CS410: Artificial Intelligence 2021 Fall
Homework 2: Constraint Satisfaction Problems
Due date: 23:59:59 (GMT +08:00), October 22 2021

1. **A* graph search**. Consider the following undirected graph shown in Figure **??** where we are searching from start state $A$ to goal state $G$. The number over each edge is the transition cost. Additionally, we are given a heuristic function $h$ as follows: $\{h(A) = 7, h(B) = 5, h(C) = 6, h(D) = 4, h(E) = 3, h(F) = 3, h(G) = 0\}$. Assume that, in case of ties, the search procedure uses an alphabetical order for tie-breaking.

   Find the sequence of nodes expanded by A* graph search algorithm, with problem-solving steps (i.e., updates for the frontier and explored set).

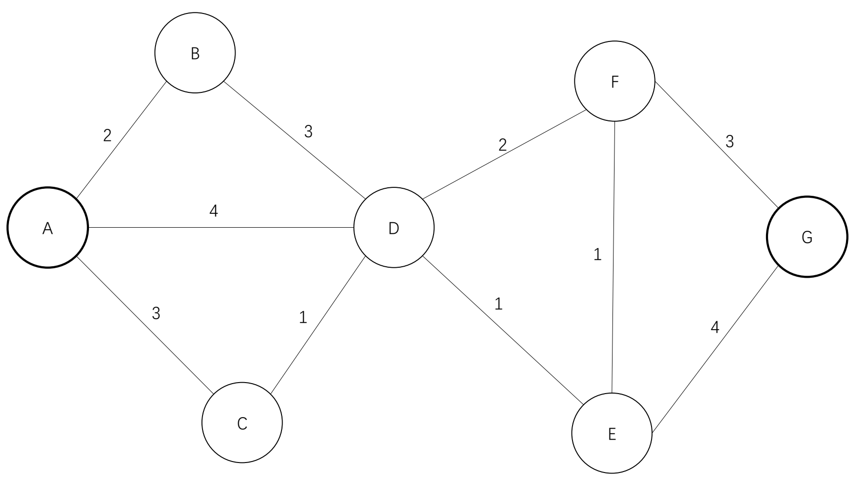



Figure 1: Problem 1.

**Solution.**
expanded: A frontier: {B(7),D(8),C(9)} explored:{ A}
expanded: B frontier: {D(8),C(9)} explored:{ A,B}
expanded: D frontier: {E(8),F(9),C(9)} explored:{A,B,D}
expanded: E frontier: {F(9),C(9),G(9)} explored:{A,B,D,E}
expanded: C frontier: {F(9),G(9)} explored:{A,B,C,D,E}
expanded: F frontier: {G(9)} explored:{A,B,C,D,E,F}

expanded: G frontier: {} explored:{A,B,C,D,E,F,G}

2. **CSP formulation**. Consider the following three problems:

   (a) Rectilinear floor-planning: find non-overlapping places in a large rectangle for a number of smaller rectangles.

   (b) Class scheduling: There is a fixed number of professors and classrooms, a list of classes to be offered, and a list of possible time slots for classes. Each professor has a set of classes that he or she can teach.

   (c) Hamiltonian tour: given a network of cities connected by roads, choose an order to visit all cities in a country without repeating any.

   Determine each as a planning problem or an identification problem and explain why. Give precise formulations for each problem as constraint satisfaction problems.

   Recall a CSP consists of three components, $X$, $D$, and $C$. Note that there are many valid formulations, but you only need to provide one.

   **Solution.**
   (a) It is an identification problem.
   $X$ = {the set of $X_i$, which means the rectangle $X_i$}
   $D_i$ = {the points $(a_i, b_i)$ as the center of the rectangle $X_i$}×{the inclination angle $k_i$}
   $C$ = {for each rectangle, all of it is in the large rectangle. And for each point $(a_i, b_i)$, $(a_j, b_j)$ , there is at most one rectangle containing this point.}

   (b) It is an identification problem.
   $X$ = {the set of $X_i$, which means the class $X_i$}
   $D_i$ = {the time $t_i$ }×{the classroom $c_i$}× {the professor $p_i$}
   $C$ = {for each pair of time and classroom $(t_i, c_i)$, there is at most one class at this pair. And for each pair of time and professor$(t_i, p_i)$, there is at most one class at this pair.}

