Neo: Learning Conceptual Knowledge by Sensorimotor Interaction with an Environment

Paul R. Cohen, Marc S. Atkin, Tim Oates
Experimental Knowledge Systems Laboratory
Department of Computer Science, LGRC, Box 34610
Carole R. Beal
Department of Psychology
University of Massachusetts, Amherst, MA 01003
{cohen,atkin,oates}@cs.umass.edu

Abstract

Recent developments in philosophy, linguistics, developmental psychology and artificial intelligence make it possible to envision a developmental path for an artificial agent, grounded in activity-based sensorimotor representations. This paper describes how Neo, an artificial agent, learns concepts by interacting with its simulated environment. Relatively little prior structure is required to learn fairly accurate representations of objects, activities, locations and other aspects of Neo's experience. We show how classes (categories) can be abstracted from these representations, and discuss how our representation might be extended to express physical schemas, general, domain-independent activities that could be the building blocks of concept formation.

Introduction

Our goal is to build a baby, or rather, an artificial agent who lives in a simulated environment and who eventually learns to think like a three year old. A virtual infant, if you like. The abilities we want our agent, which we call Neo, to develop include learning, planning, language, an organized memory containing structured knowledge, and attention. Central to these abilities is a conceptual structure, an ontology, a way to "carve the world at its joints." Given a conceptual structure, we can see a developmental path to more advanced thought, including emotion and consciousness. As Neo develops, we expect to learn much about the nature of accessible representations, the functions of categories, and the representation and roles of goals. We will elucidate the relationship between activity, attention, memory and learning. We hope to demonstrate that an agent can develop sophisticated knowledge from minimal beginnings. The goal of this paper is more modest, however: to show that Neo can acquire concepts and categories from interacting with its environment.

Categorization, as Lakoff points out, is central to human thought (Lakoff 1984). It is also central to Lakoff and Johnson's challenge to Objectivism (Johnson 1987; Lakoff & Johnson 1980; Lakoff 1984), the dominant view in Western philosophy that there is an objective way to represent the world and reason about it. Influenced by Eleanor Rosch's research on categorization, Lakoff and Johnson argue that categories are based less on objective features such as color, size, and shape, than on *interactional* properties and relationships, such as "graspable" and "fits-in-my-mouth," which characterize how an agent interacts with its environment. At the same time, AI researchers such as Agre (Agre 1988), Chapman (Chapman 1991), and Ballard (Ballard 1989) have argued for *deictic* or agent-centered representations. Lakoff and Johnson make a convincing case that adult conceptual structures are grounded in primitive interactional knowledge that could very well be acquired by infants. Thus, in contrast with Piaget's theory of developmental *stages* (Ginsberg & Opper 1988), we can now envision a continuous developmental trajectory—for conceptual knowledge, at least—beginning with simple, interactional primitives in infancy (relationships among actions and objects in the physical world, for example) and becoming more elaborate through abstraction and metaphorical extension as the agent develops (Lakoff & Johnson 1980). Indeed, Mandler's work, to which we owe much, outlines such a developmental trajectory and the empirical evidence for it (Mandler 1988; 1992). One contribution of the current paper is to show that a simulated infant can learn concepts in roughly the way Mandler suggests real infants learn.

There is a strong temptation to see in the infancy literature evidence of nativism, the idea that infants are born with conceptual structures. Obviously, babies' minds have some structure at birth, but we are anti-nativist, minimalist in our approach. We do not agree that babies must be born with theories of the physical

Agents '97 Marina del Rey CA USA ©1997 ACM

world (Carey & Gelman 1991), and in fact we show that Neo can learn concepts given very little prior structure.

Baby World

Neo is a virtual agent who lives in a simulated environment. Babyworld implements Neo's sensations, mental representations, mental and physical activities, and the behavior of objects and other agents that interact with Neo. Babyworld has two parts: one, which we call Neo, implements everything that Neo does, including learning, moving, looking, crying, and so on. The other part, called StreamsWorld, represents Neo's internal and external environment, and it implements events that happen in and around Neo and in response to Neo's actions. Notably, StreamsWorld represents Neo's sensations of its external environment and also internal states such as hunger. No distinction is made between "inside" and "outside"; Neo must learn it.

Neo senses its environment through a collection of ***streams***, which are divided into discrete time steps. On each time step t, a stream σ_i holds a ***token*** τ, that is, $\sigma_{i,t} = \tau$. Tokens represent ***sensations*** or processed percepts. For example, one token is "rattle-shape," and it is placed in the appropriate stream whenever Neo's eyes point at an object that is shaped like a rattle; that is, $\sigma_{\mathsf{sight-shape,t}}$ =rattle-shape. (We will discuss the bias this tactic engenders in a later section.) The streams that represent Neo's internal sensations include an affect stream that contains tokens such as happy and sad, a pain stream, a hunger stream, and somatic and haptic streams that are active when Neo moves and grasps.

