CPS331 Lecture: Knowledge Representation; Discussion of Newell/Simon Article

Last revised July 25, 2016

Objectives:

1. To discuss the difference between digital and non-digital representations of knowledge.
2. To introduce semantic networks
3. To introduce inheritance based on isa hierarchies
4. To introduce frames
5. To introduce scripts
6. To introduce the problem of non-monotonic information
7. To discuss the Newell/Simon PSS article

Materials:

1. Projectable of corrupted phrase
2. Projectable of Figure 2.2 from Cawsey
3. Projectable of example isa hierarchy and Demonstration isa.pro
4. Projectable of Figure 2.4 in Cawsey
5. Projectable of Winston (2e) p. 267, 268 (top part + pattern), p. 268 (story) + p. 269 (figure + first summary), p. 269 (second story)
6. Projectable of example restaurant script (Luger 6e p. 242)
7. Series of projectables for the “ABC murder”
8. Copy of Newell/Simon article (in Boden) and study guide

I. Introduction

A. Give out Newell/Simon article reading guide (for third class)

B. One of the hallmarks of intelligence is knowledge. But finding a good way to represent knowledge is not easy, for a wide variety of reasons.

1. First, there are many different types of knowledge.

a) For starters, knowledge can be classified into two broad categories:

(1) Factual knowledge ("knowledge that") - e.g.

This course is numbered CPS331
Columbus arrived in the New World in 1492
The Red Sox won the World Series in 2004, 2007 and 2013
...

(2) Procedural knowledge ("knowledge of how to") - e.g.

Driving a car
Tying ones shoes
Studying for an exam
...

b) But, beyond these broad categories there's still a lot of variety. For example, factual knowledge includes not only individual facts, but facts about relationships, etc.

2. Moreover, knowledge can be either precise or imprecise

a) For each of us, our age is a very precise piece of information. Most or all of us can give the exact day of our birth. Even if we don't know it, though, there is a precise value.

b) But now, consider concepts like "young" and "old". If I ask you "are you young?", you might well ask me "what do you mean by young?" You are a lot younger than I am; but when compared to elementary school children you are not young at all. So a question like "are you young?" is often one to which there is no simple yes-no answer.

3. Again, knowledge can be incomplete and/or uncertain. For example, an AI system for doing medical diagnosis may need to take into account the patient's age. But what if the date of birth for a particular patient is unknown or there is some question about it? How do we represent certainty in such a way as to enable us to take it into account in reasoning when appropriate?

4. Moreover, knowledge changes over time. You are currently wearing some articles of clothing and sitting in a desk in this classroom. An hour from now, you will probably not be here, but you will probably be wearing the same clothes, and the desk will be still be here. How do we represent the relationship between you, your clothes, and your desk in such a way as to cope with the changes that occur when the class ends?

5. Again, a representation for knowledge has to deal with exceptional cases. All of us know, for example, that birds fly. But we also know that there are certain exceptions - e.g. penguins and ostriches. How do we represent both the general rule and the exceptions?

6. Finally, there is the matter of representing knowledge that agents possess about the knowledge or beliefs other agents - e.g.

    a) A particular statement may be true, but some agent may not know that it is true, or it may be false but the agent may believe it is true.

    b) In either case, reasoning about the agent's behavior requires the ability to represent the agent's knowledge and beliefs, which may differ in some ways from the true facts - since the agent's behavior will be based on what it knows and/or believes to be true.

C. Classically, two key areas of concern in classical AI have been knowledge representation and search. We will look at search later in the course, but for now we intend to spend a chunk of time on classical symbolic knowledge representation.

1. In AI, it is often the case that, where a good representation is available, the solution to a problem may be fairly easy.

Example: AI has had a great deal of success with games like chess, where there are fairly straightforward ways to represent the state of a game.

2. However, many important AI problems are “hung” on the problem of representation - e.g. representing commonsense knowledge is a major stumbling block for building a natural language systems that really "understands" its input.

3. Much of what we will discuss has arisen out of work in the subfield of natural language understanding, which is where the knowledge-representation problems become particularly acute.

4. Later in the course, we will look at approaches to representing knowledge that are totally non-intuitive - e.g. neural networks modeled after the structure of the brain.

5. In some domains, it may turn out that seeking a knowledge representation is even a stumbling block - e.g. one approach in robotics uses the slogan “the external world is its own best representation”. Again, we will look at these toward the end of the course.

D. A good symbolic knowledge representation should have the following qualities (here using terminology from Rich and Knight's intro to AI book)

1. It should be able to represent all of the knowledge needed for the kind(s) of problem it is used for. (Rich/Knight calls this representational adequacy.)

   This, of course, implies that different ways of representing knowledge might be appropriate for different problems.

2. It should facilitate inferring new knowledge from existing knowledge. Rich/Knight calls this inferential adequacy).

3. It should facilitate efficient means of accessing the specific knowledge that is relevant to a particular problem. Rich/Knight calls this inferential efficiency).

4. It should support addition of new knowledge to the database as the program is running. (Rich/Knight calls this acquisitional efficiency.)

