CS2109S Tutorial 2

Adversarial Search and Local Search

(AY 25/26 Semester 2)

February 5, 2026

(Prepared by Benson)

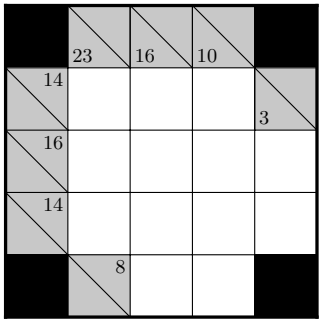
Contents

Local Search
- Q1. Kakuro

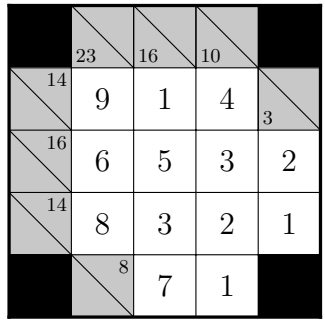
Adversarial Search
- Q2. Hunter and Rabbit
- Q3. Alpha Beta Pruning
- Q4. Alpha Beta Pruning

Bonus. Maximum and Minimum Pruning

Q1. Kakuro



(a) Initial Kakuro



(b) Solved Kakuro

Q1. Kakuro

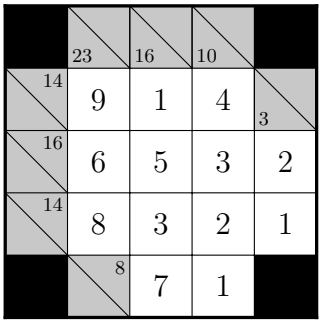
(a) Propose a state representation for the problem.

(b) What are the initial and goal states for the problem under your proposed representation?

(c) Define a reasonable transition function to generate new candidate solutions.

(d) Define a reasonable heuristic function to evaluate the "goodness" of a candidate solution. Explain how this heuristic can also be used as a goal test to determine that we have a solution to the problem.

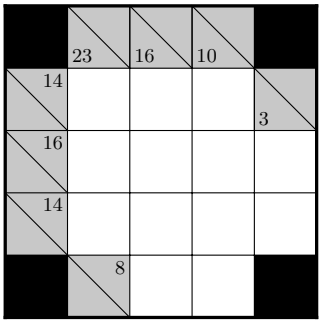
Q1. Kakuro



State Representation:

0	0	0	0	0
0	9	1	4	0
0	6	5	3	2
0	8	3	2	1
0	0	7	1	0

Q1. Kakuro



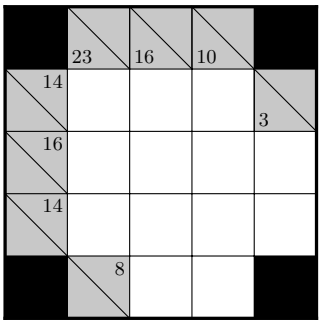
Initial State?

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

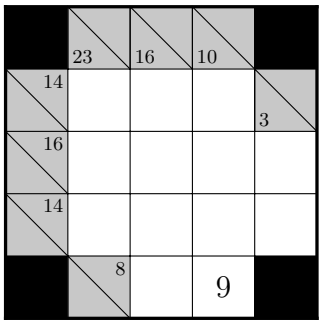
Q1. Kakuro

Transition function?: Pick a random cell and assign a random number.

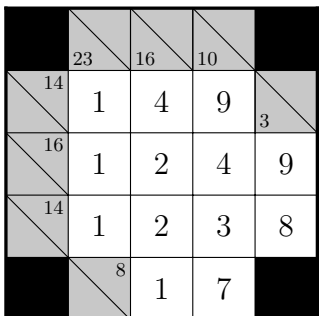
- Is the goal state reachable?
- Is it reasonably efficient?



