CE213 Artificial Intelligence – Lecture 7

Knowledge Representation & Expert Systems

What problems cannot be solved by state space search?

How important is domain knowledge in AI?

How to represent knowledge?

How to make use of knowledge in decision making?

A Brief Review of State Space Search

- Basic elements for problem solving by state space search:

  State space, Initial state, Goal state, Operators, Transition functions;

  Search strategies (not necessarily part of state space representation).

- Key issues in search strategies:

  How to select nodes for expansion during search tree construction?

  (or how to evaluate nodes/states?)

  How to evaluate search strategies?

  e.g., uniform cost search vs. A* search (in terms of 4 criteria)

  Search strategies for game playing: minimax, alpha-beta pruning, MCTS.

- What problems can or cannot be solved by state space search?

  Whether solutions to problems can or cannot be represented as a sequence of operations/actions/moves.

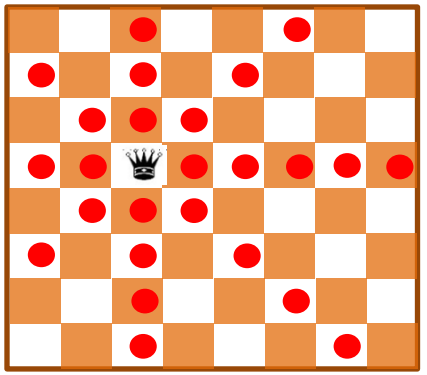
  Whether efficient search strategies are available for very complex problems.

Examples of problems that cannot be solved or are hard to be solved by State Space Search

- How can you solve problems like face recognition, medical diagnosis, financial market forecasting by state space search?

- How about the 8-queens problem?

  How to place 8 queens on an 8x8 chessboard in such a way that no queen will be attacked by any other queens.



This is a puzzle problem and can be easily represented as a state space, but the size of this state space is huge and there is go good heuristic for evaluating states.

Is it possible to solve the 8-queens problem by state space search?

- If we look for a satisfactory configuration of 8 queens on a chessboard, but not a sequence of moves:

  There would be 64 x 63 x 62 x 61 x 60 x 59 x 58 x 57 ≈ 2 x 10^{14} possible configurations. Checking whether all these possible configurations satisfy the requirement of the 8-queens problem may take several years by a computer.

- We could formalise the 8-queens problem for state space search:
  Approach 1: Use a random 8 queens configuration as initial state, and changing a queen's position as operator.
  Approach 2: Use an empty chessboard as initial state, and adding a queen to or removing a queen from the chessboard as operator.
  What search strategy would be suitable for this problem?

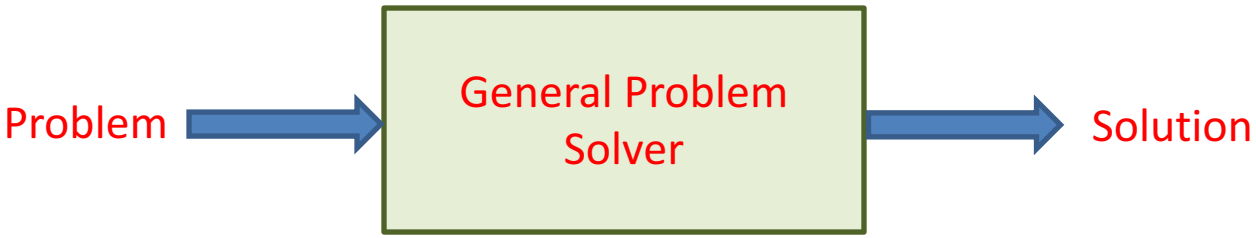
- Other approaches?

Pre-1970 AI: Heuristic Search Era

Two AI techniques:

1. Universal Problem Solving Techniques: **'Generate and Evaluate'**

   e.g., GPS (general problem solver) based on search strategies



2. Universal Learning Techniques

   e.g., Perceptrons

Both are characterised by absence of *a priori* knowledge of problem domain (There could be no knowledge available or it is difficult to use knowledge in some problems, e.g., environment understanding).

Post-1970 AI: Knowledge Based Systems Era

- The central role of ***knowledge*** in all types of intelligent activity was well recognised.
- The research emphasis of AI was shifted to study of **techniques for representing and using knowledge**.
- As a consequence, there was a 10 year eclipse of research on machine learning (This was also due to the book *Perceptrons* by Marvin Minskey, published in 1969).

knowledge vs. ***heuristic ?***

The First Influential Expert System

DENDRAL (https://en.wikipedia.org/wiki/Dendral): 1965-1972, Stanford Uni.

It was the first serious attempt to produce a system as good as a human expert in a real world problem domain (*It used a 'generate and evaluate' approach, but* was *different from state space search – no operators are available for state transition* and thus solutions cannot be a sequence of operations)

DENDRAL's Task:

To determine the **structure** of a chemical compound from **mass spectrogram** data. (There could be a very large number of possible structures)

Basic idea of mass spectrogram:

Smash the molecules up and get a histogram of fragment masses**.**

Chemical formula is known.

Atomic weights are known.