5. It should allow knowledge to be readily converted between its internal form and a form readable by humans (e.g. some English-like notation.) (This was not explicitly mentioned by Rich/Knight - it might be called explanational efficiency)

E. Because knowledge representation is so important, much work has been done on it, and many schemes have been developed. These schemes typically take the form of a knowledge representation language and associated tools for maintaining a knowledge base. These tools can highly-sophisticated.

1. First, we will look at several schemes for representing factual knowledge which have largely been developed in conjunction with work on natural language. One of the key issues addressed by these schemes is the handling of <u>relationships.</u>

2. Then, we will look at formal logic (specifically the first order predicate calculus) - which is very useful in its own right, but can also serve as an infrastructure for other schemes. Predicate calculus is used not only in AI and other areas of Computer Science, but also in philosophy and mathematics - in fact, the main course where it is taught at Gordon is in the philosophy department.

3. Next, we will look at a programming language - PROLOG - which is based on the predicate calculus

4. Then we will look at a system for representing procedural knowledge - rules.

5. Finally, we will look at some ways of dealing with problems that arise from default reasoning.

6. Later in the course, we will look at ways of approaching issues like imprecise, incomplete, and uncertain knowledge, as well as the knowledge agents have about other agents.

F. However, before moving on to these topics, we want to look at a foundational issue - the matter of digital versus non-digital representations.

II. **Digital versus Non-Digital Information**

A. As humans, we seem to be able to work with two very different kinds of information: digital information and - for want of a better term - non-digital information.

1. We sometimes use the term "digital" to describe a particular kind of devices (e.g. "digital" versus "analog" TV). But the essence of the distinction is more fundamental than that.

A digital system is one in which information is represented by symbols drawn from a finite set ("alphabet") of possibilities.

a) Natural language is a paradigm example of a digital system.

(1) Everything that can possibly be written in English is constructed from the letters of the Roman alphabet (52 if you count case distinctions) plus a few punctuation marks and other special symbols and blank space.

(2) Sometimes, it is helpful to think of the fundamental units of written English as the groupings of letters known as words.

But, though there are thousands of words in English, the number is still finite.

b) For the most part, the meaning of a sentence in English is derived from the letters (and hence words) from which it is constructed. To be sure, we sometimes convey special information by the choice of style (boldface, italic, underlined), size, or sometimes even font. But, for the most part, the essential meaning of a sentence in English comes from the particular words of which it is constructed, together with the order in which they appear.

2. On the other hand, a non-digital system does not have, at its core, a fixed "alphabet". A painting, for example, is constructed of brush strokes, no two of which are alike, and though an artist may use a small set of raw paints the colors of the actual painting typically reflect some blending of these colors.

3. Other examples - classify these as fundamentally digital or non-digital, or a hybrid:

ASK

a) A position in a chess game

b) A position in a billiards game

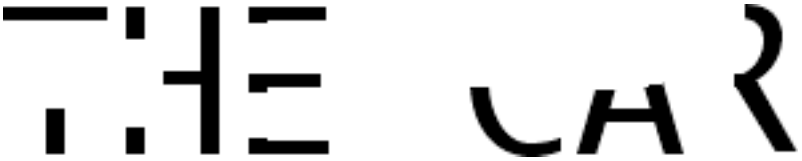
c) Taste/smell

d) Music

B. An important difference between digital and non-digital systems is robustness.

1. Digital information can be recovered even in the face of significant deterioration.

Example: PROJECT



What phrase is this?

How did you know?

ASK

2. On the other hand, when non-digital information deteriorates, something is permanently lost.

   Example: the fading of a painting

3. John Haugeland put it this way: “Consider the difference between accidentally messing up a chess game and a billiards game. Chess players with good memories could reconstruct the position perfectly (basically because displacing the pieces by a fraction of an inch wouldn’t matter). A billiards position, by contrast, can never be reconstructed perfectly, even with photographic records and finest instruments; a friendly game might be restored well enough, but jostling a tournament game would be a disaster.” (*Artificial Intelligence: The Very Idea* 57)

C. A distinction related to this is <u>failure mode</u>.

1. Non-digital information can retain some content even when much is lost. For example, given a badly faded painting it may still be possible to discern the identify of the original subject.

2. On the other hand, when a digital system fails, it typically fails catastrophically - up to a point, degraded information can be recovered perfectly, but beyond that point, typically nothing can be recovered at all. (Example: most of you have had this happen with a DVD!)

D. As the name implies, digital computers are fundamentally digital devices.

1. Internally, everything is represented by numeric codes, ultimately based on an “alphabet” of just two symbols: 0 and 1.

2. Of course, digital computers can represent non-digital information by a process of “digitizing”.

a) For example, a picture can be represented by breaking it up into discrete dots (typically 72 to 1200 dpi) called pixels. The color of each pixel can be represented by three numbers in the range 0..255, representing its redness, blueness, and greenness.

b) Sound can be represented by breaking it up into samples (e.g. at an interval of 38K or 44KHz) and then representing the intensity at each instant by a number (e.g. an integer in the range -2048 .. 2047).

c) Digitizing works because human perceptual faculties have a limited resolution.

(1) If the number of pixels used to represent an image is large enough, the human eye is unable to resolve the individual pixels and the image appears to be continuous.