→



Q1. Kakuro



A "Better" Initial State:

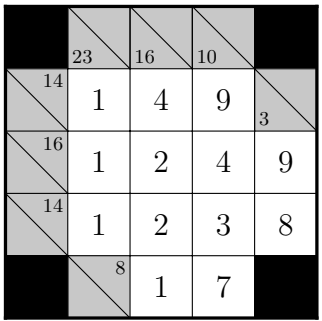
- Satisfies row constraints.
- Within a row, we can "tweak" it.

0	0	0	0	0
0	1	4	9	0
0	1	2	4	9
0	1	2	3	8
0	0	1	7	0

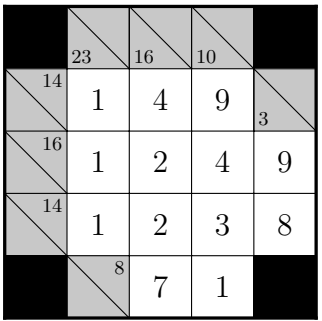
Q1. Kakuro

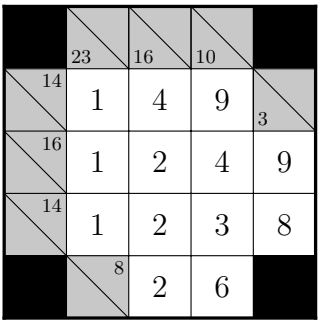
Transition function: Pick a random block on a random row. Change the arrangement of the block (swap / +1 and −1). ⇒ **Row constraints are still satisfied**.

- Is the goal state reachable?
- Is it reasonably efficient?



→

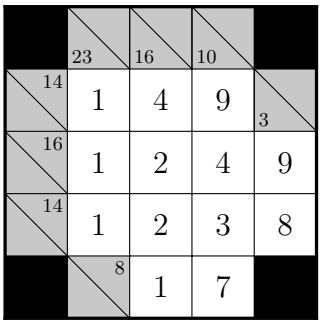




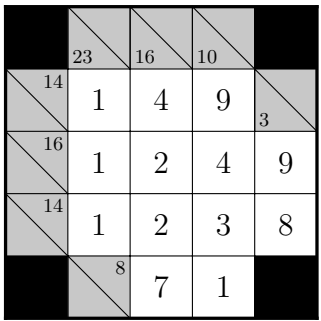
Q1. Kakuro

Heuristic function (how "good" is a state?):

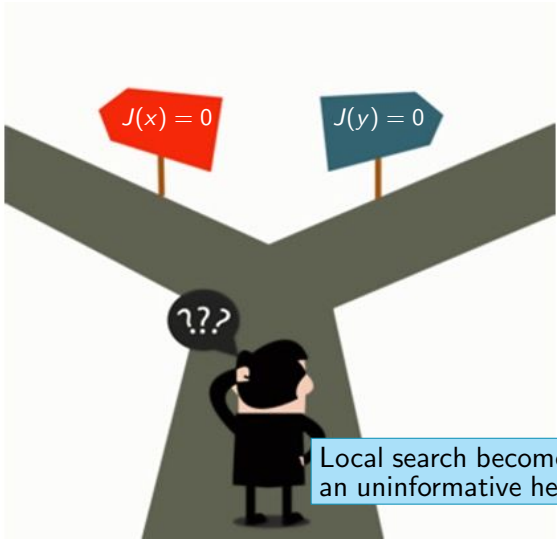
- 1 if all constraints are satisfied, 0 otherwise.
- Number of column constraints satisfied/violated.
- $|\text{current sum} - \text{expected sum}|$
- $w_1 \times |\text{current sum} - \text{expected sum}| + w_2 \times \#$ duplicate pairs



