Chapter 3: Goal-independent Learning

This research started with an existing learning algorithm [Drescher 91], and added a focus of attention mechanism, as described below, to make learning faster (requiring less computation per timestep). While Drescher's original work does include a concept similar to both tactical and strategic goals, his system does not exploit goals to guide the learning process. Further, it has no perceptual selectivity and assumes that every sensory bit might be useful all the time.[1] This approach leads to a theoretically "pure" result for the issues which Drescher was investigating, but one which is difficult to use in a real application, and which seems somewhat implausible in describing how real organisms learn.

The following discussion is an overview of the operation of the schema system; for a much more complete and detailed explanation of its intricacies, see [Drescher 91] and [Ramstad 92].

3.1 Model of the learning system

This section presents a very brief overview of the schema system [Drescher 91] to aid in understanding the focused and unfocused learning algorithms presented later. The schema system is designed to be used in building causal models of the results of actions.

1. With one very small exception, as follows. The last action taken is itself represented in the bits given to the learning algorithm (since any new schema created to represent the results of the action just taken will be a schema mentioning that action, and no other). Since *only* the last action taken is so represented to the learning algorithm, *only* the last action taken is attended to when attempting to correlate actions with their results. All other sensory inputs (e.g., proprioceptive, visual, etc) are attended to whether or not they have changed recently.

Hence, it is designed to be hooked up to some agent which is situated in an environment, and which has some sensory inputs and motor outputs.

3.1.1 The sensory system: Items

Every sensory item is represented in the simulation as a single bit. In Drescher's original algorithm, there is no grouping of these bits in any particular way (e.g., as a retinotopic map, or into particular modalities); the learning algorithm sees only an undifferentiated "bag of bits."

This means that values in the world which are not simple binary bits must somehow be turned into individual bits. Ranges, for example, are most usefully represented as a set of bits, any one of which would be asserted when the value is within some particular interval of the range.[2]

3.1.2 The motor system: Actions

The schema system assumes a known, fixed number of possible actions that the agent may take. It takes one of these actions at every timestep of the simulation, and observes which sensory items have changed after the action.

In Drescher's original implementation, the system spent most of its time (a fixed 90%) taking random actions and observing the results. The remaining 10% of its time was spent taking actions which had led to some reliable outcome before, to see if actions could be chained.[3] The work reflected in this chapter instead represents a totally random action selection strategy: the agent always picks its next action at random. In Chapter 4, in the

2. There is no particular reason why such range values could not be represented instead as, e.g., a binary number. However, such a representation, though more compact in terms of the number of sensory bits, would be somewhat slower to learn, since a schema (see Section 3.1.3) which made accurate predictions about the value might have to have a conjunctive context or result (see Section 3.1.3.2.6) which completely specified all *n* bits of the number. Generating such conjunctions would be quite timeconsuming.
3. See the composite action mechanism of [Drescher 91].

context of goal-directed focus of attention, we also investigate less random and more goal-oriented action selection strategies.

3.1.3 The knowledge base: Schemas

3.1.3.1 What is a schema?

The "facts" learned by this system are called *schemas*. They consist of a triple of the *context* (the initial state of the world, as perceived by the sensory system), the *action* taken on this iteration, and the *result* (the subsequent perceived state), which maps actions taken in a particular configuration of sensory inputs into the new sensory inputs resulting from that action. In the infant/eyehand scenario, a typical schema might therefore be, "If my eye's proprioceptive feedback indicates that it is in position (2,3) [*context*], and I move my eye one unit to the right [*action*], then my eye's proprioceptive feedback indicates that it is in position (3,3) [*result*]." A simple textual way of representing this would be VP23/EYER/VP33, as described in [Drescher 91] and [Ramstad 92].

Another typical schema might be, "If my hand's proprioceptive feedback indicates that it is in position (3,4) [*context*], and I move it one unit back [*action*], I will feel a taste at my mouth and on my hand [*result*]," or HP34/HANDB/TASTE&TACT.[4] Notice that this latter schema is *multimodal* in that it relates a proprioceptive to taste and tactile senses; the learning mechanism and its microworld build many multimodal schemas, relating touch to vision, vision to proprioception, taste to touch, graspability to the presence of an object near the hand, and so forth. It also creates *unimodal* schemas of the form illustrated in the first schema above, which in this case relates proprioception to proprioception.

Since behavior of the world may be nondeterministic (e.g., actions may sometimes fail, or sensory inputs may change in manners uncorrelated with the actions being taken), each

4. This is because the hand will move from immediately in front of the mouth to in contact with the mouth. We assume that the infant's mouth is always open, which seems empirically reasonable for most infants.

schema also records statistical information which is used to determine whether the schema accurately reflects a regularity in the operation of the world, or whether an initial “guess” at the behavior of the world later turned out to be a coincidence. (For details about how this actually works, see the next section.) This information is recorded in a schema’s *extended context* and an *extended result*, which keep these statistics for every item not yet present in the context or result.

3.1.3.2 How are schemas created? The marginal attribution mechanism

A schema is deemed to be *reliable* if its predictions of **(context, action, result)** are accurate more than a certain percentage of the time. If we already have a reliable schema, and adding some additional sensory item to the items already expressed in either its context or its result makes a schema which appears that it, too, might be reliable, we *spin off* a new schema expressing this new conjunction.[5] *Spinoff schemas* satisfy several other constraints, such as not ever duplicating an existing schema, and may themselves serve to be the basis for additional spinoffs later.

The description below, until the end of this section (Section 3.1.3.2), is quoted with a few modifications from [Ramstad 92]; that description, in turn, was a summarization of [Drescher 91]. See the latter in particular for more depth on the subtleties of the schema system and marginal attribution in particular; this description, being a summary, glosses over particular refinements which prevent certain combinatorial explosions.