(2) Representing the redness, blueness, or greenness of a pixel by one of 256 possible discrete values works because the human eye is not sensitive to smaller distinctions than this.

(3) There is an important theorem in engineering called the Nyquist-Shannon Sampling Theorem which says that a sound can be perfectly reproduced from samples taken at at least twice the rate of the highest frequency present in it. Since most human ears are unable to hear sounds above

20KHz, sampling at 40 KHz would result in a sample that is indistinguishable to the human ear from the original sound (though not necessarily to a dog!) [For technical reasons, CDs use a sampling rate of 44.1 KHz]

3. However, the digitized representation of information is not directly meaningful to humans.

   Example: project raw digitized and jpg versions of “parrots” image.

E. Historically, much of the work in AI has been based on digital representations of knowledge - though the term that is often used is “symbol system”.

1. The systems we will look at over the course of the next few weeks are all symbol systems.

2. Later in the course, we will look at alternate, non-symbolic representations for information, often inspired by biology

III.Semantic Networks

A. A semantic network is a way of representing objects and relationships between objects. A semantic network consists of nodes that represent objects, sets of objects, and properties of objects, and links that represent relationships between objects or sets or properties. It is called a semantic network because the nodes and links are intended to stand for things in “the real world”.

   Example: Figure 2.2 from Cawsey PROJECT

1. The ovals are nodes

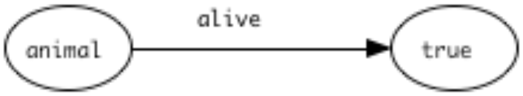
   a) Two of the nodes represent specific objects: clyde, nellie

b) Other nodes represent sets of objects: animal, reptile, mammal, head, elephant, apples

c) Still other nodes represent properties: gray, large

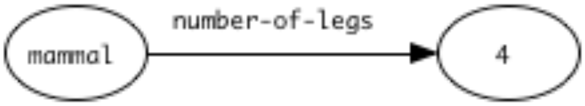
   (1) In this example, some of the properties are adjectives and some are nouns (an elephant is gray - adjective; and elephant has a head - noun)

   (2) It is also possible for properties to be simple true-false values - e.g. we might represent "an animal is alive" by

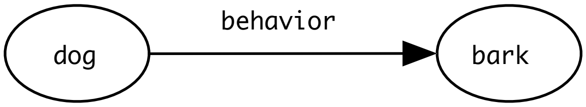


Note: true and false are often abbreviated T and F

   (3) It is also possible for properties to be numerals - e.g. we might represent "a mammal has four legs" by



   (4) Again, it is also possible for properties to be verbs - e.g. we might represent "a dog barks" by



d) No attempt is made to distinguish between different kinds of node in this example, but the distinction is an important one in practice.

   (1) The distinction between an object and set is captured in English by the use of the indefinite article or plurals.

   We say "clyde is <u>an</u> elephant"

We say “an elephant is a mammal”
or “elephants are mammals”

(But note that the plural is only used when both sides of the relationship are sets - we don’t say something like “clyde is elephants”!

(2) The distinction between an object or set on the one hand and a property on the other is captured in English by the use of the definite article with the name of the property.

We say “the size of an elephant is large”

We might say “the number of legs of a mammal is 4”

We might say “the behavior of a dog is to bark (or barking)”

2. The arrows are links. Each has a label that specifies its meaning.

a) Some of the links represent the relationship between a specific object and the set of which it is a part. These are labelled “instance”

b) Other links represent the relationship between a set of objects and a superset. These are labelled “subclass”

c) Still others represent other sorts of relationship

d) The arrows specify the direction in which the label should be read - e.g. “the size of an elephant is large”, rather than “the size of a large is elephant”

(1) In English, we typically represent both “instance” and “subclass” by the use of words like “is” or “are”.

We say “clyde is an elephant”

We say “an elephant is a mammal”
or “elephants are mammals”

(2) As we already noted, in English we typically represent a simple property relationship by “the” with the property name - e.g. “the size of an elephant is large”

(3) The “has-part” relationship in this example, is a bit different, because it relates two sets, not an object/set and a property.

(a) In this case, animal and head both refer to sets - in English we say “an animal” or “animals”, but don’t use the singular without the article. The same is true for head.

(b) The link says that every object that belongs to the set of animals has an object that belongs to the set of heads.

B. We can use a semantic network to directly answer certain kinds of questions. For example:

1. What is the colour of an elephant?
2. To what category does an elephant belong?
3. Does an animal have a head?

C. There is no universally-accepted notation for semantic networks. Different writers use widely-varying terminology.

1. In large measure, this is because a semantic network is just a pictorial way of representing something that must ultimately be represented by computer code, as we shall see later.

2. Of course, it is important, when using a semantic network, to be sure that the semantics are clear.

   For example, we understood the the “has-part” link between animal and head to mean that each animal has a different head, rather than all animals share a single head!

IV. Isa Hierarchies and Inheritance

A. One very significant usage of semantic networks is to support doing inferences based on inheritance.

1. For example, using the figure in the book, suppose we wanted to know the colour of clyde. Though there is no "colour" link attached to clyde, there is one attached to elephant and clyde is a member of the set of elephants and therefore has the properties elephants have.