The Babyworld simulator is simple and probabilistic. For example, Neo gets hungry some time after eating, it cries when it is unhappy or in pain; when Neo cries, Mommy usually visits, unless she is angry at Neo for crying, in which case she stays away. Neo falls asleep intermittently; it can move its arm and head, and grasp several objects, including three rattles, a bottle, a mobile, a bunch of metallic keys, and a knife. The latter causes pain. The rattles make noise when shaken.

Currently, Neo is incapable of anything we would call volition. If Neo's eyes alight on a rattle then Neo will grasp the rattle with some probability. However goal-directed this might appear, Neo's mind contains nothing that could be interpreted as a goal.

How Neo Learns

What does Neo's mind contain, then? Fluents, mostly. Fluents represent things that don't change, or that change in highly regular, predictable ways. Fluents are a step away from the state-based representations of AI planning research, such as situation calculus, toward script-like representations. Fluents explicitly represent events that have duration; in fact Neo learns the mean and variance of the duration of each fluent. The sound made by a rattle is a fluent, so is the sensation of holding the rattle, and so are the visual sensations of the shape and color of the rattle. Of course, the concept "rattle" has all these components, so somehow, fluents for the color, shape, sound and texture of a rattle must be linked up in a single fluent. Neo accomplishes this, building larger fluents from smaller ones, with two simple learning rules that count cooccurrences. First, if Neo notices that two fluents often start and stop simultaneously, it infers that the fluents are parts of a larger one. This rule learns fluents that represent objects and states. Second, if Neo notices that one fluent often follows another, it infers that both are parts of a larger fluent that represents an activity. The word "often" in these rules hides a statistical inference that fluents cooccur more frequently than one would expect by chance if they were independent.

The important features of fluents as representations are that they represent states or processes with temporal extent (even objects are represented as things that exist over time), they are composable, and they are learned by counting cooccurrences. Although the simplest fluents represent sensations, it's important to recognize that fluents are not identical with sensations. Sensations are tokens in streams; fluents are representations stored in memory. Streams are the locus of Neo's sensory experience, fluents are the locus of Neo's knowledge. And although Neo's earliest fluents are just copies of its sensations, they soon become aggregated and abstracted.

All the examples in this paper are from a single "run" of Neo, lasting 30,000 time steps. (A time step corresponds to one second of real time. The run we describe here thus corresponds to roughly eight hours of Neo's life.) Fluents are learned gradually: It might take hundreds or thousands of time steps to find enough cooccurrences to create a fluent, and composite fluents are obviously learned after their components.

Having said Neo combines small fluents into larger ones, we should say where the small fluents come from. One approach is to say the smallest fluents are token values that persist in streams. For example, **red** and **rattle-shape** are each persistent tokens because Neo tends to look at objects for several time steps before shifting its gaze; if Neo is currently seeing something red, then it will probably see something red on the next time step. Our first implementation of Neo had individual, persistent tokens as the smallest fluents, but there were too many ways to combine these fluents, and most produced larger fluents that didn't correspond to anything in Neo's environment. We decided that the smallest fluents should be larger than individual tokens. And if the smallest fluents contained, say, ***two*** tokens, from different streams, then we could use the covariance of streams to focus Neo's attention on pairs of tokens that “go together,” as we will now describe.

Scopes

The first things Neo learns are not fluents, but rather, pairs of streams in which to look for fluents. These pairs are called ***scopes***. A stream σ_i is said to change state at time t, denoted $\Delta(i,t)$, when $\sigma_{i,t-1} \neq \sigma_{i,t}$; that is, σ_i changes state at time t when it contains a different token at time t than it did at time $t-1$. Conversely, $\overline{\Delta}(i,t)$ means the stream doesn't change state: $\sigma_{i,t-1} = \sigma_{i,t}$. Neo learns a ***scope***, s_{ij}, when streams σ_i and σ_j change together often. Said differently, Neo learns s_{ij} when the joint event $\Delta(i,t)$ & $\Delta(j,t)$ occurs frequently relative to the joint events $\Delta(i,t)$ & $\overline{\Delta}(j,t)$ and $\overline{\Delta}(i,t)$ & $\Delta(j,t)$. To assess the relative frequencies of these events, Neo uses contingency tables like this one:

	Δ(sight-color,t)	$\overline{\Delta}$(sight-color,t)	*total*
Δ(sight-shape,t)	2996	945	3941
$\overline{\Delta}$(sight-shape,t)	826	25232	26058
total	3822	26177	29999

This says that the streams **sight-shape** and **sight-color** changed state simultaneously 2996 times, and one changed when the other didn't 945 + 826 = 1771 times. To assess the strength of association between **sight-shape** and **sight-color** we square the frequency in the first cell of the contingency table (2996) and divide by the product of the first row and first column margins (3941 and 3822, respectively). The maximum value for this statistic is 1.0, and for the table above it is $2996^2/(3941 \times 3822) = .596$.[1]

