National University of Singapore
School of Computing
CS3243 Introduction to AI

Tutorial 5: Constraint Satisfaction Problems

Issued: Week 6 Discussion in: Week 7

Important Instructions:

- **Assignment 5** consists of **Question 3** from this tutorial.
- Your solution(s) must be TYPE-WRITTEN, though diagrams may be hand-drawn.
- You are to submit your solution(s) during your **Tutorial Session in Week 7**.

Note: you may discuss the assignment question(s) with your classmates, but you must work out and write up your solution individually. Solutions that are plagiarised will be heavily penaltised.

---

1. Consider the 4-queens problem on a 4×4 chess board. Suppose the leftmost column is column 1, and the topmost row is row 1. Let Q_i denote the row number of the queen in column i, $i = 1, 2, 3, 4$. Assume that variables are assigned in the order Q_1, Q_2, Q_3, Q_4, and the domain values of Q_i are tried in the order $1, 2, 3, 4$. Show a trace of the backtracking algorithm with forward checking to solve the 4-queens problem.

2. You are in charge of scheduling for computer science classes that take place on Fridays. There are 5 classes on that day, and 3 professors who will be teaching these classes. You are constrained by the fact that each professor can only teach one class at a time.

   The classes are:

   - C_1 - Programming Methodology: 8.00am to 9.00am
   - C_2 - Discrete Structures: 8.30am to 9.30am
   - C_3 - Data Structures and Algorithms: 9.00am to 10.00am
   - C_4 - Introduction to Artificial Intelligence: 9.00am to 10.00am
   - C_5 - Machine Learning: 9.30am to 10.30am

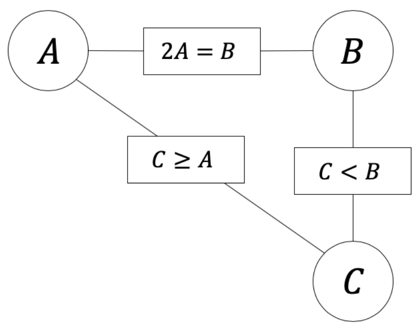
   The professors are:

- Professor Tess, who is available to teach classes C_3 and C_4.
- Professor Jill, who is available to teach classes C_2, C_3, C_4, and C_5.
- Professor Bell, who is available to teach classes C_1, C_2, C_3, C_4, and C_5.

(a) Formulate this as a CSP with each class being a variable, stating the effective domains and constraints. (For example, since C_1 and C_2 cannot be taught by the same professor, you may denote this constraint as $C_1 \neq C_2$).

(b) Specify one solution to this CSP.

3. Consider the following constraint graph.



(a) Specify the resultant domains for each variable after the application of the AC-3 algorithm on the given constraint graph.

Assume that initially, the domain of each variable is $D_A = D_B = D_C = \{1, 2, 3, 4\}$.

Further, assume that the initial queue for the AC-3 algorithm is:
$(A, B), (B, A), (B, C), (C, B), (C, A), (A, C)$, where (A, B) is at the head of the queue.

(b) With reference to the previous question, provide a valid assignment of values to A, B, and C such that the constraints are satisfied, and $A + B + C$ is minimum.

4. Consider the *item allocation problem*. We have a group of people $N = \{1, \dots, n\}$, and a group of items $G = \{g_1, \dots, g_m\}$. Each person $i \in N$ has a utility function $u_i : G \to \mathbb{R}_+$. The constraint is that every person is assigned *at most one item*, and each item is assigned to *at most one person*. An allocation simply says which person gets which item (if any).

   In what follows, you *must* use *only* the binary variables $x_{i,j} \in \{0, 1\}$, where $x_{i,j} = 1$ if person i receives the good g_j, and is 0 otherwise.

   (a) Write out the constraints: 'each person receives no more than item' and 'each item goes to at most one person', using only the $x_{i,j}$ variables[1].

   (b) Suppose that people are divided into *disjoint types* $N_1, \dots, N_k$ (think of, say, genders or ethnicities), and items are divided into *disjoint blocks* $G_1, \dots, G_\ell$. We further require that each N_p only be allowed to take no more than λ_{pq} items from block G_q. Write out this constraint using the $x_{i,j}$ variables. (Note that each N_i corresponds to the set of people who are of that person type.)

   (c) We say that player i *envies* player i' if the utility that player i has from their assigned item is strictly lower than the utility that player i has from the item assigned to player i'. Write out the constraints that ensure that in the allocation, no player envies any other player. You may assume that the validity constraints from (a) hold.

[1] You may use simple algebraic functions $-, +, \times, \div$, and numbers