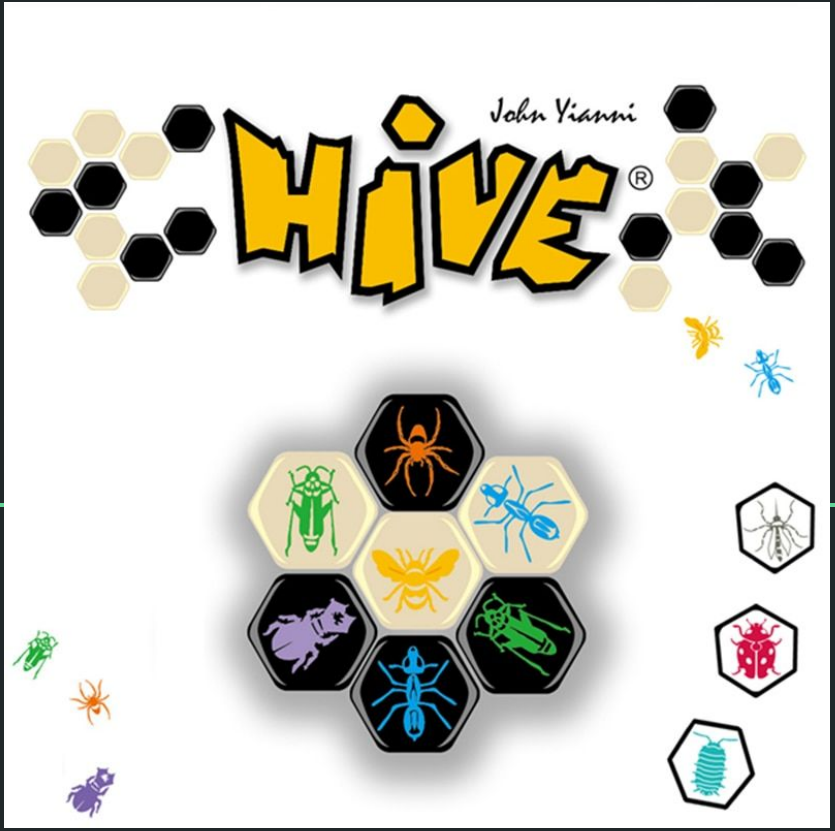
Hive Board Game AI

William Reames



Outline

- Background
- Implementing the Game
- Implementing my AI
- Demo
- Analysis
- Conclusion

Background

What is Hive?

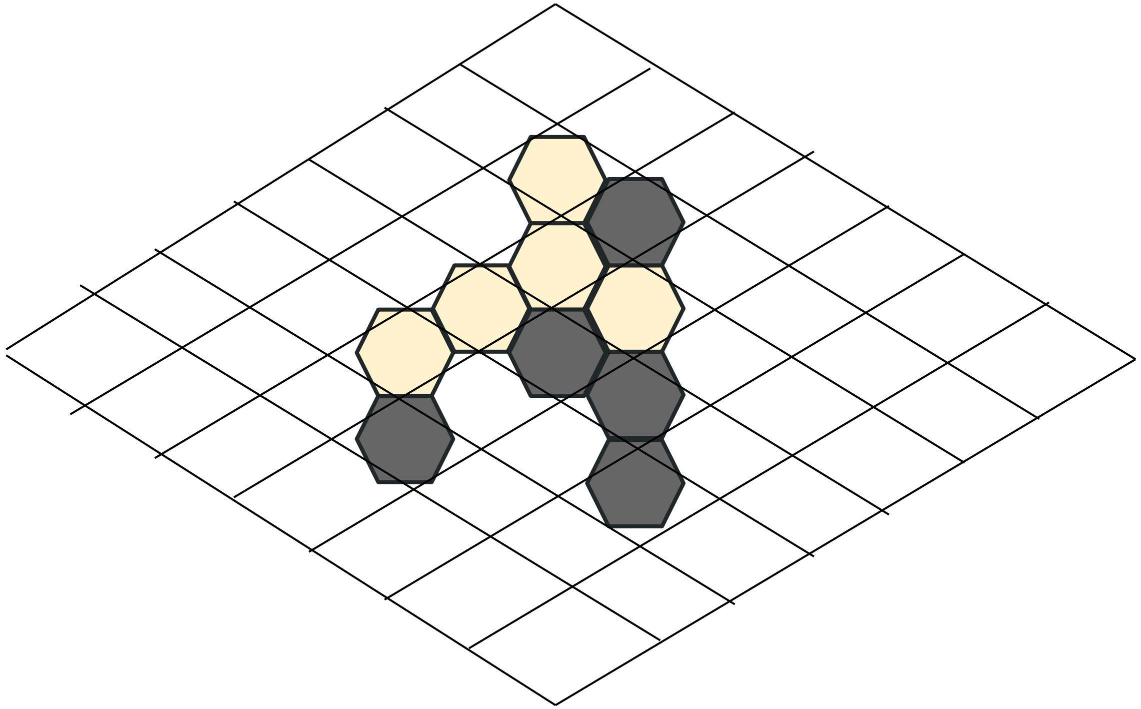
- Two player strategy-based board game
- Win by surrounding/capturing opponent’s queen bee

Project Goals/Overview

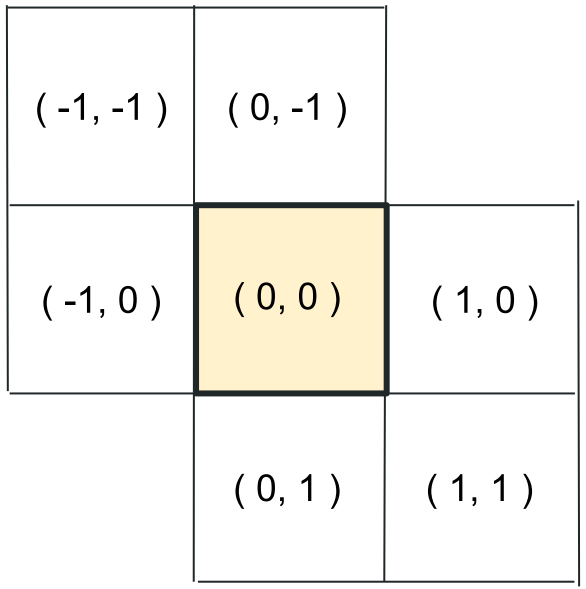
- Create an AI that can play Hive
- Developed using Python
- AI implemented through a minimax algorithm