   We call the colour property in this case an <u>inherited</u> property.

2. We can also inherit set membership. For example, from the fact that clyde belongs to the set of elephants, and elephants are a subset of animals, we can infer that clyde is an animal.

3. We can also inherit "has-part". For example, from the fact that clyde belongs to the set of animals, we can infer that clyde has a head.

4. What are some other inferences we can draw from the network?

   ASK

   (You will notice that we are discussing inheritance in the context of semantic networks, rather than frames. As the book noted, the two are really equivalent notations, but I find inheritance is more easily understood using a graphical representation)

B. In doing inferences, the "instance" and "subclass" links play a special role, since we can inherit properties by following these links. Since the verb we use in English to describe these links is "is", and the second node is always a set, for which we use the indefinite article in English, these are sometimes called by the generic name "isa" - i.e. we say "clyde isa elephant" or "elephant isa mammal".

1. Many writers actually use the term "isa" as a link label.

   a) However, writers often use it for just one of the two kinds of relationships (instance or subclass), with a different word used for the other kind of label.

   b) To make matters confusing, writers do this both ways: e.g. "isa" for "instance" with "subset" or "subclass"; or "instance" with "isa" for subclass!

   c) To avoid confusion, we will stick for now with two distinct labels.

2. A structure like this is then often called an "isa hierarchy".

C. Demo: a simple example of an isa hierarchy

PROJECT: isa hierarchy

1. What can we infer concerning snoopy?

   ASK

   DEMO isa.pro: `alli(snoopy).`

2. How about sandy? (A dog I had years ago who also was a beagle)?

   DEMO: `alli(sandy).`

D. A couple of interesting questions arise with isa hierarchies when property information is recorded for subclasses.

1. We have already noted that an instance inherits property values recorded for sets of which it is a member. Actually, we need to regard these as <u>default</u> values for an instance if no more specific information is available at a "lower" level.

Example: In the hierarchy shown, the set "animal" has the value false for the "speak" property. However, certain of the cartoon animals have the property "true" for this property (mickey, minnie, rocky, and opus - though not snoopy, garfield, nermal, or tweety).

a) It is clear how to handle inference for the speak property in the default case.

   DEMO: `alli(snoopy).`

b) But what should we do in the case of the cartoon animals? It looks like we can infer both values - but this doesn't make sense. Instead, what we want to do is define inheritance to preclude inheriting a default value if a more specific value is available lower down the hierarchy.

   DEMO: `alli(mickey).`

c) This is possible not only for a specific individual, but also for a class of individuals.

   For example, we might record that the number of legs for a mammal is 4, and for a human is 2. We might also record that humans are mammals. However, when asked the number of legs for a particular human, we should give the answer 2, not 4.

2. Another issue that arises when default values for sets are recorded is how to handle individuals or subsets who should not inherit the default value.

   Example: though cats, in general, purr, garfield does not.

   Example: though birds, in general, fly, penguins do not.

a) A situation like this might be handled by recording a negated value for the property - i.e. the fact that some property does not hold for a particular individual or subclass.

Example: in this diagram the property ¬ behavior(purr) is recorded for garfield, and ¬ behavior(fly) is recorded for penguin. (¬ is the formal logic representation for not).

As a result, the inheritance of default values for garfield or penguins is blocked.

DEMO: `alli(garfield)`

DEMO: alli(opus).

b) Of course, negated properties should only be recorded when they are needed to block unwanted inheritance. Without this restriction, one could go nuts recording negated values! (Try listing all the things that are not true of dogs, let's say!)

This builds on a notion common in AI known as negation as failure or the Closed World Assumption (often abbreviated as CWA): something is assumed to not be true unless it is explicitly recorded that it is true.

E. One thing not shown in the example is the possibility of multiple inheritance - where an individual or subset inherits from two different sets.

1. Example: We might introduce the set "pet". Then some specific individuals would belong to this set, but also to the set denoting their species - e.g.

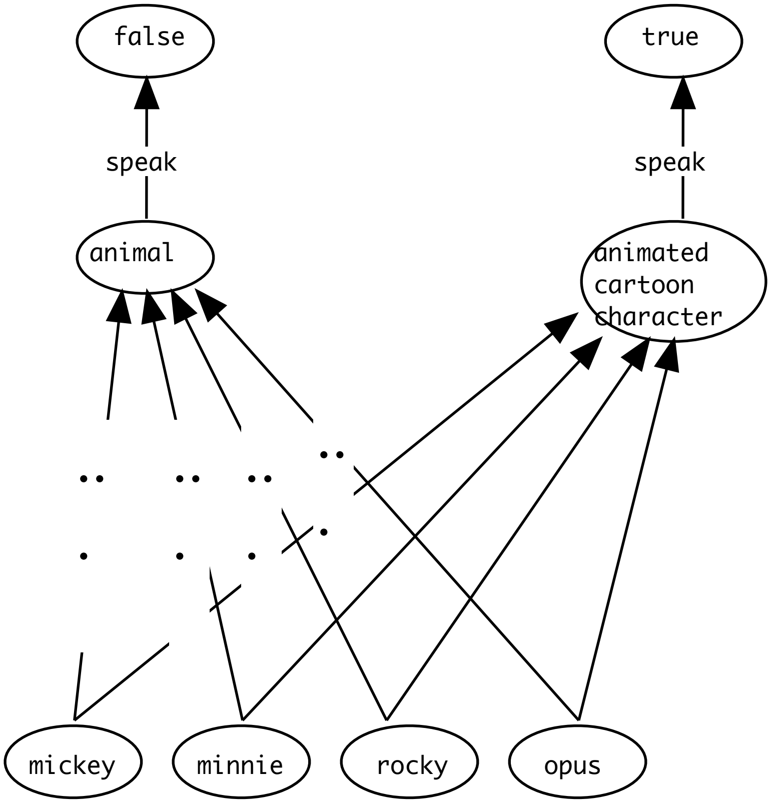
   sandy and snoopy would have "instance" links to both beagle and pet.
   garfield and nermal would have "instance" links to both cat and pet.
   tweety would have "instance" links to both canary and pet.

