

## Tutorial 4 in Cognitive Systems

In this tutorial we will consider several searching algorithms.

The algorithms are given in the appendix of the working sheet. Please, work through the corresponding algorithms and answering the questions.

1. After reading of the algorithm's pseudocode explain the following things:
  - (a) What do you think how the algorithms are working?
  - (b) What are special properties of the searching methods?
  - (c) Give possible limitations.
2. We consider several objects with the following characteristics.
  - size = {small, huge}
  - colour = {red, white, blue}
  - form = {ball, pyramid, cube}

Calculate the results using a graphical representation!

- (a) Run the *General to Specific* algorithm with negative:  $obj(\text{small}, \text{red}, \text{pyramid})$ ,  $obj(\text{huge}, \text{blue}, \text{cube})$  and positive:  $obj(\text{huge}, \text{white}, \text{ball})$ ,  $obj(\text{small}, \text{blue}, \text{ball})$ .
  - (b) Apply the *Specific to General* with positive:  $obj(\text{small}, \text{red}, \text{ball})$ ,  $obj(\text{small}, \text{white}, \text{ball})$ , and  $obj(\text{huge}, \text{blue}, \text{ball})$ .
  - (c) Run *Candidate Elimination* with positive:  $obj(\text{small}, \text{red}, \text{ball})$ ,  $obj(\text{huge}, \text{red}, \text{ball})$  and negative:  $obj(\text{small}, \text{blue}, \text{ball})$ ,  $obj(\text{huge}, \text{red}, \text{cube})$ .
3. We have a directed graph which is given in Figure 1.  
Run Dijkstra's algorithm using
    - (a) vertex a
    - (b) vertex e

as source node. Visualise the calculation steps!

## Literature

- Artificial Intelligence  
Patrick Henry Winston  
Addison-Wesley, ISBN: 0-201-60086-2
- Artificial Intelligence - A Modern Approach  
Stuart J. Russel and Peter Norvig  
Prentice-Hall, ISBN: 0-13-360124-2
- Artificial Intelligence - Structures and Strategies for Complex Problem Solving  
George F. Luger and William A. Stubblefield  
Benjamin/Cummings Publishing Company, ISBN: 0-8053-4780-1

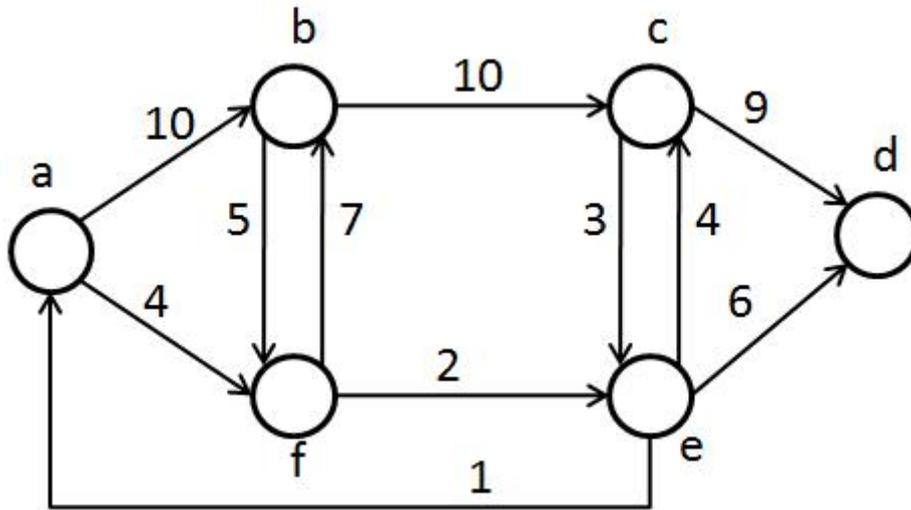


Figure 1: Directed graph (Dijkstra algorithm)

## A Version-Space-Search

### A.1 Specific to General

This searching algorithm maintains  $S$ , a set of hypotheses. It generates the maximally specific generalisation of a given training data set. This avoids overgeneralisation. The corresponding pseudo-code implementation is given in Figure 2.

```

Begin
Initialize  $S$  to the first positive training instance;
 $N$  is the set of all negative instances seen so far;

For each positive instance  $p$ 
  Begin
  For every  $s \in S$ , if  $s$  does not match  $p$ , replace  $s$  with its most specific
  generalizations that match  $p$ ;
  Delete from  $S$  all hypotheses more general than some other hypothesis in  $S$ ;
  Delete from  $S$  all hypotheses that match a previously observed negative
  instance in  $N$ ;
  End;

For every negative instance  $n$ 
  Begin
  Delete all members of  $S$  that match  $n$ ;
  Add  $n$  to  $N$  to check future hypotheses for overgeneralization;
  End;
End

```

Figure 2: Specific to General algorithm

## A.2 General to Specific

This algorithm generates a maximally general concept. For this, it maintains a set  $G$  of these hypotheses.

The corresponding programme is given in Figure 3

```
Begin
Initialize G to contain the most general concept in the space;
P contains all positive examples seen so far;

For each negative instance n
  Begin
  For each  $g \in G$  that matches n, replace g with its most general specializations
  that do not match n;
  Delete from G all hypotheses more specific than some other hypothesis in G;
  Delete from G all hypotheses that fail to match some positive example in P;
  End;

For each positive instance p
  Begin
  Delete from G all hypotheses that fail to match p;
  Add p to P;
  End;
End
```

Figure 3: General to Specific algorithm

## A.3 Candidate Elimination

If we combine the two previous programmes we get the Candidate Elimination algorithm which pseudo-code is presented in Figure 4.

## B Dijkstra Algorithm

The Dijkstra algorithm was developed by Edsger W. Dijkstra. It calculates the shortest path in a graph whose edges are weighted.

The algorithm in a pseudo-code implementation is given in Figure 5.

```

Begin
Initialize G to be the most general concept in the space;
Initialize S to the first positive training instance;

For each new positive instance p
  Begin
  Delete all members of G that fail to match p;
  For every  $s \in S$ , if s does not match p, replace s with its most specific
    generalizations that match p;
  Delete from S any hypothesis more general than some other hypothesis in S;
  Delete from S any hypothesis not more specific than some hypothesis in G;
  End;

For each new negative instance n
  Begin
  Delete all members of S that match n;
  For each  $g \in G$  that matches n, replace g with its most general specializations
    that do not match n;
  Delete from G any hypothesis more specific than some other hypothesis in G;
  Delete from G any hypothesis more specific than some hypothesis in S;
  End;

If  $G = S$  and both are singletons, then the algorithm has found a single concept that
is consistent with all the data and the algorithm halts;
If G and S become empty, then there is no concept that covers all positive instances
and none of the negative instances;
End

```

Figure 4: Candidate Elimination algorithm

```

DIJKSTRA( $G, w, s$ )
1 INITIALIZE-SINGLE-SOURCE( $G, s$ )
2  $S \leftarrow \emptyset$ 
3  $Q \leftarrow V[G]$ 
4 while  $Q \neq \emptyset$ 
5   do  $u \leftarrow \text{EXTRACT-MIN}(Q)$ 
6      $S \leftarrow S \cup \{u\}$ 
7     for each vertex  $v \in \text{Adj}[u]$ 
8       do RELAX( $u, v, w$ )

```

Figure 5: Dijkstra algorithm