Neo maintains contingency tables for all pairs of streams.[2] When the measure of association for a table exceeds a threshold, a scope is created. Table 1 shows the top ten scopes learned by Neo in a run of 30,000 timesteps (i.e., the scopes with the highest measures of association), and also the ten worst scopes. Notice that the **sight-shape** and **sight-color** streams are more highly associated than any except **do-sleep** and **sleep**. Other high-ranking scopes are **(sound voice)**, which makes sense because changes in the **sound** stream are often produced by changes in the **voice** stream; and **(tactile-mouth mouth)**, which captures the fact that when Neo starts to mouth (i.e., chew on) an object, it gets tactile sensations in its mouth. The worst scopes represent pairs of streams that are not associated. For example, there is no association between sleeping and eating, and none between moving the arm (**do-arm**) and hunger. Note that many scopes include **do-x** streams; for example, **(do-voice voice)**. These are the components of Neo's actions: the **do-voice** part represents the sensation of attempting to use the voice, and the **voice** part represents the sensory feedback from actually using the voice. Sometimes, Neo will experience **do-voice** but not **voice**; for example, Neo might try to make a sound (**do-voice**) but be unable to because it has an object in its mouth.

Before Neo learns any scopes, its world is a “blooming, buzzing confusion” of changing token values in 26 streams. Scopes “chunk” streams into covarying pairs. Without scopes, Neo has to learn fluents by searching for associated token values across all 26 streams. For example, Neo has to consider associations between **(sight-color red)** and, for instance, **(sleep asleep)**, **(hunger full)** and **(voice screaming)**. With scopes, Neo can limit its search for associations. Suppose Neo learns the scope **(sight-color sight-shape)** but it learns no other scopes relating **sight-color** to any other stream. Then, it should look for associations between **(sight-color red)** and tokens in the **(sight-shape)** stream, but it needn't look for associations in any other streams. Empirically, scopes make an enormous difference in the number and quality of the associations Neo learns. Without scopes, Neo learns many thousands of meaningless associations between token values; with them, Neo learns a few hundreds of associations that correspond to objects and activities in its environment.

Base Fluents

Whereas scopes represent the tendency of streams to change state simultaneously, Neo's smallest fluents, called *base fluents*, represent cooccurring tokens within scopes. Suppose stream σ_i contains a at time $t-1$ and b at time t. Then we say token a ***stops*** in stream i at time $t-1$, denoted $\dashv$(i,a,t-1), and token b ***starts*** in stream i at time t, denoted $\vdash$(i,b,t). Now suppose Neo turns its head and its eyes alight on a red rattle. Neo will detect two simultaneous events, $\vdash$(sight-color,red,t) and $\vdash$(sight-shape,rattle-shape,t). Sometime later, Neo might look somewhere else, which will generate two simultaneous stop events, $\dashv$(sight-color,red,v) and $\dashv$(sight-shape, rattle-shape,v). Simultaneous start events and stop events are evidence that a single object—in this case a red rattle—or a single activity, is making its presence felt in two streams. Of course, two *unrelated* events could occur simultaneously in two

[1] Neo could use other statistics, such as χ^2 and G, provided the contingency table is scaled to a constant total, preserving the proportions. (Scaling is necessary because χ^2 and G are not independent of sample size.) In practice, Neo learns the same scopes, and ranks them similarly, irrespective of how it measures association in its contingency tables.

[2] While this may seem like a lot of bookkeeping, especially as the number of streams grows, bear in mind that maintaining a contingency table is a very simple operation. All the contingency table updates could be done in unit time on a parallel machine. At higher levels of processing, operations may become more complex and require non-local data. In that case, we view it as the purpose of the *attentional mechanism* to keep processing per unit time down to reasonable level. Deciding what particular events in the environment require attention, i.e. merit being assigned computational resources, is a future direction of our research.

10 Best Scopes	**measure of association**	**10 Worst Scopes**	**measure of association**
do-sleep sleep	1.0	sleep arm-x-angle	≈ 0
sight-color sight-shape	.596	arm-x-angle arm-y-angle	≈ 0
sound voice	.533	do-sleep arm-x-angle	≈ 0
arm arm-speed	.453	do-sleep eat	≈ 0
do-mouth mouth	.326	sleep eat	≈ 0
do-voice voice	.325	tactile-skin tiredness	≈ 0
tactile-mouth mouth	.315	do-arm hunger	≈ 0
sound do-voice	.276	do-sleep arm-y-angle	≈ 0
tactile-mouth do-mouth	.274	sleep arm-y-angle	≈ 0
do-head head	.254	tactile-mouth hunger	≈ 0

Table 1: The ten best and ten worst scopes learned by Neo in 30,000 timesteps.

streams, but this sort of coincidence is less likely than the coincidence of related events.