Implementing the Game

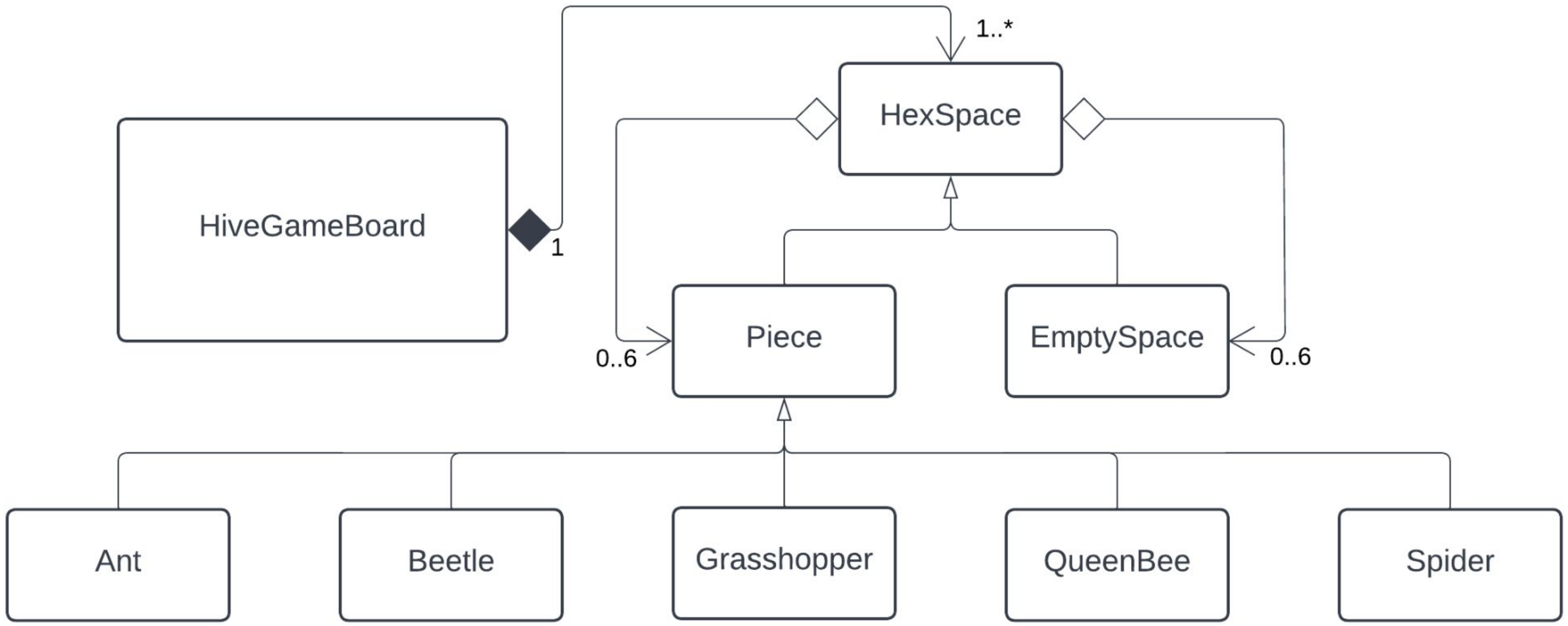
Storing a Hexagon Grid



Storing a Hexagon Grid



UML Class Diagram

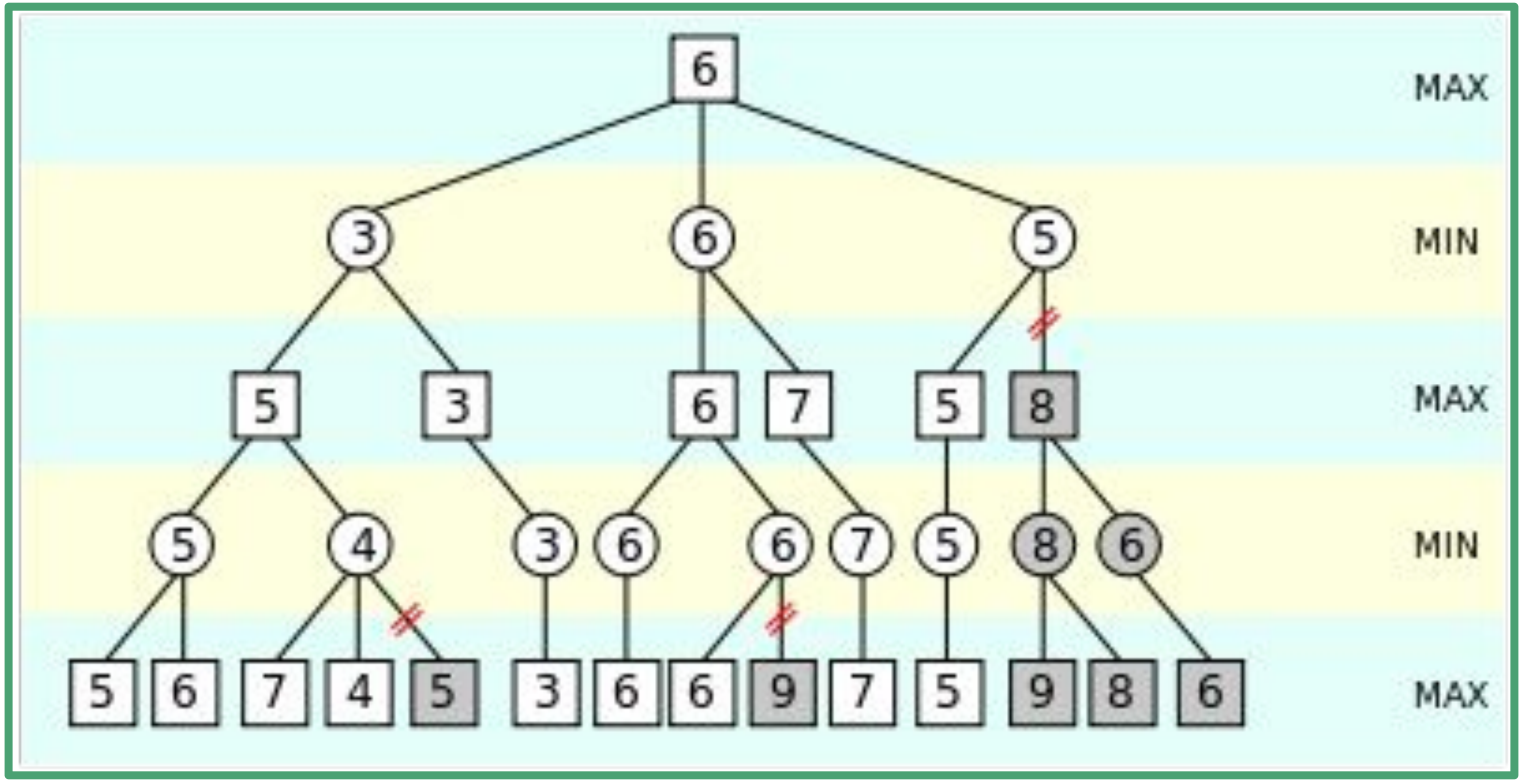


Keeping Track of Board States

- Lots of information stored in the board
- Deep copies were inefficient
- get_successor(action): store the action performed on a stack
- get_predecessor(): undo the action on top of the stack

Implementing my AI

Minimax



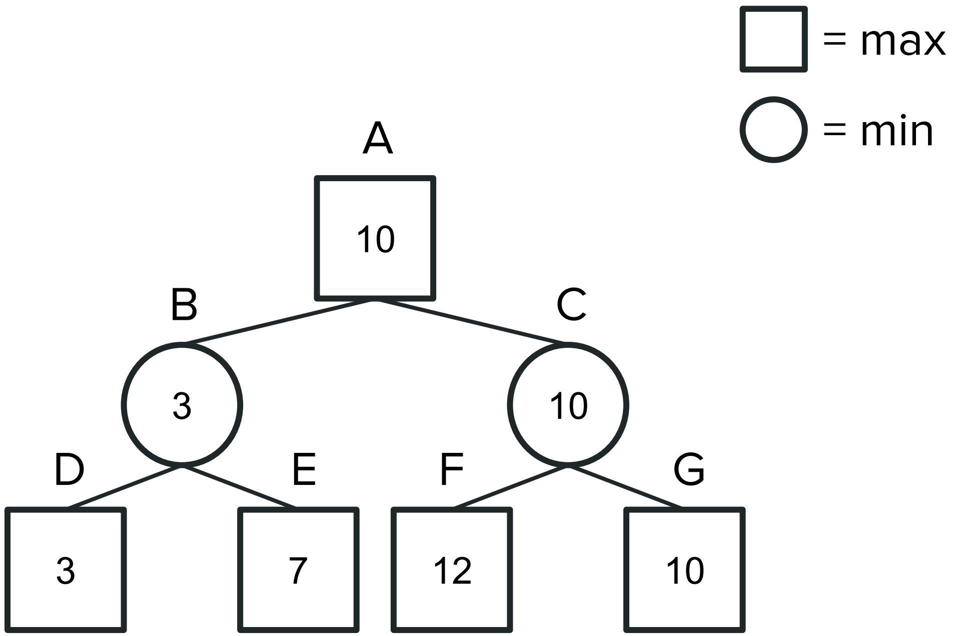
Adding Time Limitations

- Don’t want to wait forever for AI to make a move
- Cut off minimax early if it takes too long
- Use iterative deepening
- Store value found for each action at furthest depth

