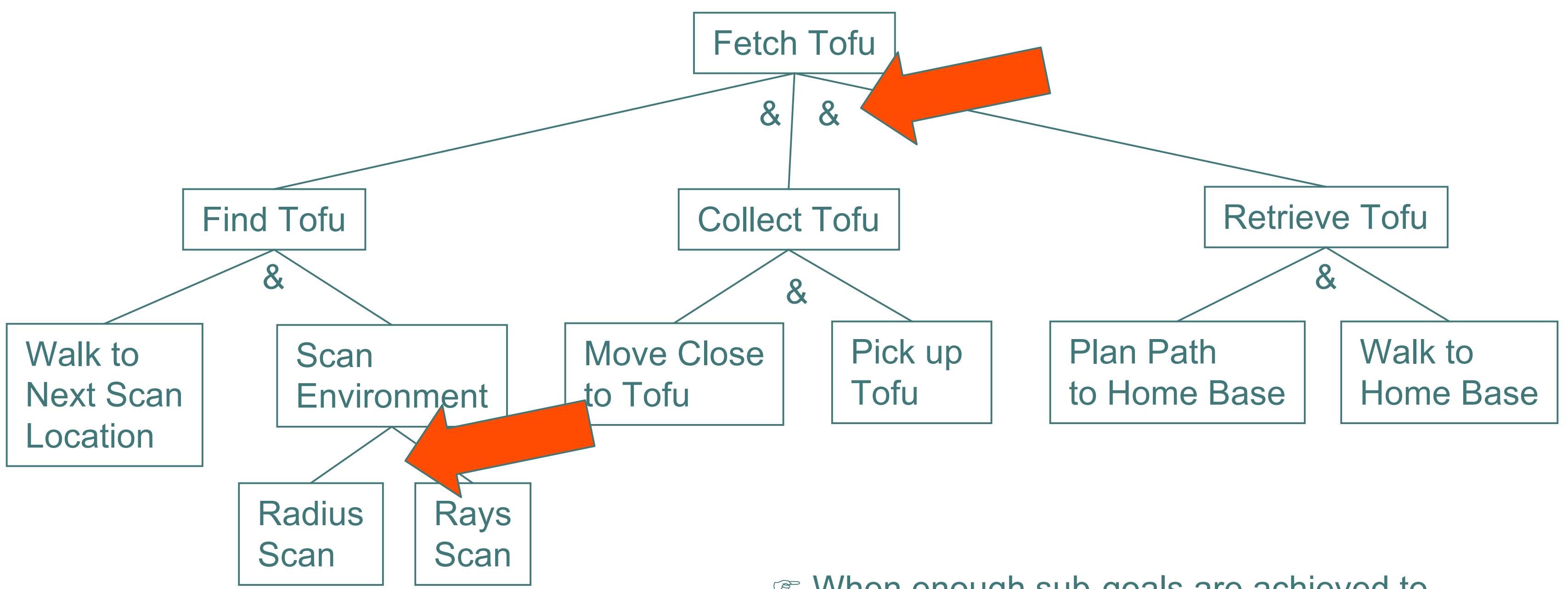
Problem Analysis

- A design example: Fetch an Object
  - The animat is tasked with retrieving a block of tofu.
- We will use goal oriented decomposition
  - Best matched to our definition of intelligence
  - Goals imply sub-goals
  - Primitive sub-goals can easily implemented by behavioral/functional components

Intelligence is the computational part of the ability to achieve goals in some world.

Problem Analysis



And/Or Goal Trees

- When enough sub-goals are achieved to achieve the goal at the root of the and/or goal tree, the goal tree is said to be <u>satisfied</u>.

