City University London

Module H800: Symbolic Artificial Intelligence

Component Guide: KBS Architectures and Modelling

Andrew Tuson (email: andrewt@soi.city.ac.uk)

Contents

1 Component Synopsis **2**
1.1 Learning Outcomes 2
1.1.1 Blackboard Systems, Frames and Semantic Nets 2
1.1.2 Case-Based Reasoning 2
1.1.3 Knowledge Modelling 2
1.2 Component Materials 3

2 Component Tutorials **4**
2.1 Tutorial 1: Frames and Semantic Networks 4
2.1.1 Frames (20 mins) 4
2.1.2 Semantic Nets (30 mins) 4
2.2 Tutorial 2: Case-Based Reasoning 5
2.2.1 Design Issues (35 mins) 5
2.2.2 CBR vs Rule-Based Systems (15 mins) 5

A Tutor's Notes **6**
A.1 Tutorial 1: Frames and Semantic Nets 6
A.1.1 Frames 6
A.1.2 Semantic Net Operations 6
A.2 Tutorial 2: Case-Based Reasoning 7
A.2.1 Design Decisions 7
A.2.2 CBR vs Rule-Based Systems 7

Chapter 1

Component Synopsis

Lecturer: Andrew Tuson

This component will aim to give the student an introduction to a number of additional knowledge representation issues: frames, semantic nets, and blackboard systems. After this the case-based reasoning architecture will be introduced as an alternative to production systems. Finally, this component will aim to give the student an introduction to knowledge modelling and its role in KBS design and the techniques available. This component consists of four lectures and two supporting tutorials

1.1 Learning Outcomes

The learning outcomes from this component will result in the student being able to accomplish the following:

1.1.1 Blackboard Systems, Frames and Semantic Nets

- Describe the blackboard system, its architecture, and its application;
- Describe frames, their components, their use as a KR method, and their (dis)advantages;
- Describe the idea of a semantic net approach to KR, how they differ from production rules, and their (dis)advantages.
- Describe and apply the conceptual graph method with respect to both KR and inference.

1.1.2 Case-Based Reasoning

- Describe the rationale behind Case-Based Reasoning (CBR).
- Describe the components of the CBR architecture.
- Describe a number of CBR design options, and their relative merits.
- Be aware of (dis)advantages of CBR compared with rule-based systems, and which should be used for a given problem.
- Be aware of the type of CBR applications, with illustrative examples, and be able to suggest CBR configo-rations for simple problems.

1.1.3 Knowledge Modelling

- Understand and discuss the view of knowledge as both a transferrable medium and codifed experience.
- Understand the need to model knowledge in KBS design, and how this can aid knowledge acquisiton.
- Describe what ontologies and problem-solving methods are and explain why they can be useful.

- Understand the aims of the CommonKADS methodology, and the advantages of a modelling-base approach to KBS design.
- Describe the models that make up CommonKADS, their function, and how they are related to each other.
- Outline the formalisms by which CommonKADS models expert knowledge.
- Describe the PragmaticKADS variant of this methodology, and why it is useful.

1.2 Component Materials

The material on frames, semantic nets, and blackboard systems can be found in most AI texts (Cawsey gives a decent introduction, and Luger and Stubblefied is especially good for details). Additional handouts on CBR and CommonKADS will be given out in the lecture. In addition, the slides will be given out at the start of the lecture. The exact coverage required for the exam will be, of course, defined by the lecture (so turn up!).

Chapter 2

Component Tutorials

2.1 Tutorial 1: Frames and Semantic Networks

CBR will be covered in next week's tutorial so could you please leave any questions till then, or if there is time at the end of this tutorial.

2.1.1 Frames (20 mins)

Consider the following frame for a car:

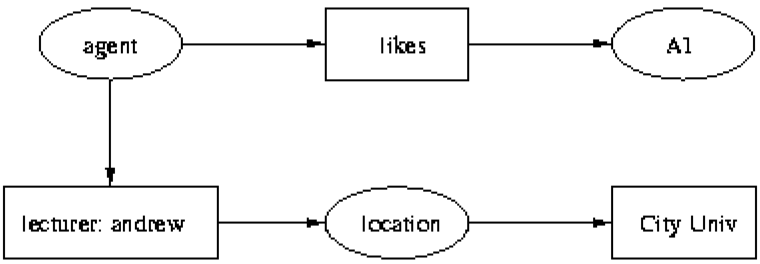
Frame	Car
Manufacturer	Ford
Wheels	4
Colour	Blue
Engine	Petrol

Construct a corresponding frame for a tank (extra slots may be required), and use this to demonstrate the multiple inheritance problem in frames by inheriting from tank and car to produce an armoured car.

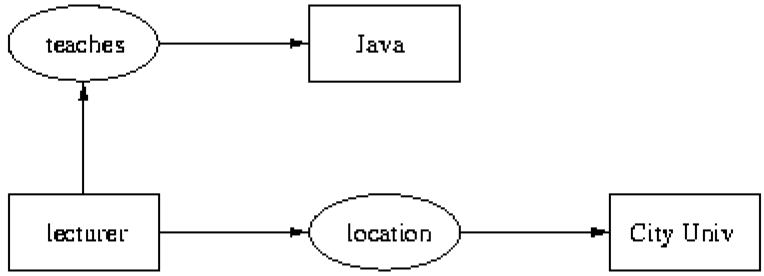
2.1.2 Semantic Nets (30 mins)

This part of the tutorial aims to get you acquainted with the mechanics of using semantic nets (conceptual graphs). Consider the two semantic nets below and perform the following operations on them:

1. **Generalise** net 1 to get net 1a.
2. **Join** the generalised net 1 (1a) with net 2 to make net 3.
3. **Simplify** net 3 to obtain net 4.