Iterative Deepening

Action	Value
A => B	3
A => C	10

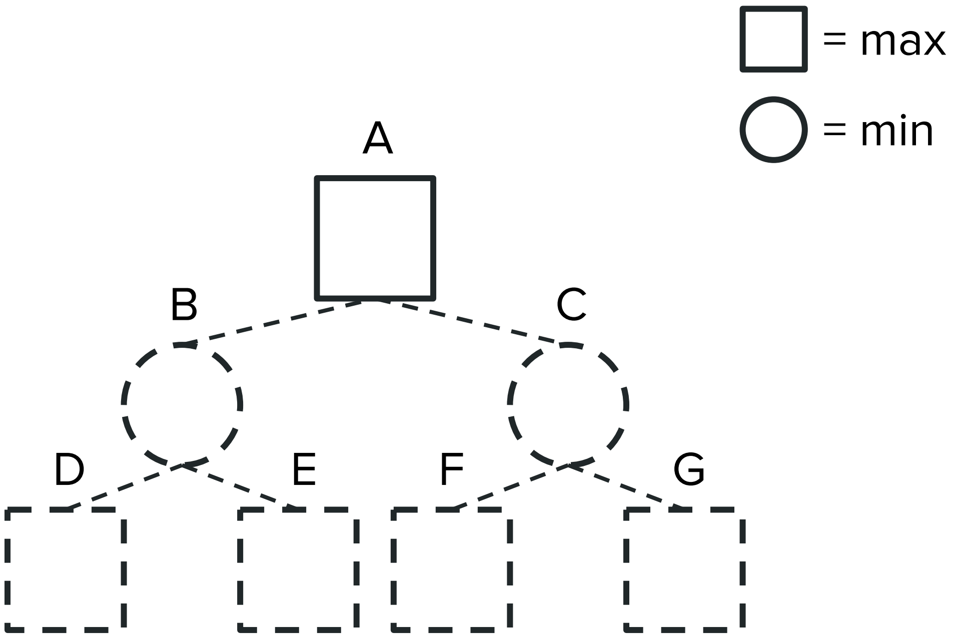
State	Sorted Actions
A	A => C, A => B
B	B => D, B => E
C	C => G, C => F



Iterative Deepening

Action	Value
A => B	3
A => C	10

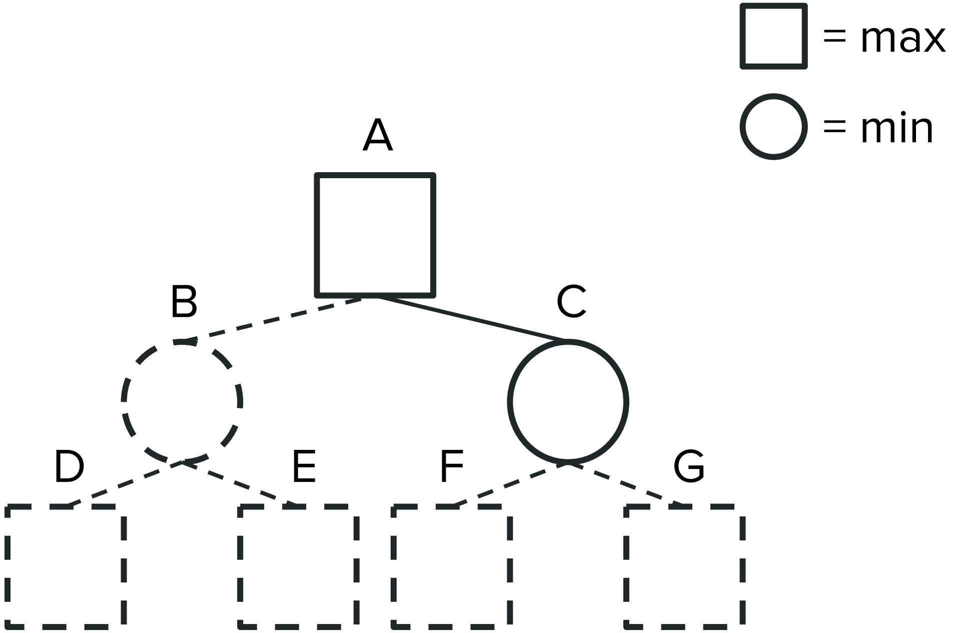
State	Sorted Actions
A	A => C, A => B
B	B => D, B => E
C	C => G, C => F



Iterative Deepening

Action	Value
A => B	3
A => C	10

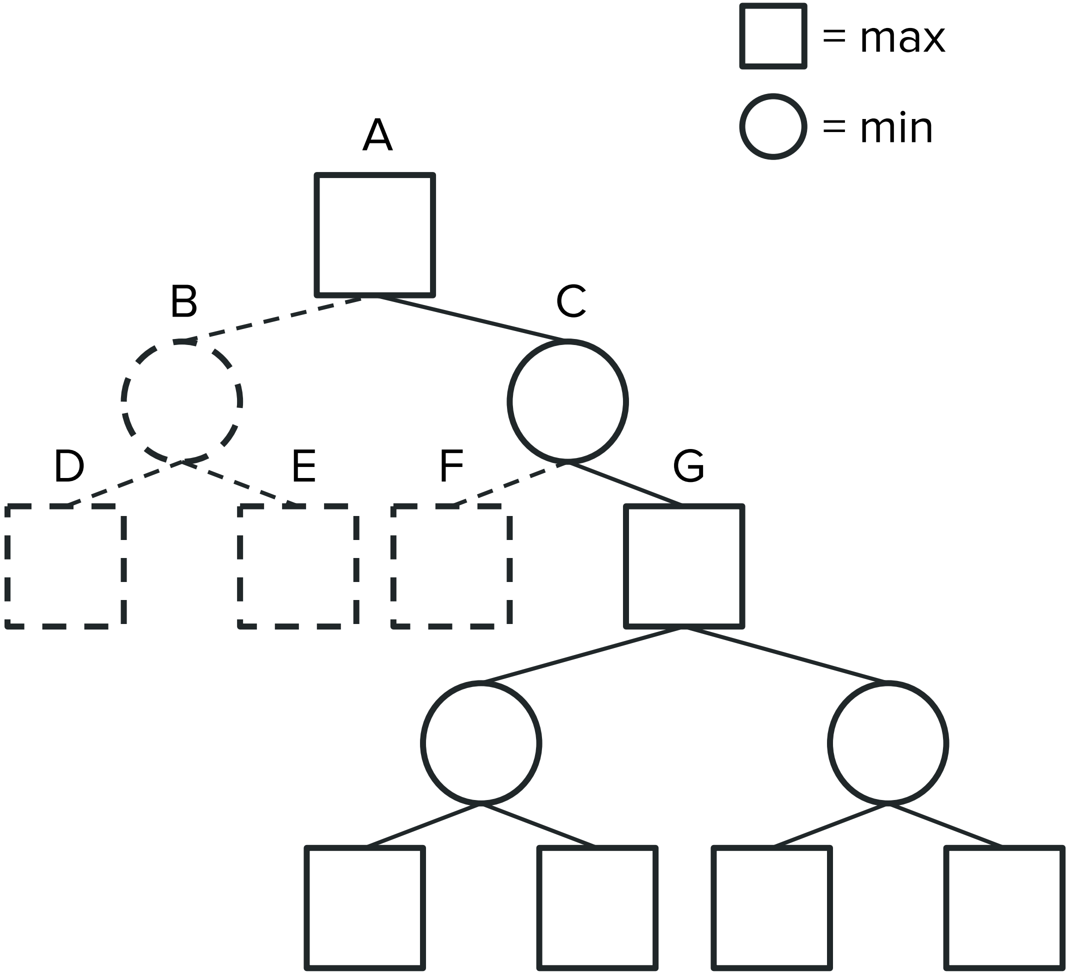
State	Sorted Actions
A	A => C, A => B
B	B => D, B => E
C	C => G, C => F



