Introduction to Cognitive Science

Assignment 1: Perception of Scenes

Due Tuesday Oct 13th, 2009, hardcopy, in class

1 Junction Labels

Assume the following set of Huffman/Clowes junction labels: (Recall that a

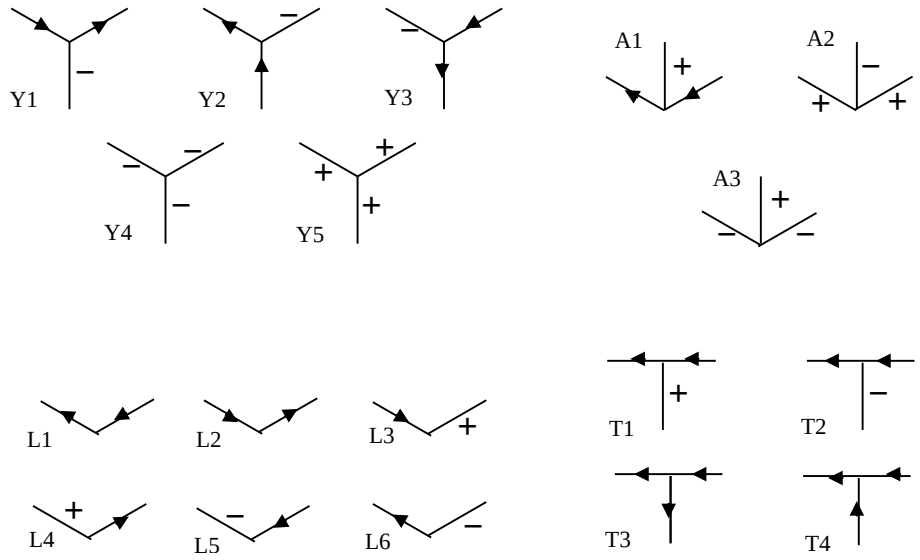


Figure 1: Huffman/Clowes Junction Labels

junction label consists of an assignment of one of the symbols $+$, $-$, $\rightarrow$ and $\leftarrow$ to each of the junction's arcs, respecively denoting convex, concave, and occulting edges.

The labels Y1 are included to offer a standard way of abbreviating the junction description: as you do this assignment, you can refer to a junction label either by drawing the associated diagram, or by providing the subscript given with it. Do whichever is more comfortable for you.

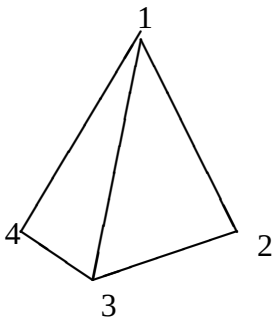
2 AC-3 Algorithm

The following is the definition of the AC-3 algorithm. (A queue is a list such that you always remove things at the left-hand end and add them at the right hand end.)

- Make a queue of all the arcs (i, j) in the picture and iterate the following procedure until the queue is empty:
  - Remove an arc (i, j) from the queue.
  - For each label l on junction i, if there is no label on j which is arc-consistent with l, remove l from i.
  - If any such label is removed from junction i, then for all arcs (k, i) in the picture except (j, i), put (k, i) on the queue if it isn't there already.

3 Example

To see how *AC-3* works, consider the following picture from the notes (Pyramid).

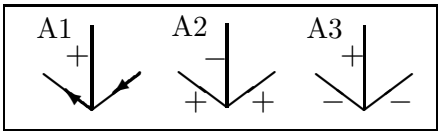


We will assume that the initial queue has the arcs in the following order:

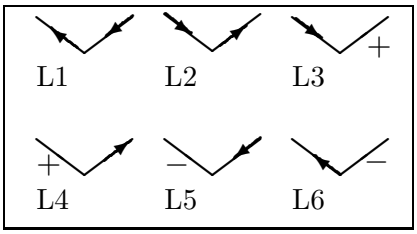
(1) $[(1,2),(2,1),(2,3),(3,2),(3,4),(4,3),(4,1),(1,4),(1,3),(3,1)]$

The first arc removed is $(1,2)$. Inspection of the labels in the figure corresponding to the indices will show that all labels on 1 are consistent with some label on 2 as shown below, (although the converse does not apply).

Labels on 1:



Labels on 2:

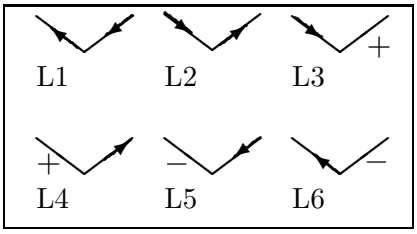


No arcs of the form $(k,1)$ are considered for adding to the queue. So the next state of the computation can be thought of in terms of the same label set as we started and the following reduced queue:

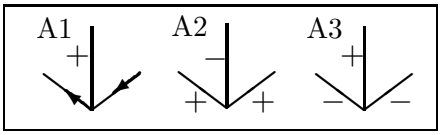
(2) $[(2,1),(2,3),(3,2),(3,4),(4,3)(4,1),(1,4),(1,3),(3,1)]$

The next arc examined is $(2,1)$. This time, two of the labels on junction 2 are not arc-consistent with any label on 1, namely L2 and L4.

