Last-Branch and Speculative Pruning Algorithms for Max^n

Nathan Sturtevant
UCLA Computer Science Department*

*soon to be University of Alberta

Problem Overview

Problem Overview

- There have been notable successes in developing expert-level 2-player games

Problem Overview

- There have been notable successes in developing expert-level 2-player games
- We'd like to develop programs to play multi-player games well

Minimax

Minimax

- Most commonly used 2-player decision rule

Minimax

- Most commonly used 2-player decision rule
- Implemented with alpha-beta pruning

Minimax

- Most commonly used 2-player decision rule
- Implemented with alpha-beta pruning
  - In best case, alpha-beta reduces tree size from b^d to $b^{d/2}$

Minimax

- Most commonly used 2-player decision rule
- Implemented with alpha-beta pruning
  - In best case, alpha-beta reduces tree size from b^d to $b^{d/2}$
  - Approach best case in practice by ordering nodes well

Max^n

Max^n

- Generalization of minimax to *n* players

Max^n

- Generalization of minimax to *n* players
  - Luckhardt and Irani, 1986

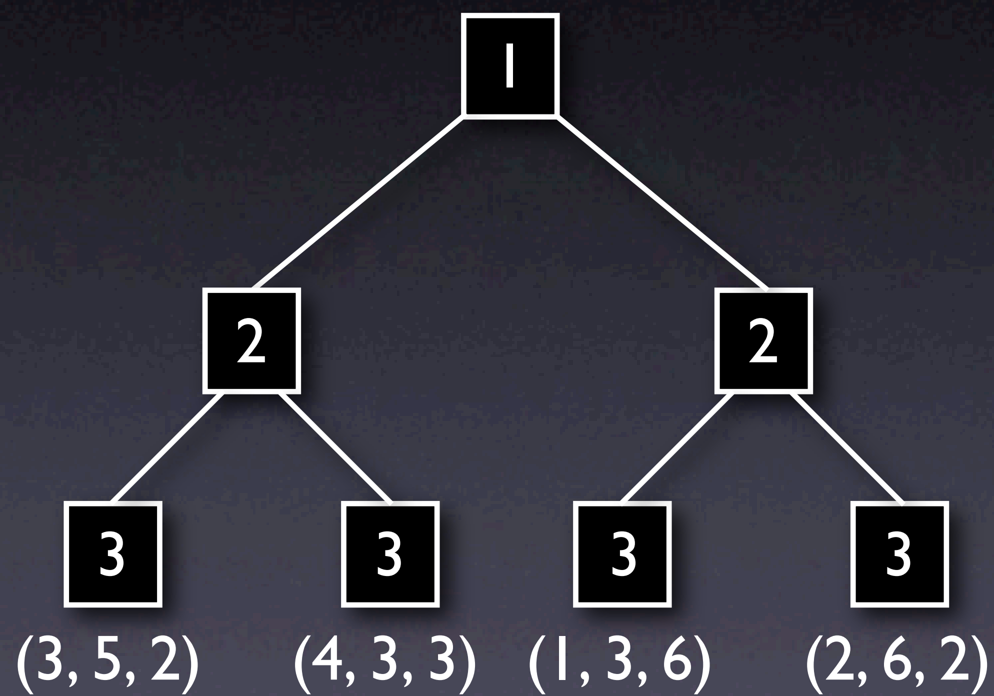
Max^n

- Generalization of minimax to *n* players
  - Luckhardt and Irani, 1986
- Static evaluation returns *n*-tuple of scores

