CS6601 Midterm - CSPs

Raymond Jia

Spring 2024 - **Solutions**

Grading*: No partial credit is awarded.

1 Constraint Satisfaction Problems (CSPs) - 24 pts

It's tennis tournament season! Your friends have been training and preparing for their tournament all winter and they're excited to show off their training, but first they have to travel to the tournament in their six seat minivan. Despite being teammates, your friends can't quite agree on who gets to sit where in the minivan (everyone wants shotgun of course!) so they've left it up to you to help them figure out where to sit in the minivan.

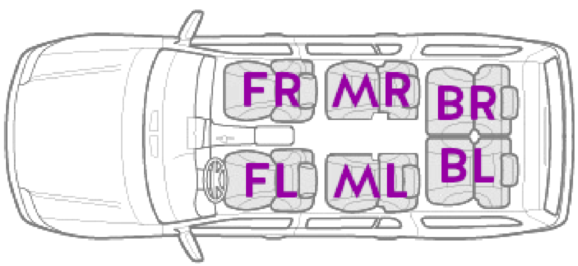

Figure 1: Minivan Seating Positions

Recognizing that this is a CSP problem, you've decided to take an algorithmic approach to solve the problem. Let's translate it into CSP terminology first:

Variables: There are **6 variables**, and they are the **six seats in the minivan**. They are labeled in Figure 1 and are as follows: **FL (Driver), FR (Navigator), ML, MR, BL, and BR**. **Note that for the abbreviation terminology, the first letter determines which* ***row*** *(**F**ront, **M**iddle, **B**ack) the seat is in and the second letter determines which* ***side*** *(**L**eft, **R**ight) the seat is within that row.*

Domains: For each variable, **6 values** can be assigned, and they are your friends: **Ava (A), Clara (C), Henry (H), Jacob (J), Leo (L), and Sara (S)**.

Below in Figure 2 is the starting set of domains for the CSP problem before any domain reduction occurs. We will refer to it as **domain state zero**.

FR: {A, C, H, J, L, S}	**MR:** {A, C, H, J, L, S}	**BR:** {A, C, H, J, L, S}
FL: {A, C, H, J, L, S}	**ML:** {A, C, H, J, L, S}	**BL:** {A, C, H, J, L, S}

Figure 2: Domain State Zero

Constraints: You've talked to your friends and collected all the different needs that they have and reduced them to a set of unary, binary, and global constraints that are easy to implement in the CSP algorithm. They are as follows:

Unary Constraints

1. FL (Driver) cannot be Ava or Jacob (They don't have valid driver's licenses).
2. FL (Driver) cannot be Henry (He broke his leg and is in crutches, can't drive).
3. FR (Navigator) cannot be Ava, Sara, or Leo (They are notorious for getting lost even when using Google Maps).
4. BL, BR (Back row seats) cannot be Clara or Leo (They're too tall, not enough leg room).
5. BL, BR (Back row seats) cannot be Henry (He broke his leg and is in crutches, can't sit in the back).

Binary Constraints

1. Henry and Jacob want to work on homework together, so they must sit in horizontally adjacent seats (same row).

Global Constraints

1. **No two seats can be assigned the same person** (We find this obvious but the computer isn't aware of this!).

1.1 CSP1 Q28 - 2 pts

How many **complete assignments** are there in total for this CSP problem? For an assignment to be complete, all variables must be assigned a single value and the assignment does not have to be consistent. *(AIMA 4th Ed. 6.1)*

Answer: $\boxed{46656}$

Solution: A complete assignment is one in which every variable is assigned a value (R&N 6.1). Note that this ignores constraints (which is handled by the definition of consistency). Since there are 6 variables and each variable's domain has 6 values that it can be assigned, there are 6^6 complete assignments or 46656. *The astute of you may have noticed that this set of complete assignments includes seats that are assigned the same value (i.e. all six seats are assigned Ava). While we instinctively understand that one person cannot possibly sit in more than one seat, the computer isn't aware of this and has to represent it as a Global Constraint of AllDiff. Since complete assignments do not require consistency (doesn't have to satisfy the outlined constraints, including AllDiff), it is possible for more than one seat to have the same person assigned.*

1.2 CSP2 Q29 - 2 pts

How many **complete and consistent assignments** are there for this CSP problem? For an assignment to be consistent, all constraints (7) must also be satisfied. *(AIMA 4th Ed. 6.1)*

Here is one example of a **complete and consistent assignment**.

FR: [C]	**MR: [J]**	**BR: [S]**
FL: [L]	**ML: [H]**	**BL: [A]**

Figure 3: Sample complete and consistent assignment

Answer: $\boxed{4}$

Solution: A consistent assignment is one that does not violate any constraints (R&N 6.1). Combined with the definition of a complete assignment, a complete and consistent assignment is then a solution to the CSP. After applying all the requirements, we find that there are only 4 complete and consistent assignments (solutions):

FR: [C]	**MR: [J]**	**BR: [S]**
FL: [L]	**ML: [H]**	**BL: [A]**

FR: [C]	**MR: [H]**	**BR: [S]**
FL: [L]	**ML: [J]**	**BL: [A]**

FR: [C]	**MR: [J]**	**BR: [A]**
FL: [L]	**ML: [H]**	**BL: [S]**

FR: [C]	**MR: [H]**	**BR: [A]**
FL: [L]	**ML: [J]**	**BL: [S]**

Figure 4: Complete and consistent assignments to this CSP problem

1.3 CSP3 Q30-Q35 - 12 pts

Node consistency occurs when all variable domains satisfy their unary constraints. What does the state of domains look like after applying node consistency to domain state zero seen in Figure 2? *(AIMA 4th Ed. 6.2.1)*

Answer:

FR: {C, H, J}	MR: {A, C, H, J, L, S}	BR: {A, J, S}
FL: {C, L, S}	ML: {A, C, H, J, L, S}	BL: {A, J, S}

Figure 5: State of domains after applying node consistency to domain state zero

1.4 CSP4 Q36 - 2 pts

CSP problems are often treated like a graph problem because the constraints can be used to make a **constraint graph**. In a constraint graph, there is at most one edge between each pair of nodes (variables) and each edge represents the set of binary constraints between that pair of nodes. To simplify the problem, we reduce the global constraint into a series of binary constraints and populate the constraint graph with our variables and edges. How many edges are in the resulting constraint graph? *We choose not to have self-edges in the constraint graph, hence the unary constraints are not represented by the constraint graph and we are really only counting merged binary constraints. (AIMA 4th Ed. 6.2.5)*

Answer: $\boxed{15}$

Solution: Because there is at most one edge between each pair of variables and the global constraint of *AllDiff* causes there to be an *AllDiff* edge between each pair of variables, we seek to count the total number of edges when each pair of variables is assigned an edge. Since there is a total 6 variables and each edge involves 2 variables, we find that the answer is ${}_6C_2$ or 15.

1.5 CSP5 Q37 - 2 pts

In order to represent all of the binary constraint between two variables, edges in the constraint graph often contain a **set of tuples of allowed values** that the two variables can take on. How many tuples are in the set for the constraint edge between the variable FL (Driver) and the variable FR (Navigator)? An example of one allowed tuple in the format of `(FL, FR)` is `(Ava, Clara)`. *Ignore unary constraints, the set of edges is only aware of binary constraints between the two variables. Also recall in CSP4 that we have reduced the global constraint into binary constraints.*

Answer: $\boxed{14}$

Solution: Ignoring unary constraints, we find that the only binary constraints that affect the set of allowed tuples are **Henry and Jacob must be in the same row** and **FL and FR must be different**. Ignoring constraints, there are 6^2 or 36 possible tuples that could make up the set. If we use complementary counting (counting the disallowed tuples and subtracting them from the total), we see that we can subtract 6 for each pair where FL and FR are assigned the same value, and then we can further subtract $4 \times 2 \times 2 = 16$ for all pairs involving one of Henry or Jacob but not the other. This would leave us with $36 - 6 - 16 = 14$ allowed tuples. If we counted it directly, we can first count the number of tuples involving Henry and Jacob (2), and then count the number of tuples that don't involve them at at all ($4 \times 3 = 12$), and altogether we would also get $2 + 12 = 14$ allowed tuples.

2 Constraint Change

The team decided to use the arrangement provided in CSP2 for the first leg of their journey. The team makes a planned rest stop at a gas station to fill up on gas and snacks and so that Leo can swap out for a fresh driver. At this point, Henry and Jacob have finished their homework and no longer need to sit adjacent to each other. We must now redefine our constraints as follows:

Unary Constraints

1. FL (Driver) cannot be Ava or Jacob (They don't have valid driver's licenses).
2. FL (Driver) cannot be Henry (He broke his leg and is in crutches, can't drive).
3. FL (Driver) cannot be Leo (He just drove for several hours and needs to rest).
4. FR (Navigator) cannot be Ava, Sara, or Leo (They are notorious for getting lost even when using Google Maps).
5. BL, BR (Back row seats) cannot be Clara or Leo (They're too tall, not enough leg room).
6. BL, BR (Back row seats) cannot be Henry (He broke his leg and is in crutches, can't sit in the back).

Binary Constraints

1. Jacob and Leo cannot sit in the same row (Jacob loves to talk but Leo is tired and wants to nap after driving for so long).
2. Jacob and Sara cannot sit in the same row (Jacob loves to talk but Sara loves peace and quiet).

Global Constraints

1. **No two seats can be assigned the same person** (We find this obvious but the computer isn't aware of this!).

2.1 CSP6 Q38 - 2 pts

After the changes above, how many **complete and consistent assignments** are there for this new CSP problem? *(AIMA 4th Ed. 6.1)*

Answer: $\boxed{16}$

Solutions:

FR: [J]	MR: {H, L}	BR: {A, S}
FL: [C]	ML: {H, L}	BL: {A, S}

FR: [H]	MR: {L, S}	BR: {A, J}
FL: [C]	ML: {L, S}	BL: {A, J}

FR: [C]	MR: {H, L}	BR: {A, J}
FL: [S]	ML: {H, L}	BL: {A, J}

FR: [H]	MR: {C, L}	BR: {A, J}
FL: [S]	ML: {C, L}	BL: {A, J}

Figure 6: Complete and consistent assignments to the updated CSP problem

2.2 CSP7 Q39 - 2 pts

Beginning with domain state zero seen in Figure 2 (node consistency has not been enforced), if we pick the variable **FL (Driver)** to assign a value to, which value(s) should we assign to FL if we use **Least Constraining Value** to make an assignment? *(AIMA 4th Ed. 6.3.1)*

Answer: A, C, H

Solution: Least Constraining Value prefers the value that rules out the fewest choices for the neighboring variables in the constraint graph (R&N 6.3.1). The neighboring variables referred to here are variables that are affected directly via a binary constraint involving the variable being assigned. As seen in Figure 7, the values of **A, C, and H** result in the ruling out of the fewest choices among all the neighboring variables in the constraint graph.

FL's Assigned Value	Remove from FR	Remove from ML	Remove from MR	Remove from BL	Remove from BR	Total
A	A	A	A	A	A	**5**
C	C	C	C	C	C	**5**
H	H	H	H	H	H	**5**
J	J, L, S	J	J	J	J	7
L	L, J, S	L	L	L	L	7
S	S, J, L	S	S	S	S	7

Figure 7: Effect of assigning different values to FL