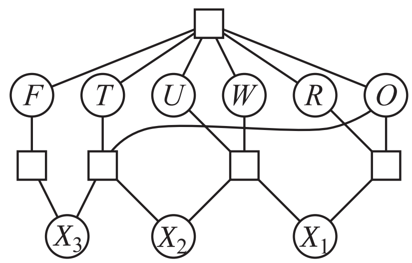
   (a) It is an identification problem.
   $X$ = {the set of $X_i$, which means the city $X_i$}$(X_n = X_0)$
   $D_i$ = {0,1,2,3,$\cdots$, n-1}
   $C$ = {for each integer $i \in \{0, 1, \cdots, n-1\}$, only one $X_i$ equals to it. And for each pair of integer$(i, i+1)$, the cities at these integer are connected.}

3. **Forward checking**. Solve the cryptarithmetic problem shown in Figure **??** step by step, using the strategy of backtracking with forward checking. Assume the variable order is $X_3 \rightarrow F \rightarrow X_2 \rightarrow X_1 \rightarrow O \rightarrow T \rightarrow R \rightarrow$

$U \to W$, and the value order is increasing. Note that different variables have different domains (e.g., the domain for $X_3$ is $\{0, 1\}$, the domain for $O$ is $\{0, 1, \ldots, 9\}$, and the domain for $F$ is $\{1, 2, \ldots, 9\}$).

$$\begin{array}{r} T\ W\ O \\ +\ T\ W\ O \\ \hline F\ O\ U\ R \end{array}$$

(a)




(b)

(a) A cryptarithmetic problem. Each letter stands for a distinct digit; the aim is to find a substitution of digits for letters such that the resulting sum is arithmetically correct, with the added restriction that no leading zeroes are allowed. (b) The constraint hypergraph for the cryptarithmetic problem, showing the *Alldiff* constraint (square box at the top) as well as the column addition constraints (four square boxes in the middle). The variables $X_1$, $X_2$, and $X_3$ represent the carry digits for the three columns.

Figure 2: Problem 3.

**Solution.**

| 1 | | X3 | F | X2 | X1 | O | T | R | U | W | | Constraint |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | start | 0, 1 | 1,2,3,4,5,6,7,8,9 | 0,1 | 0,1 | 0,1,2,3,4,5,6,7,8,9 | 1,2,3,4,5,6,7,8,9 | 0,1,2,3,4,5,6,7,8,9 | 0,1,2,3,4,5,6,7,8,9 | 0,1,2,3,4,5,6,7,8,9 | | F=X3 |
| 3 | X3=0 | 0 | empty | 0,1 | 0,1 | 0,1,2,3,4,5,6,7,8,9 | 1,2,3,4,5,6,7,8,9 | 0,1,2,3,4,5,6,7,8,9 | 0,1,2,3,4,5,6,7,8,9 | 0,1,2,3,4,5,6,7,8,9 | Wrong! | 10X3+O=2T+X2 |
| 4 | X3=1 | 1 | 1 | 0,1 | 0,1 | 0,2,3,4,5,6,7,8,9 | 2,3,4,5,6,7,8,9 | 0,2,3,4,5,6,7,8,9 | 0,2,3,4,5,6,7,8,9 | 0,2,3,4,5,6,7,8,9 | | 10X2+U=2W+X1 |
| 5 | X2=0 | 1 | 1 | 0 | 0,1 | 0,2,3,4,5,6,7,8,9 | 2,3,4,5,6,7,8,9 | 0,2,3,4,5,6,7,8,9 | 0,2,3,4,5,6,7,8,9 | 0,2,3,4,5,6,7,8,9 | | 10X1+R=2O |
| 6 | X1=0 | 1 | 1 | 0 | 0 | 0,2,3,4,5,6,7,8,9 | 2,3,4,5,6,7,8,9 | 0,2,3,4,5,6,7,8,9 | 0,2,3,4,5,6,7,8,9 | 0,2,3,4,5,6,7,8,9 | | |
| 7 | O=0 | 1 | 1 | 0 | 0 | 0 | 5 | empty | 0,2,3,4,5,6,7,8,9 | 0,2,3,4,5,6,7,8,9 | Wrong! | |
| 8 | O=2 | 1 | 1 | 0 | 0 | 2 | 6 | 4 | 0,3,5,7,8,9 | 0,3,5,7,8,9 | | |
| 9 | U=0 | 1 | 1 | 0 | 0 | 2 | 6 | 4 | 0 | empty | Wrong! | |
| 10 | U=3 | 1 | 1 | 0 | 0 | 2 | 6 | 4 | 3 | empty | Wrong! | |
| 11 | U=5 | 1 | 1 | 0 | 0 | 2 | 6 | 4 | 5 | empty | Wrong! | |
| 12 | U=7 | 1 | 1 | 0 | 0 | 2 | 6 | 4 | 7 | empty | Wrong! | |
| 13 | U=8 | 1 | 1 | 0 | 0 | 2 | 6 | 4 | 8 | empty | Wrong! | |
| 14 | O=3 | 1 | 1 | 0 | 0 | 3 | empty | | | | Wrong! | |
| 15 | U=9 | 1 | 1 | 0 | 0 | 2 | 6 | 4 | 9 | empty | Wrong! | |
| 16 | O=4 | 1 | 1 | 0 | 0 | 4 | 7 | 8 | 0,2,3,5,6,9 | 0,2,3,5,6,9 | | |
| 17 | U=0 | 1 | 1 | 0 | 0 | 4 | 7 | 8 | 0 | empty | Wrong! | |
| 18 | U=2 | 1 | 1 | 0 | 0 | 4 | 7 | 8 | 2 | empty | Wrong! | |
| 19 | U=3 | 1 | 1 | 0 | 0 | 4 | 7 | 8 | 3 | empty | Wrong! | |
| 20 | U=5 | 1 | 1 | 0 | 0 | 4 | 7 | 8 | 5 | empty | Wrong! | |
| 21 | U=6 | 1 | 1 | 0 | 0 | 4 | 7 | 8 | 6 | 3 | Right! | |

