CS 188 Summer 2024 — Midterm Review CSPs

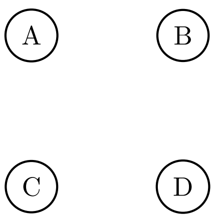
Q1. CSPs: Apartments

Four people, A, B, C, and D, are all looking to rent space in an apartment building. There are three floors in the building, 1, 2, and 3 (where 1 is the lowest floor and 3 is the highest). Each person must be assigned to some floor, but it's ok if more than one person is living on a floor. We have the following constraints on assignments:

- A and B must not live together on the same floor.
- If A and C live on *the same* floor, they must both be living on floor 2.
- If A and C live on *different* floors, one of them must be living on floor 3.
- D must not live on the same floor as anyone else.
- D must live on a higher floor than C.

We will formulate this as a CSP, where each person has a variable and the variable values are floors.

(a) Draw the edges for the constraint graph representing this problem. Use binary constraints only. You do not need to label the edges.



(b) Suppose we have assigned C = 2. Apply forward checking to the CSP, filling in the boxes next to the values for each variable that are eliminated:

A	☐ 1	☐ 2	☐ 3
B	☐ 1	☐ 2	☐ 3
C		☐ 2	
D	☐ 1	☐ 2	☐ 3

(c) Starting from the original CSP with full domains (i.e. without assigning any variables or doing the forward checking in the previous part), enforce arc consistency for the entire CSP graph, filling in the boxes next to the values that are eliminated for each variable:

A	☐ 1	☐ 2	☐ 3
B	☐ 1	☐ 2	☐ 3
C	☐ 1	☐ 2	☐ 3
D	☐ 1	☐ 2	☐ 3

(d) Suppose that we were running local search with the min-conflicts algorithm for this CSP, and currently have the following variable assignments.

A	3
B	1
C	2
D	3

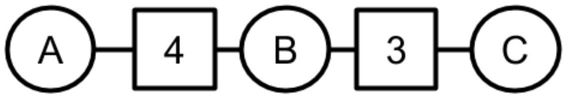
Which variable would be reassigned, and which value would it be reassigned to? Assume that any ties are broken alphabetically for variables and in numerical order for values.

.

Q2. Dudoku

Here we introduce Dudokus, a type of CSP problem. A Dudoku consists of variables and summation constraints. The circles indicate variables that can take integer values in a specified range and the boxes indicate summation constraints, which specify that the variables connected to the constraint need to add up to the number given in the box.

(a) Let's begin with linear Dudokus, where the variables can be arranged in a linear chain with constraints between adjacent pairs. For example, in the linear Dudoku below, variables A, B, and C need values assigned to them in the set $\{1, 2, 3\}$ such that $A + B = 4$ and $B + C = 3$.



(i) How many solutions does this Dudoku have?

○ 0 ○ 1 ○ 2 ○ 3 ○ more than 3

(ii) Consider the general case of a linear Dudoku with n variables $X_1, \ldots, X_n$, each taking a value in $\{1, \ldots, d\}$. What is the complexity for solving such a Dudoku using the generic tree-CSP algorithm?

○ $\mathcal{O}(nd^3)$ ○ $\mathcal{O}(n^2d^2)$ ○ $\mathcal{O}(nd^2)$ ○ $\mathcal{O}(d^n)$

(iii) One proposal for solving linear Dudokus is as follows: for each possible value i of the first variable X_1 in the chain, instantiate X_1 with that value and then run generic arc consistency beginning with the pair (X_2, X_1) until termination; keep going until a solution is found or there are no more values to try for X_1. Which of the following are true?

☐ This will correctly detect any unsolvable Dudoku.

☐ This will always solve any solvable Dudoku.

☐ This will sometimes terminate without finding a solution when one exists.

☐ The runtime is $O(nd^3)$.

(iv) Binary Dudoku constraints are *one-to-one*, meaning that if one variable in a binary constraint has a known value, there is only one possible value for the other variable. Suppose we modify arc consistency to take advantage of one-to-one constraints instead of checking all possible pairs of values when checking a constraint. Now the runtime of the algorithm in the previous part becomes:

○ $\mathcal{O}(nd)$ ○ $\mathcal{O}(nd^3)$ ○ $\mathcal{O}(n^2d^2)$ ○ $\mathcal{O}(nd^2)$

