Artificial Intelligence I 2023/2024

Week 10 Tutorial and Additional Exercises

A* & Optimisation Problems

School of Computer Science

21st March 2024

In this tutorial. . .

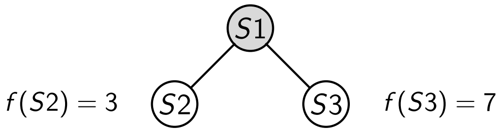
In this tutorial we will be covering

- A* search
- Terminology of optimisation problems.
- How to formally represent an optimisation problem.
- Examples of optimisation problems.

Informed Search

- **Informed search** strategies use problem-specific knowledge beyond the definition of the problem itself
- In general, informed search strategies can find solutions more efficiently compared to uninformed search
- The general approach, called best-first search, is to determine which node to expand based on an evaluation function

$$f(n) : n \to c(n)$$



- This function acts as a cost estimate: the node with the lowest cost is the one that is expanded next

A* Search

- Recall the definition of A* search.

Definition 1 (A* search)

A*-search is an informed search strategy in which

- each node, n, is evaluated using the following evaluation function

$$f(n) = g(n) + h(n)$$

  where $g(n)$ is the **cost** to reach node n and $h(n)$ is the **heuristic** of node n.

- this evaluation function is then used to provide a hierarchy between nodes, and prioritise lower-valued nodes to be expanded first.

- This is equivalent to the cost of the cheapest solution through node n.

Exercise 1

- Recall the **8-queen puzzle**.
- We start with an empty 8×8 chessboard, and the goal is to place 8 queens on the chessboard, so that there are no conflicts (no two queens attack each other).
- Recall that a queen can move any number of squares, either vertically, horizontally, or diagonally, in one move.
- Placing a queen on an empty square costs 1.
- In each state, n, there is a number of queens on the board (0-8), which had been placed in a specific order (according to the path we followed from the root to n).
- Propose possible heuristics, $h(n)$, that can be used for an implementation of A* to this problem from Definition 1.
- Evaluate and compared the performance of the proposed functions, in terms of speed and accuracy of the result.

Exercise 1: Solution

Some possible examples of heuristic functions, $h(n)$, are

1. $h_1(n)$: number of previously placed queens in the same column as the last placed queen;
2. $h_2(n)$: number of previously placed queens in the same row as the last placed queen;
3. $h_3(n)$: distance (measured in some way) of the last placed queen from the second-to-last placed queen;
4. $h_4(n)$: mean distance between the last placed queen, and all previously placed queens;
5. $h_5(n)$: number of remaining squares, so no conflict is made with an already placed queen.

Exercise 1: Solution (continued)

We next give a brief evaluation of the heuristic functions (in the same order).

1. h_1: Quick, but not very informed;
2. h_2: Quick, but not very informed;
3. h_3: Quick, but not very informed;
4. h_4: A compromise between speed and accuracy.
5. h_5: Slower, but more informed;

Exercise 1: Solution (continued)

We next give an example of the heuristic h_1, on this formulation.

- Define s_0 the initial state with the empty chessboard.
- Define $s_1, s_2, \ldots, s_{64}$ the states where the first queen is placed on $a1, a2, \ldots, h8$ respectively.
- Define $s_{65}, s_{66}, \ldots, s_{127}$ the successors of s_1, that is the states where one queen is already on $a1$, and the second queen is placed on $a2, a3, \ldots, h8$ respectively.
- Define $s_{128}, s_{129}, \ldots, s_{190}$ the successors of s_2, that is the states where one queen is already on $a2$, and the second queen is placed on $a1, a3, \ldots, h8$ respectively.
- Similarly, define the successors of all nodes $s_3, \ldots, s_{64}$.
- Repeat this labeling to define the successors for the nodes in the second level, $s_{65}, s_{66}, \ldots$, and so on, until the final level has all eight queens placed.

Exercise 1: Solution (continued)

- The root node has $g(n) = 0$ (no queens are placed) and $h(n) = 0$ (no column conflicts yet).
- All nodes, of the first level have $g(n) = 1$ (one queen is placed) and $h(n) = 0$ (no column conflicts yet).
- In the second level, node s_{65} has two queens, placed on $a1, a2$. Since there is a column conflict, we have $h(s_{65}) = 1$, and also $g(s_{65}) = 2$ (two queens are placed). Therefore, $f(s_{65}) = 1 + 2 = 3$.
- Also in the second level, node s_{127} has two queens, placed on $a1, h8$. Now $h(s_{127}) = 0$. since there are no column conflicts. Also $g(s_{127}) = 2$ (two queens are placed). Therefore, $f(s_{127}) = 0 + 2 = 2$.
- Therefore, an A* search with the heuristic h_1, will expand node s_{127} before node s_{65}.
- Do a similar computation for the other heuristics proposed.