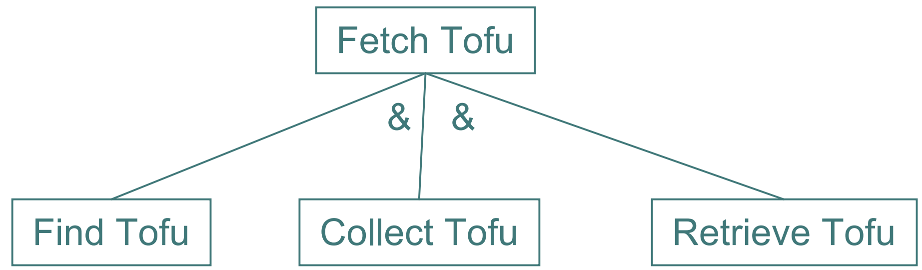
Problem Analysis

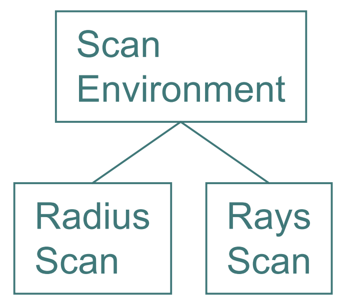
Formal Test Procedures:

```
Procedure TestNode:
if the node is trivially satisfied then
    return true;
else if the node is an AND node then
    use the TestAndNode procedure;
else if the node is an OR node then
    use the TestOrNode procedure;
```

```
Procedure TestAndNode:
for each child node
   use the TestNode procedure
if a child node returned false then
    return false;
If all child nodes returned true
   return true;
```

```
Procedure TestOrNode:
for each child node
   use the TestNode procedure;
if a child node returned true then
    return true;
```





Note: the test procedure also gives you a way to execute a goal tree - *goal reduction*

Problem Analysis

- Construct and/or goal trees for the following high-level goals:
  - Walk around a 270˚ corner
    - Initial condition: you are parallel to a wall
    - Goal: if you find a 270˚ corner, take it
  - Meet another quagent
    - Initial condition: your quagent just spotted another quagent.
    - Assumption: the other agent continues to do whatever it was doing (walking towards/away from you, standing still, etc).
    - Goal: *Anticipate* what the other agent is going to do and steer to within a click or two.

Problem Analysis

- We have the goals and their implementation...
- Some other basic questions to ask:
  - Does the problem solution require specialized knowledge?
  - If so, how do we represent that knowledge?

Knowledge Representation

A *representation* is a set of conventions about how to describe a class of things.

A *description* makes use of the conventions of a representation to describe some particular thing.

Artificial Intelligence, Patrick Winston, Addison-Wesley, 1984

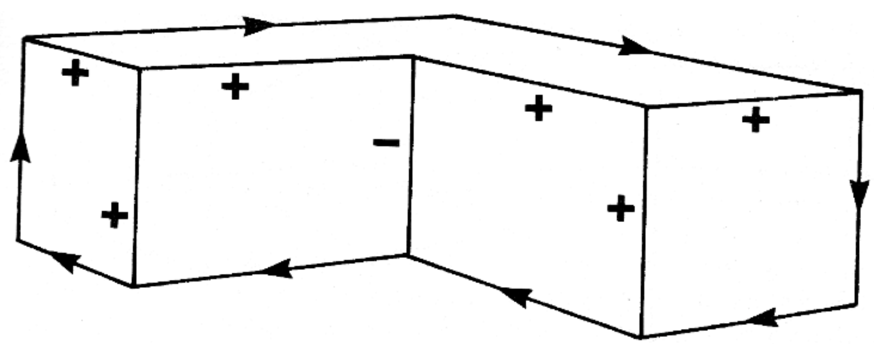
Knowledge Representation

- Good representations make the important things explicit.
- They expose the *natural constraints*.
  - Some transitions are possible, some are not.
- They are *complete*.
  - We are able say everything that needs to be said.
- They are *concise*.
  - We can say things efficiently.
- They are *transparent*.
  - We can understand what has been said.
- They facilitate *computation*.
  - We can store and retrieve information rapidly.
- They *suppress detail*.
  - We can access information at the appropriate level of abstraction.

Knowledge Representation

- Every problem has a set of *natural constraints*
  - E.g., in pathfinding we are not allowed to move instantly from one part of the space to another neither are we allowed to traverse through opaque space (e.g. walls).
- A solution must satisfy these constraints - *constraint satisfaction*
- Representations that allow for easy reasoning about constraints, *constraint propagation*, are preferred.

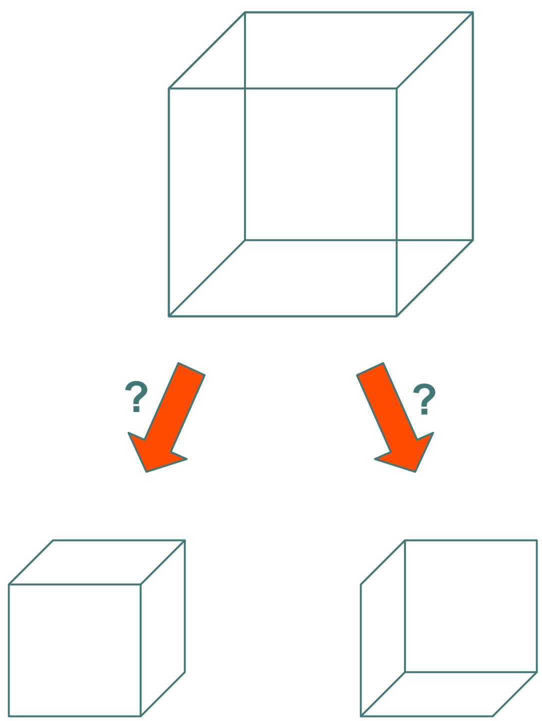
Knowledge Representation



Line drawing interpretation

- Line constraints:
  - Concave: **+**
  - Convex: **-**
  - Boundary: →
- Once we label one of the lines the constraints imposed by the real world force the interpretation of the other lines - *constraint propagation*