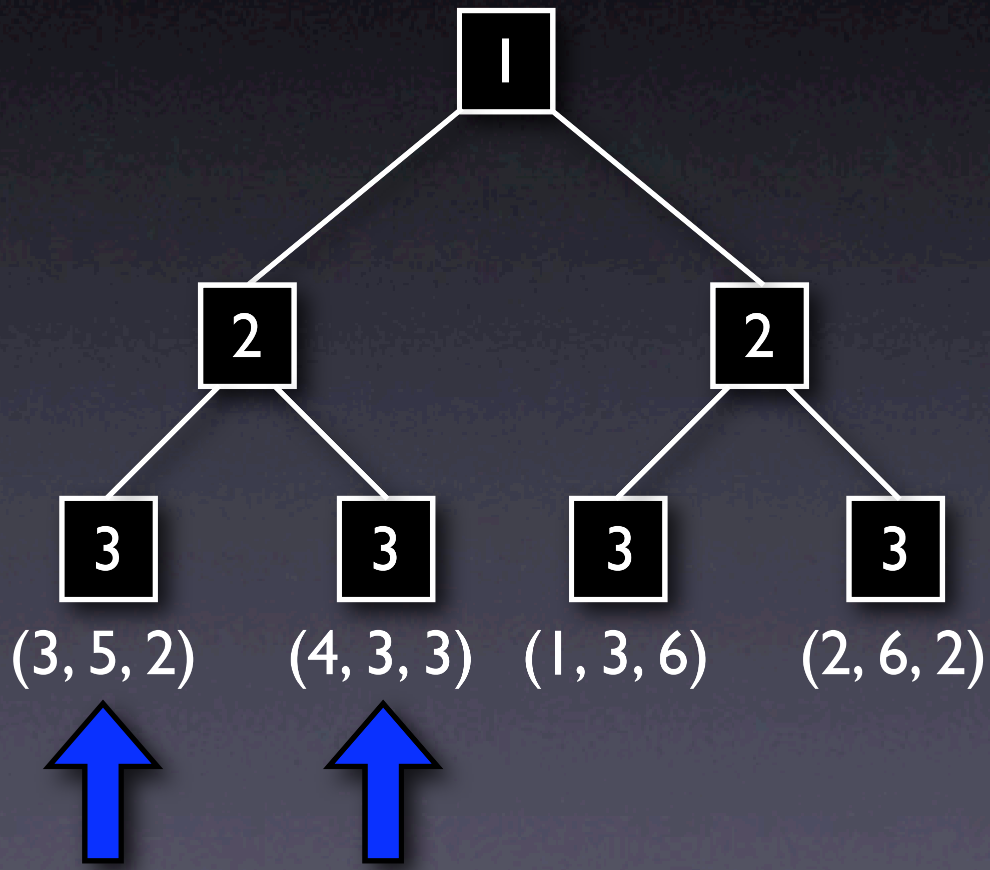
Max^n

- Generalization of minimax to *n* players
  - Luckhardt and Irani, 1986
- Static evaluation returns *n*-tuple of scores
  - Each player tries to maximize their own component of the *n*-tuple