All this **information** can be used to deduce how a molecule is put together.

The First Influential Expert System (2)

Why is it a difficult task? – Huge search space

Because atoms may be joined in many different configurations.

e.g., An organic compound with only 20 atoms could have 10^4 isomers.

Naive solution: Unconstrained "Generate and Evaluate"

Generate all possible structures (isomers).

For each, check if it is consistent with spectral data (and any other known constraints).

Computationally expensive!

The First Influential Expert System (3)

Better solution: Constrained "Generate and Evaluate"

Use ***knowledge about the problem domain*** to constrain the generation of candidate solutions.

Knowledge might include:

Substructures that molecule is known to contain.

Substructures that molecule is known not to contain.

Which bonds are more likely to break.

Result is ***far fewer candidate solutions will be generated***.

=== Benefit of using knowledge for heuristic search

Revisit the 8-Queens Problem

Naïve "Generate and Evaluate":

Generate every possible configuration of 8 queens on a chessboard.

For each, check if it satisfies requirement.

How many such configurations are there?

$64 \times 63 \times 62 \times 61 \times 60 \times 59 \times 58 \times 57 \approx 2 \times 10^{14}$

A program that can check a million configurations per second would take about 2×10^8 seconds (5.6 years) to execute.

8-Queens Problem: Adding a Constraint

Only one queen is allowed in each row.

How many configurations are now possible?

$8^8 \approx 2 \times 10^7$

Thus by applying one constraint, number of configurations to be tested is reduced by a factor of 10^7.

Execution time is reduced to about 20 seconds.

8-Queens Problem: Further Constraints

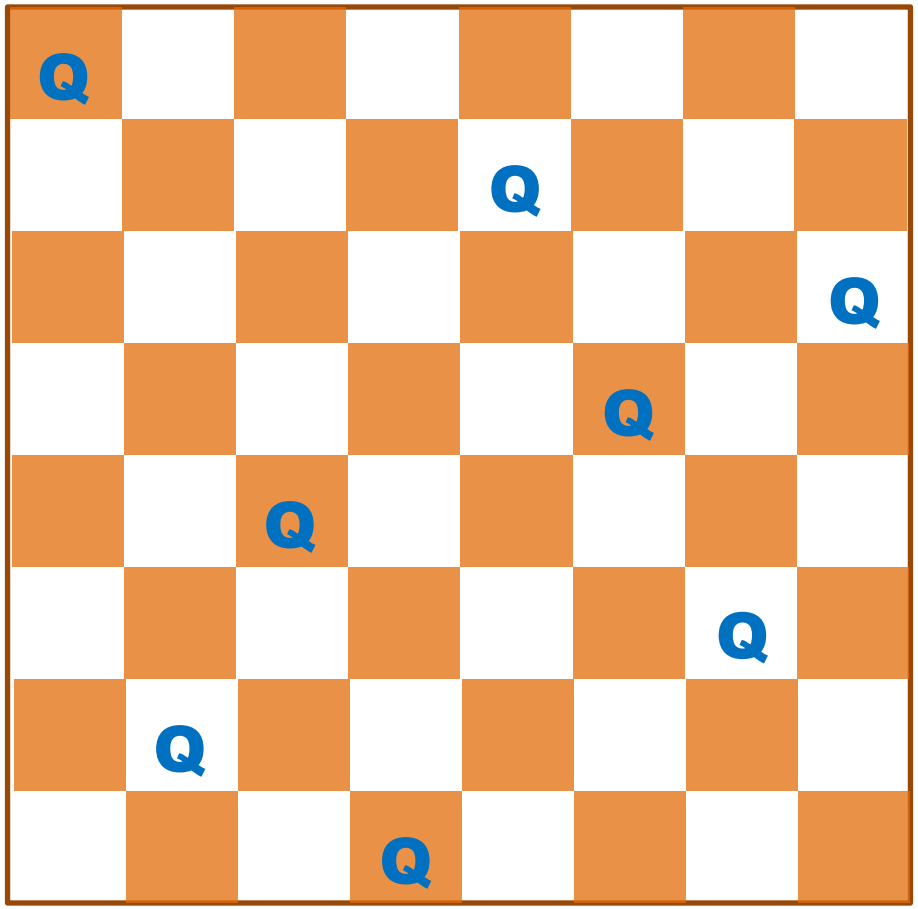
Further constraints are possible:

- Only one queen is allowed in each column.
- Only one queen is allowed in each diagonal.

Together these would constrain the generation of candidate solutions so much that only correct solutions are produced.

- The execution time will be dominated by constraint application rather than checking whether candidate solutions satisfy problem requirements.