Neo looks for associations between start and stop events within scopes. For example, having the scope **(sight-color sight-shape)**, Neo can try to associate **red** and **rattle-shape**. But if Neo lacks a scope for, say, **sight-color** and **arm-speed**, then it will never try to associate **red** with **fast**. Thus scopes prevent Neo from even considering many meaningless base fluents.

Contingency tables count the cooccurrences of start and stop events, and assess whether start and stop events happen simultaneously significantly often. For example:

	⊢(sight-color, red,t)	⊢(sight-color, $\overline{\text{red}}$,t)	*total*
⊢(sight-shape, rattle-shape,t)	65	27	92
⊢(sight-shape, $\overline{\text{rattle-shape}}$,t)	237	1931	2168
total	302	1958	2260

Of the 2260 times **sight-color** and **sight-shape** changed together, **rattle-shape** became active 92 times, both **rattle-shape** and **red** became active 65 times, and **red** became active but **rattle-shape** didn't 237 times. The conditional probability of **rattle-shape** starting clearly depends on whether something red or non-red started; these probabilities are 65/92 = .71 and 27/92 = .29, respectively. Conversely, the conditional probability of something red starting depends on whether something rattle-shaped started. In short, **red** and **rattle-shape** are associated. The strength of their association can be measured many ways; one was described in the previous section. Here, we use a modified G statistic. Because G is sensitive to sample size, all contingency tables are first scaled to maintain their proportions but have their totals equal 100. To scale the table above, each cell value and marginal total would be divided by 226. Then the G statistic is calculated for the scaled table in the usual way.

Neo accepts a base fluent when its contingency table is significant, as measured by the G statistic. Actually, the table above tells us only that **red** and **rattle-shape** often start simultaneously, we also need to establish that they often end simultaneously. For this, Neo maintains another table like the previous one for the events ⊣(sight-shape,rattle-shape,t), ⊣(sight-color,red,t), and their complements. When Neo has evidence that **red** and **rattle-shape** both start and stop together in their respective streams, and do so more often than would be expected by chance if they were independent, then it forms the base fluent **((sight-shape rattle-shape)(sight-color red))**.

To summarize the story to this point, Neo learns scopes, or pairs of streams that often change together. As soon as it has learned a scope, Neo can use it to learn base fluents, which are scopes instantiated with particular token values, such as **((sight-shape rattle-shape)(sight-color red))**. And as soon as Neo has learned some base fluents, it starts combining them into larger structures called composite fluents.

Composite Fluents

Whereas base fluents represent associations between cooccurring token values in streams, composite fluents represent cooccurring fluents. Neo currently forms two kinds of composite fluents. ***Conjunctive*** fluents are generated when fluents F_1 and F_2 start together significantly often, and they also end together significantly often. Clearly, conjunctive fluents are like base fluents. However, base fluents combine token values into fluents, whereas conjunctive fluents combine other fluents; and conjunctive fluents are not constrained by scopes. Contingency tables, like those described earlier, tabulate the frequencies of the joint events $(\vdash F_1 \ \& \ \vdash F_2)$, $(\vdash F_1 \ \& \ \vdash \overline{F_2})$, $(\vdash \overline{F_1} \ \& \ \vdash F_2)$ and $(\vdash \overline{F_1} \ \& \ \vdash \overline{F_2})$. The modified G statistic, described above, tells Neo whether the association between F_1 and F_2 is significant. If so, Neo forms the fluent **(AND F_1 F_2)**.

The second kind of composite fluent is formed when one fluent starts in the context of another one. Suppose Neo is holding a rattle, and then it starts to chew on the rattle (called "mouthing"). While it is holding the rattle, the fluent **((tactile-hand wood)(hand close))** is active, and when it starts mouthing, the fluent **((tactile-mouth wood)(do-mouth mouth))** will become active. The latter fluent starts in the context of the former. If this happens significantly often then Neo will form the ***context fluent***

```
(CONTEXT ((tactile-hand wood)(hand close))
         ((tactile-mouth wood)(do-mouth mouth)))
```

The contingency tables for context fluents are updated in a slightly different way from previous tables. When fluent F_2 starts at time $t+i$, Neo checks to see whether fluent F_1 is active, and if so, it updates the first cell of the contingency table, $(\vdash F_1, t \;\&\; \vdash F_2, t+i)$. If F_2 starts and F_1 isn't active, then Neo updates the third cell of the table, $(\overline{\vdash F_1, t} \;\&\; \vdash F_2, t+i)$. If F_1 is active but F_2 doesn't start within a window of i time steps, then Neo increments the second cell of the table, $(\vdash F_1, t \;\&\; \overline{\vdash F_2, t+i})$. The modified G statistic tells Neo whether F_2 starts in the context of F_1 more often than would be expected by chance if F_1 and F_2 were independent.

