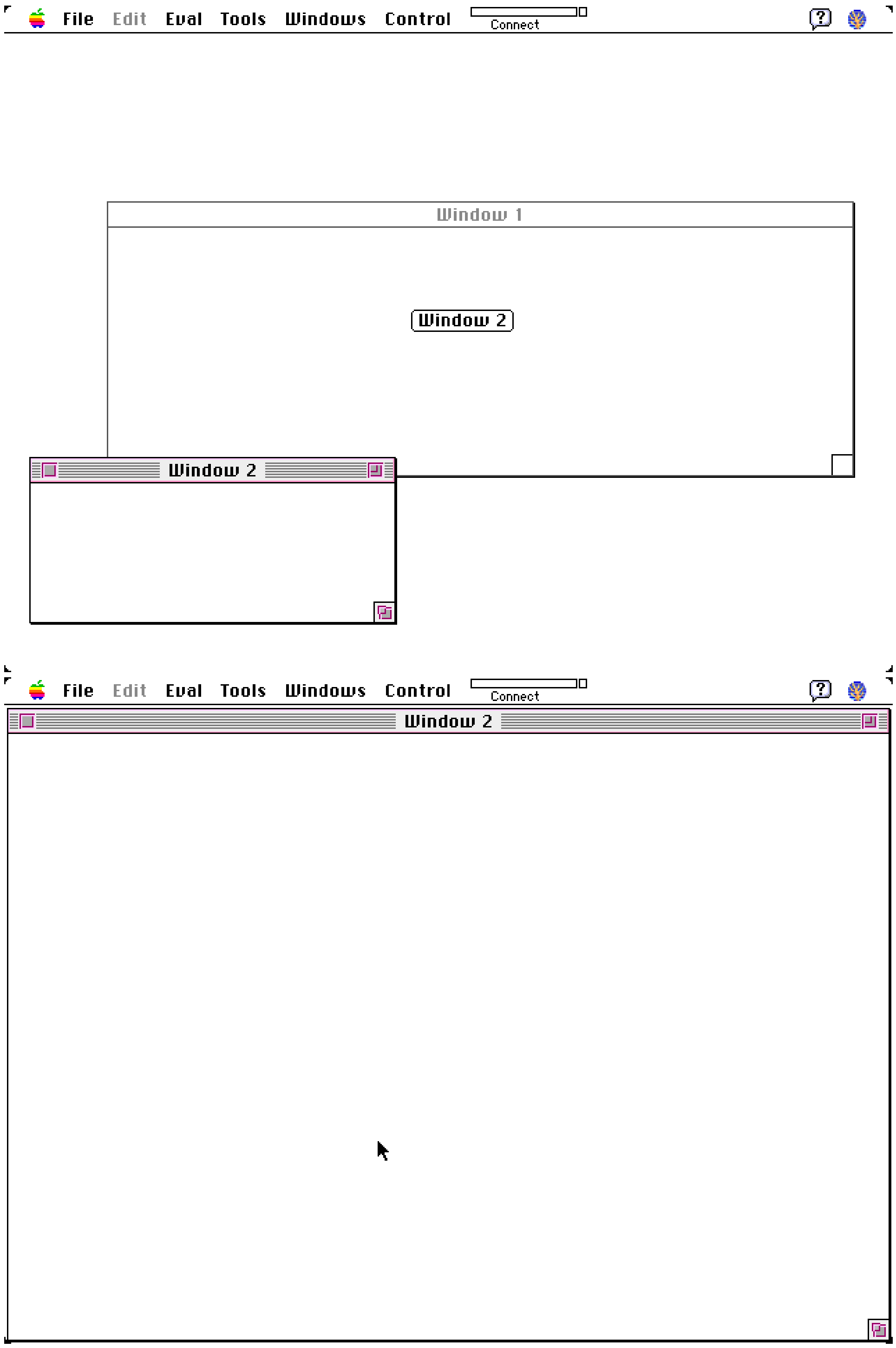

Figure 1-1: Macintosh Screen before and after a click in window 2 zoom-box

last action the agent took, and (3) the complete list of perceptions before the action was taken.

The key observation in simplifying the rule learning algorithm is that because the environment has a manifest causal structure, the preconditions sufficient to explain the postcondition are present among the conditions at the previous time-step. The task of this procedure is to isolate the right precondition from the previous perceptions list. Because of the complexity of the Macintosh environment there are many perceptions, and it is worthwhile to select the preconditions with care. The preconditions are selected at random with some guidance from heuristics that trim the search space. (The heuristics are general, not problem-specific.)

Separating the good from the bad After generating a large number of candidate rules the agents has to save the "good" rules and remove the "bad" ones. Suppose the environment the agent learns is completely deterministic. The perceptions in the current state are sufficient to determine the effects of any actions and there are no surprises. In such environments there is a set of perfect rules that never fail to predict correctly. Distinguishing good rules from bad ones is easy under these circumstances: as soon as a rule fails to predict correctly the agent can remove it.