5. For example, consider the schema VF33/EYER/FOVF22, which states that, if a particular coarse visual field bit (the one at 3,3) is on, and the agent moves the eye right, it will see a particular fine foveal visual bit turn on (the one at 2,2). If this schema is sufficiently reliable, and it also seems from experience that VF32 is usually off just before we take this action in this case, the schema system might spin off the schema ¬VF32&VF33/EYER/FOVF22 from the original schema. (Note that the notation often used in, e.g., [Ramstad 92] would say -VF32 instead of ¬VF32; we use both forms interchangeably here.) While the *original* schema must be deemed reliable to be considered for a spinoff (to prevent combinatorial explosion), the *new* one is not yet known to be reliable, and will itself be considered a spinoff candidate only if it, too, is later shown to be reliable.

3.1.3.2.1 Bare schemas

Schemas are bootstrapped from an initial set of *bare* schemas, which have neither context nor result and hence do not predict anything. There is one bare schema for each action that can be taken.

As the simulation is run, a technique known as *marginal attribution* is used to discover statistically important context and result information. This information is then used to fine-tune existing schemas by creating modified versions of them. Marginal attribution succeeds in greatly reducing the combinatorial problem of discovering reliable schemas from an extremely large search space without prior knowledge of the problem domain.

3.1.3.2.2 Result spinoffs

Many different results may occur from the execution of a given action. For every *bare* schema, the schema mechanism tries to find result transitions which occur more often with a particular action than without it. For example, a hand ends up closed and grasping an object much more often when the *grasp* action is taken than with any other action. Results discovered in this fashion are eligible to be included in a *result spinoff*—a new schema identical to its parent, but with the relevant result item included. The marginal attribution process can only create result spinoffs from bare schemas.

Specifically, each bare schema has an *extended result*—a structure for holding result correlation information. The extended result for each schema keeps correlation information for each item (primitive or synthetic). The positive-transition correlation is the ratio of the number of occurrences of the item turning on when the schema’s action has been taken to the number of occurrences of the item turning on when the schema’s action has not been taken. Similarly, the negative-transition correlation is the ratio of the number of occurrences of the item turning off when the schema’s action has been taken to the number of occurrences of the item turning off when the schema’s action has not been taken. Note that an item is considered to have turned on precisely when the item was off prior to the action and on after the action was performed, and similarly for turning off. The correlation statistics are continuously updated by the schema mechanism and weighted towards more recent data. When one of these schemas has a sufficiently high correlation with a particular item, the schema mechanism creates an appropriate result spinoff—a schema with the item positively included in the result if the positive-transition correlation is high, or a schema with the item negatively included in the result if the negative-transition correlation is high. These simple statistics are very good at discovering arbitrarily rare results of actions, especially when the statistics of the non-activated schemas are only updated for *unexplained* transitions. A transition is considered explained if the item in question was included in the result for an activated schema with high reliability (above an arbitrary threshold) and it did, in fact, end up in the predicted state.

3.1.3.2.3 Context spinoffs

For schemas which have non-negligible results, the marginal attribution mechanism attempts to discover conditions under which the schema obtains its result more reliably. To extend the example cited above, a hand ends up closed and grasping something much more often an object can be felt touching the correct part of the hand before the grasp action is executed. This information is used to create *context spinoffs*—duplicates of the parent schema, but with a new item added to its context.

Schemas with non-empty results have an *extended context*. For each item, this structure keeps a ratio of the number of occurrences of the schema succeeding (i.e., its result obtaining) when activated with the item on the number of occurrences of the schema succeeding when activated with the item off. If the state of a particular item before activation of a given schema does not affect its probability of success, this ratio will stay close to unity. However, if having the item on increases the probability of success, the ratio will increase over time. Similarly, if having the item off increases the probability of success, the ratio will decrease. If one of these schemas has a significantly high or low ratio for a particular item, the schema mechanism creates the appropriate *context spinoff*—a schema with the item positively included in the context if the ratio is high, or one with the item negatively included if the ratio is low.

There is an embellishment to the process of identifying context spinoffs. When a context spinoff occurs, the parent schema resets all correlation data in its extended context, and keeps an indication of which item (positively or negatively included) was added to its spinoff child. In the future, when updating the extended context data for the parent schema, if that item is on (if positively included in the spinoff) or off (if negatively included in the spinoff), the trial is ignored and the extended context data is not modified. This embellishment means that the parent schema has correlation data only for those trials where there is no more specific child schema, and it encourages the development of spinoff schemas from more specific schemas rather than general schemas.

Redundancy is also reduced by a further embellishment. If, at a particular moment in time, a schema has multiple candidates for a context spinoff, the item which is on least frequently is the one chosen for a context spinoff. The system keeps a *generality* statistic for each item which is merely its rate of being on rather than off—it is this statistic which is used when deciding between multiple spinoff possibilities. This embellishment discourages the development of unnecessary conjunctions when a single specific item suffices.

3.1.3.2.4 Conjunctive contexts and results

The context can be iteratively modified through a series of context spinoffs to include more and more conjuncts in the context. For a variety of reasons, but primarily to avoid combinatorial explosion, a similar process for result conjunctions is undesirable. The marginal attribution process therefore requires that result spinoffs occur only from bare schemas, and only one relevant detail can be detected and used as the result for the spinoff schema. However, conjunctive results are necessary if the schemas should be able to *chain* to a schema with a conjunctive context. (See Chapter 4 for much more about chaining schemas.) This problem is solved by adding a slot to the extended result of each bare schema for each of the conjunctions of items which appear as the context of a highly reliable schema. Statistics are kept for these in the same fashion as those kept for single items, and if one of these conjunctions is often turned on as the result of taking a given action, a result spinoff occurs which includes the entire conjunction in the result. Effectively, this process is able to produce schemas with conjunctive results precisely when such schemas are necessary for chaining.

3.1.3.2.5 Summary of marginal attribution

Schemas created by the marginal attribution process are designed to either encapsulate some newly discovered piece of knowledge about causality in the microworld (result spinoff), or to improve upon the reliability of a previous schema (context spinoff). By continuously creating new versions of previous schemas, the system iteratively improves both the reliability and the scope of its knowledge base. It is interesting to note that, once created, a schema is never removed from the system. Rather, it may be supplanted by one or more spinoff schemas which are more *useful* due to higher reliability and greater specificity.

3.1.3.2.6 Synthetic items

The schema system also defines many other concepts; we mention here briefly one other, namely *synthetic items*, that will become more important in Section 4.4.2, on page 95. This section can safely be skipped, at least until then, without loss of continuity.