Labels on 2:



Labels on 1:

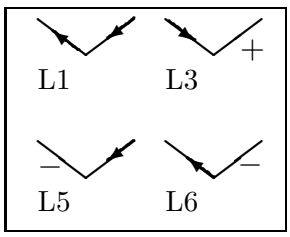


All arcs of the form $(k,2)$ except for $(1,2)$ are therefore considered for addition to the queue. However, they are all already present, so the new queue is simply the following:

(3) $[(2,3),(3,2),(3,4),(4,3)(4,1),(1,4),(1,3),(3,1)]$

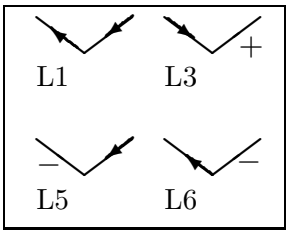
The set of labels on the picture is reduced as follows:

New Labels on 2:

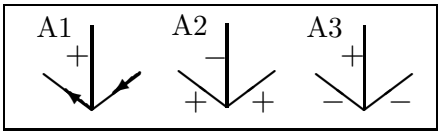


The next arc examined is $(2,3)$.

Labels on 2:

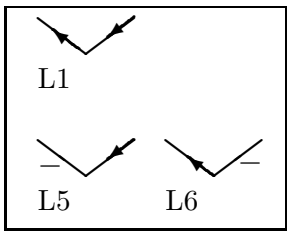


Labels on 3:



Another label on 2 is not arc-consistent with any label on 3, namely L3, so the labels are as follows:

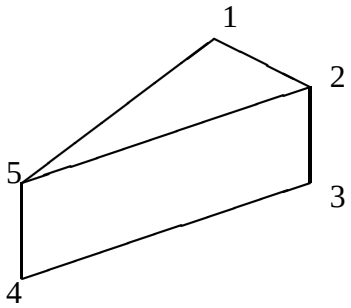
New Labels on 2:



There are several more steps shown in the course slides.

4 What you Have to Do:

1. This exercise asks you to either simulate by hand or implement in your favourite programming language, the algorithm *AC-3* presented in this handout. Show all states that the algorithm goes through in computing the labelling for the following example (cheese).



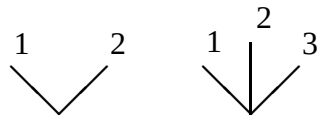
Use one of the following initial queues based on the last digit of your University of Edinburgh student number:

#	Initial Queue
0	$[(1,2),(1,5),(2,3),(2,5),(2,1),(3,4),(3,2),(4,5),(4,3),(5,1),(5,2),(5,4)]$
1	$[(5,4),(1,2),(1,5),(2,3),(2,5),(2,1),(3,4),(3,2),(4,5),(4,3),(5,1),(5,2)]$
2	$[(5,2),(5,4),(1,2),(1,5),(2,3),(2,5),(2,1),(3,4),(3,2),(4,5),(4,3),(5,1)]$
3	$[(5,1),(5,2),(5,4),(1,2),(1,5),(2,3),(2,5),(2,1),(3,4),(3,2),(4,5),(4,3)]$
4	$[(4,3),(5,1),(5,2),(5,4),(1,2),(1,5),(2,3),(2,5),(2,1),(3,4),(3,2),(4,5)]$
5	$[(4,5),(4,3),(5,1),(5,2),(5,4),(1,2),(1,5),(2,3),(2,5),(2,1),(3,4),(3,2)]$
6	$[(3,2),(4,5),(4,3),(5,1),(5,2),(5,4),(1,2),(1,5),(2,3),(2,5),(2,1),(3,4)]$
7	$[(3,4),(3,2),(4,5),(4,3),(5,1),(5,2),(5,4),(1,2),(1,5),(2,3),(2,5),(2,1)]$
8	$[(2,1),(3,4),(3,2),(4,5),(4,3),(5,1),(5,2),(5,4),(1,2),(1,5),(2,3),(2,5)]$
9	$[(2,5),(2,1),(3,4),(3,2),(4,5),(4,3),(5,1),(5,2),(5,4),(1,2),(1,5),(2,3)]$

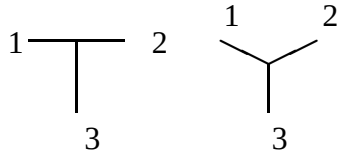
Use the notation for the labelled graph and the queue from the discussion in this assignment and/or the paper, and making the same assumption that items are removed from the front, and added to the back, of the queue.