Formal optimisation problem

- Recall the canonical form of an optimisation problem:

$$\begin{array}{rlr} \text{maximise/minimise} & f(\mathbf{x}) & \\ \text{subject to} & g_i(\mathbf{x}) \leq 0, & i = 1, \ldots, m \\ & h_j(\mathbf{x}) = 0, & j = 1, \ldots, n \end{array}$$

- $\mathbf{x}$ is the vector of *design variables*.
- f is the *objective function*, e.g. the cost or quality of a solution.
- $g_1, \ldots, g_m$ are the *inequality constraints* and $h_1, \ldots, h_n$ are the *equality constraints*.
- In a *multi-objective* optimisation problem, there are more than one objective functions, e.g. $f_1, f_2, \ldots, f_k$.

Formal optimisation problem (continued)

Some more definitions:

- Each value of $\mathbf{x}$ is a *solution* to the optimisation problem.
- The *search space* consists of all possible solutions.
- A solution that satisfies the constraints is called *feasible*. A solution that does not satisfy the constraints is called *infeasible*.

Exercise 2

Consider the following problem:

- A company makes square boxes and triangular boxes. Square boxes take 2 minutes to make and sell for a profit of 4. Triangular boxes take 3 minutes to make and sell for a profit of 5. No two boxes can be created simultaneously. A client wants at least 25 boxes including at least 5 of each type in one hour. What is the best combination of square and triangular boxes to make so that the company makes the most profit from this client?
- Formalize this problem as a canonical optimisation problem, but consider it is ok for the objective to be a function to be maximised instead of minimised. Identify the design variables, the objective function and the constraints.

Exercise 2: Solution

- Let $x_1 \in \mathbb{N}$ be the number of square boxes and $x_2 \in \mathbb{N}$ be the number of triangular boxes. These are the design variables. Write $\mathbf{x} = (x_1, x_2)$. The formal optimisation problem is the following:

$$\begin{array}{ll} \text{maximise} & 4x_1 + 5x_2 \\ \text{subject to} & 2x_1 + 3x_2 \leq 60 \\ & x_1 \geq 5 \\ & x_2 \geq 5 \\ & x_1 + x_2 \geq 25 \end{array}$$

- Let us find the objective function and the constraints so that they follow the canonical formulation.

Exercise 2: Solution (continued)

- Objective function:

$$f(\mathbf{x}) = 4x_1 + 5x_2$$

- Constraints:

$$\begin{aligned} g_1(\mathbf{x}) &= 2x_1 + 3x_2 - 60 \\ g_2(\mathbf{x}) &= 5 - x_1 \\ g_3(\mathbf{x}) &= 5 - x_2 \\ g_4(\mathbf{x}) &= 25 - x_1 - x_2 \end{aligned}$$

  There are no equality constraints, so no $h_1, h_2, \ldots$ functions.

- With these definitions, the canonical problem can be written

$$\begin{array}{ll} \text{maximise} & f(\mathbf{x}) \\ \text{subject to} & g_i(\mathbf{x}) \leq 0, \qquad i = 1, 2, 3, 4 \end{array}$$

Exercise 3

Consider the following problem:

- A woman makes pins and earrings. Each pin takes 1 hour to make and sells for a profit of 8. Each earring takes 2 hours to make and sells for a profit of 20. She wants to make exactly as many pins as earrings. She has 40 hours and wants to have made at least 20 items, including at least 4 of each item. How many each of pins and earrings should the woman make to maximise her profit?
- Formalize this problem as a canonical optimisation problem, but consider it is ok for the objective to be a function to be maximised instead of minimised. Identify the design variables, the objective function and the constraints.

Exercise 3: Solution

- Let $x_1 \in \mathbb{N}$ be the number of pins and $x_2 \in \mathbb{N}$ be the number of earrings. These are the design variables. Write $\mathbf{x} = (x_1, x_2)$. The formal optimisation problem is the following:

$$\begin{array}{rl} \text{maximise} & 8x_1 + 20x_2 \\ \text{subject to} & x_1 + 2x_2 \leq 40 \\ & x_1 + x_2 \geq 20 \\ & x_1 \geq 4 \\ & x_2 \geq 4 \\ & x_1 = x_2 \end{array}$$

- Let us find the objective function and the constraints so that they follow the canonical formulation.