Iterative Deepening

Action	Value
A => B	3
A => C	10

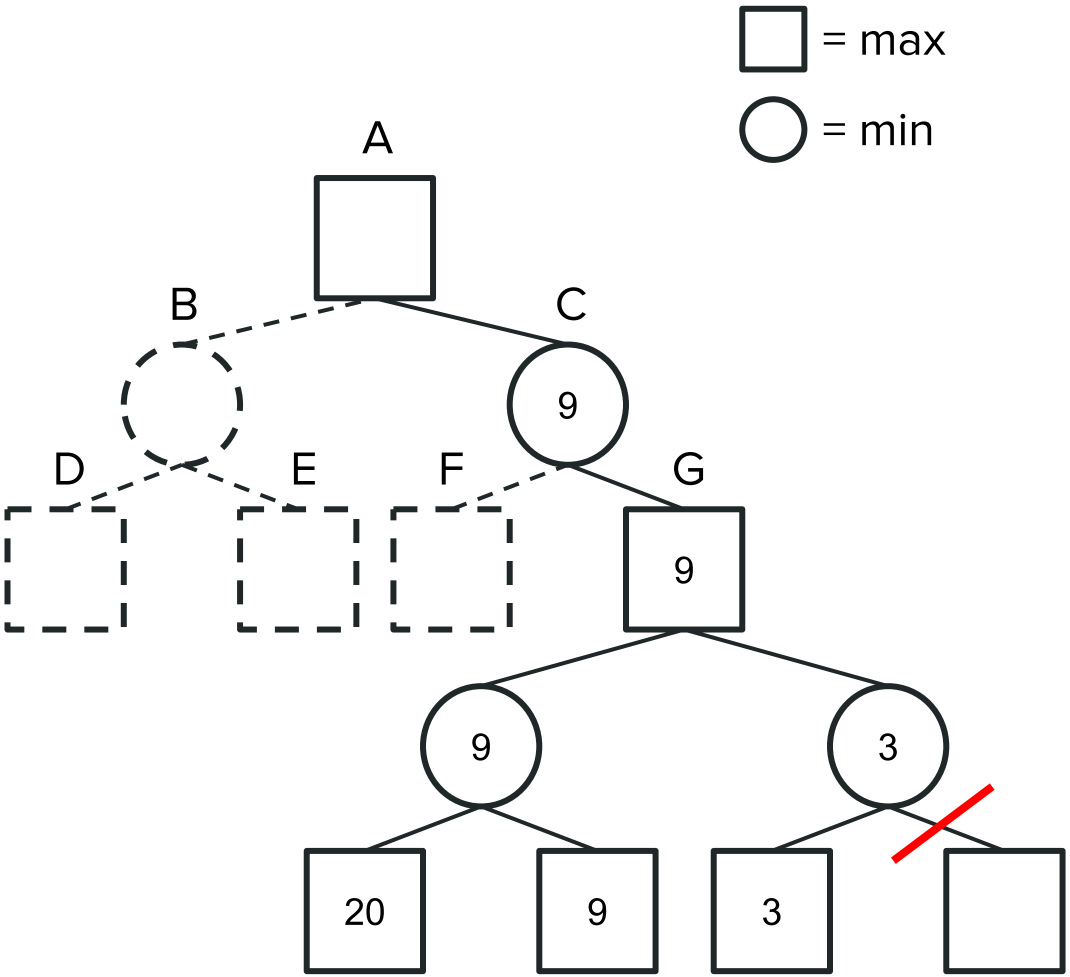
State	Sorted Actions
A	A => C, A => B
B	B => D, B => E
C	C => G, C => F



Iterative Deepening

Action	Value
A => B	3
A => C	10

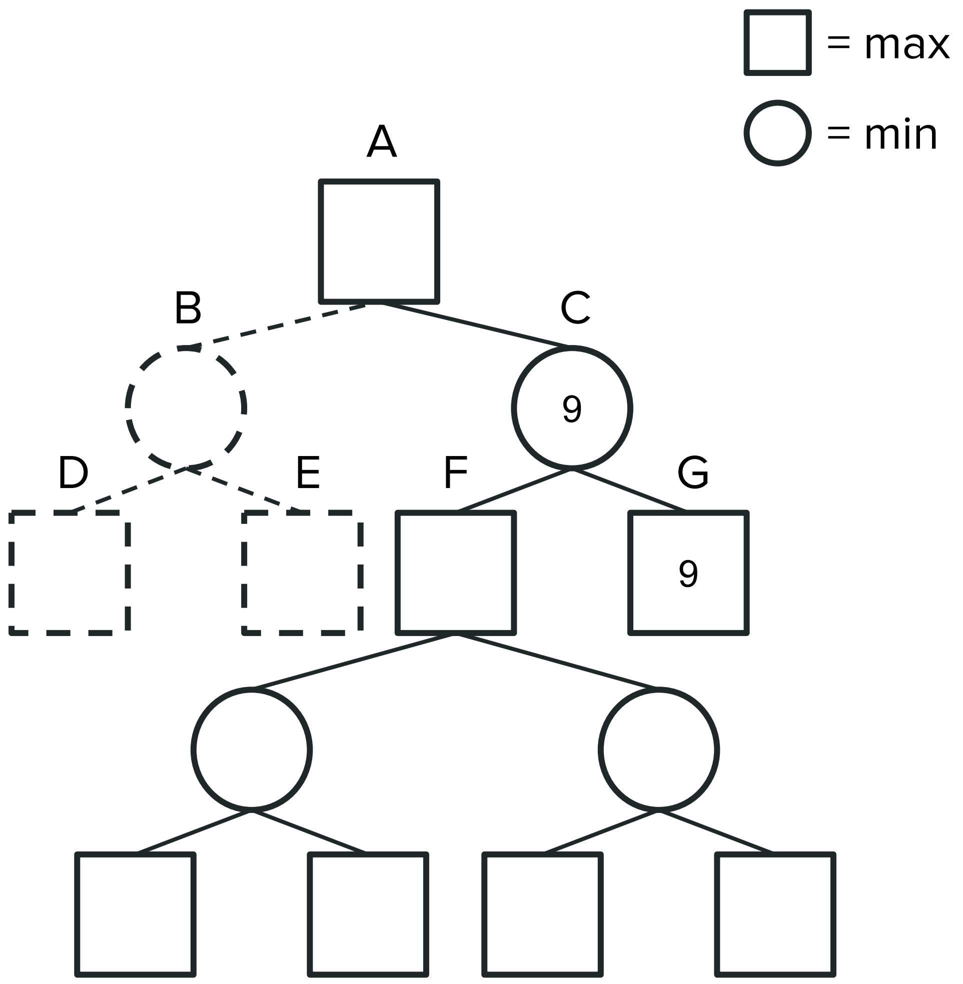
State	Sorted Actions
A	A => C, A => B
B	B => D, B => E
C	C => G, C => F