There are certain concepts that primitive items are unable to express, for example, that a particular object is present at a particular location while the glance orientation is such that the object is out of view. The schema mechanism contains a facility for building *synthetic items*—items which, when on, indicate that a particular unreliable schema, if activated, would succeed. Suppose a schema saying /[MOVE GLANCE ORIENTATION TO VP01]/VOVF02 (note that this schema has no context) is very reliable if some object in the microworld is in the correct position. However, this object spontaneously moves around between a few different positions and so, on average, is only in the correct position some of the time. Notably, this schema, if activated and successful, will continue to be very reliable for some period of time (equal to the duration that the object remains in that position), even though on average it is normally not very reliable. To discover such situations, the schema mechanism keeps a *local consistency* statistic which indicates how often the schema succeeds when its last activation was successful. If a schema is unreliable but highly locally consistent, the mechanism constructs a synthetic item—an item which, when on, indicates that the schema (its *host schema*), if activated, would succeed. Effectively, such an item, when on, predicts what the result of activating the host schema would be. For a variety of reasons (see [Drescher 91]), synthetic items are fundamentally very different from primitive items and express concepts which are inexpressible through any conventional combination of primitive items.

Primitive items get their state directly from the microworld. On the other hand, the schema mechanism itself must maintain and update the state of all synthetic items. (The rules for how this update is accomplished are complicated and not explained here.) The use of synthetic items effectively allows the schema mechanism to invent new concepts—concepts which are not expressed well by the microworld or cannot be expressed by conjunctions of boolean values at all.

3.2 Improving learning via focus

If we are to talk about improving the computational effort of learning, we must first be clear about what happens during learning, and how to describe how much work is performed for various modifications of the basic learning algorithm.

3.2.1 The work of learning

Given the action model learning algorithm described above, at each clock tick, we must do two things:

- First, we must update various statistics reflecting what just happened; this is the "perception" part of the learning algorithm. A focus mechanism dictates which sensory items will be attended to[6] ($Stat_i$), for which schema numbers ($Stat_s$).
- Second, we must decide whether to spin off a new schema; this is the "learning" part of the algorithm, and here the focus mechanism dictates which item numbers to check for reliability ($Spin_i$) for which schema numbers ($Spin_s$).

Thus, our choice of these four sets of numbers determines which sensory items and schemas are used in either updating our perception of the world, or deciding when a correlation has been learned. $Stat_i$ and $Stat_s$ determine the perceptual selectivity, while $Spin_i$ and $Spin_s$ determine the cognitive selectivity of the agent, as shown in the table below)

	Statistics	**Spinoffs**
Which items?	$Stat_i$	$Spin_i$
Which schemas?	$Stat_s$	$Spin_s$

6. In other words, will have their associated statistics reflecting frequency of occurrence updated in all schemas which mention them.

The algorithm does the vast majority of its work in two $O(n^2)$ loops (one loop for updating statistics, reflecting what is currently perceived, and one loop for deciding whether to spin off new schemas, reflecting learning from that perception). The number of items and the number of schemas selected at any given clock tick determines the amount of work done by the learning algorithm in that tick. Thus, if $Work_{Stat}$ is the work done during the statistical-update part of the algorithm (e.g., perceiving the world) of any one clock tick, and $Work_{Spin}$ is similarly the amount of work done in deciding which spinoff schemas to create, then the work done during either one is the product of the number of items attended to and the number of schemas attended to, or:[7]

$$Work_{Stat} = \|Stat_i\| \bullet \|Stat_s\|$$

$$Work_{Spin} = \|Spin_i\| \bullet \|Spin_s\|$$

This means that the total work per step (clock tick) is simply the sum of these individual pieces, and that the total work over some particular number of steps is simply the sum of the work during the individual steps:[8]

$$Work_{Step} = Work_{Stat} + Work_{Spin}$$

$$Work_{Total} = \sum_{steps} Work_{Step}$$

Thus, the behavior of $Work_{Step}$ over time tells us how well the algorithm will do at keeping up with the real work, e.g., how much it slows down as the number of iterations (which is proportional, though not in a particularly simple way, to the number of schemas) increases.

7. Where $\|x\|$ denotes the cardinality of the set x.
8. There is no particular end to this series of learning steps. In this research, the number of steps taken was limited by the amount of time available to learn, or by the approximate number of schemas that were desired for a run, and so forth. In a real agent, we might turn off the learning system once the agent has shown that it is competent (see Chapter 4) to perform some set of tasks—or not. After all, if we were ever to turn off the learning system, the agent would fail to change its internal world model even if the external world were to change.

The discussion above looked at the work per step, e.g., per clock tick of the simulation. Another way of evaluating the utility of the various algorithms is to examine the amount of work performed per schema learned (either reliable schemas or all schemas; the former being perhaps the more useful metric):

$$Work_{Step}SchemaRel = \frac{Work_{Step}}{SchemaRel_{Step}}$$

which is simply the amount of work performed during some steps (*Work*$_{Step}$) divided by the number of reliable schemas generated by that work (*SchemaRel*$_{Step}$).[9] A similar definition for total schemas over total work is straightforward:[10]

$$Work_{Step}SchemaTotal = \frac{Work_{Step}}{SchemaTotal_{Step}}$$

An algorithm which determines these choices is thus described by the pairs $<<Stat_i,Stat_s>,<Spin_i,Spin_s>>$; we will call each element a *selector*.

A little more terminology will enable us to discuss the actual selectors used. $Schema_{max}$ is the number of schemas currently learned. $Item_{max}$ is the number of sensory items. $Item_n$ is the value of sensory bit *n*, and $Item_{n,t}$ is the value of that item at some time *t*. $SchemaDep_sOn_i$ denotes some schema *s* which is dependent upon (e.g., references in its context or result) some item *i*.