Exercise 3: Solution (continued)

- Objective function:

$$f(\mathbf{x}) = 8x_1 + 20x_2$$

- Constraints:

$$\begin{aligned} g_1(\mathbf{x}) &= x_1 + 2x_2 - 40 \\ g_2(\mathbf{x}) &= 20 - x_1 - x_2 \\ g_3(\mathbf{x}) &= 4 - x_1 \\ g_4(\mathbf{x}) &= 4 - x_2 \\ h_1(\mathbf{x}) &= x_1 - x_2 \end{aligned}$$

- With these definitions, the canonical problem can be written

$$\begin{array}{lll} \text{maximise} & f(\mathbf{x}) & \\ \text{subject to} & g_i(\mathbf{x}) \leq 0, & i = 1, 2, 3, 4 \\ & h_j(\mathbf{x}) = 0, & j = 1 \end{array}$$

Exercise 4

Consider the following problem:

- A student works at the library and as a tutor. They must work for at least 5 hours per week at the library and 2 hours per week as a tutor, but they are not allowed to work more than 20 hours per week total. They earn 15 per hour at the library and 20 per hour at tutoring. They prefer working at the library, so they want to have exactly twice as many library hours as tutoring hours. If they need to make exactly 360 in one week, how many hours should they work at each job to minimise their total working hours and meet their preferences?
- Formalize this problem as a canonical optimisation problem, but consider it is ok for the objective to be a function to be maximised instead of minimised. Identify the design variables, the objective function and the constraints.

Exercise 4: Solution

- Let $x_1 \in \mathbb{N}$ be the number of library hours per week and $x_2 \in \mathbb{N}$ be the number tutor hours per week. These are the design variables. Write $\mathbf{x} = (x_1, x_2)$. The formal optimisation problem is the following:

$$\begin{array}{rl} \text{minimise} & x_1 + x_2 \\ \text{subject to} & x_1 \geq 5 \\ & x_2 \geq 2 \\ & x_1 + x_2 \leq 20 \\ & x_1 = 2x_2 \\ & 15x_1 + 20x_2 = 360 \end{array}$$

- Let us find the objective function and the constraints so that they follow the canonical formulation.

Exercise 4: Solution (continued)

- Objective function:

$$f(\mathbf{x}) = x_1 + x_2$$

- Constraints:

$$\begin{aligned} g_1(\mathbf{x}) &= 5 - x_1 \\ g_2(\mathbf{x}) &= 2 - x_2 \\ g_3(\mathbf{x}) &= x_1 + x_2 - 20 \\ h_1(\mathbf{x}) &= x_1 - 2x_2 \\ h_2(\mathbf{x}) &= 15x_1 + 20x_2 - 360 \end{aligned}$$

- With these definitions, the canonical problem can be written

$$\begin{array}{lll} \text{minimise} & f(\mathbf{x}) & \\ \text{subject to} & g_i(\mathbf{x}) \leq 0, & i = 1, 2, 3 \\ & h_j(\mathbf{x}) = 0, & j = 1, 2 \end{array}$$

Exercise 5

Consider the following problem:

- A telescope manufacturer produces three types of telescopes; reflector, refractor and catadioptric. A reflector takes 3 hours to make and is sold for 500. A refractor takes 2 hours to make and is sold for 300. A catadioptric takes 4 hours to make and is sold for 600. The manufacturer wants to earn exactly 2000 profit in one day, while making an equal number of reflectors and refractors and at least 1 of each type. What is the best combination of telescope types to minimise the working hours?
- Formalize this problem as a canonical optimisation problem, but consider it is ok for the objective to be a function to be maximised instead of minimised. Identify the design variables, the objective function and the constraints.

Exercise 5: Solution

- Let $x_1 \in \mathbb{N}$ be the number of reflectors, $x_2 \in \mathbb{N}$ the number of refractors and $x_3 \in \mathbb{N}$ the number of catadioptrics per day. These are the design variables. Write $\mathbf{x} = (x_1, x_2, x_3)$. The formal optimisation problem is the following:

$$\begin{array}{rl} \text{minimise} & 3x_1 + 2x_2 + 4x_3 \\ \text{subject to} & x_1 \geq 1 \\ & x_2 \geq 1 \\ & x_3 \geq 1 \\ & 3x_1 + 2x_2 + 4x_3 \leq 24 \\ & 500x_1 + 300x_2 + 600x_3 = 2000 \\ & x_1 = x_2 \end{array}$$

- Let us find the objective function and the constraints so that they follow the canonical formulation.