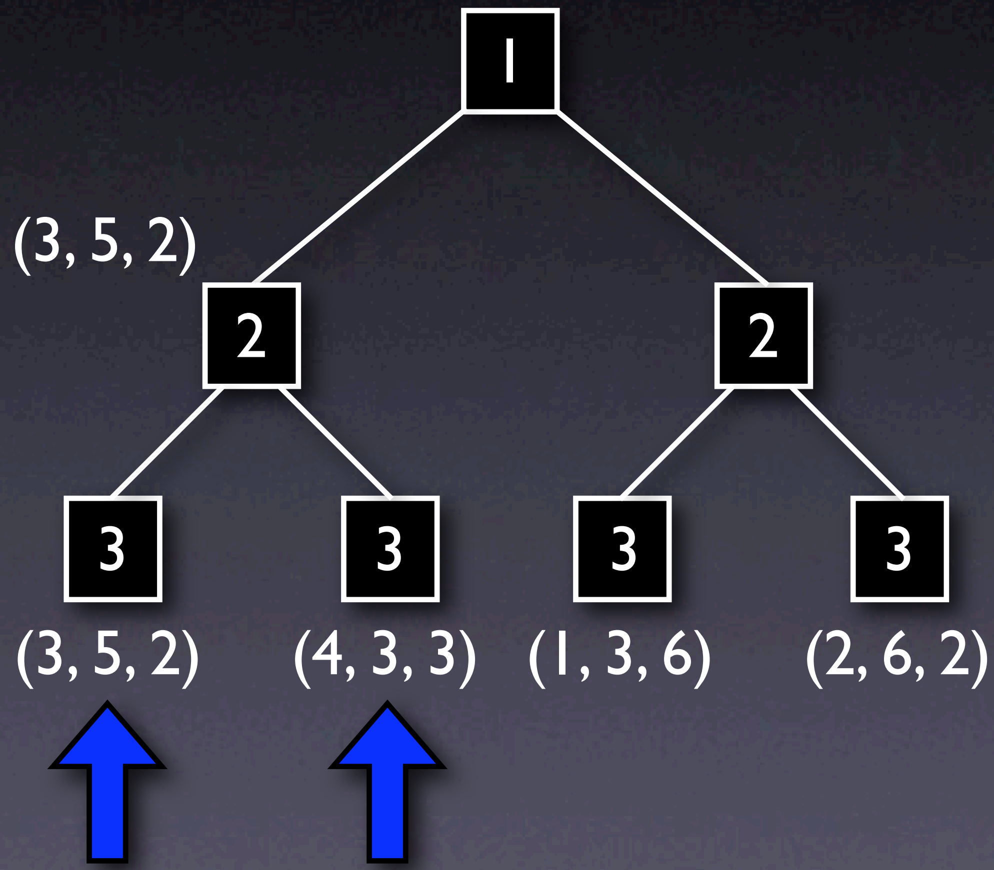
Max^n Decision Rule



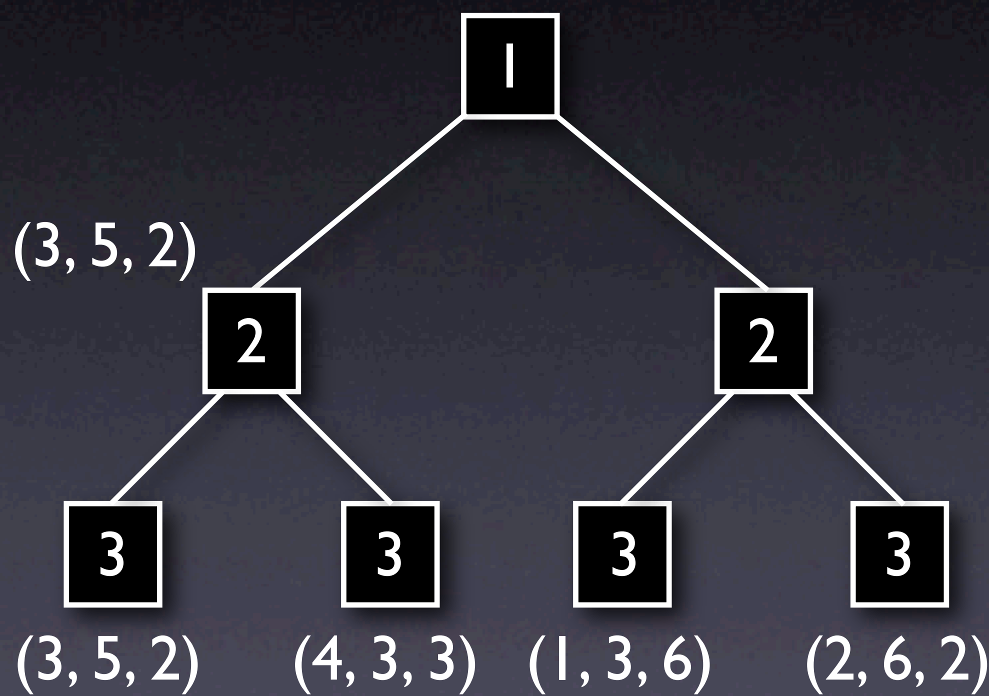