9. Note a peculiar detail here. It is possible for a schema that was formerly thought to be reliable to be later decided to be unreliable. This could happen if the world has changed in the meantime, or if some not-very-correct correlation happened often enough to push the schema over the arbitrary threshold from not being considered reliable to being considered reliable, and then later data pushed the schema's reliability back down. Hence, it is possible for the number of reliable schemas to *decrease* during a single step, and this is not an altogether infrequent occurrence. This means that, while the *average* of *Work*$_{Step}$*SchemaRel* is positive, the instantaneous value might be negative if *Step* is a single step or a small number of steps.

10. Since the *total* number of schemas (as opposed to *reliable* schemas) can never decrease, this number must always be nonnegative.

3.2.2 The basic learning algorithm

In Drescher's basic algorithm, every possible sensory bit *before* an action taken by the "infant" was correlated with every possible sensory bit *after* the action, for *every* schema that has been created so far. In other words, $Stat_i$ and $Spin_i$ use the selector **all item numbers**, or **AIN**:

$Stat_i = Spin_i = \mathrm{AIN}$, where

$\mathrm{AIN} = \{n | 0 \le n \le Item_{max}\}$

and $Stat_s$ and $Spin_s$ use the selector **all schema numbers**, or **ASN**:

$Spin_i = Spin_i = \mathrm{ASN}$, where

$\mathrm{ASN} = \{n | 0 \le n \le Schema_{max}\}$

This means that the basic algorithm does a tremendous amount of work in the two n **x** m inner loops, where n is the size of the set of items in use, $\|\mathrm{AIN}\|$, and similarly m is the size of the set of schemas in use, $\|\mathrm{ASN}\|$. Hence,

$Work_{Stat} = Work_{Spin} = \|\mathrm{AIN}\| \bullet \|\mathrm{ASN}\|$

This algorithm can eventually learn a large number of facts about the world in this way, but it runs slowly, and becomes increasingly slow as the number of known facts (e.g., schemas) increases. Furthermore, if we were to increase the number of sensory bits available (e.g., by putting a higher-resolution camera in an agent using this technique), the work involved would increase in direct proportion to the number of added sensory bits, even if none of these bits ever changed.

3.2.3 The focused algorithm

Various pruning techniques help a great deal over the basic approach. The most successful of the approaches examined, which we shall call the *focused* algorithm, takes the following tack (why these particular parameters were chosen is explained at the end of this section):

- *Perceptual selectivity*. When updating statistics, only consider sensory items which have changed very recently (last two clock ticks) and only in schemas which make predictions about those items.
- *Cognitive selectivity*. When deciding whether to spin off a schema (make a new fact), only consider sensory items which have changed in the last clock tick, and only consider schemas which have had their statistics changed in the last clock tick (such schemas can only have had their statistics changed if they themselves made predictions involving sensory items which themselves have changed).

Put more precisely, the items used were as follows. $Stat_i$ used the **all changed items in history**, or **CINIH** selector (where the word “history” refers to a timeline of prior events, of some chosen length or *horizon*, and in this case of length 2):

$Stat_i = \mathrm{CINIH}_H$, where

$$\mathrm{CINIH}_H = \mathrm{AIN} \cap \{\exists (0 < T \le H) \mid Item_{n,t} \ne Item_{n,t-T}\}$$

while $Spin_i$ used a specialization of this, in which the history is only the very last event, which we shall call the **changed item numbers**, or **CIN**, selector for compactness:

$Spin_i = \mathrm{CIN}$, where

$$\mathrm{CIN} = \mathrm{AIN} \cap \{Item_{n,t} \ne Item_{n,t-1}\} = \mathrm{CINIH}_1$$

Similarly, the schemas used were as follows. Consider the set **all bare schemas**, or **ABS:**[11]

$$\mathrm{ABS} = \mathrm{ASN} \cap \{\forall (0 \le i \le Item_{max}) \mid \neg SchemaDep_n On_i\}$$

To define $Stat_s$, we add to these **schemas dependent upon changed items**, to get:

$Stat_i = \mathrm{ABSPSDUCI}_H$, where

$$\mathrm{ABSPSDUCI} = \mathrm{ABS} \cup \{SchemaDep_n On_{i \in \mathrm{CIN}}\}$$

This selector is a special case of the more general one (which uses an arbitrary-length history), in that it uses a history of length 1. The general case, of course, is:

$$\mathrm{ABSPSDUCIIH}_H = \mathrm{ABS} \cup \{SchemaDep_n On_{i \in \mathrm{CINIH}_H}\}$$

Finally, $Spin_s$ is defined as schemas with recently updated statistics, or

$Spin_i = \mathrm{SWRUS}$, where

$$\mathrm{SWRUS} = \mathrm{ASN} \cap \{SchemaDep_n On_i \mid (i \in \mathrm{CIN})\}$$

Adding these changes amounts to adding some simple lookup tables to the basic algorithm that track which items were updated in the last clock tick, and, for each item, which schemas refer to it in their contexts or results. These tables are then used to determine which sensory items or schemas will be participants in the statistical update or spinoffs. Keeping the tables updated requires a negligible overhead on the basic algorithm.[12]

3.2.3.1 How has focus changed the computational effort?

The computational effort in the two $n \times m$ major loops is reduced by these selectors as follows:

11. Recall, from Section 3.1.3.2, on page 42, that bare schemas make no predictions about anything, and that there is one of these per action at the start of any run.

$$Work_{Stat} = \|\mathrm{CINIH}_2\| \bullet \|\mathrm{ABSPSDUCI}\|$$

$$Work_{Spin} = \|\mathrm{CIN}\| \bullet \|\mathrm{SWRUS}\|$$

In any run which generates more than a trivial number of schemas or has more than a handful of sensory bits, this is a dramatic reduction in the complexity, as shown in Figure 8, on page 63. Another way to look at it is as follows:

- The *unfocused* algorithm allows the work of learning to grow as the full cross-product of the *total* number of sensory bits (items) and the *total* number of predictions we make about the world (schemas).
- In the *focused* algorithm, the work of learning instead grows as the product of the amount of *change* in the world times the number of predictions we make about *those items which changed.*

If the world were such that *every* item changed at *every* step, and we had (somehow) managed to make a prediction about *every* item in *every* schema, then these two algorithms would behave identically. However, this does not describe very many plausible worlds in which we might want an agent to do learning, nor is it plausible that every prediction the agent may want to make about the world needs to mention every possible sensory bit the agent can perceive.

