NUS | Computing
National University of Singapore

Eric Han
eric_han@nus.edu.sg
https://eric-han.com

Computer Science

T03 – 12 Sep 2024

Week 5

CS2109s TG35,36

Content I

1 Adversarial Search in Tic-Tac-Toe

2 Minimax

3 Nonogram

4 Alpha-Beta Pruning

Section 1: **Adversarial Search in Tic-Tac-Toe**

Tic-Tac-Toe - Use the minimax to determine the first move of the player.

$$Eval(n) = P(n) - O(n), \quad \text{where } P(n), O(n) \text{ are the no. of winning lines}$$

Recap

1. What is the `MINIMAX` algorithm? Why is it used?
2. What are the ingredients needed to setup a minimax problem?

Question 1

Tic-Tac-Toe - Use the minimax to determine the first move of the player.

$$Eval(n) = P(n) - O(n), \quad \text{where } P(n), O(n) \text{ are the no. of winning lines}$$

Recap

1. What is the `MINIMAX` algorithm? Why is it used?
2. What are the ingredients needed to setup a minimax problem?
   - Actors, Actions, Leaf Costs
3. What is the impact of choosing min/max in our computation?
4. [@] When was `MINIMAX` famously used in AI?

Question 1

Tic-Tac-Toe - Use the minimax to determine the first move of the player.

$$Eval(n) = P(n) - O(n), \quad \text{where } P(n), O(n) \text{ are the no. of winning lines}$$

Recap

1. What is the `MINIMAX` algorithm? Why is it used?
2. What are the ingredients needed to setup a minimax problem?
   - Actors, Actions, Leaf Costs
3. What is the impact of choosing min/max in our computation?
4. [@] When was `MINIMAX` famously used in AI?
   - IBM Deep Blue versus Garry Kasparov in Chess.

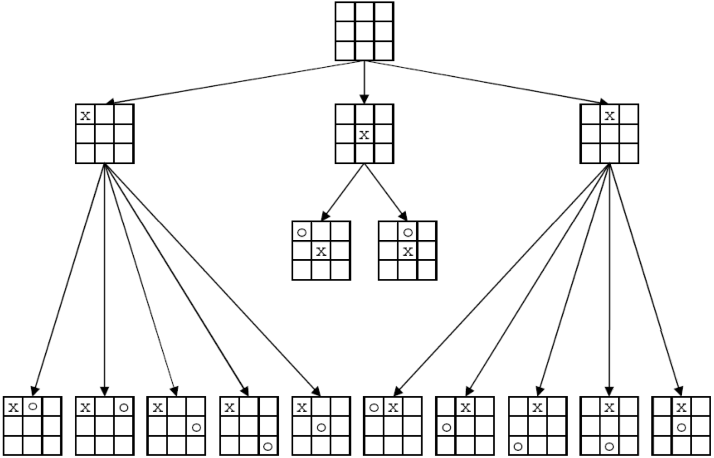

Figure 1: What is the move of the player?