8-Queens Problem: A Solution



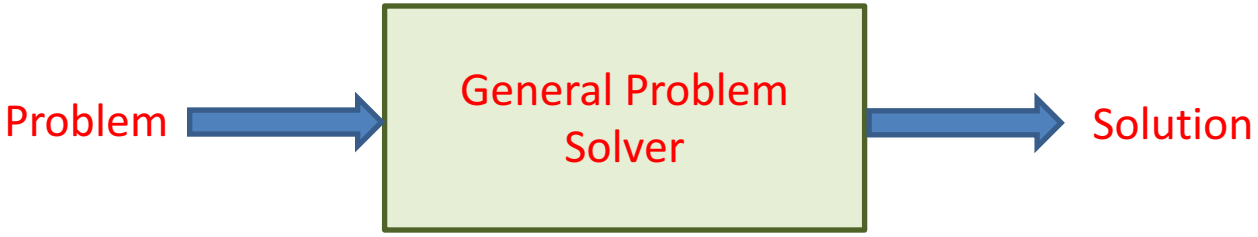
(There are 92 distinct solutions to the 8-queens problem.)

Lessons from DENDRAL and 8-Queens Problem

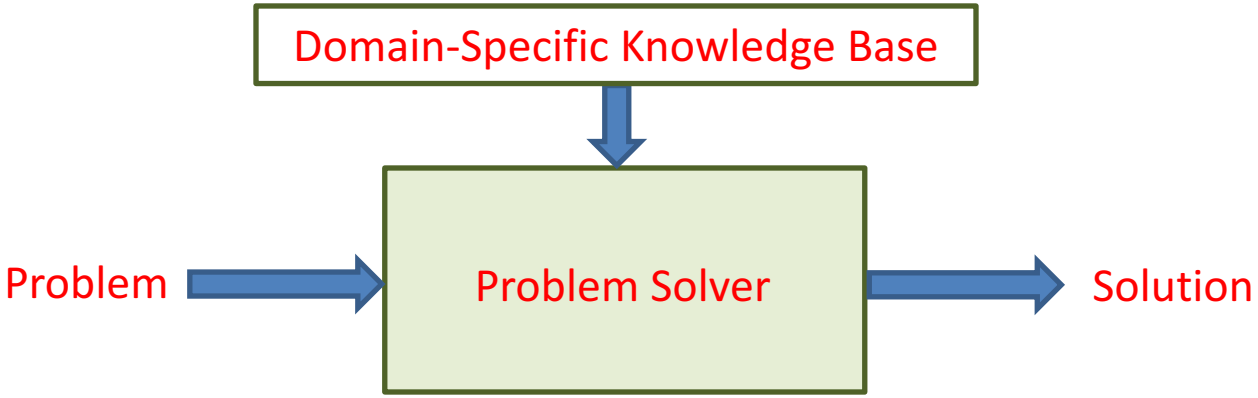
- Searching is easy but time consuming.
- The "intelligence" lies in **constraining the search** so the time taken is reasonable.
- Putting in the constraints is harder than programming the basic search.
- Constraints embody domain-specific knowledge in the problem solving process.

Another Problem Solving Framework

All of the lessons suggest that we need to replace



with



And this suggests that we need a method which makes it easy to supply and use domain-specific knowledge.

Production System

Basic Idea (simple but powerful)

Codify knowledge as a set of condition (situation or state) - action rules:

e.g.,

IF IT IS RAINING THEN PUT UP YOUR UMBRELLA

(condition) (action)

Such rules are called "***Production Rules***".

A set of production rules constitutes a main part of a "***Production System***" (the other main part is rule interpreters, the topic of next lecture).

In state space search, there are states and operations for state transition. In production systems, there are states and actions that can be state transition or others such as making a decision or drawing a conclusion.

Production Systems (2)

A production rule is essentially an *ordered pair*.

The **condition component** is often called "antecedent", "left hand side" or simply "LHS".

The **action component** is often called "consequent", “right hand side” or simply "RHS" .

Note the similarity to a logical implication:

antecedent $\Rightarrow$ consequent

(IF) (THEN)

Advantages of Production Systems

Procedural Representation (easy to program)

The knowledge is encoded in a form that indicates *how it is to be used*.

Compare the production rule

IF it is raining THEN put up your umbrella

with

An umbrella is a device to keep the rain off.

Both represent the same knowledge, but the production rule makes it easy to program.

The latter is an example of a ***declarative representation***.

Advantages of Production Systems (2)

Modularity

Each chunk of knowledge is encoded as a separate rule.

Thus it is comparatively **easy to add or remove** individual pieces of knowledge.

This is comparable to adding or removing individual statements from a program in Java, Python or C++.

Frequently the order of the rules has no significance.

Again, this is comparable to conventional languages.

Advantages of Production Systems (3)

Explanation Facility

For many production systems, it is easy to add a facility enabling the system to explain its reasoning to the user.

Similar to getting a statistical package to explain how it had calculated a result.

Similarity to Human Cognition - Naturalness

It has been claimed that the way a production system operates has a closer resemblance to human cognition than the behaviour of other computer languages does.

Production systems also have some disadvantages - we will talk about these later (e.g., incompleteness, curse of dimensionality).

Summary

State space search can not solve all the problems

Knowledge is useful and necessary for problem solving

DENDRAL as an example, the first influential expert system.

Production system for knowledge representation

Condition (situation or state) – action rules

Procedural representation

Knowledge represented in a form that indicates how it can be used

Modularity, Explanation facility, Similarity to human cognition

Production system vs. state space representation?

How to use production rules to solve problems? – Expert systems