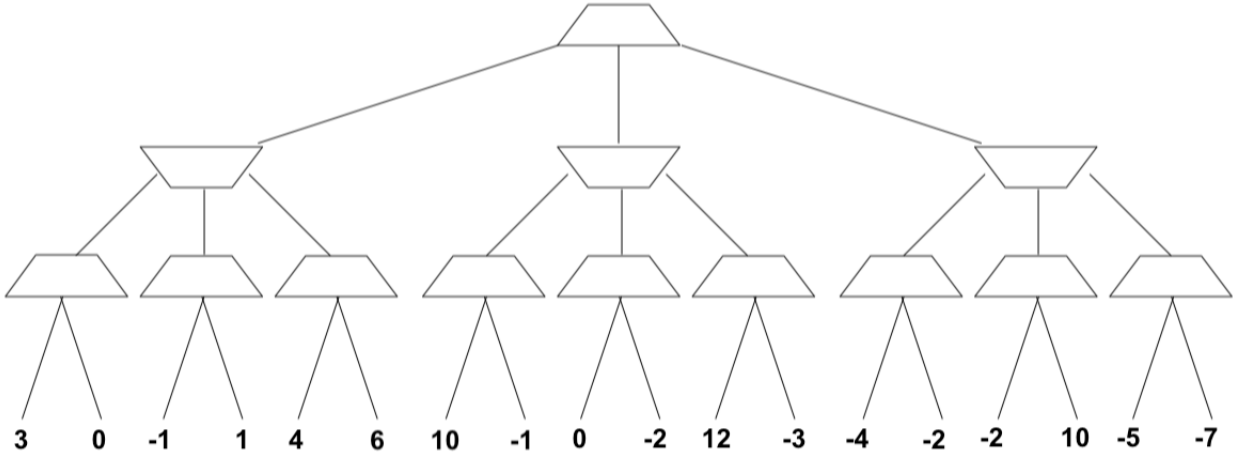
1 Adversarial Search

Consider the following game tree, where the root node is a maximizer. Using alpha beta pruning and visiting successors from left to right, record the values of alpha and beta at each node. Furthermore, write the value being returned at each node inside the trapezoid. Put an 'X' through the edges that are pruned off.



2 Discussion Questions

(a) What is the difference between Forward Checking and AC-3?

(b) Why would one use the following heuristics for CSP?

(i) Minimum Remaining Values (MRV)

(ii) Least Constraining Value (LCV)

3 CSP: Air Traffic Control

We have five planes: A, B, C, D, and E and two runways: international and domestic. We would like to schedule a time slot and runway for each aircraft to either land or take off. We have four time slots: 1, 2, 3, 4 for each runway, during which we can schedule a landing or take off of a plane. We must find an assignment that meets the following constraints:

- Plane B has lost an engine and must land in time slot 1.
- Plane D can only arrive at the airport to land during or after time slot 3.
- Plane A is running low on fuel but can last until at most time slot 2.
- Plane D must land before plane C takes off, because some passengers must transfer from D to C.
- No two aircrafts can reserve the same time slot for the same runway.

(a) Complete the formulation of this problem as a CSP in terms of variables, domains, and constraints (both unary and binary). Constraints should be expressed implicitly using mathematical or logical notation rather than with words. Make sure to specify variables, domains, and constraints.

For the following parts, we add the following two constraints:

- Planes A, B, and C cater to international flights and can only use the international runway.
- Planes D and E cater to domestic flights and can only use the domestic runway.

(b) The addition of the two constraints above alters the CSP. Specifically, the domain does not need to include the runway type since this information is carried by the variable, and the binary constraints have changed. Determine the new domain and complete the constraint graph for this problem given the original constraints and the two added ones.

(c) What are the domains of the variables after enforcing arc consistency? Begin by enforcing unary constraints. (Cross out values that are no longer in the domain.)

(d) Arc-consistency can be rather expensive to enforce, and we believe that we can obtain faster solutions using only forward-checking on our variable assignments. Using the Minimum Remaining Values heuristic, perform backtracking search on the graph, breaking ties by picking lower values and characters first. List the (variable, assignment) pairs in the order they occur (including the assignments that are reverted upon reaching a dead end). Enforce unary constraints before starting the search.

List of (variable, assignment) pairs:

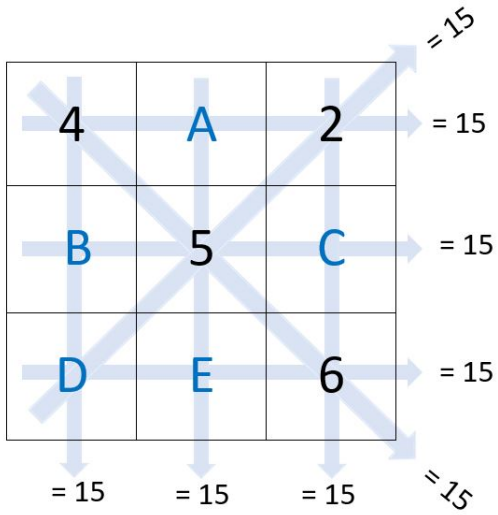
(You don't have to use this table)

A	1	2	3	4
B	1	2	3	4
C	1	2	3	4
D	1	2	3	4
E	1	2	3	4

4 CSP: Magic Square

A magic square is an $n \times n$ grid where each entry is unique and contains one of $\{1, ..., n^2\}$. It has that every row, column, and diagonal sum to the same number.

In this problem, we'll solve a 3×3 magic square by formulating it as a CSP. Each row, column, and diagonal in the 3×3 magic square must sum to 15. We have already filled out some of the numbers for you, but the letters in blue still need to be filled in.



(a) Complete the formulation of this problem as a CSP in terms of variables, domains, and constraints (both unary and binary). Constraints should be expressed implicitly using mathematical or logical notation rather than with words. Make sure to specify variables, domains, and constraints.

Hint: You do not need to create variables for the squares already provided.

(b) Draw the binary constraint graph for this problem. For simplicity, you may ignore the "alldiff" constraint that all variables are unique when creating this graph.

(c) Use the binary constraint graph to run the AC-3 algorithm and find a solution to the magic square.

5 MRV and LCV in Action

Ayush, Simrit, and Josep want to paint "15-281" on the fence tomorrow in honor of their favorite class. They have the following paint collections at home:

- Ayush = Red, Yellow, Green
- Simrit = Red, Yellow, Pink
- Josep = Red, Pink

Each of them will contribute exactly one bucket of paint from their collections such that no two TAs bring the same paint color.

Ayush suddenly remembers going over CSPs in lecture, and suggests formulating this problem as a CSP to determine an assignment of each TA to a paint color so that all three chosen paint colors are different. Simrit isn't fully convinced yet that LCV and MRV will help speed up a constraint satisfaction problem, so Ayush asks for your help to convince Simrit.

(a) Let's use the minimum remaining values (MRV) and least constraining value (LCV) heuristics to assign TAs to paint colors. Recall that the MRV heuristic determines which *variable* to assign, while the LCV heuristic determines which *value* to assign to that variable to. How many times will we backtrack to a previous assignment? Assume we break ties in rainbow order.

(b) Suppose we use a new set of heuristics to assign TAs to paint colors: maximum remaining values (instead of MRV) and most constraining value (instead of LCV). How many times will we backtrack to a previous assignment?