Net 1 is above, and net 2 is below.



2.2 Tutorial 2: Case-Based Reasoning

Note this was adapted from part of the Jan 1999 H800 exam paper

2.2.1 Design Issues (35 mins)

Consider what factors would matter in deciding to give someone credit. Assume that the bank has already built a database of previous applications along these lines. How would you implement a CBR system for the above problem? Make explicit reference to each of the components of the CBR architecture.

2.2.2 CBR vs Rule-Based Systems (15 mins)

Give three criteria for deciding whether to use a rule-based or CBR system, With these criteria in mind, along with any others you feel applicable, choose which of the two above approaches you would adopt and why for the problem above and why.

Appendix A

Tutor's Notes

A.1 Tutorial 1: Frames and Semantic Nets

I'll just put outline answers down here. Luger and Stubblefield give a good description of what you need to do these kind of exercises.

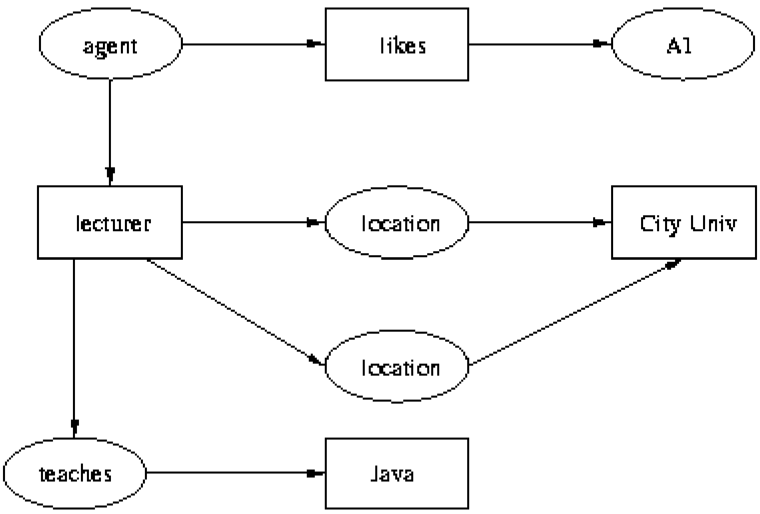
A.1.1 Frames

In this case, any reasonable frame definition for a tank will suffice. However, tanks would have the WHEELS solt filled by 'tracks', and the ENGINE solt filled by 'diesel'. In this case which definition do you use - the one for a car or the one for a tank?

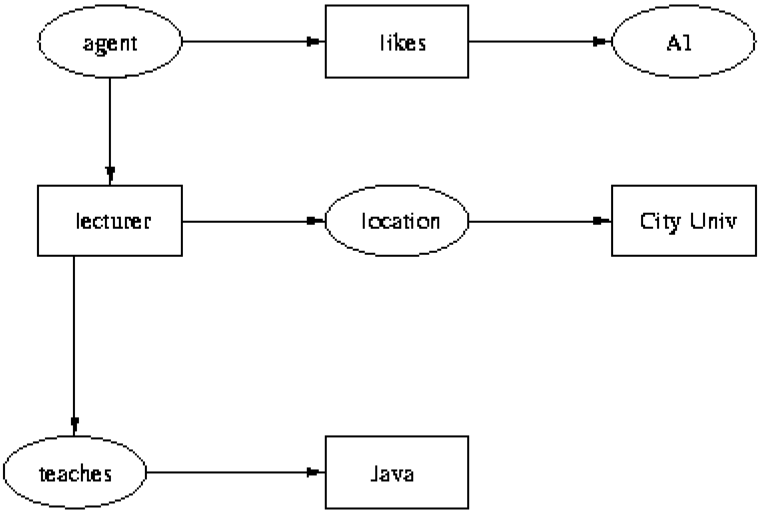
A.1.2 Semantic Net Operations

The generalised version of net 1 (net 1a) replaces 'agent:andrew' with 'agent' — in other words we are not talking about a specific agent.

The result of combining net 1a with net 2 is is given below (net 3):



Of course there is some redundancy here so this can be simplified to give net 4 below:



A.2 Tutorial 2: Case-Based Reasoning

Note this was adapted from part of the Jan 1999 H800 exam paper

A.2.1 Design Decisions

An outline of the model for CBR is as follows. The students need to address design decisions for each of indexing, retrieval, adaptation, and where to source a suitable database of cases. These decisions need to be explicitly related to the above example problem for full marks. As decision tree can be derived from a database of past loan cases, and it should be possible to just use this database (assuming of course that it is available). It would also be reasonable to assume that the data would be well warehoused with inconsistencies, etc. removed (otherwise rule induction will obtain a good-quality decision tree).

A decision tree will also be of use for constructing the indexing scheme. If the indexing scheme is attribute based, the attributes in the tree and their position should indicate which attributes are of importance and their relative usefulness. However, the indexing scheme can be based around a decision tree to directly organise the cases (this is usually more efficient). Adaptation would be unnecessary in this case as the solution is merely an accept/reject answer.

A.2.2 CBR vs Rule-Based Systems

The first part is really bookwork - there are a number of reasons why CBR may be preferred (though the Watson CBR book, pp.46-48, is the best source for a discussion of the pros and cons). Three examples are given below:

- Ease of knowledge acquisition when rules are not easily available.
- Ease of updating - introducing new cases is analogous to learning and can increase the system's capabilities (c.f. CAVIER).
- In some situations, explanation in terms of a previous case is more understandable than a procedural description of how the solution was arrived at.

In the remainder for the question the aim here is not to give a "right" answer but instead to relate and apply the criteria to the problem at hand and use them to justify a conclusion. The student should note that there are no clear-cut answers and instead should suggest questions that need to be asked to decide the issue.