2. We might also do this with sets - e.g. conceivably, instead of linking the individuals snoopy, sandy, garfield and nermal to pet, we could just link "dog" and "cat" to pet (assuming we have no

strays!) (Of course, even if we did this tweety would still need to link to both pet and canary because there are such things as wild canaries.)

3. One tricky issue that arises is how do we handle a situation in which an individual inherits conflicting values?

   Example: Suppose we introduce the set "animated cartoon character". Then, for a subset of our creatures, we might get something like this:



   What shall we report for the "speak" property of any of these creatures?

   Although the problem is tricky in general, in this case we might resolve it by using a rule of "inheriting from the nearest ancestor".

V. **Frames**

A. One issue that arises with the use of semantic networks is that the basic structure is, in some sense, <u>too</u> flexible. In order to use a semantic network in reasoning, it is desirable to have more specificity about the types of links allowed. This has become particularly important in natural language work, as we shall see later in the course, but we will look at one approach to handling this now.

B. One of the things intelligent beings like us do with the entities in our world is we organize them into categories. For each category of entity, we tend to want to know certain things.

1. Example: someone tells us about a car he/she just purchased. What things might we ask about it?

   ASK

   Some possibilities: Manufacturer, model, year, and color.

2. Example: we learn that a friend of ours has just had a baby. What things might we want to know?

   ASK

   Name, sex, birth weight, birth height

C. This leads to notation use in AI that has come to be called “frames of context” or more simply <u>frames</u>.

1. A frame is a structure describing some instance or set. Different pieces of information are represented as values of the various <u>slots</u> in the frame.

   Example: The book shows how the a portion of a semantic net similar Figure 2.1 might be represented using frames.

   PROJECT: Cawsey Figure 2.1 again

   Cawsey: Figure 2.4.

a) Two of the frames (Clyde and Nellie) represent individuals. The other two (Mammal and Elephant) represent sets. What is the clue that distinguishes the two kinds of frame?

   ASK

   Frames for individuals have an “instance” slot

b) Note how some of slot values for sets are default values (indicated by a star in the figure) while others are absolute (e.g. warm-blooded for mammal or has-trunk for elephant).

c) Note how some of the slot values actually refer to other frames - thus maintaining the isa hierarchy (these are called “subclass” or “instance” in the example - though maybe “subclass-of” or “instance-of” would be clearer.)

d) Though not shown in this example, it may also be the case that other slots contain references to other frames, rather than simple values - e.g. the fact that a mammal has a head (from the original semantic net) might be represented by a relationship between the Elephant frame and the Head frame.

   ....
   has_part   Head
   ...

   Another relationship in the example could be represented this way as well (and likely would be, to avoid proliferating slot names)

   ASK

   has_trunk would more likely be has_part Trunk

2. One of the major benefits of using frames is that one can develop a <u>schema</u> for each type of frame, with specific named slots that need to be filled in for an instance of the scheme.

3. Historically, frames in AI developed in parallel with OO in software engineering. Thus, if you are familiar with OO in CS, it may have struck you that an AI frame has many similarities to an object in OO - though AI sometimes uses a more flexible representation (e.g. a network of nodes) rather than the more rigid object structure of most OO languages.

   (In this lecture, we will use the traditional AI terminology, though many concepts also appear in OO under different names.)

D. One thing that is often characteristic of knowledge in AI programs is that it is partial. For example, for some particular auto, we may know the make, model, and color, but not the year.

1. In this case, the :year slot might contain an indication that the value is unknown.

2. When a value is put in a slot, we say that the slot is <u>filled in</u>. This may occur at the time the frame is created, or later as further information about the object becomes known.