12. *Sensory*: Updating the list of items which have changed recently (where *recently* is defined by the horizon in use) runs in time linear with the number of items.

Cognitive: Each time we spin off a schema, we must update the table that maps items to schemas which depend upon them, in order to properly reflect the dependence of the schema on the items. This update runs in time linear with the number of items mentioned by the schema, and this number is very small anyway (less than half a dozen or so in the runs described).

Result 1: The work per spinoff is linear, which means it is negligible compared to the rest of the algorithm, which runs overall in approximately $O(n^2)$ time.

Result 2: We must perform the cognitive step above for each spinoff. Spinoffs become somewhat more frequent per clock tick as the number of clock ticks increases (e.g., if there are more schemas in the knowledge base, the number of new, spinoff schemas we are likely to create is higher). Thus, there is a slow growth in the overall work per clock tick to keep these tables updated, which grows per clock tick as the number of spinoffs per clock tick grows.

The problem of learning here is still $O(n^2)$, of course. We are still correlating *some* sensory bits with *some* predictions. However, the magnitude of this correlation has been decreased by an amount reflecting the behavior of the world and the characteristics of our predictions about that world.

3.2.3.2 Why were these parameters chosen?

The perceptual and cognitive strategies above place a high value on novel stimuli. Causes which precede their effects by more than a couple of clock ticks are not attended to. In the world described above, this is perfectly reasonable behavior. If the world had behaviors in it where more prior history was important, it would be necessary to attend further back in time to make schemas which accurately predicted the effects of actions.

The actual temporal horizons[13] used were determined empirically. For example, several runs using horizons for parameters of 1, 2, 3, 5, 10, 20, and 100 were tried, for both horizons (in other words, the cross-product of most of the combinations), and it was observed that the increase in learning was negligible (though not zero) above the values chosen. In the particular case of the sensory update horizon, values smaller than two tended to cause the statistical machinery to malfunction, missing most transitions (e.g., it became difficult to perceive that some particular bit did *not* change after some particular action).

Those readers familiar with the full schema mechanism described in [Drescher 91] may wonder about the interactions of synthetic items and this temporal horizon, particularly synthetic items employing composite actions. The runs investigated did not tend to generate large numbers of synthetic items (although see Section 4.4.2, on page 95, for some other remarks on this subject). If synthetic items *with composite actions* (which are used, for example, to represent object persistence [Drescher 91] and whose values might change

13. As specified in Section 3.2.3, the horizons were two clock ticks when doing perceptual update, and one clock tick when finding candidates for spinoffs.

due to events arbitrarily far in the past) were much more common, the temporal horizons used would most likely require some adjustment upwards. However, as Drescher points out [Drescher 94], it might suffice to represent intermediate states that keep track of the effects of the past events, so that only the temporally local values of those intermediate states need be attended to. However, this has not been investigated here.

Note that the above problem with synthetic items and the temporal horizons would *only* be true, however, if such synthetic items had actions which were composite—and this implementation, which lacks composite actions, cannot ever generate such synthetic items. Since all synthetic items generated in *this* implementation are therefore noncomposite, looking arbitrarily far back in history is not necessary.

These particular strategies also place a high value on a very specific spatial locality. Even sensory items that are very near items which have changed are not attended to. Since this microworld only has objects which are one bit wide, and the actions which involve them are, e.g., touch (which requires contact), this strategy works well.[14] In a world where actions had effects farther away than a single pixel in the visual field, or which contained objects subtending more pixels in the visual field, for example, such a strategy would have to be modified.

14. Selectors which attended to the unchanged items in a spatial “halo” around changed items were found to be less efficient, in terms of work per reliable schema, than the selectors described here. A different microworld (such as one with spatially larger objects, or different types of actions available to the agent) might require selectors that attended to a wider radius of (unchanged) sensory items around items which actually changed, in which case such “haloing” would be necessary to reliably learn the effects of actions.

3.3 Results

3.3.1 Evaluation of the focused algorithm

Two crucial questions that must be addressed concern how much the system learns *(completeness)*, and whether what it learns actually reflects its experience *(correctness)*. We must evaluate whether these focus of attention mechanisms impair that learning in any way. After all, one way to decrease the work of learning would be to simply ignore the world completely—but the resultant gain in speed could hardly be said to be worthwhile, because nothing would be learned.

The discussion below is based on results from the infant/eyehand scenario. Results from the Hamsterdam scenario have been comparable. However, since the majority of the goal-independent mechanism was developed using the infant/eyehand scenario, the Hamsterdam results were primarily restricted to simple verification of similar savings for similar mechanisms, and a relatively smaller number of experiments were performed. (The infant/eyehand scenario had literally dozens of experimental runs performed while the algorithms were being developed, from which, e.g., the chart shown in Figure 8, on page 63, is only a small sample. The Hamsterdam scenario tried out the resultant algorithms for verification and demonstrated that they worked acceptably there, too, but was not as exhaustively sampled as was the infant/eyehand scenario.)

3.3.1.1 Completeness

The schema system generates thousands of schemas in runs of reasonable duration, for instance, runs of ten or twenty thousand iterations have generated over 7000 schemas. How is one to know what all of these facts really represent? The state of the knowledge base is critically dependent upon prior knowledge: a more-detailed schema can only be generated from a less-detailed one, so any change in the learning mechanism which changes which schemas are generated leads to rapidly-diverging sets of generated schemas. While all may say *approximately* the same thing, the fine details of exactly which facts are learned will tend to be different. It would be possible to run enough iterations so that almost every possible fact that *is* true about the microworld could be *learned* to be true, but this is an unreasonably large amount of computation (the total number of learned schemas plotted over time appears to have an asymptote at least in the tens of thousands of schemas, even for this simple world).

3.3.1.1.1 Manual evaluation methods

Manual inspection of the schemas generated by these runs was employed as a first cut at establishing that alternative focus mechanisms were not substantially decreasing completeness, and tools were developed for examining how many schemas, representing what general categories of facts (e.g., unimodal visual field, multimodal across various modalities, etc) were being learned. By comparing rough totals of different types of generated schemas, one could obtain at least some assurance that some particular class of schemas was not being systematically omitted.

