# World Models

## 1 Analogical and logical representations

When an agent stores previous percepts in a memory, we can think of the contents of this memory as a *model* of the world. In AI, when we're building models of the world, we choose between *analogical representations* and *logical representations*. There's no clear-cut distinction between the two, but it's a useful distinction to make because they have different strengths and weaknesses.

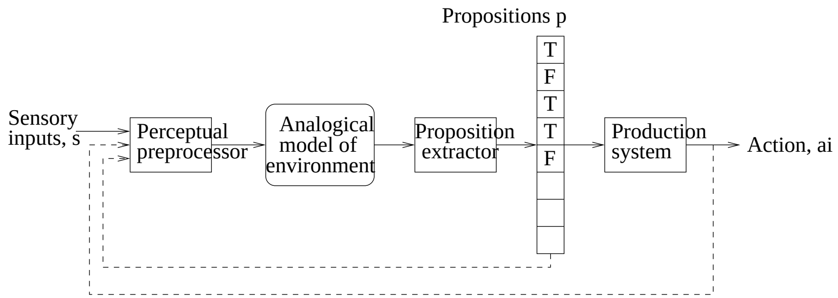
- In an *analogical representation*, we use a data structure that gives a strong structural similarity between the representation and the world being modelled. Maps are good examples of analogical representations.
- In a *logical representation*, the world is described by statements in some language but the syntax of the statements is not a reflection of the structure of the world. Modelling the world as a vector of T or F propositions is an example of a logical representation. Describing the world using first-order predicate logic (see next lecture) is another example.

One might say that an analogical representation is an *analogue* of the world; a logical representation is a *description* of the world.

**Class exercise.** What do you think the relative strengths and weaknesses of the two representations are?

## 2 Example using an analogical representation

Let's take an example of an agent whose memory stores a model of the world using an analogical representation. There are many ways of organising such an agent. The diagram shows one of these, in the case of an agent whose action function is implemented using a production system:




The inputs to the perceptual preprocessor are the agent's sensory inputs, $\mathbf{s}$, the previous vector of truth-values, $\mathbf{p}$, and the action the agent has just executed. These are used to update the analogical model. Note how the agent's actions will not only affect the world, they will also update the model. For example, if the agent decides to execute a move action, then it needs to both move in the world and update the model so that its new position in the world is reflected in the model.

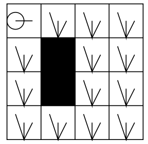


The truth or falsity of propositions are then determined from the model. These propositions are used by the production system.

The example agent we will design is an 'intelligent' lawn-mower. The agent inhabits a grid-like lawn. It has eight touch sensors mounted around its body, so it can detect obstacles on the lawn. However, it has no way of sensing whether a part of the lawn has been mown already or not. But, by keeping track, in its memory, of where it has been, it can nevertheless know which parts of the lawn it has already mown.

An analogical model of this world can take the form of a two-dimensional array. Cells of the array represent locations in the environment. At any time, the cells in the model can contain one of five values: 'O' if the corresponding location in the world contains an obstacle; 'A' if the corresponding location in the world is where the agent is currently located; 'N' if the corresponding location in the world has not yet been mown; 'M' if the corresponding location in the world has been mown; and 'U' if the status of the corresponding location in the world is unknown. We'll also indicate the orientation of the agent: 'AN' indicates that the agent is facing North; 'AE', 'AS' and 'AW' are similar for East, South and West.

Suppose the world is as shown in the left-hand diagram. And suppose the agent knows its initial location. But suppose the agent knows nothing else about the world. Therefore, its initial model of the world is as shown in the right-hand diagram.



| | | | |
|---|---|---|---|
| AE | U | U | U |
| U | U | U | U |
| U | U | U | U |
| U | U | U | U |

Remember, your God-like perspective lets you see the left-hand diagram; but the agent knows only what is shown in the right-hand diagram.

Our agent starts off, then, in a position of considerable ignorance. (In some cases, we, as designers of an agent, might choose to give the agent a less under-specified model of the world. We might even give it a fully-specified map of the world, showing the location of all objects. Obviously, the more complete the initial model, the better.)

The proposition extractor will set the truth-values of the following three propositions:

- $p_0$ will be T iff at least one of the cells to the North, East, South or West of the agent in the model contains 'N';
- $p_1$ will be T iff the cell in front of the agent in the model contains 'N';
- $p_2$ will be T iff the cell in front of the agent in the model contains 'M'.

Here are the condition-action rules:

**if** $p_1$ **then** `Move`
**if** $p_0 \wedge \neg p_1$ **then** `Turn(RIGHT, 2)`
**if** $p_2$ **then** `Move`
**if** $\neg p_2$ **then** `Turn(RIGHT, 2)`

(Whether these are good lawn-mowing rules, I don't know.) The agent will turn clockwise until directly ahead of it there is a cell that it knows to be not yet mown ('N'), and it will move forward to that cell. If there is no such cell, it will turn clockwise until directly ahead of it there is a cell that has previously been mown ('M'), and it will move forward to that cell.