Iterative Deepening

Action	Value
A => B	3
A => C	10

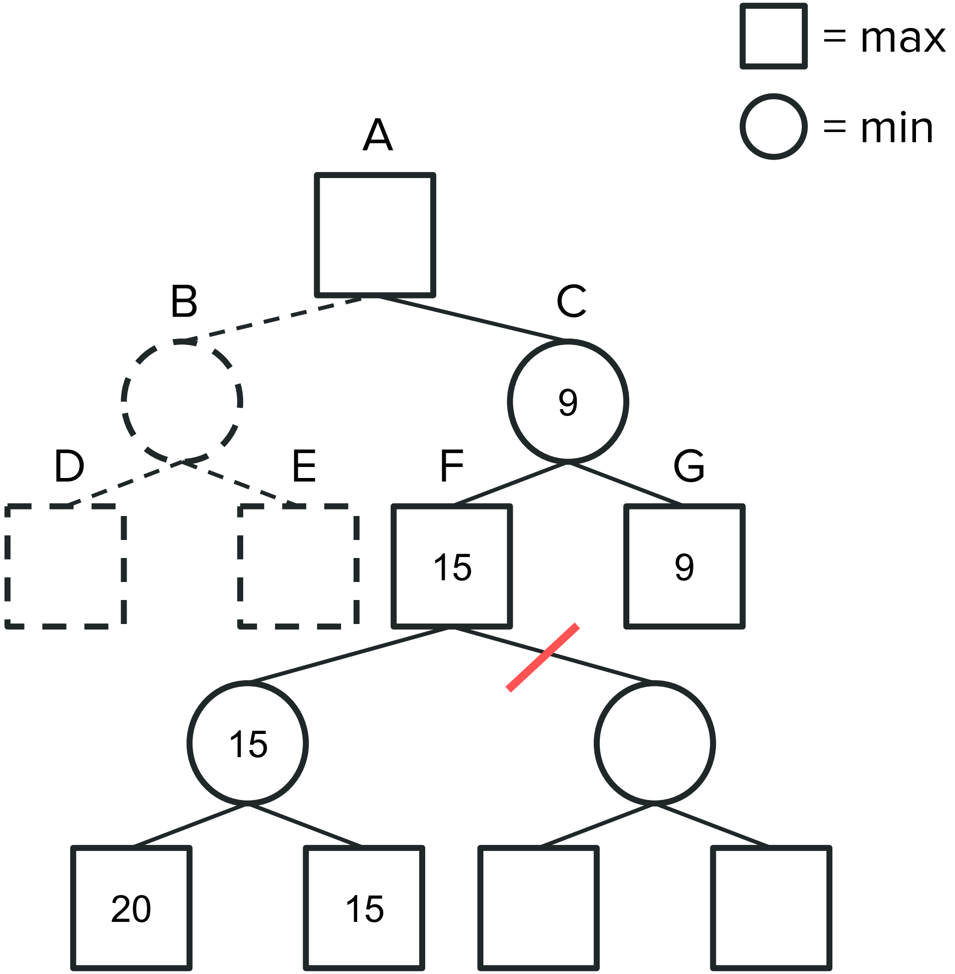
State	Sorted Actions
A	A => C, A => B
B	B => D, B => E
C	C => G, C => F



Iterative Deepening

Action	Value
A => B	3
A => C	10

State	Sorted Actions
A	A => C, A => B
B	B => D, B => E
C	C => G, C => F

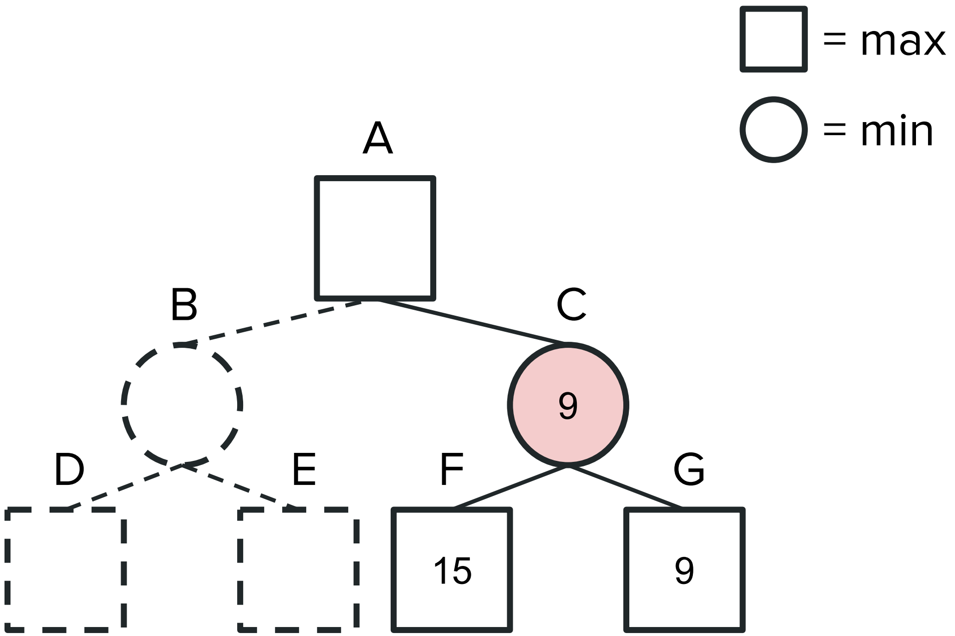


Iterative Deepening

Action	Value
A => B	3
A => C	9

<= Update

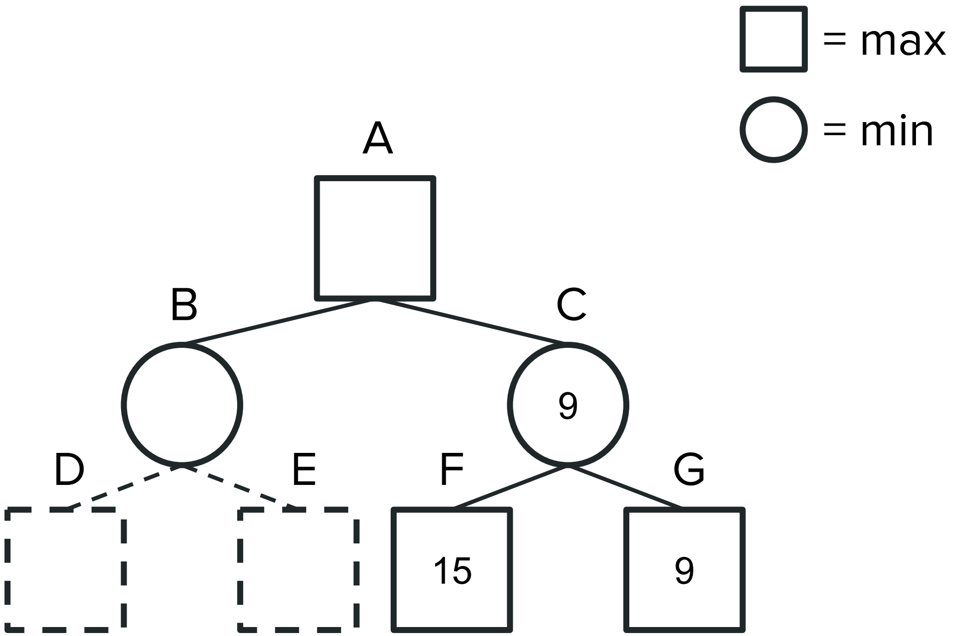
State	Sorted Actions
A	A => C, A => B
B	B => D, B => E
C	C => G, C => F