Max^n Decision Rule



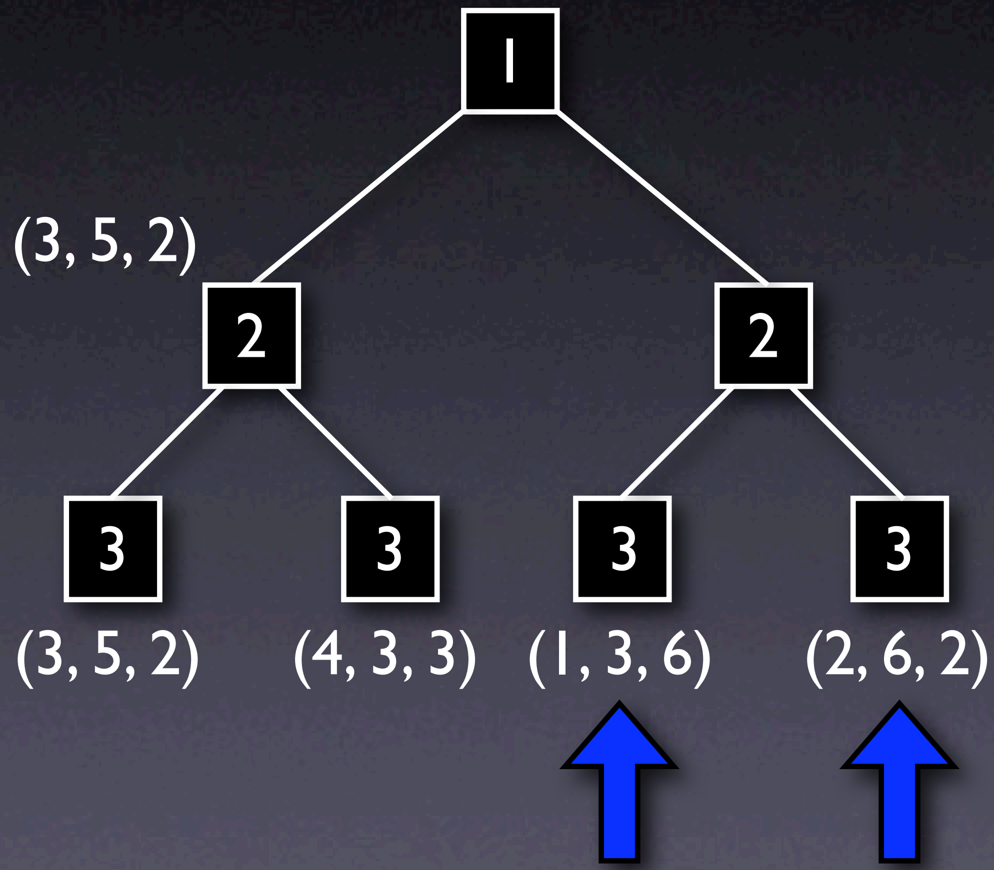
Max^n Decision Rule



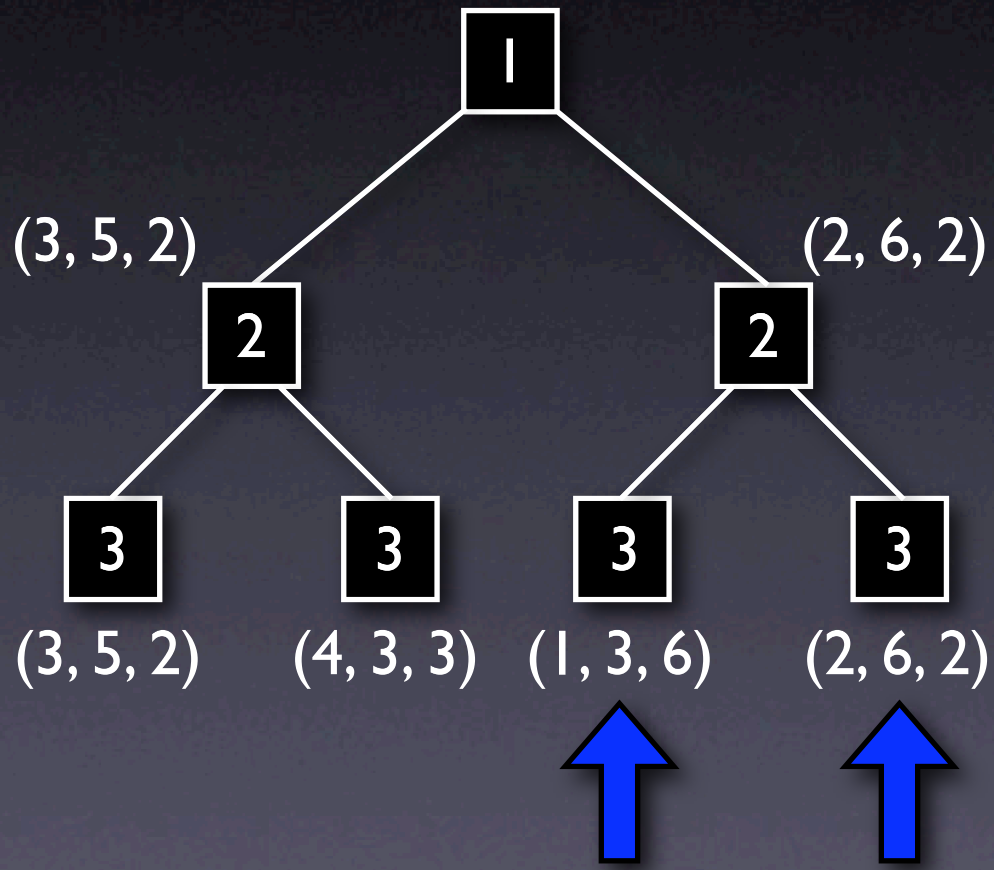