Answer 1a

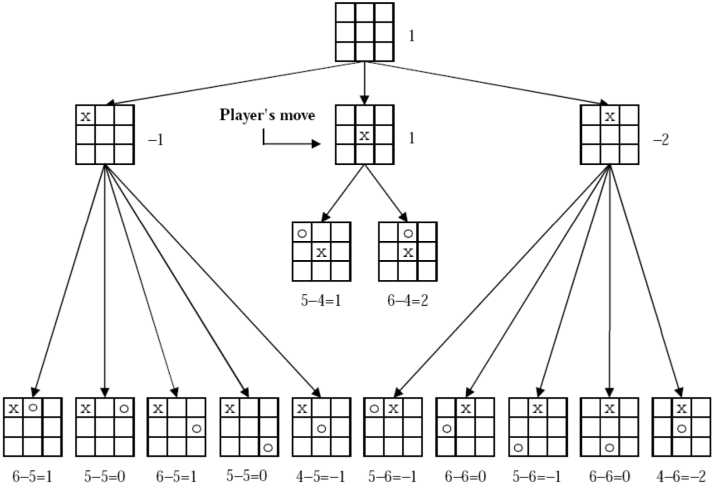

Figure 2: First move 2-ply deep search space

Question 1b

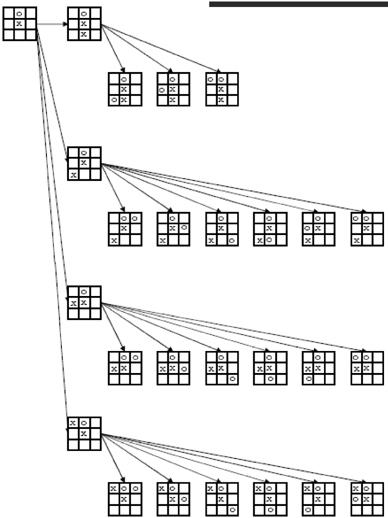

Figure 3: What is the move of the player?

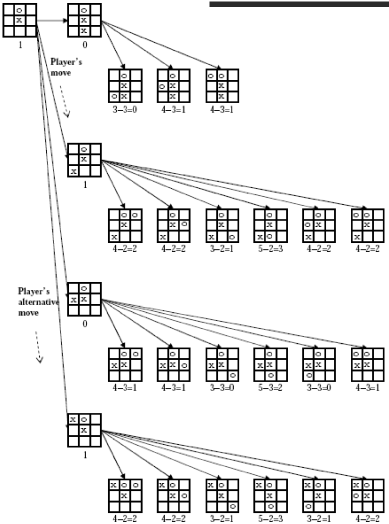

Figure 4: Second move 2-ply deep search space solution

Section 2: **Minimax**

Question 1 [G]

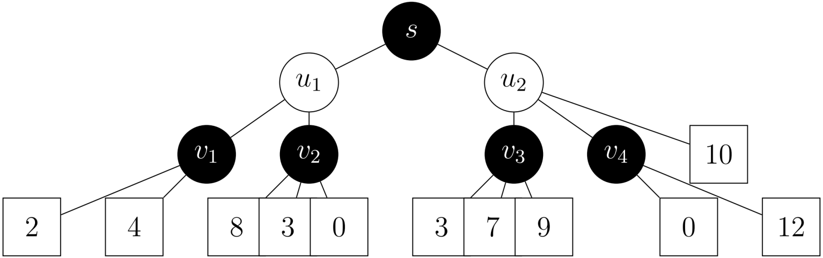

Figure 5: Alpha-Beta Tree

Run through the α-β:

a. Right to Left
b. Left to Right

Then determine if the effectiveness of pruning depends on iteration order.

Recap

1. What does α-β do?
2. What kind of efficiency do you gain?

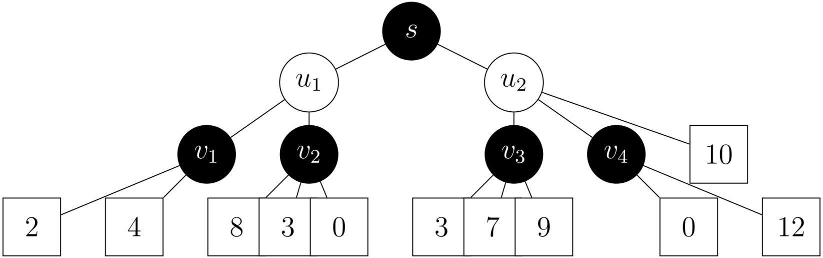

Figure 5: Alpha-Beta Tree

Run through the α-β:

a. Right to Left

b. Left to Right

Then determine if the effectiveness of pruning depends on iteration order.

Recap

1. What does α-β do?
2. What kind of efficiency do you gain?
   - Static evaluation and move generation.
3. What is deep cutoff?