(b) Branching Dudokus

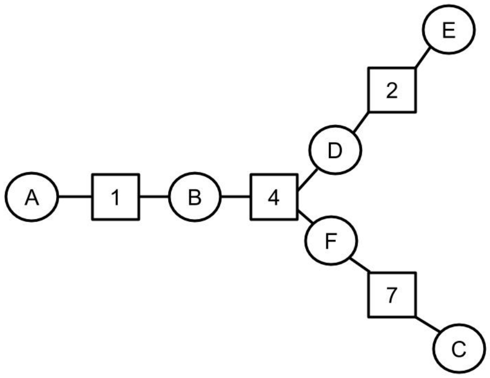


Figure 1: Example A

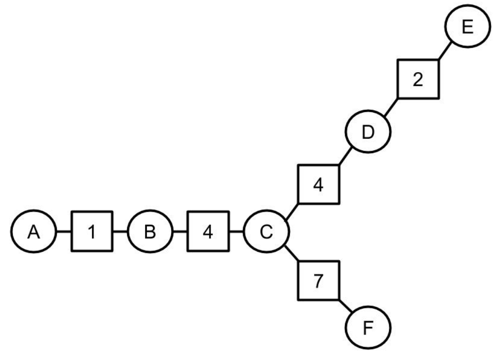


Figure 2: Example B

A branching Dudoku is one in which multiple linear chains are joined together. Chains can be joined at a summation node, as in example A above, or at a variable, as in example B above. Recall that a cutset is a set of nodes that can be removed from a graph to ensure some desired property. Which of the following are true?

☐ Dudoku A is a binary CSP.

☐ Dudoku B is a binary CSP.

☐ Dudoku A is a tree-structured CSP.

☐ Dudoku B is a tree-structured CSP.

☐ If variables B, D, and F are merged into a single megavariable, Dudoku A will be a tree-structured CSP.

☐ If variables A and E are merged into a single megavariable, Dudoku B will be a tree-structured CSP.

☐ The minimum cutset that turns Dudoku A into a set of disjoint linear Dudokus contains 3 variables.

☐ The minimum cutset that turns Dudoku B into a set of disjoint linear Dudokus contains 1 variable.

(c) Circular Dudokus

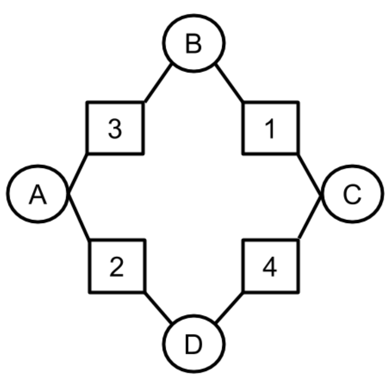
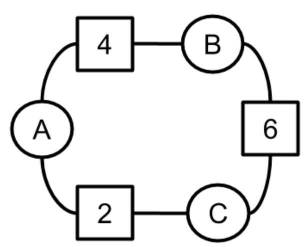


Figure 3: Example 1



Example 2

(i) The figure above shows two examples of circular Dudokus. If we apply cutset conditioning with a minimal cutset and a tree-CSP solver, what is the runtime for a circular Dudoku with n variables and domain size d?

○ $\mathcal{O}(d^{n-1})$ ○ $\mathcal{O}(n^2d^2)$ ○ $\mathcal{O}(nd)$ ○ $\mathcal{O}(nd^3)$

(ii) Suppose that the variables in the circular Dudokus in the figure are assigned values such that all the constraints are satisfied. Assume also that the variable domains are integers in $[-\infty, \infty]$. Now consider what happens if we add 1 to the value of variable A in each Dudoku and then re-solve with A fixed at its new value. Which of the following are true?

☐ Dudoku 1 now has no solution.

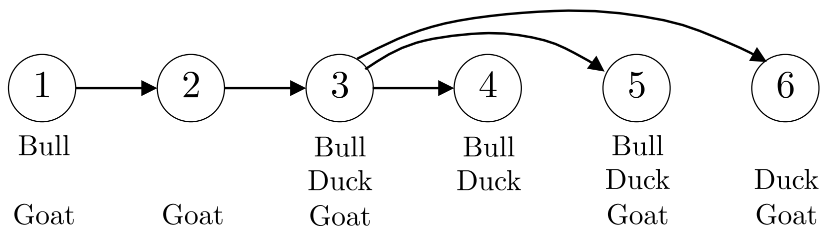
☐ Dudoku 2 now has no solution.

(iii) What can you conclude about the number of solutions to a circular Dudoku with n variables?

Q3. Farmland CSP

The animals in Farmland aren't getting along and the farmers have to assign them to different pens. To avoid fighting, animals of the same type cannot be in connected pens. Fortunately, the Farmland pens are connected in a tree structure.

(a) Consider the following constraint diagram that shows six pens with lines indicating connected pens. The remaining domains for each pen are listed below each node.



After assigning a bull to pen 5, enforce arc consistency on this CSP considering only the *directed* arcs shown in the figure. What are the remaining values for each pen?

Pen	Values
1	
2	
3	
4	
5	Bull
6	

(b) What is the computational complexity of solving general tree structured CSPs with n nodes and d values in the domain? Give an answer of the form $O(\cdot)$.

(c) This True/False question is worth 1 points. Leaving a question blank is worth 0 points. **Answering incorrectly is worth -1 points.**

(i) [*true* or *false*] If root to leaf arcs are consistent on a general tree structured CSP, assigning values to nodes from root to leaves will not back-track if a solution exists.

(d) Given 3 animal types, what is the most number of pens a tree structure could have, such that the computational complexity to solve the tree CSP is no greater than the computational complexity to solve a fully connected CSP with 10 pens?