So the answer is $(F, O, T, R, U, W) = (1, 4, 7, 8, 6, 3)$

4. **AC-3**. Consider the following CSP:

- Variables: $A, B, C, D$
- Domain: $\{1, 2, 3\}$
- Constraints: $A \neq B, B > C, C < D$

Solve the problem using the strategy of backtracking search with AC-3 algorithm. Give the problem-solving steps, specifying each assignment when backtracking and the consequence of each pop operation in AC-3 (i.e., what values you cross off and which arcs you push to the queue).

Suppose the value order is descending and the variable order is alphabetical, which both queue initialization and neighbor consideration follow (i.e., the queue is initialized as $A \to B, B \to A, \dots$).

| | A | B | C | D | | |
|---|---|---|---|---|---|---|
| start | 123 | 123 | 123 | 123 | | |
| A=1 | 1 | 123 | 123 | 123 | queue: B->A | add: B->A |
| B->A | 1 | 23 | 123 | 123 | queue: C->B | add: C->B |
| C->B | 1 | 23 | 12 | 123 | queue: B->C, D->C | add: B->C, D->C |
| B->C | 1 | 23 | 12 | 123 | queue: D->C | |
| D->C | 1 | 23 | 12 | 23 | queue: C->D | add: C->D |
| C->D | 1 | 23 | 12 | 23 | queue: empty | |
| B=2 | 1 | 2 | 12 | 23 | queue: A->B, C->B | add: A->B , C->B |
| A->B | 1 | 2 | 12 | 23 | queue: C->B | |
| C->B | 1 | 2 | 1 | 23 | queue: D->C | add: D->C |
| D->C | 1 | 2 | 1 | 23 | queue:empty | |
| C=1 | 1 | 2 | 1 | 23 | queue:empty | |
| D=2 | 1 | 2 | 1 | 2 | queue:C->D | add:C->D |
| C->D | 1 | 2 | 1 | 2 | queue:empty | Success! |

5. **K-consistency & tree structure**. Consider the following three questions:

   (a) Give a concrete CSP example to show that $k$-consistency does not imply $(k+1)$-consistency for some $k \geq 2$.

   (b) Give a concrete CSP example to show that $k$-consistency does not imply $(k-1)$-consistency for some $k \geq 3$.