Exercise 5: Solution (continued)

- Objective function:

$$f(\mathbf{x}) = 3x_1 + 2x_2 + 4x_3$$

- Constraints:

$$\begin{aligned} g_1(\mathbf{x}) &= 1 - x_1 \\ g_2(\mathbf{x}) &= 1 - x_2 \\ g_3(\mathbf{x}) &= 1 - x_3 \\ g_4(\mathbf{x}) &= 3x_1 + 2x_2 + 4x_3 - 24 \\ h_1(\mathbf{x}) &= 500x_1 + 300x_2 + 600x_3 - 2000 \\ h_2(\mathbf{x}) &= x_1 - x_2 \end{aligned}$$

- With these definitions, the canonical problem can be written

$$\begin{array}{rll} \text{minimise} & f(\mathbf{x}) & \\ \text{subject to} & g_i(\mathbf{x}) \leq 0, & i = 1, 2, 3, 4 \\ & h_j(\mathbf{x}) = 0, & j = 1, 2 \end{array}$$

Exercise 6

The following is the well-known *knapsack problem*:

- We want to load a knapsack with items. There are N items that can be loaded, and item i has a weight w_i, and a profit p_i. We need to decide which items to load, so as to maximise the total profit of loaded items, while keeping the total weight at most W.
- Formalize this problem as a canonical optimisation problem. Identify the design variables, the objective function and the constraints.
- Hint: Use N design variables, each of which is either 0 or 1.

Exercise 6: Solution

There exist more than one equivalent formulations. Here is one:

- Let $x_1, \ldots, x_N$ be variables where x_i equals 1 if item i is loaded, and 0 if not. These are the design variables.
- Write $\mathbf{x} = (x_1, \ldots, x_N)$, $\mathbf{p} = (p_1, \ldots, p_N)$, and $\mathbf{w} = (w_1, \ldots, w_N)$. The formal optimisation problem is:

$$\begin{array}{ll} \text{maximise} & \mathbf{x}^T\mathbf{p} \\ \text{subject to} & \mathbf{x}^T\mathbf{w} \leq W \end{array}$$

- Recall that $\mathbf{x}^T\mathbf{p} = x_1p_1 + x_2p_2 + \cdots + x_Np_N$ and similarly $\mathbf{x}^T\mathbf{w} = x_1w_1 + x_2w_2 + \cdots + x_Nw_N$. They can be computed using algorithm 1.
- Let us find the objective function and the constraints so that they follow the canonical formulation.

Exercise 6: Solution (continued)

- Objective function:

$$f(\mathbf{x}) = \mathbf{x}^T\mathbf{p}$$

- Constraints:

$$g(\mathbf{x}) = \mathbf{x}^T\mathbf{w} - W$$

- With these definitions, the canonical problem can be written

$$\begin{array}{ll} \text{maximise} & f(\mathbf{x}) \\ \text{subject to} & g(\mathbf{x}) \leq 0 \end{array}$$

Exercise 6: Solution (continued)

An algorithm to compute the objective function $f(\mathbf{x})$.

Algorithm 1: Compute $\mathbf{x}^T\mathbf{p}$.

```
Input: $\mathbf{x} = (x_1, \ldots, x_N)$: design variables, $\mathbf{p} = (p_1, \ldots, p_N)$: profits
Output: $T$: value of $f(\mathbf{x})$
1 $T \leftarrow 0$;
2 for $i = 1, \ldots, N$ do
3     $T \leftarrow T + x_i p_i$
4 return $T$
```

Up next. . .

Advanced Material

Advanced Exercise 1

- In this exercise, we consider an extension to the knapsack problem, known as the *quadratic knapsack problem*, where pairs of items are also accounted for.
- We want to load a knapsack with items. There are N items that can be loaded, and item i has a weight w_i, and a profit p_i. Also, for all $i \neq j$, if both items i and j are loaded, we gain a further profit q_{ij} (we have $q_{ij} = q_{ji}$ for all $i \neq j$). We need to decide which items to load, so as to maximise the total profit of loaded items, while keeping the total weight at most W.
- Formalize this problem as a canonical optimisation problem. Identify the design variables, the objective function and the constraints.
- Hint: Use N design variables, each of which is either 0 or 1.

Advanced Exercise 1: Solution

There exist more than one equivalent formulations. Here is one:

- Let $x_1, \ldots, x_N$ be variables where x_i equals 1 if item i is loaded, and 0 if not. These are the design variables.
- Write $\mathbf{x} = (x_1, \ldots, x_N)$, $\mathbf{p} = (p_1, \ldots, p_N)$, and $\mathbf{w} = (w_1, \ldots, w_N)$. The formal optimisation problem is:

$$\begin{aligned} \text{maximise} \quad & \mathbf{x}^T \mathbf{p} + \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} x_i x_j q_{ij} \\ \text{subject to} \quad & \mathbf{x}^T \mathbf{w} \leq W \end{aligned}$$

- Let us find the objective function and the constraints so that they follow the canonical formulation.