3. Several mechanisms are commonly used in AI programs to deal with incomplete information:

   a) We have already seen the notion of default values for slots - values to be assumed to hold for a particular instance if the slot has not been filled in. This may be handled by a constructor or “when created” procedure, since the default value is written into the slot when the frame is created unless some other value is known.

   b) “If-needed” or “when read” procedures for slots - an if-needed procedure is a procedure to be invoked to attempt to infer the value for a slot (based on information in other slots) if the information is needed at some point. (Of course, such a procedure might fail to produce a value, too, if the information it needs is missing.)

c) “If-added” or “when-written” procedures for slots - an if-added procedure is a procedure to be invoked when a formerly empty slot is filled in. This procedure may fill in other slots based on the new knowledge.

d) Such procedures are sometimes referred to as “daemons” [where the term in this context does not refer to fallen angels, but rather picks up a Unixism referring to background programs that are not run under the direct control of a user. Note that the spelling is daemon].

e) Example: suppose we were to create a schema for a student-frame, describing a college student.

   (1) What slots might be present in such a frame?

   ASK

   (2) One slot in this frame might be the student's majors (:majors). We might attach the default value "Undecided" to this slot.

   (3) Another slot in this frame might be his/her grade-point average (:gpa). This slot would initially be vacant when a student frame is created, since students don't have a gpa until they've taken at least one course. Further, the value in this slot would change at the end of each semester in which the student takes at least one course.

      (a) Since the gpa calculation involves a fair amount of computation, we might choose to not do it regularly. Instead, we might associate with the gpa slot an if-needed procedure that calculates the gpa when someone tries to use it. (Presumably, this procedure would then fill in the slot with the calculated value so that recomputation is not necessary the next time.) Of course, this procedure would have to return something like null if the course history information is lacking.)

(b) Since any change to the student's course history would change his/her gpa, we might associate with the courses-taken slot an if-added procedure that invalidates the gpa slot (sets it back to unknown) any time a change is made to the courses-taken slot. This will force a fresh computation of gpa the next time it is needed. (Or, at higher computational cost, we might have the if-added procedure re-compute the gpa and put the new value in the gpa slot, though this might involve a lot more computation if grades are entered one at a time.)

f) Note that default values, if-needed procedures, and if-added procedures for slots are really properties of the schema, rather than of the actual frame itself.

(1) Some writers call these <u>attachments</u> to the slots to distinguish them from the actual slot values.

(2) Another term sometimes used is <u>facets</u>. A given slot may have a value facet, a default facet, an if-needed facet etc.

E. An example of how frames can be used in problem-solving.

1. In his intro AI text, Patrick Henry Winston discussed the use of frames to model the structure of stereo-typed news stories, such as disaster stories. This method has been used in actual programs.

2. In this case, the frame structure, together with if-needed procedures or each slot, provides a mechanism for extracting key features from a story.

PROJECT: Winston p. 267, 268 (top part + pattern)

3. This mechanism could digest a news story and produce a summary like the following:

   PROJECT: p. 268 (story) + p. 269 (figure + first summary)

4. Of course, it could also produce some strange results:

   PROJECT: p. 269 (second story)

VI. **Scripts**

A. Historically, a knowledge representation scheme closely related to frames was called scripts. Scripts were frame-like structures used when there is a time-order relationship between the slots.

   Though scripts, per se, are an old idea not currently used, they underlie some more modern approaches and also were the stimulus for the famous “Chinese Room” argument mentioned in chapter 1 of your text - and presented by Searle in an article we will read later in the course. So, we’ll look at them briefly.

   1. Example: Consider the following story (from Rich/Knight):

      "John went out to a restaurant last night. He ordered steak. When he paid for it, he noticed that he was running out of money. He hurried home since it had started to rain."

      Based on the above story, did John eat dinner last night?

   2. Of course, the story does not explicitly say that John actually ate the dinner. The fact that we routinely answer yes to such a question implies that we are using a structure of knowledge with some default information that fills in for things not explicitly stated. Though the story does not say that John actually ate dinner, it does say that he went to a restaurant, ordered steak, and paid for it. Anyone hearing the story would assume that he ate the steak.

3. If you think about typical human story-telling, you realize that we seldom fill in all the details of a story. Rather, we tell some of the details, relying on our listeners to fill in the rest.

   a) For example, in the story we just looked at, it is not necessary to explicitly say that John ate the steak; and, in fact, it would sound a bit strange if we said so. To see this, consider the following variant of the story:

      "John went out to a restaurant last night. He ordered steak. When the waiter brought it to the table, John ate it. When he paid for it, he noticed that he was running out of money. Then he hurried home since it had started to rain."

   b) The only time we would probably incorporate the act of eating into the story would be if there was something unusual about the way he ate it. Consider the following variant, which does sound natural:

      "John went out to a restaurant last night. He ordered steak. When the waiter brought it to the table, John bolted it down. When he paid for it, he noticed that he was running out of money. Then he hurried home since it had started to rain."savors a steak.

B. One approach to dealing with familiar patterns like going out to a restaurant was to make use of structures called SCRIPTS that described routine event. Those who worked with scripts generally assumed that something similar is part of the way we understand stories.

   1. These scripts contained various slots pertaining to the sequence of actions that normally occur in such an event.

      PROJECT Example Script

a) Discuss use of conceptual dependency theory primitives

b) Discuss provision for alternatives

2. It was assumed that when we listen to a story, something early in the story triggers a mental script that we use to help us understand what we are hearing. For example, in the story we just considered the opening sentence "John went out to a restaurant last night" would trigger the restaurant script. In like manner, a natural language understanding program could use keywords to trigger appropriate scripts.