We'll look at the effects of a few sense/plan/act cycles.

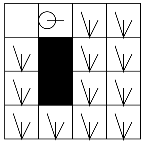
Sense:

| | | | |
|---|---|---|---|
| AE | N | U | U |
| N | O | U | U |
| U | U | U | U |
| U | U | U | U |

Plan:

Move

Act:



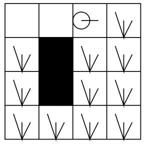
Sense:

| | | | |
|---|---|---|---|
| M | AE | N | U |
| N | O | N | U |
| U | U | U | U |
| U | U | U | U |

Plan:

Move

Act:



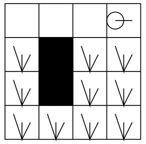
Sense:

| | | | |
|---|---|---|---|
| M | M | AE | N |
| N | O | N | N |
| U | U | U | U |
| U | U | U | U |

Plan:

Move

Act:



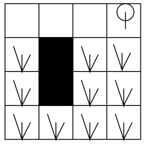
Sense:

| | | | |
|---|---|---|---|
| M | M | M | AE |
| N | O | N | N |
| U | U | U | U |
| U | U | U | U |

Plan:

Turn(RIGHT, 2)

Act:



Some cycles later…

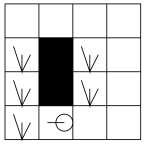
Sense:

| | | | |
|---|---|---|---|
| M | M | M | M |
| N | O | N | M |
| U | O | N | M |
| U | N | AW | M |

Plan:

Move

Act:



Later still…

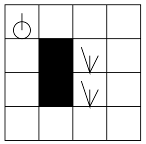
Sense:

| | | | |
|---|---|---|---|
| M | M | M | M |
| AN | O | N | M |
| M | O | N | M |
| M | M | M | M |

Plan:

Move

Act:





And even later…

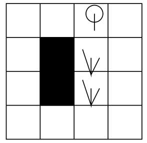
Sense:

| | | | |
|---|---|---|---|
| M | M | AE | M |
| M | O | N | M |
| M | O | N | M |
| M | M | M | M |

Plan:

Turn(RIGHT, 2)

Act:



**Class exercise.** Consider an obstacle-free rectangular lawn, with the agent starting in the top-left. The agent, using the rules above, will mow the whole lawn. Explain why.

**Class exercise.** There are worlds that the agent would never successively mow in their entirety. Can you come up with any?

## 3 Dynamic environments

The lawn-mowing agent eventually builds such a complete model that it has no need to sense the world at all. At this point, in a *static* environment, the agent could switch off its sensors and rely entirely on its model. But this assumes that changes to the environment are due only to the actions of the agent.

In general, environments are *dynamic*: the world changes independently of our agent. For example, grass keeps growing; a rolling ball continues to roll until it has lost all energy; and, perhaps most importantly, other agents act upon the world.

This brings the problem of keeping the model up-to-date. This can be done in two ways. Either it can be done by revisiting and re-sensing parts of the environment, or it can be done through knowledge and reasoning: you might be able to *predict* what parts of the environment will be like on the basis of knowledge that you possess (e.g. that grass re-grows), and so you might update your model using these predictions. However, predictions can be wrong, so the model no longer provides information that has perfect certainty. Let's discuss predictions in more detail.

The easiest prediction to make is that things haven't changed. For example, we might predict that an object is wherever it was when we last sensed it. For many objects, this is a correct prediction. But, of course, for others, it will be wrong. Even if we equip our agent with the ability to make only this easy prediction, we have to accept that in dynamic environments the longer it is since a memorised percept was sensed and stored, the more unreliable that memorised percept becomes. Beyond a certain point, the memorised percept will have become so unreliable that it should no longer be used and that part of the world needs to be re-sensed.

Suppose instead we want to make more sophisticated predictions. Two requirements become apparent:

- We need to be able to represent general knowledge about the world, and reason with that knowledge. As we discussed in answer to the class exercise at the start of this lecture, analogical representations tend to be less expressive and therefore less able to represent the general knowledge that we need, and their reasoning methods tend to be ad hoc; logical representations are better able to satisfy our requirements.
- An agent's belief state will now be the set of states that are compatible with both the agent's percepts and predictions made on the basis of general knowledge and memorised percepts. Again as we discussed in answer to the class exercise at the start of this lecture, it is more cumbersome to represent indefiniteness, incompleteness and uncertainty analogically than logically.

In the next few lectures, then, we will look at logical representations in more detail. (Ideally, we also need to make these representations probabilistic, but that is an issue that we will overlook.)