If you are writing a program, you will want to represent the arcs and junctions in the picture as data structures (such as Lisp lists, Prolog facts, C arrays, etc). Since junctions may be rotated in a picture, you will need an ordering convention for associating list items, array elements etc. with particular neighbouring junctions. (If you are hand-simulating, you don't need to think about this.) One such convention is the following:

A(rrow) junctions and (el)L-junctions are mapped onto their neighbours in the picture by a "clockwise point-down convention", relative to the following normalised orientation for the junctions:



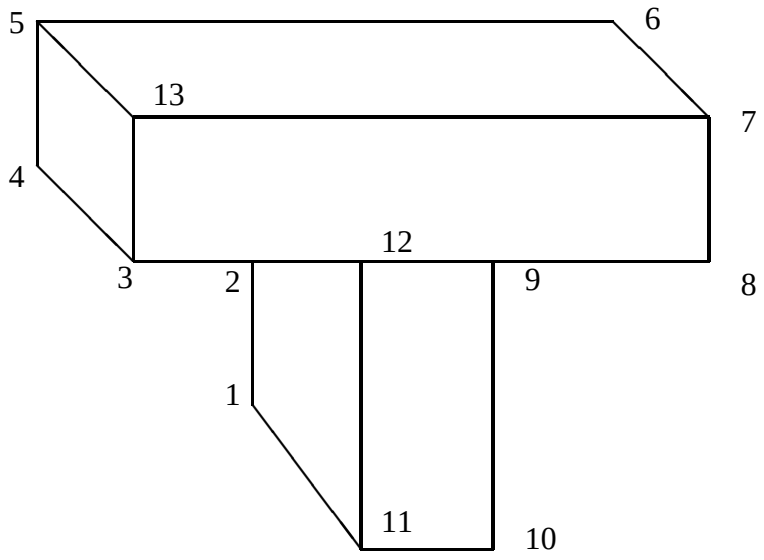
T-junctions and Y-junctions are mapped by a "clockwise right-way-up convention", relative to the following normalised orientations:



You can then think of each junction in the picture as a pair including an index number, and a list of junction labels (which changes). Each junction label for a junction i is a list of triples (i, j, l) where j is a neighbouring junction and l is a line-label. In most programming languages, a convenient notation for line-labels is to use `plus, minus, in,` and `out` for the four labels, where `in` and `out` represent direction of the arrow with respect to a junction, and where two arcs (i, j) and (j, i) are defined to be consistent if both are labelled `plus`, or both are `minus`, or one is `in` and the other `out`.

The picture also determines the set of arcs between neighbours i, j that constitutes the initial queue. If you want your program to be fully general you should make the arcs into *triples* (i, j, ac) and (j, i, ac), where *ac* is the predicate or test for arc-consistency defined at (3) on page 18 in the paper. In the line labelling task, all arcs are subject to the same constraint, but in general the arcs in a constraint satisfaction problem may be associated with different constraints.

2. Assume that all lines on the periphery of the following figure (BigT) are labelled as obscuring edges, so that all peripheral junctions have only the label that is consistent with that assumption.



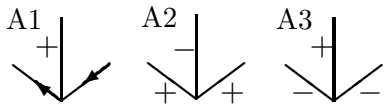
Compute (or guess) the final labelling produced by *AC-3* in application to

the example. (You do **not** have to show a trace of the algorithm.) How many distinct interpretations does the labelling permit?

3. Now assume that the lines on the periphery of the same figure *BigT* can have any labeling that is consistent with the junctions involved. Compute (or guess) the final labelling produced by *AC-3* in application to the figure, *BigT*.

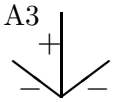
   Notice that the figure is ambiguous under this assumption, and it is quite hard to tell how many distinct interpretations there are.

   Your result should have left junction 11 ambiguous between the following three labels.

   You will probably find one of these junction labels surprising, but all three are possible. (If you didn't get them all, you should check your computation for other mistakes).

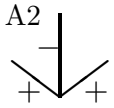
   (a) First assume that junction 11 is of type

   (a1) How many interpretations of the whole figure are compatible with this assumption? Say in English what the interpretation(s) say the figure represents in 3-D.

   (a2) Are they all equally likely to be correct if this is a representation of a real scene? If not why not?

   (b) Now assume that junction 11 is of type

   (b1) How many interpretations of the whole figure are compatible with *this* assumption? Say in English what the interpretation say the figure represents in 3-D.

(b2) Speculate concerning the reason(s) why none of these interpretations are among the ones that human viewers naturally come up with for this figure.

(b3) How could the program or simulation be made more human-like in this respect?

Note: Write your name, student number, and email address at the top of your homework. If you have multiple pages, staple them together. Credit will be given for presentation and completeness of information.