Iterative Deepening

Action	Value
A => B	3
A => C	9

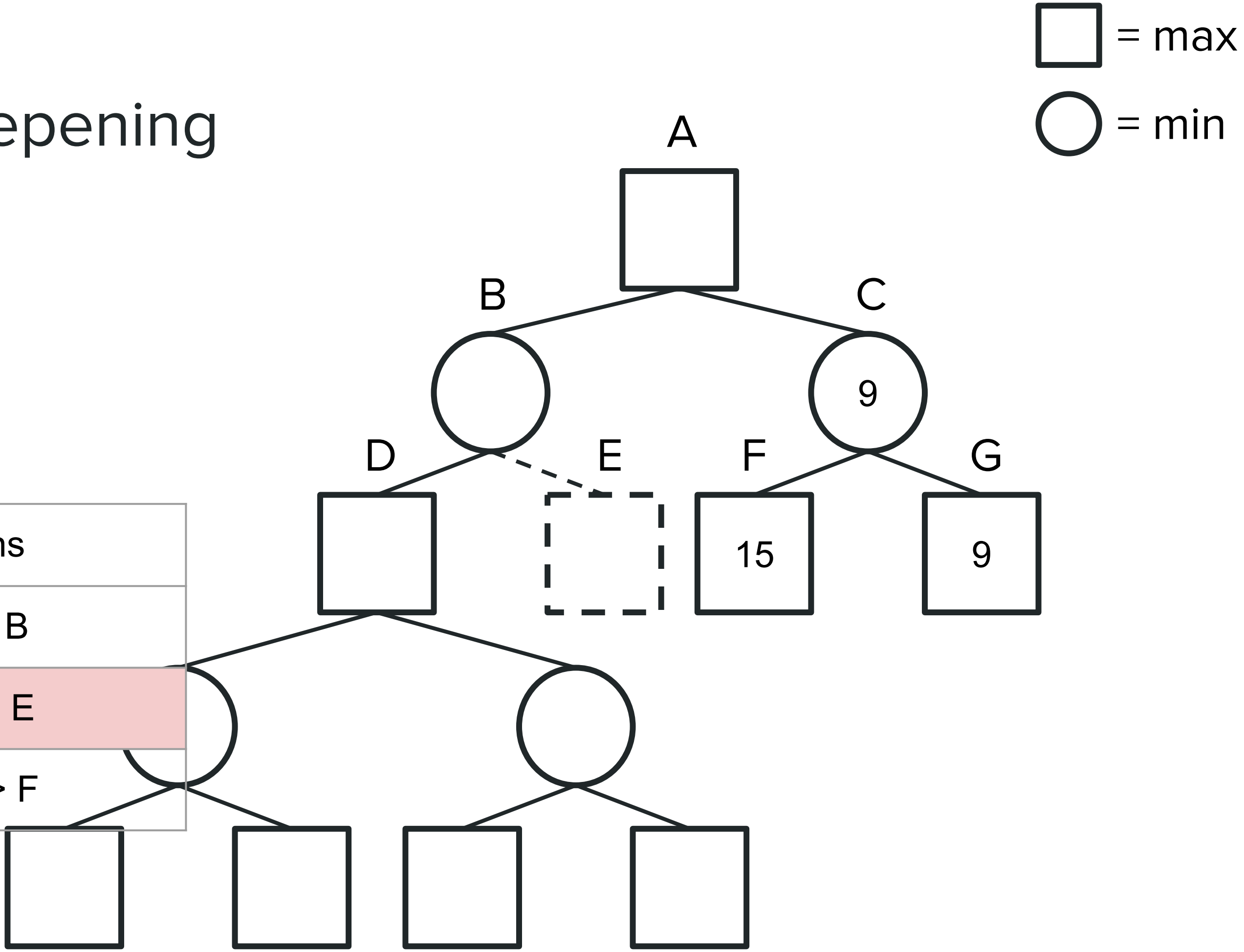
State	Sorted Actions
A	A => C, A => B
B	B => D, B => E
C	C => G, C => F



Iterative Deepening

Action	Value
A => B	3
A => C	9

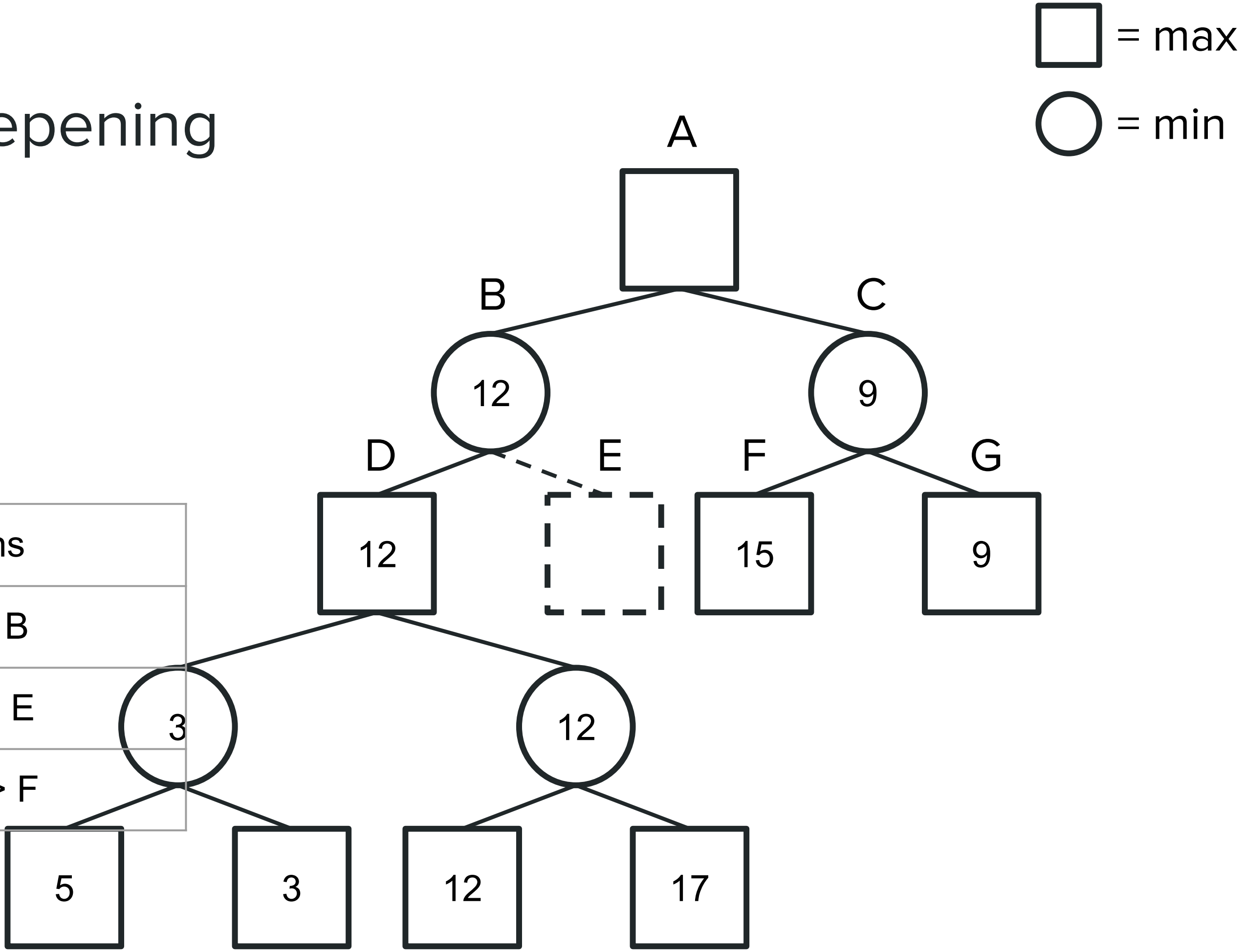
State	Sorted Actions
A	A => C, A => B
B	B => D, B => E
C	C => G, C => F