Advanced Exercise 1: Solution (continued)

- Objective function:

$$f(\mathbf{x}) = \mathbf{x}^T\mathbf{p} + \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} x_i x_j q_{ij}$$

- Constraints:

$$g(\mathbf{x}) = \mathbf{x}^T\mathbf{w} - W$$

- With these definitions, the canonical problem can be written

$$\begin{array}{ll} \text{maximise} & f(\mathbf{x}) \\ \text{subject to} & g(\mathbf{x}) \leq 0 \end{array}$$

Advanced Exercise 1: Solution (continued)

- The objective function can also be written in compact form.
- Define $\mathbf{Q}$ to be an $N \times N$ symmetric matrix, whose i-th diagonal element equals p_i, for all i, and its ij-th off-diagonal element equals $q_{ij}/2$, for all $i \neq j$. That is

$$\mathbf{Q} := \begin{bmatrix} p_1 & q_{12}/2 & q_{13}/2 & \cdots & q_{1N}/2 \\ q_{21}/2 & p_2 & q_{23}/2 & \cdots & q_{2N}/2 \\ q_{31}/2 & q_{32}/2 & p_3 & \cdots & q_{3N}/2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ q_{N1}/2 & q_{N2}/2 & q_{N3}/2 & \cdots & p_N \end{bmatrix}.$$

- The objective function will then be

$$f(\mathbf{x}) = \mathbf{x}^T \mathbf{Q} \mathbf{x}.$$

- The expression $\mathbf{x}^T \mathbf{Q} \mathbf{x}$ is called *quadratic form*.

Advanced Exercise 2

Consider the routing problem formulation, and in particular the formulation of the explicit constraint $h(\mathbf{x})$. It is possible to change the formulation of this constraint so that the constraint function will not just check whether the explicit constraint is or is not violated, but will also count the number of non-existent direct paths used by the solution. So, the function would retrieve 0 if all direct paths used by the solution are between neighbouring cities, and it would retrieve the number inexistent direct paths otherwise. Create this function.

Advanced Exercise 2 (continued)

PS: For now, you may create it either in the form of pseudocode for the purpose of this exercise, or in the form of mathematical equations, as an aid to your learning of the topic. However, later on in assessments, you may be required to understand and write mathematical equations.
PS: You will see later on in the module that counting the number of non-existent direct paths used by the solution can be useful when we use optimisation algorithms to solve optimisation problems with constraints.

Advanced Exercise 2: Solution

In pseudocode, the function could be written as:

Algorithm 2: Violate neighbour Constraint

```
Input: $\mathbf{x}$, $a$, $b$, $D$
Output: $num$
$num \leftarrow 0$;
if $D_{a,x_1} = -1$ then
    $num \leftarrow num + 1$
if $D_{x_{size(\mathbf{x})},b} = -1$ then
    $num \leftarrow num + 1$
for $i = 1$ to $size(\mathbf{x}) - 1$ do
    if $D_{x_i,x_{i+1}} = -1$ then
        $num \leftarrow num + 1$
return $num$
```

Advanced Exercise 2: Solution (continued)

As an equation, it could be written as:

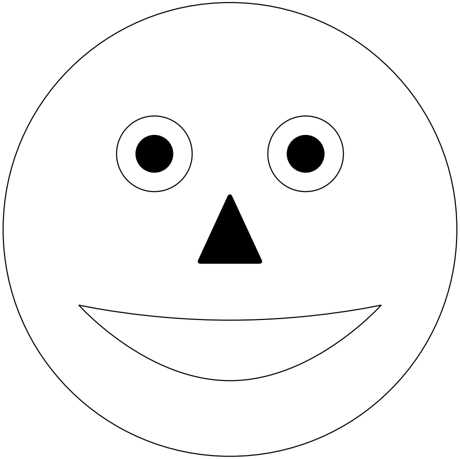
$$h(\mathbf{x}) = \mathbf{1}(D_{a,x_1} = -1) + \mathbf{1}(D_{x_{size(\mathbf{x})},b} = -1) + \sum_{i=1}^{size(\mathbf{x})-1} \mathbf{1}(D_{x_i,x_{i+1}} = -1)$$

where $\mathbf{1}(a)$ is a function called "indicator" function that returns 1 if the condition a is satisfied and 0 otherwise. You can see this function as "indicating" whether the condition is true or false by outputting 1 or 0, respectively.

Q&A

Any questions?

Until the next time. . .



Thank you for your attention!