Max^n Decision Rule



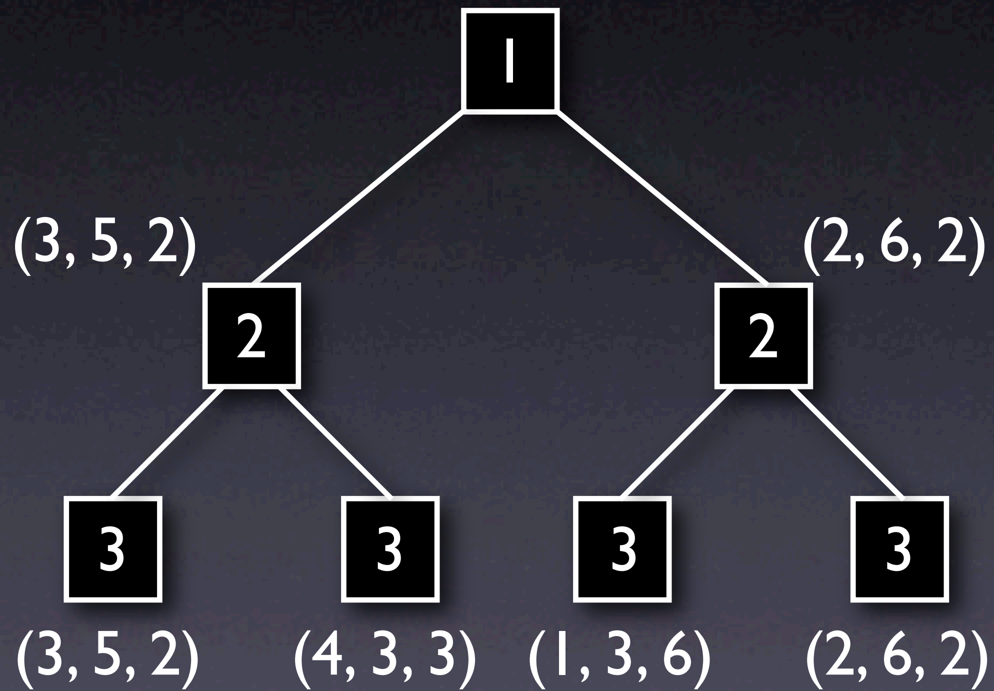
Max^n Decision Rule



