Questioning Assumptions about Search

Using memory to avoid search: book moves

Costs of doing search: sorting the queue

1. Initialize the list of states to be considered with a node that contains the initial state of the problem.

2. Repeat the following:

   a. If the list is empty, then announce failure.

   b. Choose the first node in the list.

   c. If the node chosen refers to a goal state, then return the corresponding sequence of actions.

   d. Expand the node.
      Add all of its successors to the list,
      ordered by the heuristic function.

Time spent keeping the queue in order might be better spent considering more nodes.

Knowledge Representation

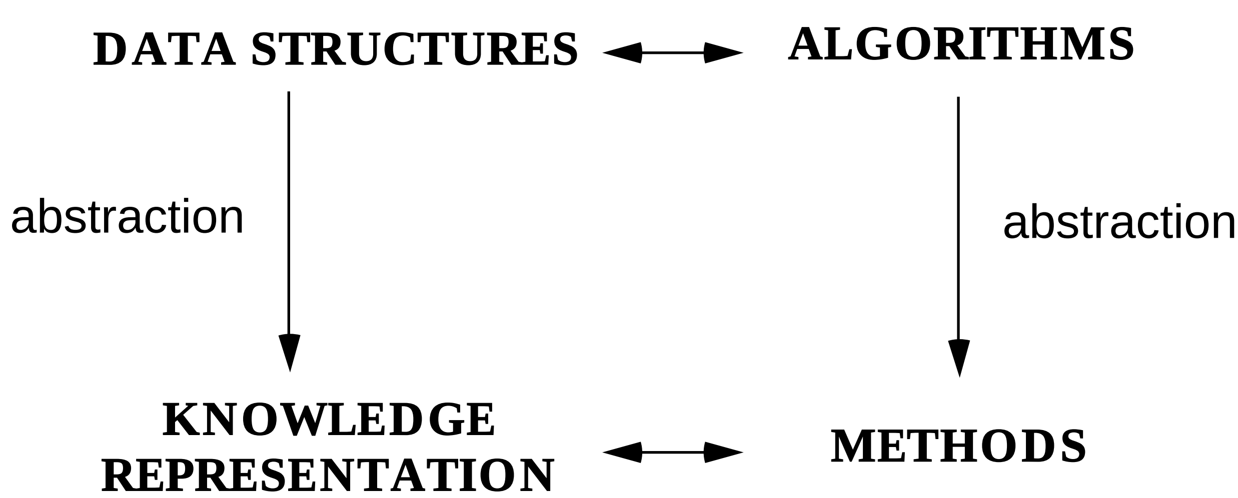
Sometimes, we know something about the world.

Why should we search for an answer?

Search is a naive way to reach a goal. In most situations, though, we know something about our domain and task. We would like to think about what we are doing, so that we can do it better.

This requires some way to...

- write knowledge in a form usable by a program
- use knowledge to solve the problem



Representing Knowledge in a Program

One truth from human experience:

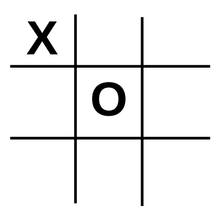
Described appropriately, a problem is almost solved.

So, how can we describe the world to our programs?

- natural language, such as English

  “Well, see, I have, like, an X in the upper corner and he has, you know, like an O in the middle.”

- iconic representation

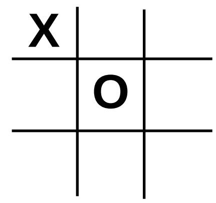


- feature-based representation

  [X | — | — | — | O | — | — | — | —]

An Exercise in Writing Knowledge

Assume that we use an iconic representation of states in the world.



How would you represent these pieces of information?

- "Two Xs in a row is good for me."
- "My opponent is Bobby Fischer."
- "Bobby always plays in the side squares if he can."

Then assume a feature-based representation, with an array of features.

[X | — | — | — | O | — | — | — | —]

Now how would you represent the same information?

Representing Knowledge in a Program

Natural language is powerful enough to represent any knowledge our program might need, but it is wordy and ambiguous.

Iconic and feature-based representation can be concise and precise, but they are not generally powerful enough to represent everything a program needs to know.

For example:

- “Two Xs in a row is good for me.”
- “My opponent is Bobby Fischer.”
- “Bobby always plays in the side squares if he can.”
- “Bobby will not eat professors.”
- “Bobby is alone in the boat or on the shore.”

The Components of a Knowledge Representation

A knowledge representation (KR) is a description written using a notation.

A KR consists of four parts:

• vocabulary	terms
• syntax	legal combinations of terms
• manipulation rules	ways to use sentences in the process of reasoning
• semantics	what sentences *mean*

For the sort of "network" shown for the Missionaries and Cannibals puzzle...

• vocabulary:	labeled nodes and links
• syntax:	node [- link - node]
• manipulation rules:	traversing the network
• semantics:	each nodes refers to a possible state in the world...

More on Knowledge Representations

What does this sentence mean?

Eugene is a professor.

What can sentences in a language mean?

Anything—depending on the meanings of the terms and the combinations.

"Eugene"	== the country we call Canada
"is"	== includes
"professor"	== the region we call Alberta

The Semantics of a Representation

How can we know what a sentence means?