Chains

Context fluents can be chained together to form multi-fluent sequences. Consider the previous context fluent and the following one:

```
(CONTEXT ((tactile-mouth none) (voice cry))
         ((tactile-hand wood) (hand close))
```

These fluents share a common fluent, **((tactile-hand wood) (hand close))**, so they may be composed into a chain:

```
(CHAIN ((tactile-mouth none) (voice cry))
       ((tactile-hand wood) (hand close))
       ((tactile-mouth wood)(do-mouth mouth)))
```

In words, Neo had nothing in its mouth and was crying, then it grabbed something wooden, then it started to mouth something wooden. Now consider another, very similar, chain:

```
(CHAIN ((tactile-mouth none) (voice cry))
       ((tactile-hand plastic) (hand close))
       ((tactile-mouth plastic)(do-mouth mouth)))
```

The only difference between these chains is the object that Neo grabs and mouths: in the first case it is wooden, in the second, plastic. We may form a ***class*** of things that Neo can grab and mouth. The chains don't say exactly which objects are in the class, but we know they are either wood or plastic, and they are graspable, and mouthable.[3]

Classes

Note that "graspable" and "mouthable" are ***interactional*** properties (Lakoff 1984) that characterize Neo's activities in its environment. Unlike "texture"—wood or plastic—they are fundamentally ***subjective***. What's graspable by one agent isn't necessarily graspable by another. Whereas texture is an inherent property of an object, graspable is a property of the object ***and the agent*** who may try to grasp it. Objective features such as texture have gotten a bad name because they appear inadequate for conceptual activities such as forming classes and judging similarity; for instance, it is difficult, perhaps impossible, to define a category in terms of necessary and sufficient objective features. One is tempted by the conjecture that categories might be defined in terms of necessary and sufficient interactional features, instead. However, we will try to show that categories are best defined in terms of ***activities***, and the apparent superiority of interactional features is due to them describing activities better than objective features.

Consider a conceptual activity such as judging whether a cup and a ladle are similar or different. We immediately want to ask, "Similar or different in what context?" As devices for transferring liquid, cups and ladles are similar; as containers to drink from, they require different motor schemas; as something to serve coffee in at an elegant dinner, they aren't similar. Note that the required context in which we judge similarity is often an activity, often purposeful, and often agent-centered. What's central to judging the similarity of objects is knowing which activities the objects are involved in. Activities seem to ***select*** which features of objects are relevant to judgments of similarity; these features will sometimes be objective, often interactional.

Why would Neo even want to have concepts, in particular classes? In a nutshell, because they capture the structure of Neo's environment. Classes enable Neo to make predictions about never before seen situations. If Neo had formed the class of "rattle", and was given a new rattle that resembled its old ones in some way, it could immediately infer that this new object can be grasped, held, and may even make a noise when shaken. All the ***entailments*** of "rattle," all the things a rattle can be used for and how it is expected to behave, are immediately transferable to this new object. One can also imagine that once a class has been established, a symbol could be fairly easily attached to it. This symbol makes it possible for an agent to communicate its experience succinctly, with just one "word". It can also reason about its experiences, for example during planning, at a level higher and more easily manageable than the raw sensory data.

Neo's activities are not purposeful because Neo has

[3] In general, we don't know that the object being grabbed is identical with the object being mouthed, but the Neo simulator is simple enough that this general problem doesn't arise. It will, eventually.

no goals, but even so, Neo's activities—represented as chains—are a basis for judging similarity and forming some classes. In fact, although Neo learns chains, we are responsible for using these chains to identify features and form classes. This is how we do it: We match up chains that have the same stream names in the same order, creating an ***abstract chain***, then form classes of the token values. Consider these chains:

```
(CHAIN ((do-arm resting) (arm resting))
       ((do-hand close) (hand close))
       ((tactile-mouth wood) (mouth mouthing)))

(CHAIN ((do-arm resting) (arm resting))
       ((do-hand close) (hand close))
       ((tactile-mouth plastic) (mouth mouthing)))

(CHAIN ((do-arm move-rt) (arm move-rt))
       ((do-hand close) (hand close))
       ((tactile-mouth wood) (mouth mouthing)))

(CHAIN ((do-arm move-rt) (arm move-rt))
       ((do-hand close) (hand close))
       ((tactile-mouth plastic) (mouth mouthing)))
```

Looking only at stream names (e.g., **do-arm, arm, do-hand**, ...) we see that all these chains are instances of this abstract chain: **(do-arm arm) → (do-hand hand) → (tactile-mouth mouth)**. What we're seeing in the four chains, above, is two activities:

resting arm →
closing hand →
mouthing something
right-moving arm →
closing hand →
mouthing something

and the "something" in each activity is either wood or plastic. It may not be immediately apparent how these activities identify classes of objects, but in fact the objects that can participate in these activities are just those objects that can be grasped, mouthed, and are either wood or plastic. We know, because we built the Neo simulator, that these objects include Neo's rattles and bottles, but not the mobile, Mommy, or Neo's own hand. (Incidentally, if Neo had run longer, it might have learned that its keys can also participate in the abstract chain, above, in which case the instantiated chains would have included a fluent **((tactile-mouth metallic)(mouth mouthing))**.) The point is that the class of objects that can participate in an activity is identified by interactional and objective features—graspable, mouthable, wooden, plastic or metallic.