Another manual method of checking the results employed *n*-way comparisons of the generated schemas themselves. The **(context, action, result)** triple of each schema can be represented relatively compactly in text (ignoring all the statistical machinery that also makes up a schema); by sorting the schemas generated in any particular run into a canonical order, and then comparing several runs side-by-side, one can gain an approximate idea of how different runs fared. Figure 7, on page 60, demonstrates a tiny chunk from a 5-way comparison of a certain set of runs, in which 5 somewhat-different runs were compared for any large, overall changes to the types of schemas generated.[15]

3.3.1.1.2 Automatic evaluation methods

Manual methods are tedious and error-prone. Furthermore, the underlying reason that an agent learns is to aid it in the pursuit of its goal. This means that a sensible evaluation strategy is to ask if the agent has, indeed, learned enough to accomplish goals that it was unable to accomplish before learning.

A simple way to establish what the agent knows is therefore to use the generated schemas as parts of a plan, *chaining* them together such that the result of one schema serves as the context of the next, and to build these chains of schemas until at least one chain reaches from the initial state of the microworld to the goal state. If we can build at least one such chain, we can claim that the agent *knows* how to accomplish the goal in that context; the shortness of the chain can be used as a metric as to *how well* the agent knows.[16] For this task, the schemas to be used should be those deemed *reliable*, e.g., those which have been true sufficiently often in the past that their predictions have a good chance of being correct. Simply employing *all* schemas, reliable or not, will lead to many grossly incorrect chains. (A more complete description of the chaining system, and its use in evaluating the results of learning, is deferred until Chapter 4, where the generated chains are also used in goal-directed learning and behavior.)

At the start, no facts about the world are known, hence no chain of any length can be built. However, after a few thousand schemas are built (generally between 1000 and 5000),

15. The layout of this chart, and its ordering, is not accidental. This is, after all, a manual evaluation method; it depends on the ability of the human visual system to pick out aberrations in patterns. Large holes or gaps in the columns merit closer attention, allowing the effort of checking carefully to be limited to only a small number of cases.

16. Note that the small size of the microworld and the small number of actions possible at any given timestep mean that even a random walk through state space has a significant chance of accomplishing the goal, if we are willing to wait long enough; hence, a path which is *close to optimal*, rather than one which exists at all, should be our metric for whether learning has succeeded. See Section 4.4.3, on page 99, for a comparison of the length of the chains built for a typical goal versus the average length of a random-walk to the goal.

```
-vf23/eyer/fovf00___________________________________________________________________________________________________________________________________________________
-vf23/eyer/fovf11___________________________________________________________________________________________________________________________________________________
-vf23/eyer/fovf12___________________________________________________________________________________________________________________________________________________
-vf23/eyer/fovf22___________________________________________________________________________________________________________________________________________________
-vf23/eyer/fovf33___________________________________________________________________________________________________________________________________________________
_________________________________-vf24/eyef/-fovf00_______________-vf24/eyef/-fovf00________________________________________________________________________________
_________________________________-vf24/eyef/-fovf01_______________-vf24/eyef/-fovf01_______________-vf24/eyef/-fovf01_______________-vf24/eyef/-fovf01______________
_________________________________-vf24/eyef/-fovf11_______________-vf24/eyef/-fovf11_______________________________________________________________________________
_________________________________-vf24/eyef/-fovf12_______________-vf24/eyef/-fovf12_______________________________________________________________________________
_________________________________-vf24/eyef/-fovf22_______________-vf24/eyef/-fovf22_______________________________________________________________________________
_________________________________-vf24/eyef/-fovf23_______________-vf24/eyef/-fovf23_______________-vf24/eyef/-fovf23_______________-vf24/eyef/-fovf23______________
_________________________________-vf24/eyef/-fovf33_______________-vf24/eyef/-fovf33_______________________________________________________________________________
_________________________________-vf24/eyef/-vf23_________________-vf24/eyef/-vf23_________________-vf24/eyef/-vf23_________________-vf24/eyef/-vf23________________
_________________________________-vf24/eyel/(-vp12&-vf34)_________-vf24/eyel/(-vp12&-vf34)_________________________________________________________________________
_________________________________-vf24/eyel/-vp10_________________-vf24/eyel/-vp10_________________________________________________________________________________
_________________________________-vf24/eyel/vf44____________________________________________________________________________________________________________________
-vf30/eyeb/vp01____________________________________________________________________________________-vf30/eyeb/vp01__________________-vf30/eyeb/vp01_________________
_________________________________-vf30/eyer/-vp11_________________-vf30/eyer/-vp11_________________________________________________________________________________
___________________________________________________________________________________________________-vf31/eyeb/-fovr00_______________-vf31/eyeb/-fovr00______________
___________________________________________________________________________________________________-vf31/eyeb/-fovr01_______________-vf31/eyeb/-fovr01______________
___________________________________________________________________________________________________-vf31/eyeb/-fovr03_______________-vf31/eyeb/-fovr03______________
___________________________________________________________________________________________________-vf31/eyeb/-fovr11_______________-vf31/eyeb/-fovr11______________
___________________________________________________________________________________________________-vf31/eyeb/-fovr12_______________-vf31/eyeb/-fovr12______________
___________________________________________________________________________________________________-vf31/eyeb/-fovr22_______________-vf31/eyeb/-fovr22______________
___________________________________________________________________________________________________-vf31/eyeb/-fovr23_______________-vf31/eyeb/-fovr23______________
___________________________________________________________________________________________________-vf31/eyeb/-fovr30_______________-vf31/eyeb/-fovr30______________
___________________________________________________________________________________________________-vf31/eyeb/-fovr33_______________-vf31/eyeb/-fovr33______________
___________________________________________________________________________________________________-vf31/eyeb/-vf32_________________-vf31/eyeb/-vf32________________
-vf31/eyeb/vf31_____________________________________________________________________________________________________________________________________________________
___________________________________________________________________________________________________-vf31/eyef/-vf30_________________-vf31/eyef/-vf30________________
_________________________________-vf31/eyef/-vf44_________________-vf31/eyef/-vf44_________________________________________________________________________________
_________________________________-vf31/eyer/-fovb00_______________-vf31/eyer/-fovb00_______________________________________________________________________________
___________________________________________________________________________________________________-vf31/eyer/-fovb01_______________-vf31/eyer/-fovb01______________
___________________________________________________________________________________________________-vf31/eyer/-fovb02_______________-vf31/eyer/-fovb02______________
_________________________________-vf31/eyer/-fovb03_______________-vf31/eyer/-fovb03_______________________________________________________________________________
___________________________________________________________________________________________________-vf31/eyer/-fovb10_______________-vf31/eyer/-fovb10______________
___________________________________________________________________________________________________-vf31/eyer/-fovb13_______________-vf31/eyer/-fovb13______________
_________________________________-vf31/eyer/-fovb20_______________-vf31/eyer/-fovb20_______________________________________________________________________________
___________________________________________________________________________________________________-vf31/eyer/-fovb23_______________-vf31/eyer/-fovb23______________
_________________________________-vf31/eyer/-fovb30_______________-vf31/eyer/-fovb30_______________________________________________________________________________
___________________________________________________________________________________________________-vf31/eyer/-fovb31_______________-vf31/eyer/-fovb31______________
___________________________________________________________________________________________________-vf31/eyer/-fovb32_______________-vf31/eyer/-fovb32______________
_________________________________-vf31/eyer/-fovb33_______________-vf31/eyer/-fovb33_______________________________________________________________________________
___________________________________________________________________________________________________-vf31/eyer/-vf21_________________-vf31/eyer/-vf21________________
___________________________________________________________________________________________________-vf32&vf33/eyer/fovf00___________-vf32&vf33/eyer/fovf00__________
___________________________________________________________________________________________________-vf32&vf33/eyer/fovf11___________-vf32&vf33/eyer/fovf11__________
___________________________________________________________________________________________________-vf32&vf33/eyer/fovf12___________-vf32&vf33/eyer/fovf12__________
___________________________________________________________________________________________________-vf32&vf33/eyer/fovf22___________-vf32&vf33/eyer/fovf22__________
___________________________________________________________________________________________________-vf32&vf33/eyer/fovf33___________-vf32&vf33/eyer/fovf33__________
_________________________________-vf32/eyeb/-vf20_________________-vf32/eyeb/-vf20_________________-vf32/eyeb/-vf20_________________-vf32/eyeb/-vf20________________
```

