CS3243 Introduction to Artificial Intelligence

AY2021/2022 Semester 1

Tutorial 5: Constraint Satisfaction Problems

Important Instructions:

- **Assignment 5** consists of **Question 5** and **Question 6** from this tutorial.
- Your solutions for the above questions may be handwritten or typewritten, but handwritten solutions must be legible for marks to be awarded. If you are submitting handwritten solutions, please append the question paper in front of your solutions.
- You are to submit your solutions on LumiNUS by **Week 6 Monday (4 October), 2359 hours**, under **Tutorial Submission** → **Tutorial 5**.

Note: you may discuss the content of the questions with your classmates, but you must work out and write up your solution individually - solutions that are plagiarised will be heavily penalised.

Tutorial Questions

(1) Consider the following constraint satisfaction problem:

Variables:	X, Y, Z
Domains:	$D_X = D_Y = D_Z = \{0, 1, 2, 3, 4\}$
Constraints:	$C_1 := (X = Y + 1)$
	$C_2 := (Y = 2Z)$

Assuming that assign is empty and var $= Y$, perform inference on these constraints.

Solution: We start by popping Y from the queue and perform inference on the constraints of Y.

(a) Pop Y:

- From C_1, $X \neq 0$ is added to the inference.
- From C_2, $Z \neq \{3, 4\}$ is added to the inference.

Since both X's and Z's domains are updated, we add them to the queue (i.e. queue now contains $\{X, Z\}$).

(b) Pop X:

- From C_1, $Y \neq 4$ is added to the inference.

Since Y's domain is updated, we add Y to the queue (i.e. queue now contains $\{Z, Y\}$).

(c) Pop Z:

- From C_2, $Y \neq \{1, 3\}$ is added to the inference.

Since the queue contains Y, we do not add it back (i.e. queue now contains $\{Y\}$).

(d) Pop Y:

- From C_1, $X \neq \{2, 4\}$ is added to the inference.
- From C_2, $Z \neq 2$ is added to the inference.

Since both X's and Z's domains are updated. we add them to the queue (i.e. queue now contains $\{X, Z\}$).

(e) Pop X:

- No inference added (i.e. queue now contains $\{Z\}$).

(f) Pop Z:

- No inference added (i.e. queue is now empty).

From the inference above, we derive the following:

- $X \neq \{0, 2, 4\}, D_X = \{1, 3\}$
- $Y \neq \{1, 3, 4\}, D_Y = \{0, 2\}$
- $Z \neq \{2, 3, 4\}, D_Z = \{0, 1\}$

Note that of course not all of these values are compatible: we need to choose one of them, and then the remaining values will be chosen correspondingly (e.g. $X = 1 \implies Y = 0 \implies Z = 0$).

(2) Consider the following constraint satisfaction problem: there are four variables x_1, x_2, x_3, x_4 whose domain is $D = \{1, 2, 3\}$. They have the following constraints:

$$x_1 + x_2 \leq 3 \qquad x_1 + x_3 \geq 4$$
$$x_2 + x_3 \leq 3 \qquad x_3 + x_4 \leq 2$$

Apply the inference algorithm taught in lecture to this problem. Assume that assignment is initially empty, and var $= x_2$.

Solution:

Variable popped/inferred	Domain update	Queue
x_2	$x_1 = \{1, 2\}$ $x_2 = \{1, 2, 3\}$ $x_3 = \{1, 2\}$ $x_4 = \{1, 2, 3\}$	$\{x_1, x_3\}$
x_1	$x_1 = \{1, 2\}$ $x_2 = \{1, 2\}$ $x_3 = \{2\}$ $x_4 = \{1, 2, 3\}$	$\{x_3, x_2\}$
x_3	$x_1 = \{2\}$ $x_2 = \{1\}$ $x_3 = \{2\}$ $x_4 = \{\varnothing\}$	$\{x_2, x_1\}$

Since the domain of x_4 is empty, x_4 is unable to take up any values. Hence, the inference algorithm terminates and returns *failure*.

(3) Explain why it might be a good heuristic to choose the variable that is the *most constrained* but the value that is the *least constraining* in a CSP search.

Solution: The *most constrained variable* heuristic makes sense because it chooses a variable that is (of all other things being equal) likely to cause a failure, and it is more efficient to fail as early as possible (thereby reducing the search space). The *least constraining value* heuristic makes sense because it allows the most opportunities for future assignments to avoid conflict.

(4) Consider the *item allocation problem*. We have a group of people $N = \{1, \ldots, n\}$, and a group of items $G = \{g_1, \ldots, g_m\}$. 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:

(i) each person receives no more than one item

Solution:

$$\forall i \in N : \sum_{g_j \in G} x_{i,j} \leq 1$$

(ii) each item goes to at most one person

Solution:

$$\forall g_j \in G : \sum_{i \in N} x_{i,j} \leq 1$$

using only the variables $x_{i,j}$.[1]

(b) Suppose that people were divided into *disjoint types* $N_1, \ldots, N_k$ (e.g. say, genders or ethnicities), whereas items were divided into *disjoint blocks* $G_1, \ldots, G_l$. 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 variables $x_{i,j}$. (Note that each N_i corresponds to the set of people who are of that person-type.)

Solution:

$$\forall p \in \{1, ..., k\}, q \in \{1, ..., l\} : \sum_{i \in N_p} \sum_{g_j \in G_q} x_{i,j} \leq \lambda_{pq}$$

(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.

Solution: Note that for this constraint, the definition requires that the allocation is valid, so you will need to add the constraints from (a) to make either of the definitions below meaningful.

$$\forall i, i' \in N, \forall g_j, g_{j'} \in G : (x_{i,j} \wedge x_{i',j'}) \implies u_i(g_j) \geq u_i(g_{j'})$$

or

$$\forall i, i' \in N : \left(\left(\sum_{g_j \in G} x_{i,j} u_i(g_j) \right) > 0 \right) \implies \left(\left(\sum_{g_j \in G} x_{i,j} u_i(g_j) \right) \geq \left(\sum_{g_j \in G} x_{i',j} u_i(g_j) \right) \right)$$

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