→



Q1. Kakuro

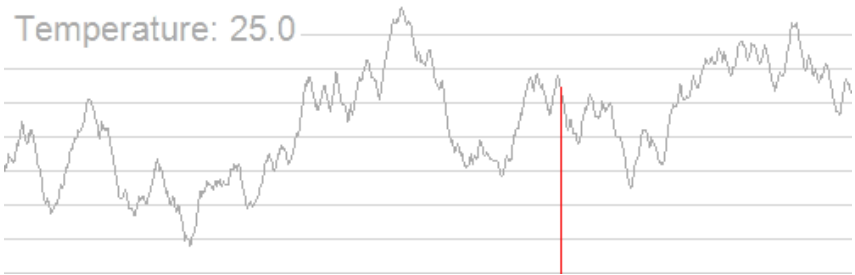


Local search becomes “wild guessing” with an uninformative heuristic function.

Q1. Kakuro

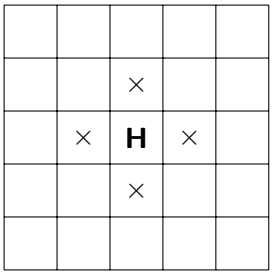
(f) Local search is susceptible to local minimas. Describe how you can modify your solution to combat this.

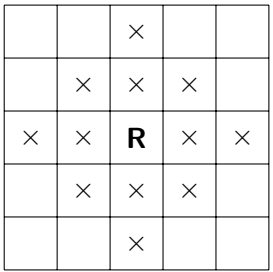
- Introduce random restarts by repeating local search from a random initial state.
- Use simulated annealing or stochastic beam search.



Q2. Hunter and Rabbit

Two-player game: The hunter wants to catch the rabbit.





(a) Explain why a depth-limited version of minimax is required for this game.

- 1000 turns – search tree grows exponentially.

(b) Propose a reasonable evaluation function.

- $\pm$`MAX` at terminal states; otherwise measure the Manhattan distance.

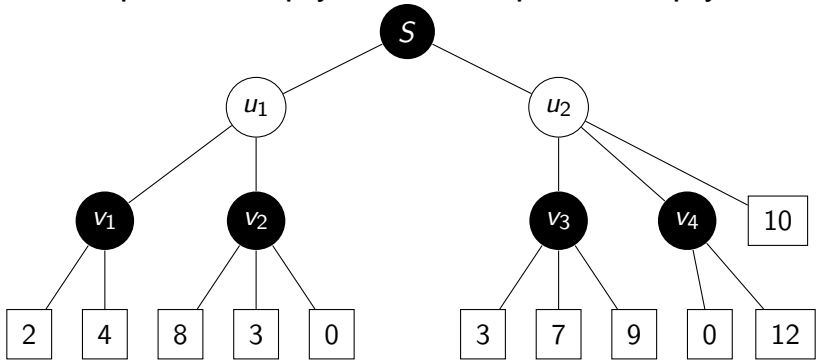
Alpha Beta Pruning

- α = Best already explored option along path to the root for MAX
  β = Best already explored option along path to the root for MIN
- We always wish to "reduce" the range $[\alpha, \beta]$. Hence, the MAX player increases α and the MIN player reduces β.
- Once $\alpha \geq \beta$, the current subtree isn't relevant anymore and we can prune the remaining branches.

Q3. Alpha Beta Pruning

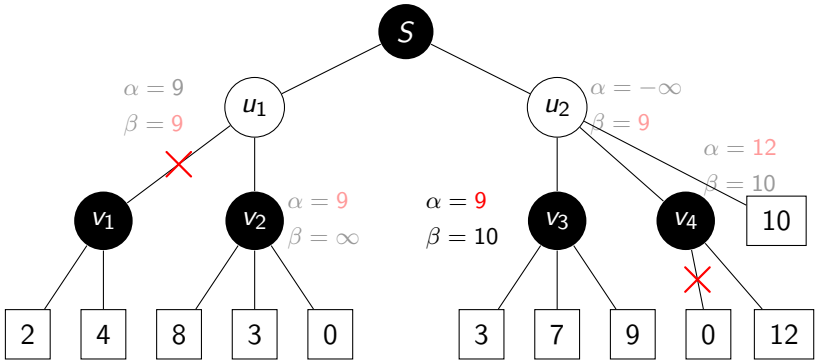
(a) Assume that we iterate over nodes from right to left; mark with an 'X' all arcs that are pruned by α-β pruning, if any.