Answer 1a

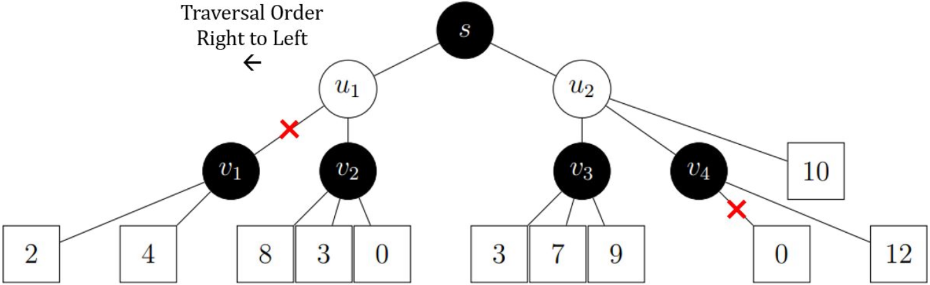

Figure 6: Right to left

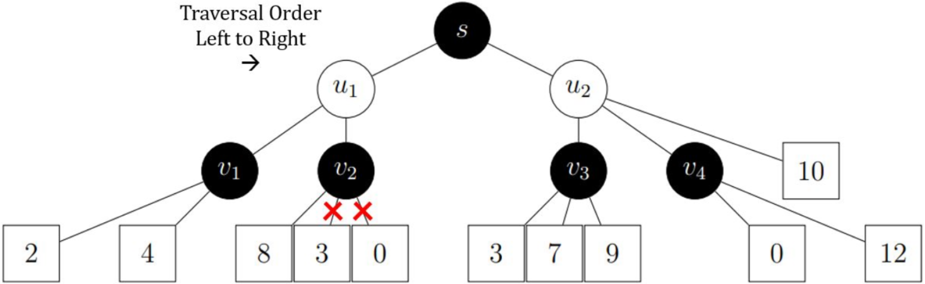

Figure 7: Left to right

Section 3: **Nonogram**

Question 1

Nonogram, aka Paint by Numbers, is a puzzle where cells are colored or left blank according to the numbers at the side of the grid.

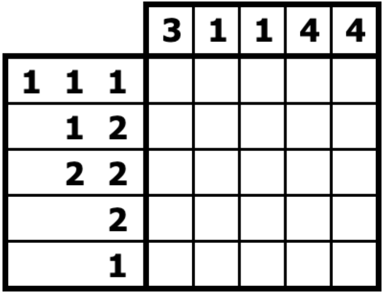

Figure 8: Inital

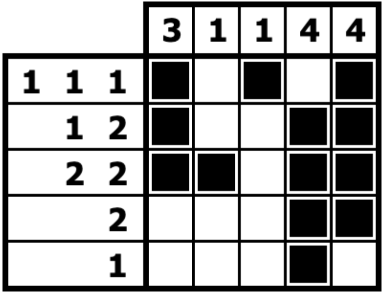

Figure 9: Solved

Recap

1. What are the ingredients needed for informed search?
2. What are the ingredients needed for local search?
3. What are the objectives for informed/local search?

1. What are the ingredients needed for informed search?
2. What are the ingredients needed for local search?
3. What are the objectives for informed/local search?

Summary

Un/Informed Search (Path): State space, Initial, Final, Action, Transition

- Uninformed: BFS, UCS, DFS
- Informed: GBFS, A*

Local Search (Goal): Inital state, Transition, Heuristic/Stopping criteria

- Hill Climbing, Sim. Annealing, Beam, Genetic...

Adversarial Search: Actors, Actions, Leaf Costs

- Minimax, Alpha-Beta

Question 1a [G]

Having learnt both informed search and local search, you think that local search is more suitable for this problem. Give 2 possible reasons why informed search might be a bad idea.

Question 1a [G]

Having learnt both informed search and local search, you think that local search is more suitable for this problem. Give 2 possible reasons why informed search might be a bad idea.

Answer

- We are only interested in the final solution.
- Search space is large $O(2^{n \times n})$ for a $n \times n$ grid.
- May not be solvable? In that case we can get a config that minimize violations.

Question 1b/c/d/e [G]

Find a formulation for **Local Search**.

Question 1b/c/d/e [G]

Find a formulation for **Local Search**.

Answer

$n \times n$ boolean matrix, where each element is either true (if the corresponding cell is colored) or false (if the corresponding cell is not colored).

- **Inital state** is an $n \times n$ boolean matrix with every row having random permutations of boolean vector satisfying row constraints, while the rest of the entries are set to false.
- **Transition**: we can pick a random row and generate the list of neighbors with the corresponding row permuted satisfying row constraints.
- **Heuristic/Stopping criteria**: number of instances where the constraints on the column configurations are violated.