3. Once a script has been selected, a program using a script could maintain a pointer to the script which it advances as different items are encountered.

   a) As appropriate, slots can be filled in with information provided in the narrative. For example, the narrative may provide details as to where the customer sat, what he ordered, etc.

   b) Of course, typically many items in the script will not be mentioned in a given narrative. Thus, the pointer to the script will advance by jumps from one item explicitly mentioned to the other. Items skipped over may be assumed to have occurred by default.

   Example: in our first story, the pointer would skip over most of Entering to Ordering and then would skip over Eating to go directly to Exiting. But we would assume by default that Eating occurred.

4. Though scripts are an old idea not currently used, as I noted earlier they did serve as a stimulus for a classic article by Searle we will read shortly.

VII. **Dealing with Non-Monotonic Information**

A. We now turn to a separate, totally different problem - the problem of dealing with information that is <u>non-monotonic.</u>

1. Traditional logic systems are MONOTONIC. That is, the sum total of what we know always never decreases; once a piece of knowledge is inferred, it always remains true.

2. Real world problems can call for NON-MONOTONIC systems. In such systems, some or all of the conclusions reached are DEFEASIBLE - that is, they are subject to later being revoked.

3. One of the major reasons why non-monotonic logic is needed is because of the use of DEFAULT REASONING.

a) Example: Suppose I tell someone that tweety is a bird, and then I ask if tweety can fly. What would the person say?

ASK

b) However, if I now tell the person that tweety is a penguin, the answer would have to change.

(1) In essence, what was used was a rule like the following:

"If X is a bird, then assume X can fly, unless you have a reason to think otherwise."

(2) The acquisition of a new piece of knowledge - that tweety is a penguin, invalidated a former inference - that tweety can fly. That is, the inference "tweety can fly" is non-monotonic.

4. Non-monotonic knowledge arises any time we do reasoning that involves logical negation (not). This because we typically regard something we don't know to be true as false - an assumption we mentioned earlier which is known technically as the Closed World

Assumption (CWA) If we make such an assumption, the conclusion that “not something” holds (and any inferences based on it) based on absence of knowledge is falsified if we learn that “something” is true.

B. Default reasoning plays a major role in human intelligence. Rather than remembering long lists of facts about every individual entity we know about, we remember information about CLASSES of entities, which information is inherited by the members of the class by default - in the absence of information to the contrary. However, we must always be prepared to deal with entities that form an exception to the rule for the class to which they belong.

C. Non-monotonic reasoning is still very much an open research problem, and several different approaches have been proposed.

D. Just to show how this problem might be approached, we will briefly consider a very simple example using a simple approach: a justification-based truth maintenance system (JTMS). (The intention here is not to learn how the problem can be solved, but simply to enhance understanding of the difficulty of the problem!)

1. In a JTMS, our database consists of a collection of nodes, each of which is in one of two states: IN or OUT.

a) An IN node represents something that is currently believed to be true - based either on

(1) External knowledge (corresponding to a fact in traditional monotonic logic).

(2) An inference based on the value of other nodes

b) An OUT node represents something that is currently believed to be false - based either on

(1) Explicit knowledge that it is not true.

(2) A node about which we have no knowledge that allows us to conclude that it is IN - and hence we assume it to be OUT. (The closed world assumption)

c) Of course, a node can change states if we either acquire explicit knowledge about it, or the state of some other node changes, leading to a changed inference about it.

2. Associated with each node is a list of justifications - i.e. bases on which the node can be believed to be true. Each justification, in turn, consists of a list of other nodes that must be IN, and a list of nodes that must be OUT.

   Example: Consider the following murder mystery (adapted from Quine and Ullian, 1978 - quoted by Rich/Knight)

   “Let Abbot, Babbit, and Cabot be suspects in a murder case, because they are beneficiaries of the victim’s will. Abbot has an alibi, in the register of a respectable hotel in Albany. Babbit also has an alibi, for his brother-in-law testified that Babbit was visiting him in Brooklyn at the time. Cabot pleads alibi too, claiming to have been watching a ski meet in the Catskills, but we have only his word for that

   a) We can represent some general rules for reasoning about murder cases like this:

      possible(X) in: suspect(X), out: has-alibi(X)

      suspect(X) in: beneficiary(X)

      has-alibi(X) in: registered-far-away(X)

      has-alibi(X) in: seen-far-away-by(X, Y), out: lying(Y)

      [We don’t regard a person’s claim to be doing something else as an alibi unless the claim is otherwise confirmed]

   b) If we make the closed-world assumption, we can represent our initial body of facts as follows, where + in the justification for a node means that it depends on the node being in, and - means it depends on the node being out.

PROJECT

(The arcs connecting pairs of edges indicate that these are anded - i.e. both conditions must hold)

3. When a node changes state, the nodes which have it on their justification lists are checked, and may also have their states changed (which in turn may change other nodes ...)

   a) If we now learn that the register in the hotel in Albany was forged, we might change the registered in albany node for abbot to OUT, which in turn would change the has-alibi node for abbot to OUT, which in turn would change the possible node for abbot to IN.