NOTE: Black node represents MAX player, white node represents MIN player.



Q3. Alpha Beta Pruning

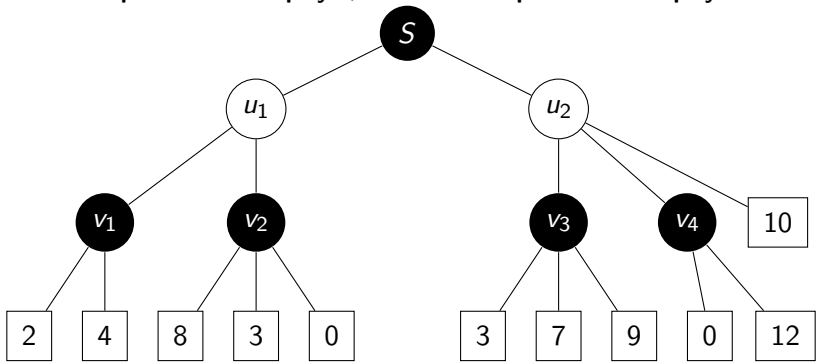
Solution:



Q3. Alpha Beta Pruning

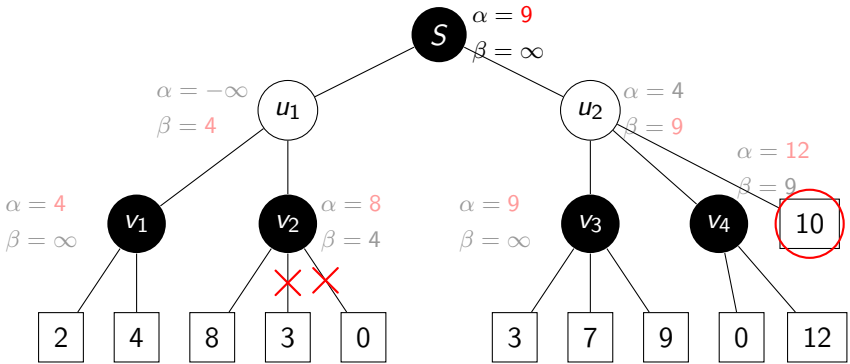
(b) Show that the pruning is different when we instead iterate over nodes from left to right. Your answer should clearly indicate all nodes that are pruned under the new traversal ordering.

NOTE: Black node represents MAX player, white node represents MIN player.



Q3. Alpha Beta Pruning

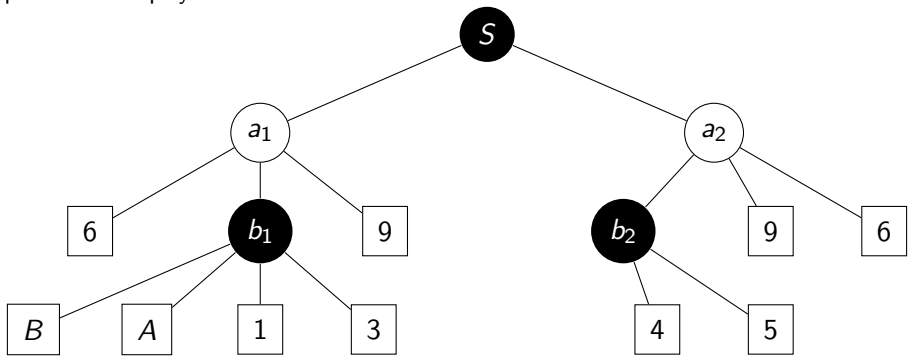
Solution:



Q4. Alpha Beta Pruning

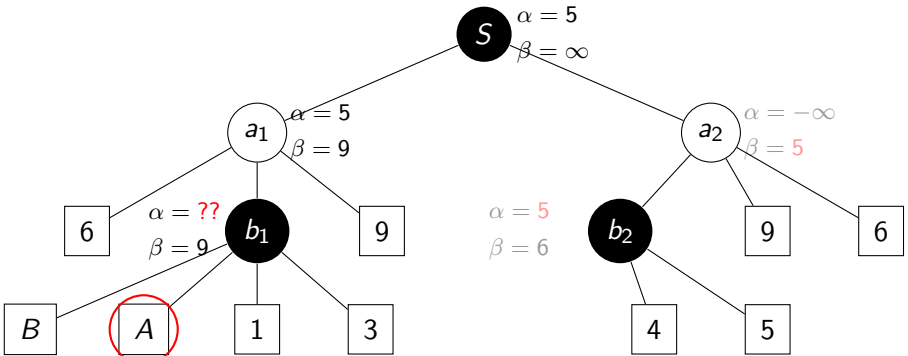
(a) In order for node B to NOT be pruned, what values can node A take on?

NOTE: Assume that we iterate over nodes from right to left. Black node represents MAX player, white node represents MIN player.



Q4. Alpha Beta Pruning

Solution:

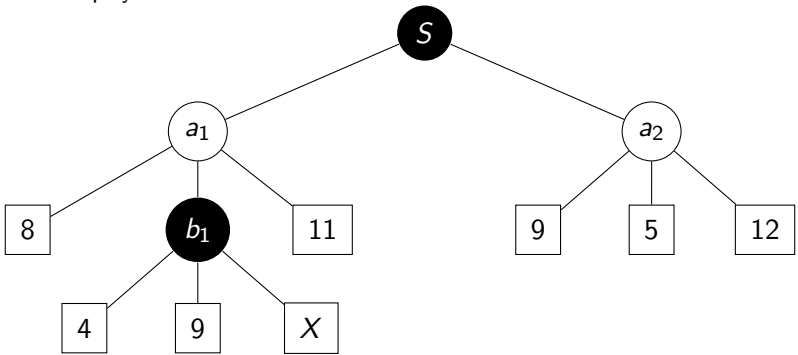


Node B is NOT pruned $\Rightarrow \alpha < \beta \Rightarrow A < 9$

Q4. Alpha Beta Pruning

(b) In order to prune the leftmost two children of node b_1, what values can node X take on?

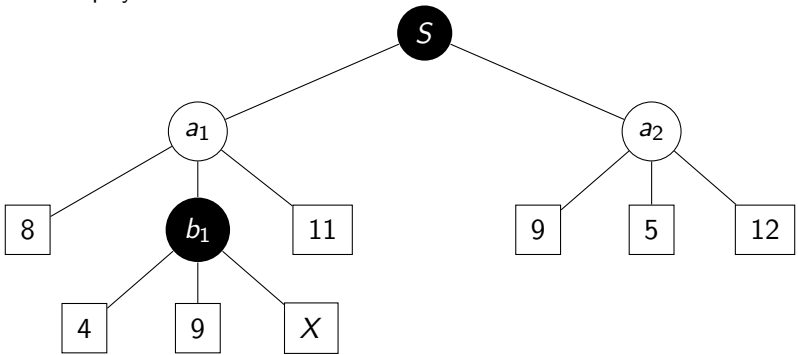
NOTE: Assume that we iterate over nodes from right to left. Black node represents MAX player, white node represents MIN player.