Question 1f [G]

Local search is susceptible to local minima. Describe how you can modify your solution to combat this.

Question 1f [G]

Local search is susceptible to local minima. Describe how you can modify your solution to combat this.

Answer

- Introduce random restarts by repeating local search from a random initial state
- Simulated annealing search to accept a possibly `bad` state with a probability that decays over time
- beam search to perform k hill-climbing searches in parallel.

Section 4: **Alpha-Beta Pruning**

Question 1 [@]

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

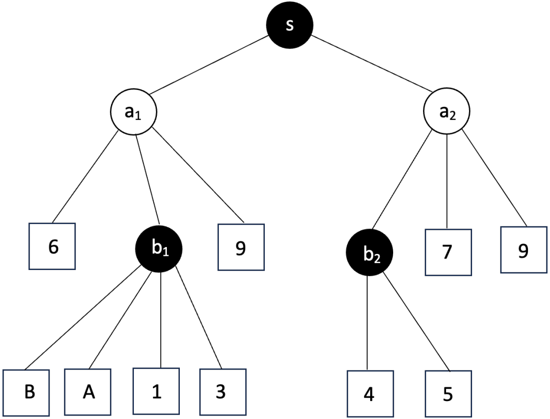

Figure 10: Find A so the B is not pruned.

```
< S -inf inf
        < a2 -inf inf
        > a2 -inf 9
        < a2 -inf 9
        > a2 -inf 7
        < a2 -inf 7
                < b2 -inf 7
                > b2 5 7
                < b2 5 7
                > b2 5 7
        > a2 -inf 5
> S 5 inf
```

```
< S 5 inf
        < a1 5 inf
        > a1 5 9
        < a1 5 9
                < b1 5 9
                > b1 5 9
                < b1 5 9
                > b1 5 9
                < b1 5 9
                > b1 Pruned val >= beta: 9 >= 9
        > a1 5 9
        < a1 5 9
        > a1 5 6
> S 6 inf
```

Pruned when $A \geq 9$, Not pruned when $A \leq 8$

To help you further your understanding, not compulsory; Work for EXP!

Tasks

1. Trace Manually/Use code Figure 11 to see the full capability.
   a. Some code implemented in https://github.com/eric-vader/CS2109s-2425s1-bonus
2. How can we benefit from α-β's efficiency?

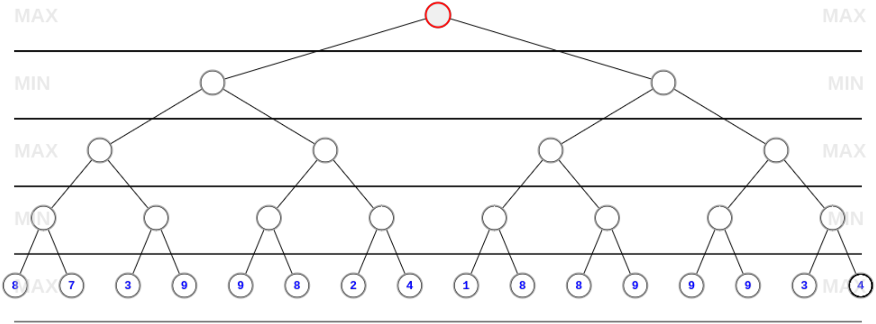

Figure 11: Alpha-Beta Example (Credit MIT)

Additional Readings

1. MIT Lecture - https://youtu.be/STjW3eH0Cik?si=YcnrXUJko5jjLzB0
2. IBM Deep Blue - https://www.sciencedirect.com/science/article/pii/S0004370201001291
3. Game Theory Concepts Within AlphaGo - https://towardsdatascience.com/game-theory-concepts-within-alphago-2443bbca36e0
4. What Game Theory Reveals About Life, The Universe, and Everything - https://youtu.be/mScpHTIi-kM?si=CLagrjz3WVi-EkXG

Buddy Attendance Taking

1. [@] and Bonus declaration is to be done here; You should show bonus to Eric.
2. Attempted tutorial should come with proof (sketches, workings etc...)
3. Random checks may be conducted.
4. Guest student should come and inform me.

Figure 12: Buddy Attendance: https://forms.gle/q5Secb3dHshmXNXd7

References I