Although Neo is currently not capable of it, we can imagine showing Neo a novel object, say, a small wooden block, that Neo would classify with its rattles, bottles and keys on the basis of being graspable and mouthable, and wooden, plastic or metallic. Similarly, if we show Neo a graspable but immovable wooden object, Neo might put it in the same class as its crib bars. Again, this classification would be made on the basis of activities. When Neo interacts with the new object it would form a chain that includes **((do-hand close) (hand close)) ((do-arm move-rt)(arm not-move))**—after grasping, Neo tries to move its arm but it cannot. Assuming that Neo has learned a similar chain in its interactions with the crib bars, it will be able to form the class of objects that can be grasped but cannot be moved.

What Neo ***can*** do is less spectacular, but promising. One of the most interesting abstract chains is only two fluents long: **((do-head head) → (sight-shape sight-color))**. This chain identifies many of the physical objects in Neo's environment. For example,

```
(CHAIN ((do-head lookup)(head lookup))
       ((sight-shape mobile-like)
                    (sight-color green)))
```

This chain says, when Neo directs its head to look up, and actually looks up, then it will see the green mobile. Here are two more examples:

```
(CHAIN
    ((do-head look+90)(look +90))
    ((sight-shape crib-like)(sight-color white)))
(CHAIN
    ((do-head look-90)(look -90))
    ((sight-shape crib-like)(sight-color white)))
```

In words, when Neo looks either to its extreme left (+90) or extreme right (-90) it sees its crib.

Remarkably, fluents that represent the color and shape of objects do not appear in any other chains learned by Neo; in other words, the **((do-head head) → (sight-shape sight-color))** chain represents an activity that defines the class of objects based on their appearance after a head turn.

Classes of actions can be learned in the same way. For example, here are four chains that Neo learned quickly and observed frequently:

```
(CHAIN ((do-arm move-up) (arm move-up))
       ((sight-movement fast) (arm-speed fast)))

(CHAIN ((do-arm move-dn) (arm move-dn))
       ((arm-movement fast) (arm-speed fast)))

(CHAIN ((do-arm move-rt) (arm move-rt))
       ((sight-movement fast) (arm-speed fast)))

(CHAIN ((do-arm move-lf) (arm move-lf))
       ((sight-movement fast) (arm-speed fast)))
```

The abstract chain is **(do-arm arm) → (sight-movement arm-speed)**, and it defines a class of actions in which, in the context of an arm movement, Neo sees the arm moving fast.

Physical Schemas

We have seen how fluents can be used to define activities (by abstracting regularities in streams over time), and how a set of related fluents can be used to implicitly define a category such as "graspable object." This type of conceptual knowledge is still relatively specific to Neo's capabilities and the particulars of Neo's environment. What if it were possible to abstract an even more general kind of fluent that represents activities or classes ***across domains***? We call such data structures ***physical schemas***, and postulate that they are central to the development of a cognitive agent. Their uses include providing a basis for communication between agents (by establishing a common frame of reference), forming the building blocks for further abstract categories, and bridging the gap between an agent's sensorimotor behavior and its higher level cognitive skills.

Conisder as an example for a physical schema the notion of "containment". It looks deceptively simple, but actually covers a great number of situations, at varying levels of abstraction. In terms of an agent's activities, there is a difference between containing a block in a box (which will generally stay there if left alone) and containing a flock of sheep within an unfenced field, which requires constant intervention by the agent. Different contexts will require different containers: A cage will contain an animal, but not a liquid. There are even such concepts as containing a thought within one's head, which doesn't even involve a palpable object. Lakoff and Johnson have proposed that abstract concepts are understood by ***metaphorical extension*** from basic schemas such as this one. You understand what it ***means*** to contain a thought within your head by relating it to the process of containing something within your hand, for example. In this view, people communicate and reason using such metaphors, which are derived from the physical structures of our bodies and the way we interact with the world. By giving us the ability to redescribe our world in more familiar terms, they make understanding it possible.

Could a concept like containment be represented with fluents? We believe so, although a great deal of work clearly remains to be done. The key is noticing that once again, physical schemas like all other concepts are defined via an agent's activities. Containing something could be thought of as being able to push it back into a specified state should it move out of that state. If Neo had simple fluents for "moving" and "pushing" (simple since they correspond to basic effector actions, and are easily described physically), it should be able to learn that when it can successfully and repeatedly keep an object within a state by pushing it (applying force to it), it has "contained" that object.