Knowledge Representation



- A line drawing without constraints is difficult to interpret.
  - Can you see the big cube flipping back and forth between the two perspectives?
  - What constraints should be put on big cube to represent the left bottom cube? The right bottom cube?

Knowledge Representation

- Symbolic constraint propagation offers a plausible explanation for *one* human information processing phenomenon, it offers also a good way to structure computer solutions.
- Processing constraints and regularities in the world make it possible for entities to be intelligent, be it human or machines.

Knowledge Representation

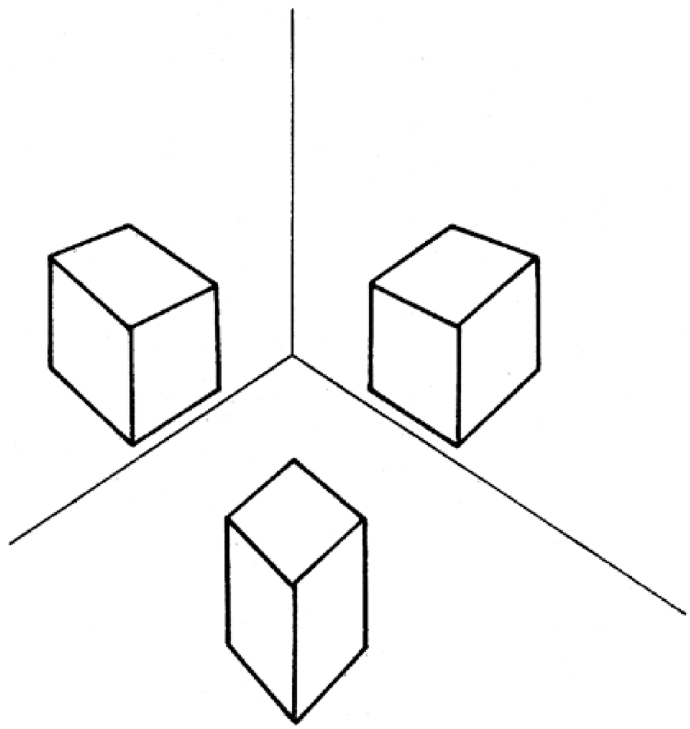


Figure 3-19. Without shadows, there are several ways to interpret a cube: it may be suspended or it may be attached to a floor or a wall by one of its hidden faces.

- Shading is also a real world constraint we use to interpret the world.
- Without shading the picture to the left is difficult to interpret.

Knowledge Representation

- A representation is a set of conventions about how to describe a class of things
- Representations used in AI:
  - Symbols
  - Attribute-value representation
  - Frames
  - Semantic networks
  - List

Knowledge Representation

- Symbolic representation
  - Imperative languages
    - Weak
    - Groups of symbols with values assigned to them
    - Data structures are used to convey some relationships between the symbols - e.g. classes in OOP
  - Symbolic programming languages
    - The symbol itself becomes the carrier of knowledge/meaning, e.g. predication in logic programming.
    - A program is a collection of these symbols and a way to reason about them.
    - This makes Prolog so attractive as an AI language.

