FAI HW1

Problem 1

a. Depth-first search
Order of state expansion: $S \to A \to C \to D \to B \to G$
Final path: $S \to A \to C \to D \to G$

b. Breadth-first search
Order of state expansion: $S \to A \to B \to D \to C \to G$
Final path: $S \to D \to G$

c. Uniform cost search
Order of state expansion: $S \to A \to B \to C \to D \to G$
Final path: $S \to A \to C \to G$

d. Greedy search with the heuristic h shown on the graph
Order of state expansion: $S \to D \to G$
Final path: $S \to D \to G$

e. A* search with the same heuristic
Order of state expansion: $S \to A \to D \to C \to G$
Final path: $S \to A \to C \to G$

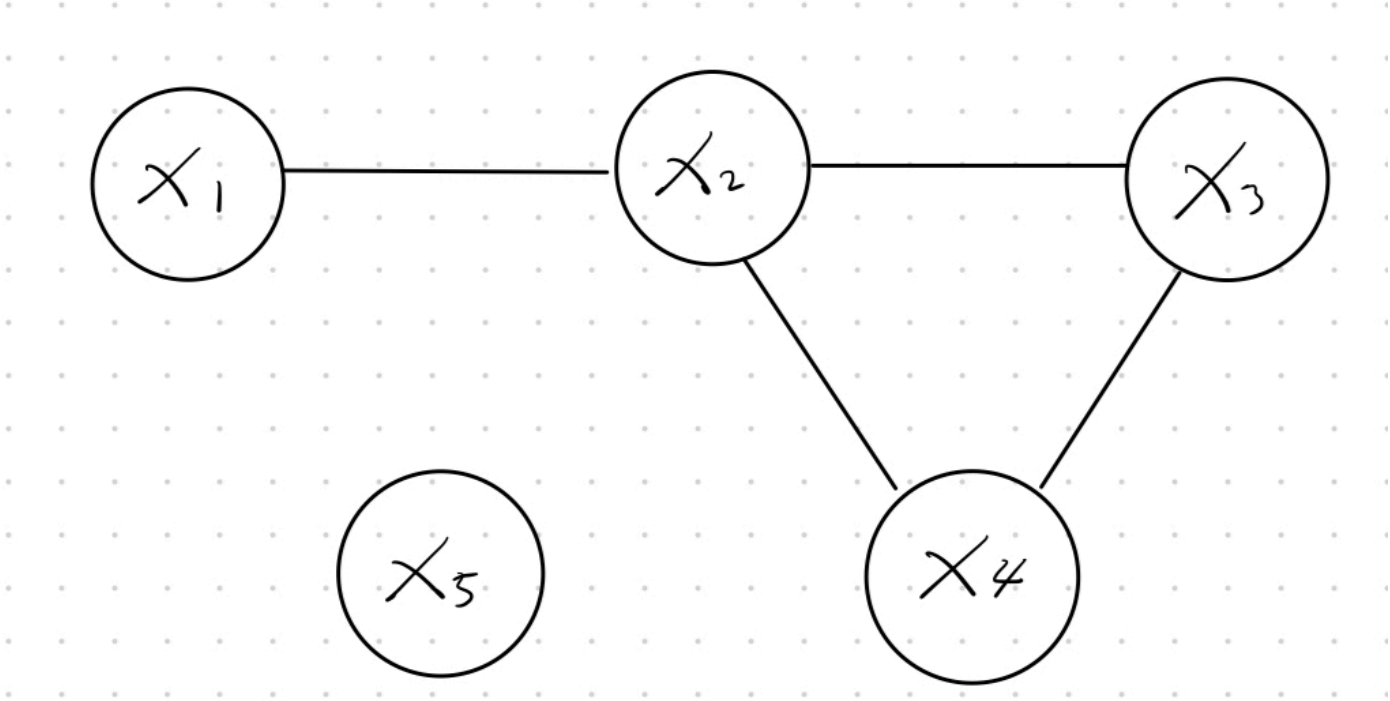
Problem 2

The bug is the timing of "ADD successor to closed". In this implementation, any successor will be added to ***closed*** whenever it is enqueued, which might yield a suboptimal solution. The correct implementation is to add a node to ***closed*** whenever it is dequeued (popped from the ***fringe***).

Problem 3

(1) Formulate this problem as a CSP. Describe the variables, domains and constraints.
Variables: X_i, i from 1 to 5.
Domains: {A, B, C}
Constraints: $X_1 \neq X_2$, $X_2 \neq X_3$, $X_2 \neq X_4$, $X_3 \neq X_4$, $X_1 \in \{A, C\}$, $X_2 \in \{A\}$, $X_3 \in \{B, C\}$, $X_4 \in \{B, C\}$, $X_5 \in \{A, B\}$

(2) Draw the constraint graph associated with your CSP.