Like containment, all physical schemas share the property of being domain general, grounded in physical activities and physical relationships such as "move," "push," and "in," while providing a very abstract concept. Further examples are "support", "control", and "cause". The total number of such schemas may not be very large. It is our hypothesis that these concepts are so generally useful that they will emerge in any intelligent agent placed in any reasonable domain.

What Neo Knew and What It Learned

It is sometimes argued that real babies are born with "faculties" for physical and spatial reasoning (Carey & Spelke 1994), language (Pinker 1995), even reasoning about living things (Keil 1994). Nobody believes that much can be learned without constraints from perceptual systems, effectors, and prior mental structures, so the question is not "whether" but "how much." Although we eventually expect to show that Neo can acquire a rich conceptual structure from minimal beginnings, at present, we can show no such thing, because Neo's conceptual structure is quite poor—Neo learned hundreds of fluents and chains but these produced very few classes of objects and activities—and Neo's prior structure is not insignificant. Let us review what Neo was born knowing:

Streams and Tokens. Neo experiences its world through streams, and streams contain tokens such as **red** and **hungry**.

A Notion of Events. Neo's learning methods all focus on start ($\vdash$) and stop ($\dashv$) events. Static streams or fluents don't interest Neo's learning methods.

Methods to Learn Scopes and Base Fluents. Neo is born with a method to find correlations among start and stop events in streams. The resulting structures, called scopes, constrain the simplest fluents Neo learns. Base fluents are statistically significant instantiations of scopes with tokens. Neo is born with a method, based on contingency tables, for learning base fluents.

Methods to Learn Composite Fluents. The method for learning conjunctive fluents (and base fluents) is based on the idea that simultaneity is rare, so when it is observed, it probably has a single cause. The method for learning context fluents and chains is based on the idea that events sometimes follow others more often than expected by chance if they were unrelated, so when this is observed, the events are probably causally related. The learning methods for base, conjunctive and context fluents all rely on contingency table analysis, but the tables are set up slightly differently.

Now consider what Neo learned:

- It learned that most of the regularity in its environment takes place in 30 pairs of streams, less than 10% of the $(26 \times 25)/2 = 325$ pairs of streams that it might have focused on.

- It learned base fluents corresponding to the shape and color of every object in its environment.

- It learned the permanent locations of the green mobile (directly overhead) and the crib bars (to the extreme left and right of its field of view). No other objects have permanent location, because Neo can move them, but interestingly, it learned locations for objects that it let sit for a long time.
- It learned activities, such as grasping an object and mouthing it, or moving its arm and seeing its arm move.
- It almost learned conditions. For example, it learned a chain that includes **...((do-hand open) (hand open)) ((tactile-mouth skin) (mouth mouthing))**, but it has no way to learn that the first fluent is a condition for the second—that the hand ***must*** be open to be mouthed.
- It learned chains from which we, the authors, abstracted classes that make sense in Neo's environment, such as the class of objects that can be grasped and mouthed, and the class of activities that end in seeing the arm moving fast.

Keep in mind that Neo's actions are largely random: when it grabs an object it ***can*** mouth it, but it's just as likely to drop it, or move its head. Using the conceptual structure learned to guide further knowledge acquisition is the next step in our research program. The only structure in Neo's actions is provided by conditions (e.g., it cannot mouth an object it hasn't grasped, and it cannot mouth its hand unless the hand is open) and by a handful of simple behavioral dependencies built into the simulator (e.g., it sometimes grabs what it looks at, and it cries if it gets hungry). Keeping in mind also that Neo ran for only 30,000 time steps, it seems to us that it learned quite a lot.

Related Work

Attempts to build complete cognitive systems such as the one we are proposing have been rare; some examples follow. Andreae viewed his **PURR-PUSS** system as a general approach to building an intelligent agent (Andreae 1977). It resembles Neo in that it is based on few general design principles, primarily trying to predict its future sensory experiences and exploring those aspects of its environment that it cannot fully explain. Unfortunately, though, **PURR-PUSS** was limited by its somewhat rigid approach and the lack of available computing power at the time. Soar (Laird, Newell, & Rosenbloom 1987) is also viewed by its designers as an architecture for intelligence and a theory of cognition. However, Soar requires a lot of built-in structure, e.g. production rules and ultimate goal states, and places no value on learning through interaction. The learning it does, called ***chunking***, is limited to storing problem solving experiences for future use. Neo, on the other hand, attempts to build structures that will help it ***explain*** its environment. Brooks' Cog project has a lot in common with Neo, both in motivation and general design principles (Brooks & Stein 1993; Brooks 1996). He views cooccurrence of events as key to learning, and emphasizes the importance of embodiment for the emergence of human-like behavior and conceptual knowledge. We differ from Brooks, however, in that we do not feel that a physical embodiment—as opposed to simulation—is absolutely necessary.