Q4. Alpha Beta Pruning

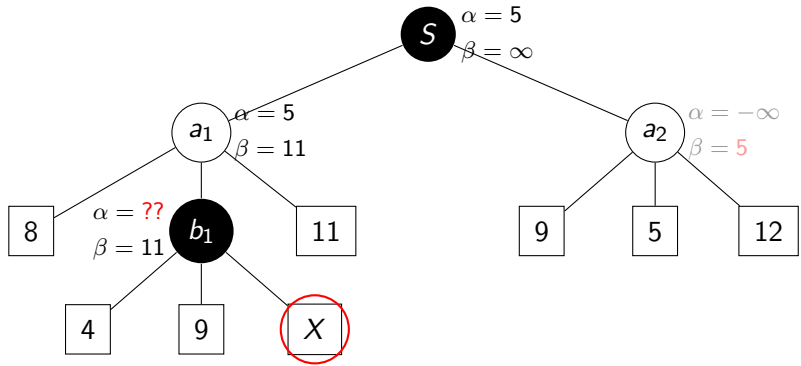
(b) In order to prune the leftmost two children of node b_1, what values can node X take on?

NOTE: Assume that we iterate over nodes from right to left. Black node represents MAX player, white node represents MIN player.



Q4. Alpha Beta Pruning

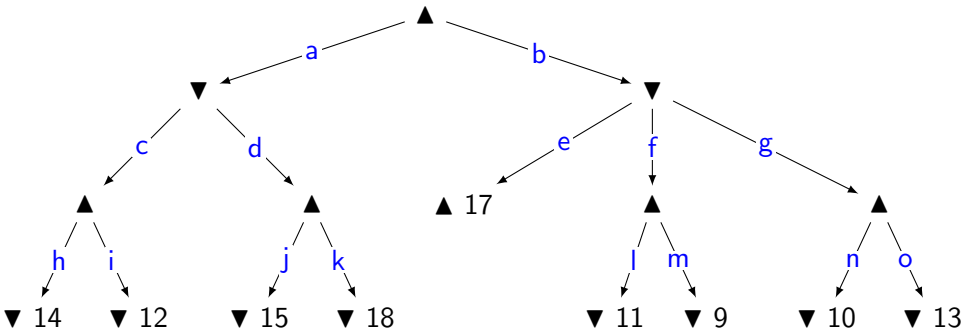
Solution:



Nodes 4, 9 are pruned $\Rightarrow \alpha \geq \beta \Rightarrow X \geq 11$

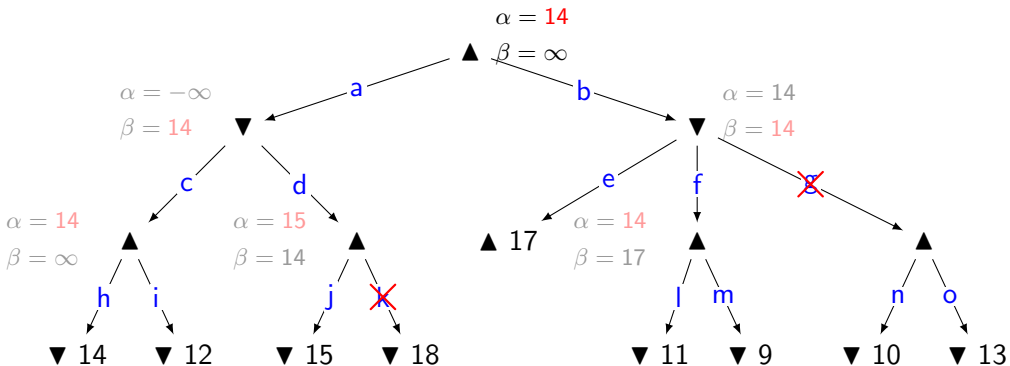
(Mini-)Practice

Poll: Suppose we traverse this tree with DFS from left to right. Select all the link(s) that would be pruned by alpha-beta.



(Mini-)Practice

Solution:



Bonus. Maximum and Minimum Pruning

Extra Slide

Leaf nodes take values which are either 0 or 1. Assume that we iterate over nodes from left to right. Write down the values of the leaf nodes where the **minimum** / **maximum** number of leaf nodes get skipped.