Knowledge Representation

- Object-Attribute-Value
  - Can often be represented as a table

	Distance	Presence
Left_obstacle	4.0	known
Right_obstacle	?	unknown

Attribute

Object

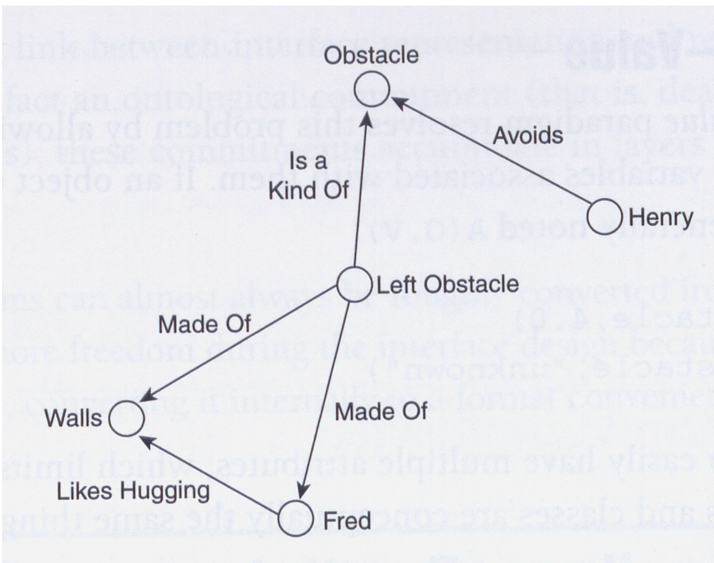
Value

Knowledge Representation

- Frames
  - Records composed of slots and pointers
  - Stores values of objects and relationships between objects
  - When programming in Java/C++ you can use classes to achieve the same goal
  - Used in applications in cognitive modeling

```
Frame-left-obstacle:
   distance:     4.0
   presence:     known
   pointer:      (frame-fred)(frame-joe)
```

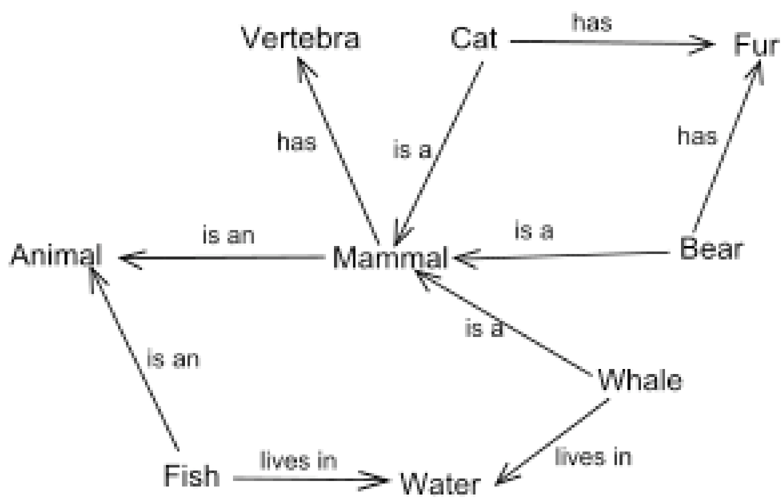
Knowledge Representation



- Semantic Networks
  - Stores the relationships between objects in a graph-like manner
    - Nodes represent concepts
    - Edges describe relationships
  - Interesting relationship to logic programming
    - the edges in a semantic network can be viewed as predicates.

Knowledge Representation

- Another semantic network (Wikipedia)



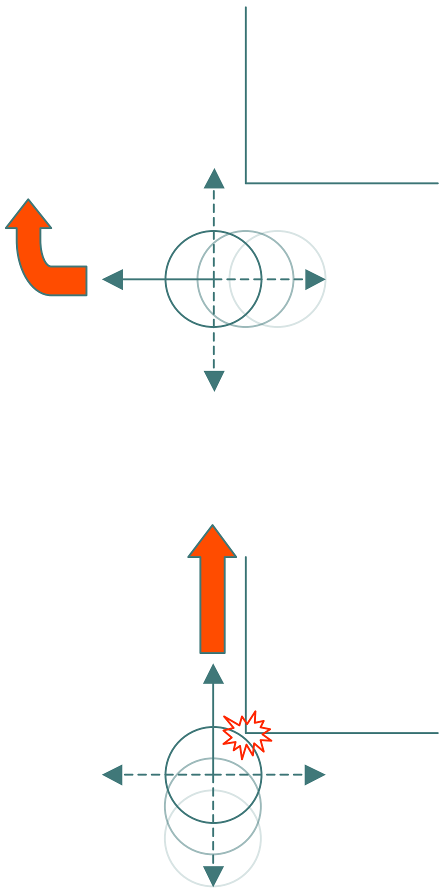
Knowledge Representation

- Lists
  - Similar objects are gathered together in a list
  - We can envision list operators almost like the operators from set theory: union, intersection, member-of, complement, *etc.*
  - Also considered a symbolic representation.
  - Powerful and natural representation, we keep lists of things, sort lists, *etc*.

Mammals = [dog, giraffe, elephant, mouse]
Shopping = [milk, coffee, bread, eggs]

Lisp = **Lis**t **P**rocessor – most famous AI language based on list representation.

Hints: Embodiment



- When turning a corner you will need to take the size of the body into account.
- Embodiment poses natural constraints onto your navigation problem.

Assignments

- Reading Chaps 1-10 (except Chap 4)
- Quiz Tuesday 2/16
  - Covers everything up to this point