Unfortunately the class of deterministic environments is too restrictive. Most environments of interest do not have *completely* manifest causal structure. For example, in the Macintosh environment one window can cover another window completely, and if the top window is closed the hidden window is surprisingly visible. To cope with a small degree of surprise the rule reliability measure must be probabilistic.

The difficulty in distinguishing between good and bad probabilistic rules is that at any time the rule has some *estimated* reliability from its evaluation in the world. The agent must decide if the rule is good or bad based on this estimate rather than the true reliability of the rule. This problem is common in statistical testing, and several "goodness" tests are known. The rule learning algorithm in this thesis uses the sequential ratio test (Wald 1947) to decide if a rule is good or bad.

Mysteries In most environments some situations occur rarely. The algorithm uses mysteries to learn about rare situations. The agent remembers a rare situations as a mysteries, and then "re-plays" the mysteries, i.e., repeatedly tries to explain these situations. Mysteries are similar to the world model component of the Dyna architecture (Sutton 1991), which the agent can use to improve its goal directed model.

For the complete algorithm see chapter 3. Chapter 3 also contains rule learning results in the Macintosh environment, and shows that under the rule learning algorithm converges to a good model of the environment.

1.5.2 Learning New Concepts

Concept learning gives the agent a higher level understanding of its environment that is not ground in the perceptual inputs. A world model composed of rules, where the preconditions and postcondition are perceptual conditions, are overly specific. Such rules remember conditions that were correlated to an effect in the past, but do not necessarily capture the cause of the effect.

For example, in the screen situation of figure 1-2, *window 1* disappears and *window 2* becomes active. There is a simple rule to explain that *window 1* disappears:

() → click-in *window 1* close-box → *window 1* is not visible.

(No preconditions are needed since clicking in *window 1*'s close-box implies that the close-box exists which implies that *window 1* is active.) Many rules explain why *window 2* became active, Among them

window 2 is visible → click-in *window 1* close-box → *window 2* is active

window 2 interior is visible → click-in *window 1* close-box → *window 2* is active