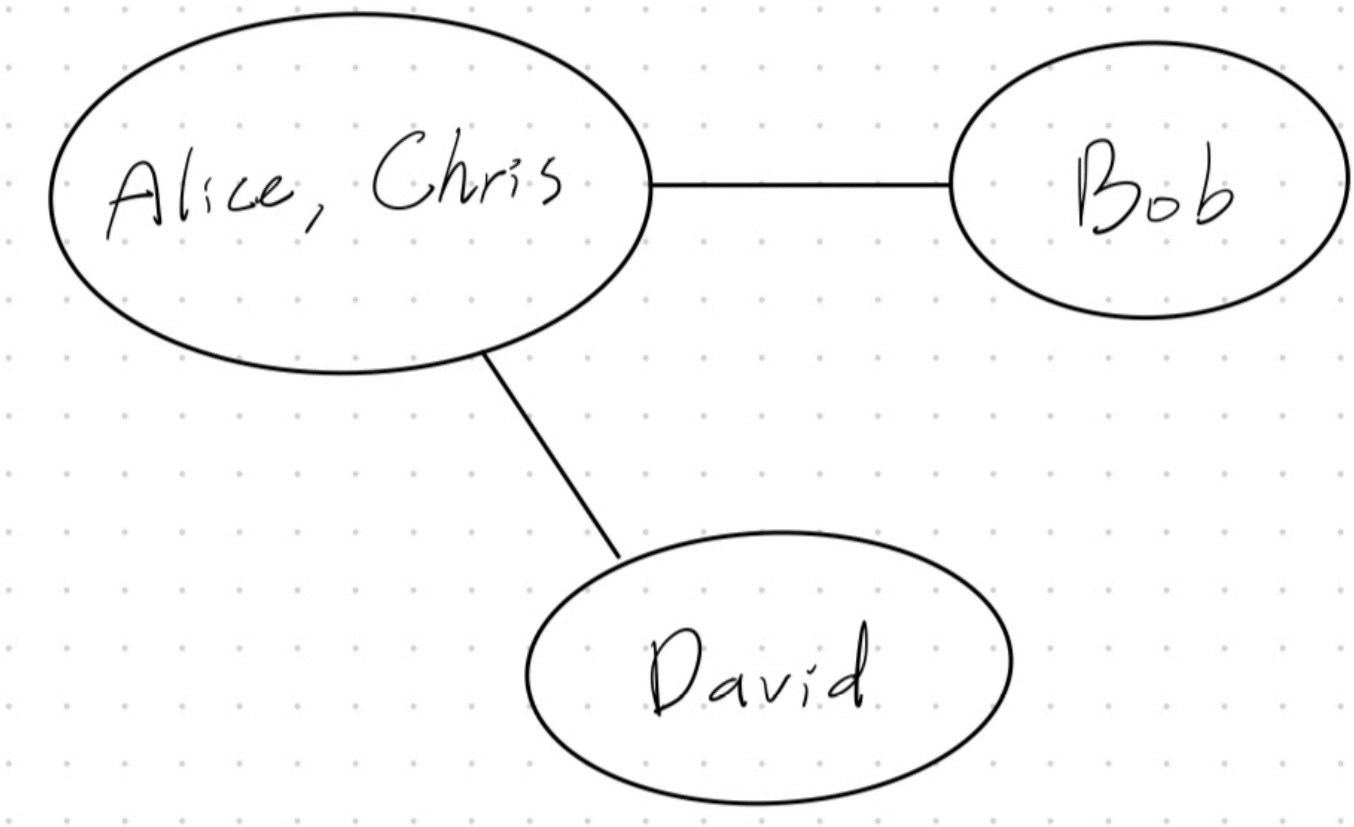
(3) Show the domains of the variables after running arc-consistency on this initial graph (after having already enforced any unary constraints).
$X_1 \in \{C\}$
$X_2 \in \{A\}$
$X_3 \in \{B, C\}$
$X_4 \in \{B, C\}$
$X_5 \in \{A, B\}$

(4) Give one solution to this CSP.
$(X_1, X_2, X_3, X_4, X_5) = (C, A, B, C, A)$

(5) Your CSP should look nearly tree-structured. Briefly explain (one sentence or less) why we might prefer to solve tree-structured CSPs.
Since tree-structured CSPs only have to perform one pass of removing backward and one pass of assigning forward, which reduces the time complexity from $O(d^n)$ to $O(nd^2)$.

Problem 4

a. Draw the constraint graph for this CSP.



b. Run the basic backtracking search. Use alphabetical order to both select unassigned variables and iterate over values. Write down the food assignment.
$(Alice, Bob, Chris, David) = (pizza, ramen, pizza, ramen)$

c. Assume that no variables have been assigned values yet. When running one iteration of forward checking, which value(s) will be removed for each variable if we assign "pizza" to Alice. Write down "None" if no values will be removed.
Bob: "pizza" will be removed.
Chris: "quesadillas" and "ramen" will be removed.
David: "pizza" will be removed.