   PROJECT

   b) Again, if we learn that cabot was not actually a beneficiary of the victim's will, that would change the beneficiary node for cabot to OUT, which in turn would change the possible node for cabot to OUT.

   c) What would happen if we added the following to our general rules?

   possibly-lying(X)   in: related-to-suspect(X)

   ASK

   PROJECT

4. Note, however, that a node may have more than one justification associated with it, so that even if one justification is invalidated another may still hold.

a) Going back to our original situation, suppose we also had another witness who saw babbit in brooklyn. Then our situation would look like this:

PROJECT

The absence of an arc indicates that we have two pairs of anded conditions that are <u>orred</u> - i.e. the node is considered justified if both nodes in either pair is satisfied

b) Now, even if we discounted the brother-in-law’s testimony because of his relationship to babbit, we would still not regard babbit as the possible murderer.

PROJECT

VIII. **Discussion of Newell/Simon Article**

A. This article discusses a hypothesis that is foundational to the symbolic approach to AI.

1. As such, it is controversial in the field.

2. In particular, some researchers have explicitly rejected this hypothesis. (In fact, one of the articles we will read later in the course does so.)

B. What is the occasion for this presentation? (When was it given)?

10th ACM Turing Award Lecture - 1975

C. Newell and Simon claim that each science has a central hypothesis. Examples?

"The cell doctrine in biology"

"Plate tectonics in geology"

"The germ theory of disease"

"The doctrine of atomism"

D. What do they say is the central hypothesis for Computer Science?

The PSSH

1. What is this hypothesis?

   "A physical symbol system has the necessary and sufficient means for general intelligent action."

2. What do they mean by "a symbol"?

3. What do they mean by a physical symbol system?

   "A physical symbol system consists of a set of entities, called symbols, which are physical patterns that can occur as components of another type of entity called an expression (or symbol structure.) Thus, a symbol structure is composed of a number of instances (or tokens) of symbols related in some physical way (such as one token being next to another.) At any instant of time the system will contain a collection of these symbol structures. Besides these structures, the system also contains a collection of processes that operate on expressions to produce other expressions; processes of creation, modification, reproduction and destruction. A physical symbol system is a machine that produces through time an evolving collection of symbol structures. Such a system exists in a world of objects wider than just these symbolic expressions themselves."

4. They also use the terms "designation" and "interpretation" concerning expressions (symbol structures). What do they mean by these terms?

"Designation. An expression designates an object if, given the expression, the system can either affect the object itself or behave in ways dependent on the object."

"Interpretation. The system can interpret an expression if the expression designates a process and if, given the expression, the system can carry out the process."

(Note that these are operational definitions)

E. Newell and Simon claim that a PSS has the necessary and sufficient means for general intelligence.

1. What do they mean by "necessary"?

   "By 'necessary', we mean that any system that exhibits general intelligence will prove upon analysis to be a physical symbol system."

   (Hence you and I are, ultimately, manifestations of a PSS)

2. What do they mean by "sufficient"?

   "By 'sufficient', we mean that any physical symbol system of sufficient size can be organized further to exhibit general intelligence."

3. What do they mean by "general intelligent action"?

   "By 'general intelligent action' we wish to indicate the same scope of intelligence as we see in human action."

F. How was the PSSH developed?

G. What evidence do they offer to support the hypothesis

1. Positive empirical evidence

   a) The existence of hundreds of physical symbol systems exhibiting some degree of intelligent behavior, with continual improvement in performance with time. They write: "The basic paradigm for the initial testing of the germ theory of disease was: identify a disease; then look for the germ. An analogous paradigm has inspired much of the research in artificial intelligence: identify a task domain calling for intelligence; then construct a program for a digital computer that can handle tasks in that domain ... "

   b) The modeling of human symbolic behavior. They write "the search for explanations of man's intelligent behavior in terms of symbol systems has had a large measure of success over the past twenty years, to the point where information processing theory is the leading contemporary point of view in cognitive psychology."

      (Note: recall that this talk was given in 1975. How would this claim hold up today?)

2. Negative evidence

   As negative evidence, they cite "the absence of specific competing hypotheses as to how intelligent activity might be accomplished - whether by man or machine." In particular, they claim that competing psychological theories are "sufficiently vague so that it is not terribly difficult to give them information processing interpretations, and thereby assimilate them to the symbol system hypothesis." [p. 120]

H. Your thoughts

1. Do you find the evidence convincing? (Why or why not?)

   ASK

2. What do you think about the “necessary” aspect of the PSS?

   ASK

3. What do you think about the “sufficient” aspect?

   ASK

I. Newell and Simon also offer a second hypothesis in this talk

1. What is it called

   the Heuristic Search Hypothesis.

2. What does it claim?

   "The solutions to problems are represented as symbol structures. A physical symbol system exercises its intelligence in problem solving by search - that is, by generating and progressively modifying symbol structures until it produces a solution structure"

   [Note: their later discussion emphasizes that they mean informed, heuristic search, or even strong search that goes straight to the answer without ever making a wrong turn.]

3. Newell and Simon regard this latter hypothesis as a further characterization of the type of physical symbol system likely to exhibit intelligent behavior.