The principle of interactivity in learning has been finding growing support in recent years even in AI circles (e.g. (Kuipers & Byun 1991; Mataric 1992)). Drescher has taken Piaget's principles of elaboration and accommodation to heart (Drescher 1991). His system elaborates "context–action–result" triples into ***schemas*** that resemble the fluent representation we use, although they are somewhat less general and not as directly grounded in sensory experience. The agent forms new concepts by creating "synthetic items" to denote unknown aspects of the world that are responsible for some regularity in the observed schemata.

There is a vast amount of work in the field of Artificial Life describing aspects of animal behavior and animal interaction within societies. Into this broad area falls Bates' work on artificial characters (Bates 1992; Reilly & Bates 1992). There are even some very realistic and fairly complete models of individual animals, e.g. fishes (Terzopoulos, Tu, & Grzeszczuk 1994). We do not place too much emphasis on modeling animal or human behavior, however. Instead, we are trying to develop a general mechanism for the acquisition of abstract conceptual knowledge, and show how such knowledge might be useful to the agent.

Conclusion

The goal of the Neo project is to build a virtual infant that learns many of the cognitive skills that we expect from a three-year old. Underlying these skills is a conceptual structure, an ontology, a way to "carve the world at its joints." This conceptual structure identifies classes, and supports judgments of similarity. Following Lakoff, Johnson, Mandler and others, our position is that concepts are based in activities. Neo's fluents represent objects, states, dependencies and activities. We believe that they can be extended to learn very abstract and general "physical schemas". We were able to identify classes by examining Neo's learned activities, thus providing the first evidence from this project that conceptual structure can be learned by interacting with the environment.

Acknowlegements

This research is supported by DARPA under contract No. N66001-96-C-8504. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements either expressed or implied, of the Defense Advanced Research Projects Agency, or the U.S. Government.

References

Agre, P. E. 1988. The dynamic structure of everyday life. MIT Artificial Intelligence Laboratory Report 1085.

Andreae, J. H. 1977. *Thinking With The Teachable Machine*. Academic Press.

Ballard, D. H. 1989. Reference frames for animate vision. In *IJCAI 89*, 1635–1641.

Bates, J. 1992. The nature of characters in interactive worlds and the Oz project. Technical Report CS-92-200, School of Computer Science, Carnegie Mellon University.

Brooks, R. A., and Stein, L. A. 1993. Building brains for bodies. MIT AI Lab Memo #1439.

Brooks, R. A. 1996. From earwigs to humans. To appear in Robotics and Autonomous Systems.

Carey, S., and Gelman, R. 1991. *The epigenesis of mind: Essays on biology and cognition*. Erlbaum.

Carey, S., and Spelke, E. 1994. Domain-specific knowledge and conceptual change. In Hirschfeld, L. A., and Gelman, S. A., eds., *Mapping the Mind*. Cambridge University Press.

Chapman, D. 1991. *Vision, Instruction and Action*. MIT Press.

Drescher, G. L. 1991. *Made-Up Minds*. MIT Press.

Ginsberg, H. P., and Opper, S. 1988. *Piaget's Theory of Intellectual Development*. Prentice Hall. 3rd Edition.

Johnson, M. 1987. *The Body in the Mind*. University of Chicago Press.

Keil, F. 1994. The birth and nurturance of concepts by domains. In Hirschfeld, L. A., and Gelman, S. A., eds., *Mapping the Mind*. Cambridge University Press.

Kuipers, B., and Byun, Y.-T. 1991. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Robotics and Autonomous Systems* 8:47–63.

Laird, J. E.; Newell, A.; and Rosenbloom, P. S. 1987. Soar: An architecture for general intelligence. *Artificial Intelligence* 33:1–64.

Lakoff, G., and Johnson, M. 1980. *Metaphors We Live By*. University of Chicago Press.

Lakoff, G. 1984. *Women, Fire, and Dangerous Things*. University of Chicago Press.

Mandler, J. M. 1988. How to build a baby: On the development of an accessible representational system. *Cognitive Development* 3:113–136.

Mandler, J. M. 1992. How to build a baby: II. Conceptual primitives. *Psychological Review* 99(4):587–604.

Mataric, M. J. 1992. Integration of representation into goal-driven behavior-based robots. *IEEE Journal of Robotics and Automation* 8(3):304–312.

Pinker, S. 1995. *The Language Instinct*. HarperCollins.

Reilly, W. S., and Bates, J. 1992. Building emotional agents. Technical Report CS-92-143, School of Computer Science, Carnegie Mellon University.

Terzopoulos, D.; Tu, X.; and Grzeszczuk, R. 1994. Artficial fishes with autonomous locomotion, perception, behavior, and learning in a simulated physical world. In Brooks, R. A., and Maes, P., eds., *Artificial Life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems*, 17–27. MIT Press.