Iterative Deepening

Action	Value
A => B	3
A => C	9

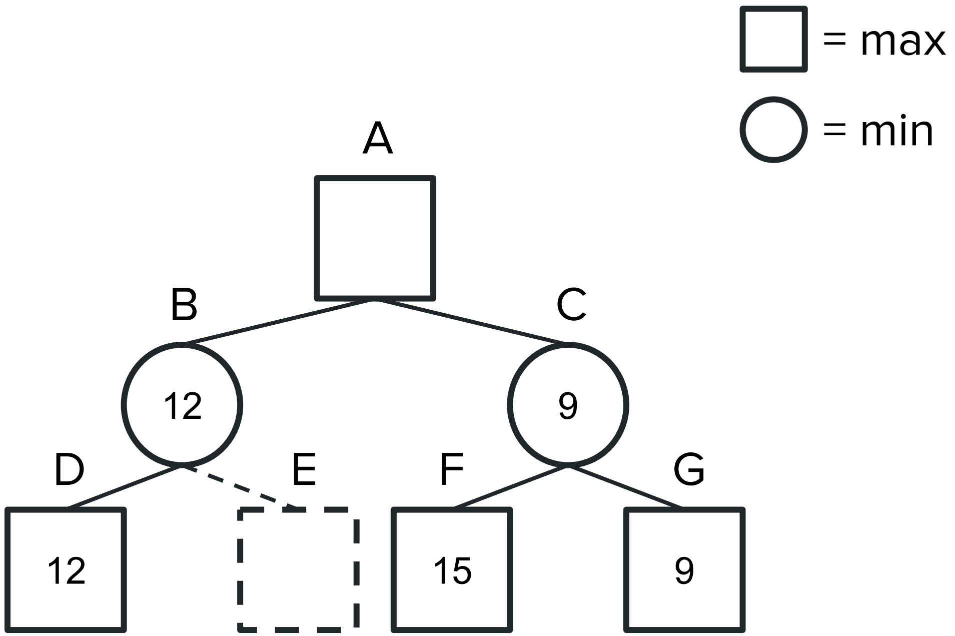
State	Sorted Actions
A	A => C, A => B
B	B => D, B => E
C	C => G, C => F



Iterative Deepening

Action	Value
A => B	3
A => C	9

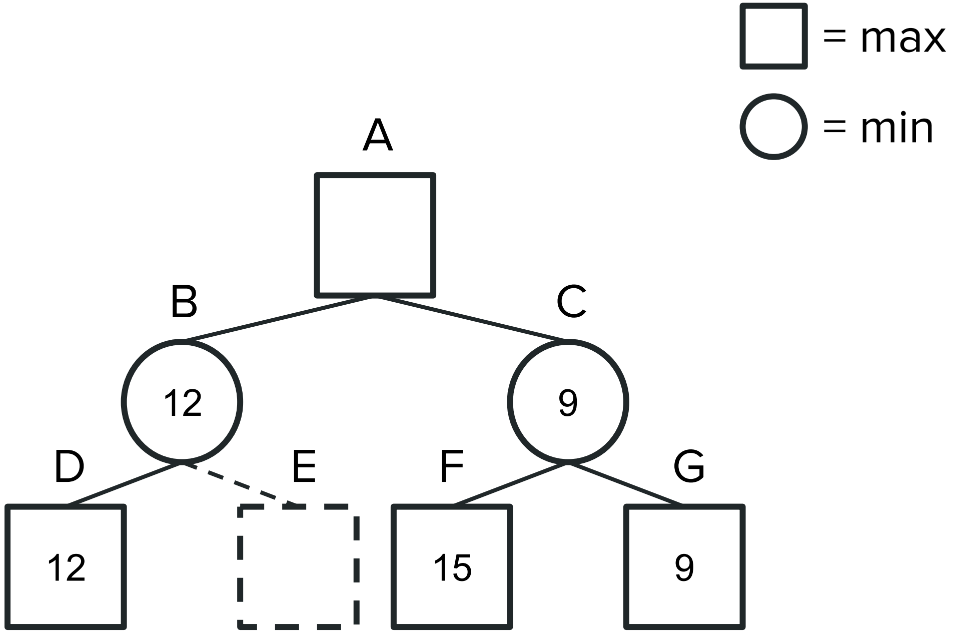
State	Sorted Actions
A	A => C, A => B
B	B => D, B => E
C	C => G, C => F



Iterative Deepening

Action	Value
A => B	3
A => C	9

State	Sorted Actions
A	A => C, A => B
B	B => D, B => E
C	C => G, C => F



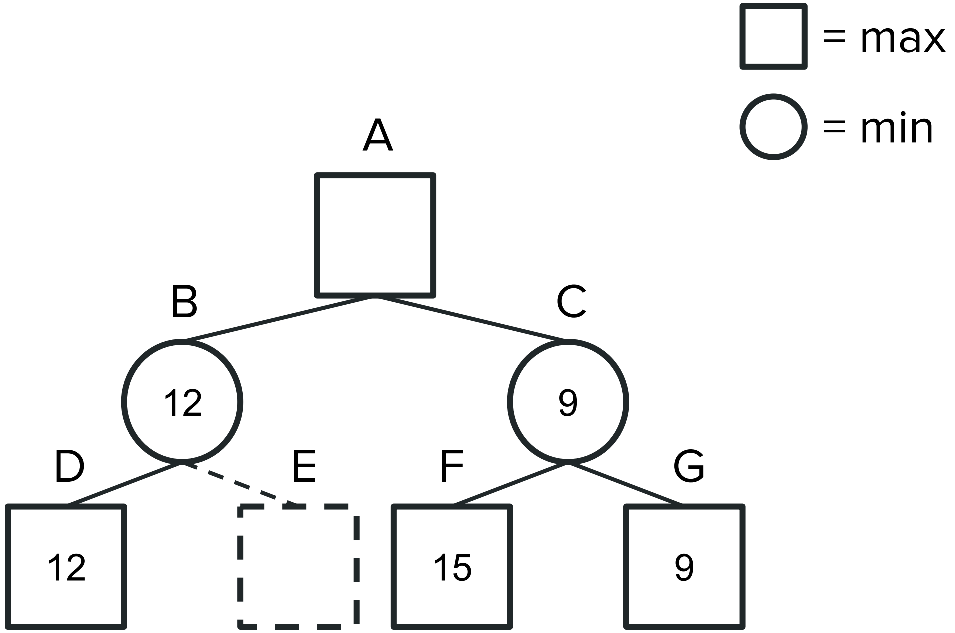
Wait! Time’s up!!!

Iterative Deepening

Action	Value
A => B	3
A => C	9

<= Don't update!