Figure 7: A tiny chunk of a 5-way comparison of generated schemas

Five columns of schemas, sorted alphabetically by their printed representations, are shown here side-by-side, horizontally aligned. Small holes are fine, but large holes could indicate a potentially missing class of schemas.

most starting states can plausibly chain to a simple goal state, such as centering the visual field over the hand, in a close-to-optimal number of steps.

3.3.1.1.3 Completeness results

Given this mechanism, how well did the focus of attention mechanisms fare? Quite well. In general, given the same approximate number of generated schemas, both the basic and focused approaches cited above learned “the same” information: they could both have plausibly short chains generated that led from initial states to goals. Both the chaining mechanism described above, and manual inspection, showed no egregious gaps in the knowledge or particular classes or types of facts that failed to be learned.

As shown in Figure 8, on page 63, and explained in Section 3.3.2, on page 62, the focused approach tended to require approximately twice as many timesteps to yield the same number of schemas as the unfocused approach. This means that a real robot which employed these methods would require twice as many experiments or twice as much time trundling about in the world to learn the same facts. However, the reduction of the amount of computation required to learn these facts by between one and two orders of magnitude[17] means that the processor such a robot must employ could be much smaller and cheaper—which would probably make the difference between having it onboard and not. This is even more compelling when one realizes that these computational savings get bigger and bigger as the robot learns more facts.

3.3.1.2 Correctness

The statistical machinery of the schema mechanism goes to great pains to avoid being fooled by occasional coincidence. Only if some change in the state of the world is positively correlated with an action more often than it is negatively correlated, and only if we have seen enough instances of both the event happening after some specific action and the

17. For runs of this length, e.g., 1000-2000 schemas generated.

event *not* happening in the absence of the action, *and* if the event is unexplainable by any other schemas, will the mechanism conclude that the action is truly the cause of the event. (This is an overview of the marginal attribution mechanism described in Section 3.1.3.2, on page 42, and in [Drescher 91].)

Because of this, the only way that any learning algorithm which uses this system could learn “incorrect” facts (e.g., correlations that do not, in fact, reflect true correlations in the world) would be to *systematically* exclude relevant evidence that indicates that a schema that is thought to be reliable is in fact unreliable. No evidence of this was found in spot checks of any test runs. It is believed (but not proven) that none of the focused algorithms described can lead to such systematic exclusion of relevant information: the mechanism may miss correct correlations (such is the tradeoff of having a focus of attention in the first place), but it will not miss only those correlations that would tend to otherwise invalidate a schema thought to be reliable.

3.3.2 Comparison of the different strategies

Figure 8, on page 63, presents partial results from several runs with different choices of selectors. Only the most salient combinations of selectors were included in this table. Of those, the rows in boldface will be discussed below; the non-boldfaced rows are included to give a feel for how different choices can influence the results.

The results in this table were all produced by runs 5000 iterations long. Similar runs of two or three times as long have produced comparable results, with correspondingly greater increases in β (see below).