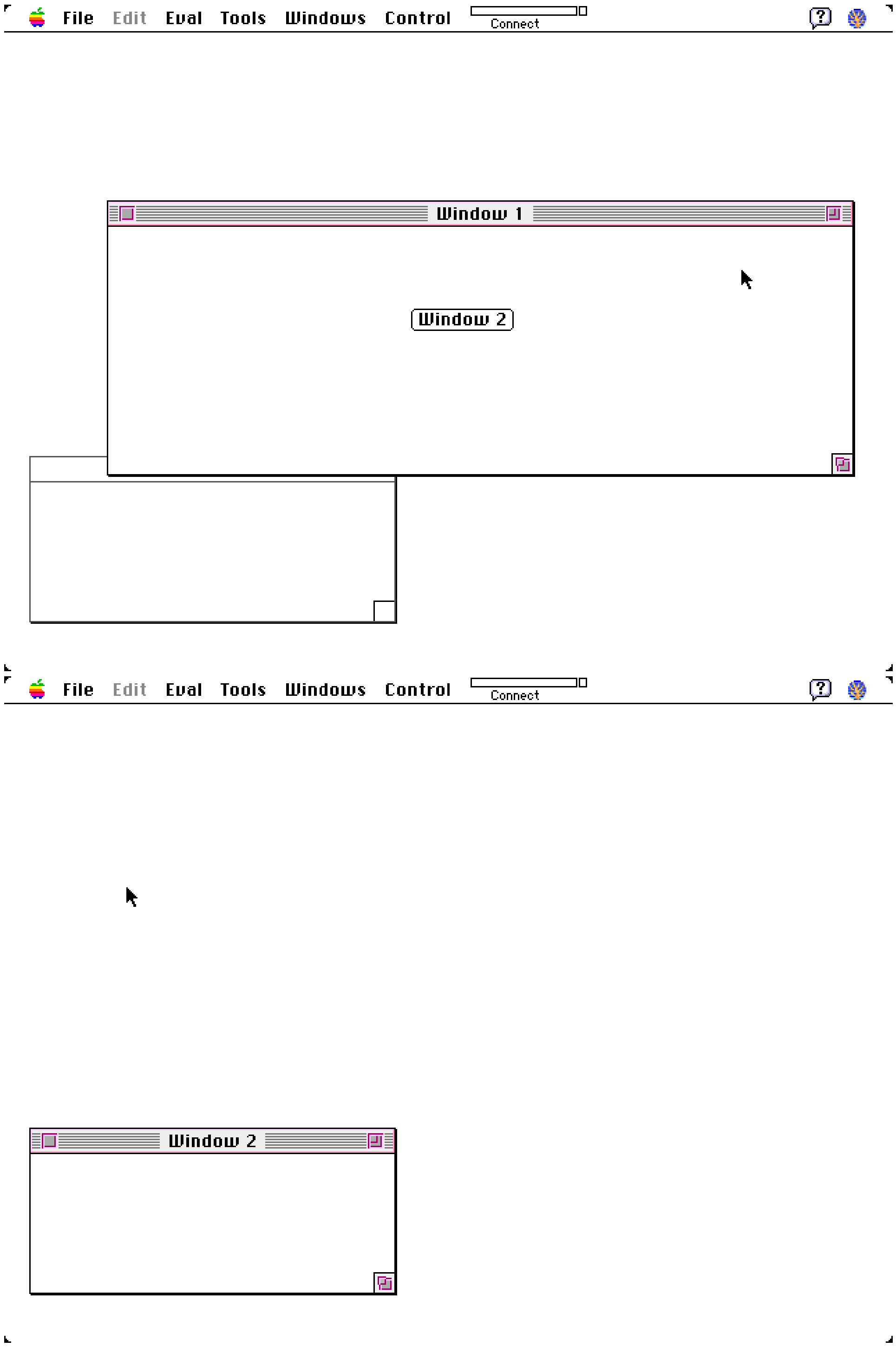

Figure 1-2: Macintosh Screen before and after a click in window 1 close-box

window 2 title-bar is visible → click-in *window 1* close-box → *window 2* is active.

Clearly most of these rules are redundant since whenever *window 2* is visible and not active it has an interior and a title-bar, etc. ?) makes the distinction between correlated conditions, such as the second and 3rd rules above, and true causality, such as the first rule. Correlating perceptions is a type of concept learning which addresses the problem of redundant rules and finding the cause of an effect.

To learn correlated perceptions the agent first learns causal rules such as

preconditions → NOACTION → postcondition

when no action is taken. These rules mean that when the preconditions are true the postcondition is also true in the same state. Notice that a set of noaction rules defines a graph in the space of perceptions. The agent finds correlated perceptions by looking for strongly connected components in this graph.

The second type of concept learning addresses the problem of rules that are specific to particular instances in the environment. For example, consider a room with three light switches, the agent learns rules that explain how each of the light switches works, but when the agent moves to a different room with different light switches it has to learn how these light switches work. Instead, the agent should learn that there is a concept *light switch* and some rules that apply to all light switches. Similarly in the Macintosh environment, there is a concept *window* and rules that apply to all windows.

The agent learns general concepts by finding similar rules and generalizing over the objects in those rules. When it generalizes over objects it adds the attributes of the objects to the preconditions of the general rule. For example consider the rules

() → click-in *window 1* close-box → *window 1* is not visible

() → click-in *window 2* close-box → *window 2* is not visible.

These rules are similar and indicate that the objects *window 1* close-box and *window 2* close-box should be generalized. The agent adds the precondition that the attributes that the general object is a close-box and that this new object is part of the generalized window object. The generalized rule becomes

$$object_x \text{ is an active window} \wedge object_y \text{ is a close-box} \wedge object_y \text{ is part of } object_x \rightarrow \\ \text{click-in } object_y \rightarrow object_x \text{ is not visible.}$$

See chapter **??** for the complete algorithm for generalizing rules.

1.5.3 Evaluating the World Model

While the agent learns a world model we need to evaluate how good the model is. There are three ways of evaluating a world model: to compare it with a correct model, or to test it as a predictor or as a basis for planning. This section examines the plausibility of each method of evaluation in turn.

The first form of evaluation is to compare the learned model with a correct model. In this form of evaluation the set of learned rules is compared with a set of a priori known rules and the evaluation returns the percentage of correct rules the agent learned. This method suffers from two drawbacks: (1) it assumes that a single correct model exists, and (2) someone (I) must encode the correct model manually. In many environments any number of non-identical world models are equally good, and the number of rules in a world model often prohibits manually coding the model. Thus, in this thesis, the world model is not compared with the "right" model, but examples of learned rules are given.

The second form of model evaluation is to predict the next state of the world given the current state and an action. This thesis uses this method of evaluation primarily. The prediction algorithm predicts postconditions from all the applicable rule, and assumes no change as a default when no rules apply (as discussed in section 1.3).

The final form of model evaluation is to test the agent's ability to achieve a goal. This algorithm uses a backward chaining search to find action sequences that achieve goals. The planning algorithm is simplistic and not efficient enough for general use, but it is sufficient to demonstrate that the world model has the knowledge to achieve the goal.

1.6 Overview

The remainder of this thesis contains an extended discussion of the algorithms mentioned in this chapter with results from experiments in the macintosh environment. Chapter 2 discusses the Macintosh environment and how the agent perceives the Macintosh screen. I present the rule learning algorithm in complete detail in chapter 3, along with a proof that in environments with manifest causal structure this algorithm converges to a desirable world model. Chapters **??** and **??** contain the two concept learning algorithms: finding correlated perceptions and generalizing rules.

In addition, this thesis present results on picking a good expert from a sequence in chapter **??**. This direction of research is tangentially related to the research on learning world. It stems from the work on deciding if rules are good. In this research, we can examine experts (for example the experts may be rules) one at a time and we want to discard bad experts and keep good ones (experts that make few mistakes). Much like rule learning the experts come from an unknown distribution and their error probability (or reliability) is unknown. Unlike rules, we cannot assume how good the best experts are. Chapter **??** presents an algorithm that finds an expert that is almost as good as the best expert we expect to find if the experts error probabilities were known, in the same length of time.