Wintersemester 2006/2007 Bernd Neumann

Übungen zur Vorlesung: Wissensbasierte Systeme

Blatt 3

Exercise 3.1:
There are at least two ways to represent the crossword puzzle shown below as a constraint satisfaction problem.

The first is to represent the word positions (A1, A2, A3, D1, D2, and D3) as variables, with the set words as possible values. The constraints are that where the words intersect the letter is the same.

The second is to represent the nine squares as variables. The domain of each variable is the set of letters of the alphabet, {a, b, ..., z}. The constraints are that there is a word in the word list that contains the corresponding letters. For example, the top-left square and the center-top square cannot both have the value a, because there is no word starting with aa.

(a) Give an example of pruning due to domain consistency, using the first representation (if one exists).

(b) Give an example of pruning due to arc consistency, using the first representation (if one exists).

(c) Are domain consistency plus arc consistency adequate to solve this problem using the first representation? Explain.

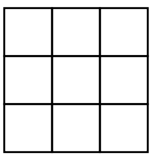
(d) Give an example of pruning due to domain consistency, using the second representation (if one exists).

(e) Give an example of pruning due to arc consistency, using the second representation (if one exists).

(f) Are domain consistency plus arc consistency adequate to solve this problem using the second representation?

(g) Which representation leads to a more efficient solution using consistency-based techniques? Give the evidence on which you are basing your answer.

Word list:
add, ado, age, ago, aid, ail, aim, air, and, any, ape, apt, arc, are, ark, arm, art, ash, ask, auk, awe, awl, aye, bad, bag, ban, bat, bee, boa, ear, eel, eft, far, fat, fit, lee, oaf, rat, tar, tie.



Exercise 3.2:

Tic-tac-toe is a game played on a 3 x 3 grid. Two players, X and O, alternately place their mark in an unoccupied position. X wins, if, at its turn, it can place an X in an unoccupied position to make three X's in a row. For example, from the state of the game

```
X O O
  X
X   O
```

X can win by moving into the left middle position.

Fred, Jane, Harold, and Jennifer have all written programs to determine if, given a state of the game, X can win in the next move. Each of them has decided on a different representation of the state of the game. The aim of this exercise is to compare their representations.

Fred decided to represent a state of the game as a list of three rows, where each row was a list containing three elements, either x, o, or b (for blank). Fred represents the above state as the list

[[x, o, o], [b, x, b], [x, b, o]].

Jane decided that each position on the square could be described by two numbers, the position across and the position up. The top left X is in position pos(1, 3), the bottom left X is in position pos(1, 1), and so forth. She then decided to represent the state of the game as a pair ttt(XPs, OPs) where XPs is the list of X's positions and OPs is the list of O's positions. Thus Jane represented the above state as

ttt([pos(1, 3), pos(2, 2), pos(1, 1)], [pos(2, 3), pos(3, 3), pos(3, 1)]).

Harold and Jennifer both realized that the position on the tac-tac-toe board could be represented in terms of a so-called magic square:

```
6 7 2
1 5 9
8 3 4
```

Based on this representation, the game is transformed into one where the two players alternately select a digit. No digit can be selected twice, and the player who first selects three digits summing to 15 wins.

Harold decides to represent a state of the game as a list of nine elements, each of which is x, o, or b, depending on whether the corresponding position in the magic square is controlled by X, controlled by O, or is blank. Thus Harold represents the game state above as the list:

[b, o, b, o, x, x, o, x, b].

Jennifer decides to represent the game as a pair consisting of the list of digits selected by X and the list of digits selected by O. She represented the state of the game above as

magic([6, 5, 8], [7, 2, 4]).

(a) For each of the four representations, write the relation x_can_win(State), with the intended interpretation that X can win in the next move from the state of the game State.

(b) Which representation is easier to understand? Explain why.

(c) Which representation is more efficient to determine a win?

(d) Which representation results in the simplest algorithm to make sure a player doesn't lose on the next move, if such a move exists?

(e) Which do you think is the best representation? Explain why. Can you suggest a better representation?