State	Sorted Actions
A	A => C, A => B
B	B => D, B => E
C	C => G, C => F



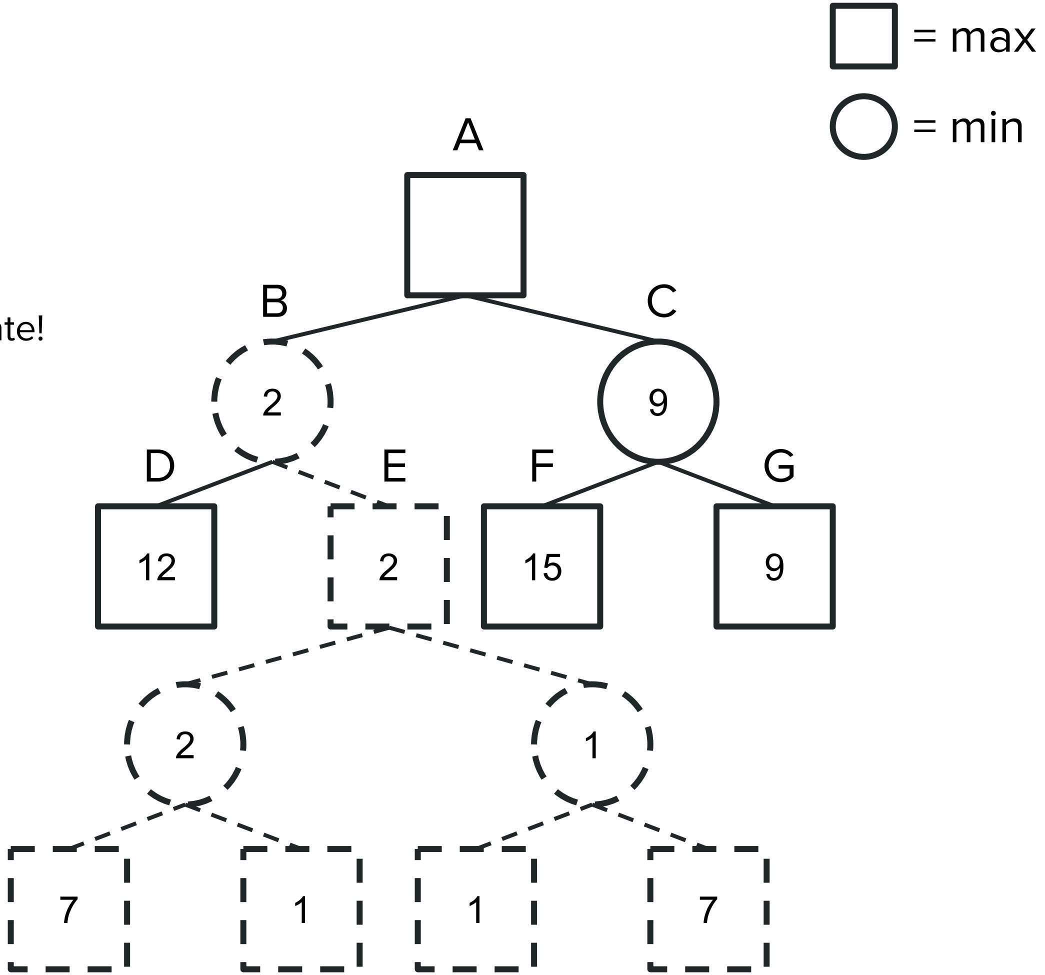
Wait! Time's up!!!

Iterative Deepening

Action	Value
A => B	3
A => C	9

<= Don’t update!

State	Sorted Actions
A	A => C, A => B
B	B => D, B => E
C	C => G, C => F

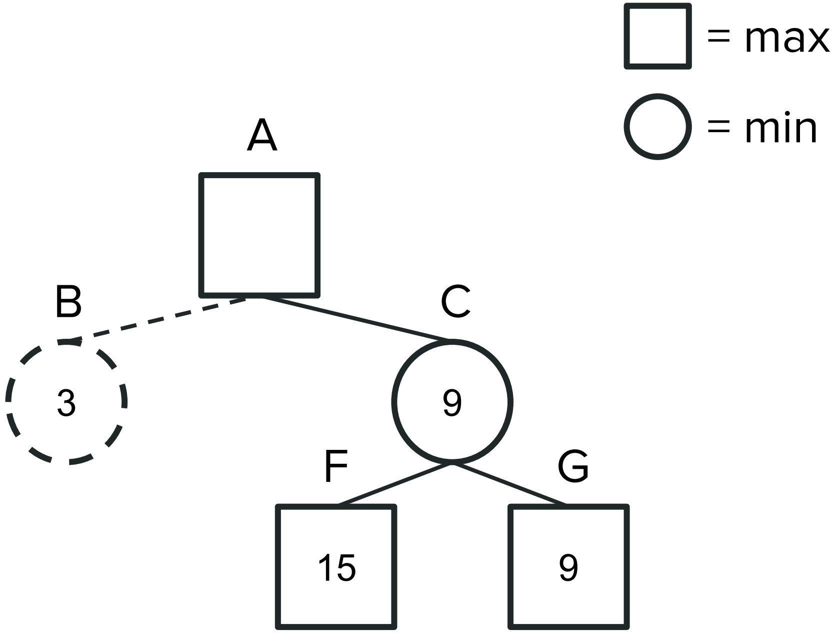


Iterative Deepening

Action	Value
A => B	3
A => C	9

<= Don't update!

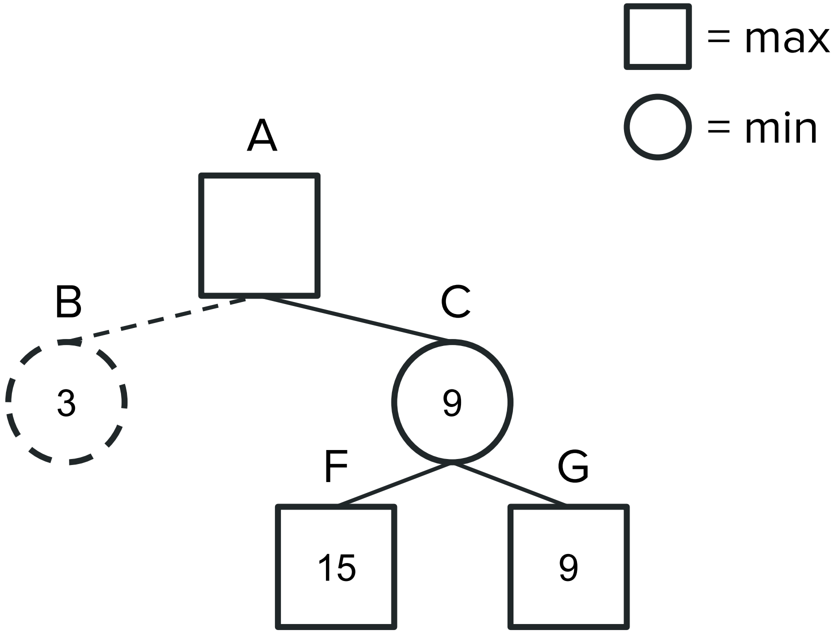
State	Sorted Actions
A	A => C, A => B
B	B => D, B => E
C	C => G, C => F



Iterative Deepening

Action	Value
A => B	3
A => C	9

State	Sorted Actions
A	A => C, A => B
B	B => D, B => E
C	C => G, C => F



Result: Action A => C

Additional Improvements

- Immediately return winning moves
- Check one move further if there are 5 pieces around the opponent’s Queen Bee

Utility Function

Utility	**Value**
● Allied Pieces around Opponent’s Queen Bee (QB)	Really Good!
● Enemy Pieces around Opponent’s QB ● Allied Pieces that *can* move to Opponent’s QB	Good
● Allied Pieces that *cannot* move to Opponent’s QB	Bad
● Pieces that cannot move	Really Bad

Demo

https://github.com/wdreames/hive_board_game_ai

Analysis and Conclusion

Analysis

- AI is able to beat me! (sometimes)
- AI can look 2 turns in the future (4 ply)
- Reduces number of actions to process from millions to a couple hundred thousand

What I learned

- Iterative deepening with minimax
- Setting a time limit for minimax

Conclusion

- Successfully created an AI that can play Hive
- Implemented minimax with iterative deepening
- Fairly happy with the results of this project
- Github repo: https://github.com/wdreames/hive_board_game_ai

Questions?