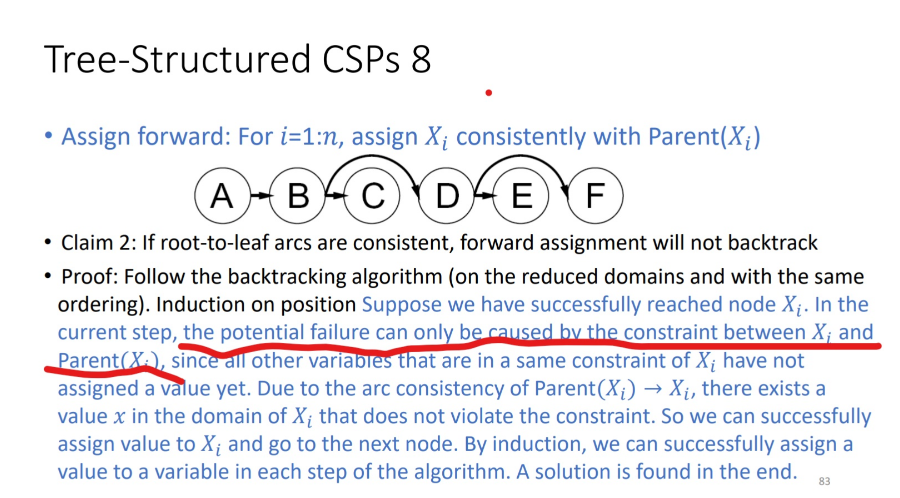
   (c) Why graphs with cycles can not be applied with the algorithm for tree-structured CSPs (introduced in Page 77-83, Lecture 4)? What step in the analysis fails? Why this step holds for trees? Give an example to explain the failure.

**Solution.**

(a) Considering a K4, their domains are all {R,G,B}. It is 3-consistent because for each 3 points$X_i, X_j, X_k$, $\exists$ 2 points $X_i, X_j$, $s.t. \forall X_i, X_j$, $X_k$ can be the third color different from $X_i, X_j$, which proves the 3-consistency. But it isn't 4-consistent, because when $X_i = R, X_j = G, X_k = B$, $X_m$ is unable to assign. So it isn't 4-consistent.

(b) Considering a K3, $X_i \in \{1\}, X_j \in \{1\}, X_k \in \{0, 1\}$. The constraint is that the sum of all $X$ is an odd number. Then the K3 is 3-consistent because when we choose $X_i, X_j$, no matter we assign them, there exists a value so that $\sum_{a=i,j,k} X_i$ is odd. So it is 3-consistent.
But when we choose only $X_i, X_j$.The sum of their value will not be an odd number. So the example is not 2-consistent.

(c) Because if it is a graph, the contradiction may not only be between a node and it's parent node but farther, like another leaf node.
This step in the analysis fails:



# Tree-Structured CSPs 8

- Assign forward: For $i$=1:$n$, assign $X_i$ consistently with Parent($X_i$)

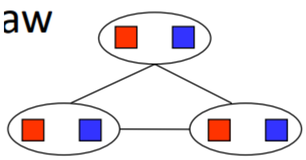


- Claim 2: If root-to-leaf arcs are consistent, forward assignment will not backtrack
- Proof: Follow the backtracking algorithm (on the reduced domains and with the same ordering). Induction on position Suppose we have successfully reached node $X_i$. In the current step, the potential failure can only be caused by the constraint between $X_i$ and Parent($X_i$), since all other variables that are in a same constraint of $X_i$ have not assigned a value yet. Due to the arc consistency of Parent($X_i$) $\rightarrow X_i$, there exists a value $x$ in the domain of $X_i$ that does not violate the constraint. So we can successfully assign value to $X_i$ and go to the next node. By induction, we can successfully assign a value to a variable in each step of the algorithm. A solution is found in the end.

83

This step holds for trees because every node has only one parent node or none.

**Example:**



We consider a K3. D={R,B}
First, we randomly choose a root node, A, without losing generality. Then we have two arcs $A->B, A->C, B->C$.
They all satisfy the arc-consistency. So the remove backward ends.
When the assign forward begins, we find that no matter how we assign A and B, we cannot find a proper C. That is because C has two parents A and B, which cause the contradiction.