The first four columns of the table show the particular selectors in use for any given run; the top row shows those selectors which correspond to the basic (Drescher) algorithm,

Algorithm				Learning			Work required			Facts per work unit		
Spinoff selectors		Statistic selectors		Schemas			Inner loops ($x10^6$)			Reliable schemas over		
Items	Schemas	Items	Schema	Total	Rel	T/R	Spin	Stat	Both	Spin	Stats	Both
AIN	**ASN**	**AIN**	**ASN**	**1756**	**993**	**1.77**	**533**	**533**	**1066**	**1.9**	**1.9**	**0.9**
AIN	ASN	CINIH	ABSPSDUCI	1135	403	2.82	398	12	410	1.0	33.6	**1.0**
AIN	ASN	AIN	ABSPSDUCI	1110	518	2.14	391	55	446	1.3	9.4	**1.2**
CIN	**ASN**	**AIN**	**ASN**	**1693**	**948**	**1.79**	**44**	**524**	**568**	**21.5**	**1.8**	**1.7**
CIN	**SWRUS**	**AIN**	**ASN**	**1395**	**791**	**1.76**	**2**	**463**	**466**	**316.4**	**1.7**	**1.7**
CIN	ABSPSDUCI	AIN	ASN	1622	924	1.76	15	510	525	61.6	1.8	**1.8**
CIN	ASN	AIN	ABSPSDUCI	1110	506	2.19	33	54	87	15.3	9.4	**5.8**
CIN	ABSPSDUCI	AIN	ABSPSDUCI	1110	506	2.19	10	54	64	50.6	9.4	**7.9**
CIN	ABSPSDUCI	CINIH	ASN	1366	643	2.12	13	64	77	49.5	10.0	**8.4**
CIN	ASN	CINIH	ABSPSDUCI	1136	399	2.85	34	12	46	11.7	33.3	**8.7**
CIN	**SWRUS**	**AIN**	**ABSPSDUCI**	**1102**	**498**	**2.21**	**1**	**53**	**54**	**415.0**	**9.4**	**9.2**
CIN	SWRUS	CINIH	ASN	1353	688	1.97	2	64	66	275.2	10.8	**10.3**
CIN	ABSPSDUCI	CINIH	ABSPSDUCI	1136	399	2.85	10	12	22	40.7	33.3	**18.3**
CIN	**SWRUS**	**CINIH**	**ABSPSDUCI**	**1134**	**398**	**2.85**	**1**	**12**	**13**	**331.7**	**33.2**	**30.2**

Figure 8: Summary of goal-independent results

The names for the algorithms used in learning are explained in Section 3.2.1 through Section 3.2.3. Results from the rows in boldface are discussed in this section. The non-boldfaced rows are not discussed in the text, but are included for additional context. This table is a sampling; in all, in excess of 30 different combinations of selectors were investigated.

The top line is effectively the "unfocused" case, as in [Drescher 91]; the bottom line is considered the "best" or most tightly-focused case.

while the bottom row shows the most highly-focused algorithm, as described in Section 3.2.3, on page 52.

The table is sorted in order by its last column, which shows number of *reliable* schemas generated during the entire 5000-iteration run, divided by the amount of total work required. For conciseness, we shall call the value in this column β, which is defined as:

$$\text{Facts per work unit} = \beta = \frac{SchemaRel_{Total}}{Work_{Total} \times 10^{-6}}$$

where the multiplication by 10^{-6} is simply to normalize the resulting numbers to be near unity, given the millionfold ratio between work units and number of schemas generated.

The bold rows in the table show successive changes to the selectors used, one at a time. The top row is the basic algorithm, which shows that about a billion total inner loops were required to learn 1756 schemas, 993 of which were reliable, which gives a β of 0.9.

Note that, because the world is stochastic (for example, the two “inanimate” objects occasionally move from one square to a neighboring square, approximately every few hundred clock ticks), one might imagine that there would be variance in the number and reliability of schemas generated between two runs, even if they use the same strategy. In fact, this is true, but the variance is quite low: out of a run of two or three thousand schemas with the same strategy and different seeds for the random number generator (hence different random behaviors in the world), the difference in the number of schemas generated is generally less than ten. In other words, the number of schemas generated is usually within 1% between runs using the same algorithm. Further, the types of schemas generated also match each other quite closely, as determined by *n*-way comparisons between runs, using the techniques discussed in Section 3.3.1.1.1, on page 58. (The exact schemas generated will, of course, be different, as discussed in Section 3.3.1.1, on page 57).

Let us examine changes to the selectors for spinoffs, which determine the cognitive selectivity, or what is attended to in learning new schemas from the existing schemas. When we change $Spin_i$ from **AIN** to **CIN** (e.g., from all item numbers to those whose items changed at the last clock tick), the amount of work drops by about a factor of two, while the number of generated schemas barely decreases. This means that virtually all schemas made predictions about items which changed in the immediately preceding clock tick (e.g., that corresponding to the action just taken), hence looking any further back in time for them costs us computation without a concomitant increase in utility.

Changing $Spin_s$ from **ASN** to **SWRUS** (e.g., from all schema numbers to those whose statistics were recently updated), given that $Spin_i$ is already using the selector **CIN**, yields a small improvement in β (not visible at the precision in the table), and also a small improvement in the ratio of reliable to total schemas. (Were $Spin_i$ not already **CIN**, the improvement would be far more dramatic, as demonstrated in runs not shown in the table.) Note, however, the enormous decrease in the amount of work done by the spinoff mechanism when $Spin_s$ changes from **ASN** to **SWRUS**, dropping from 10% of $Work_{Total}$ to 1%.

Next, let us examine the effects of perceptual selectivity. Changing $Stat_s$ from **ASN** to **ABSPSDUCI** (e.g., from all schema numbers to all bare schemas plus schemas dependent upon changed items) increases β by a factor of 5.4, to 9.2, by decreasing the amount of work required to update the perceptual statistics by almost an order of magnitude. In essence, we are now only bothering to update the statistical information in the extended context or extended result of a schema, for some particular sensory item in some particular schema, if the schema depends upon that sensory item.

Finally, examine the last bold row, in which $Stat_i$ was changed from **AIN** to **CINIH** (e.g., from all item numbers to all item numbers whose items changed in the last two clock ticks). β increases by a factor of 3.2, relative to the previous case, as the amount of statistical-update work dropped by about a factor of four. We are now perceiving effectively only those changes in sensory items which might have some bearing in spinning off a schema which already references them.

Note that each successive tightening of the focus has some cost in the number of schemas learned in a given number of iterations. This means that, e.g., a real robot would require increasingly large numbers of experiments in the real world to learn the same facts. However, this is not a serious problem, since, given the focus algorithm described here, such a robot would require only about twice as many experiments, for any size run, as it would in the unfocused algorithm. This means that its learning rate has been slowed down by a small, roughly constant factor, while the computation required to do the learning has dropped enormously.