- equivalence semantics

  ... by relating the terms to something we already know

  - example: “Eugene is a professor.”
  - example: using meaningful variable names

- descriptive semantics

  ... describing what each sentence means
  (this requires another language we both understand)

  - example: an English description of a program

- procedural semantics

  ... by looking at what an agent **does** with the sentence

  - example: Run the program. Interpret its behavior.

.

Agents that Use Knowledge

We would like to give our programs a KR scheme that is:

- computable
- complete
- concise
- fast to process
- and *transparent*

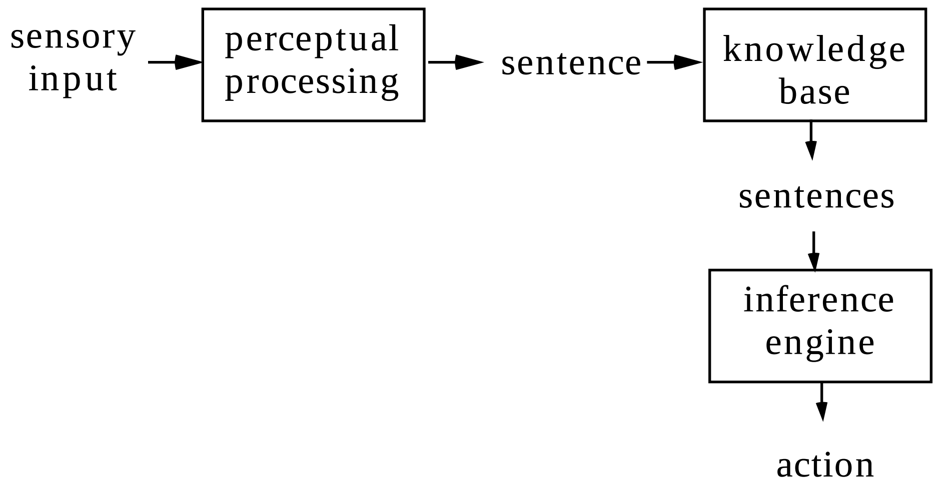
A knowledge-based agent:

- represents knowledge in some language
- reasons about the world
  by manipulating sentences written in the language
- acts based on the results of reasoning

The Architecture of a Knowledge-Based Agent

The basic process of the agent is:

- Input a percept **p**.
- Translate **p** into a sentence **s**.
  Add **s** to the knowledge base.
- Infer what to do next, action **a**,
  by manipulating the knowledge base.
- Execute **a**.



On Knowledge Representation

Generally, the goal of KR is to ...

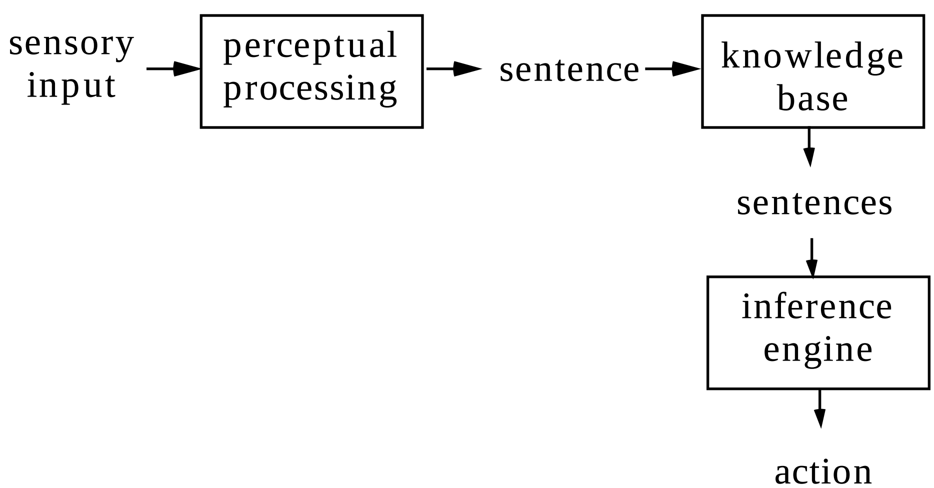
a. make it possible to describe situations in the world.

(writing knowledge down)

b. make it possible to reason over these descriptions.

(using knowledge)

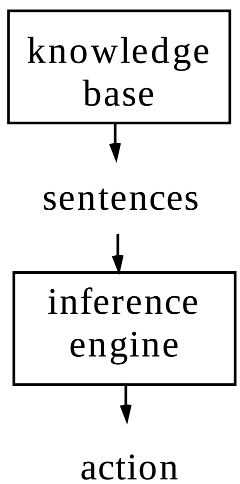
To the extent that (a) makes it easy for us to do (b),
"the problem is almost solved".



Styles of Knowledge Representation

We have open to us an important trade-off:

- put most of the knowledge in the sentences
- put most of the knowledge in the procedures



This trade-off is the center of a long-running open question in KR research...

Logic
as a Knowledge Representation

Why use logic as a model of thought?

- Logic is a deeply-studied, well-understood, concise model of knowing and reasoning.
- Logic is formal, and so easily computed.
- Most logics reach only correct answers.

Does that sound like how humans think?

Does it make sense to use humans as a model here?