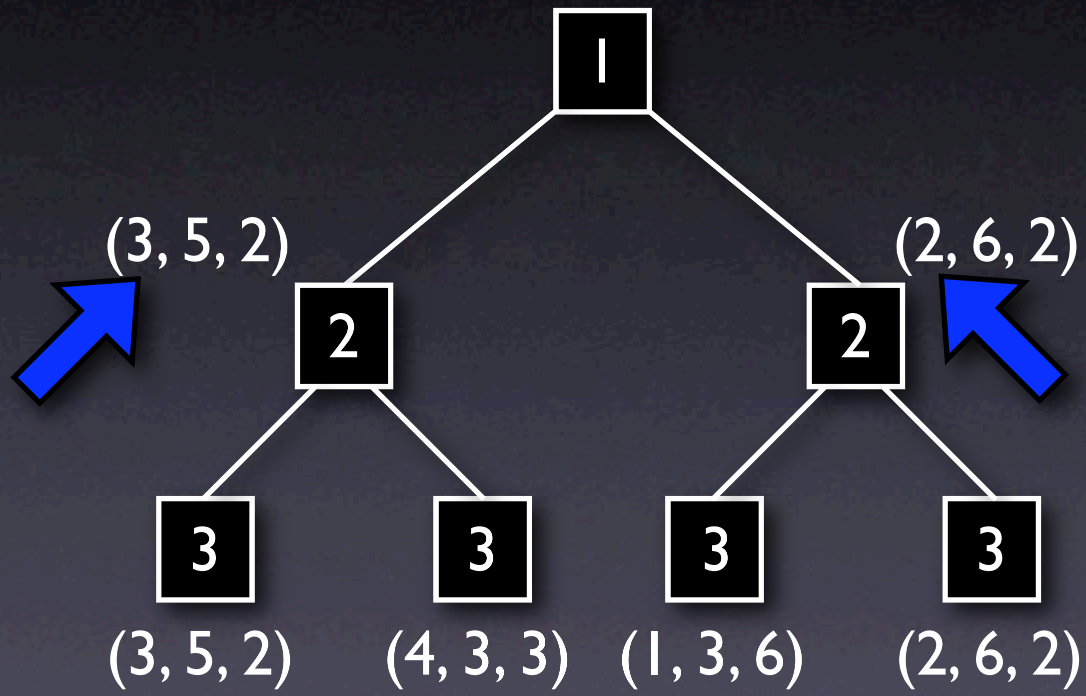
Max^n Decision Rule



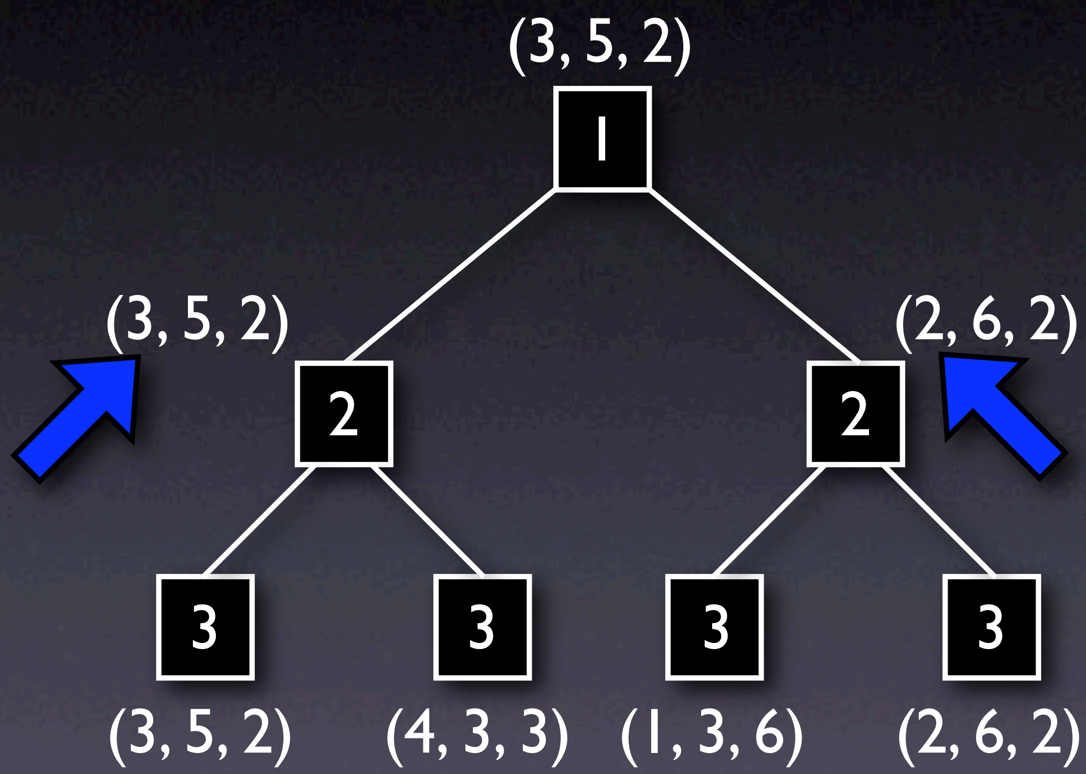
Max^n Decision Rule



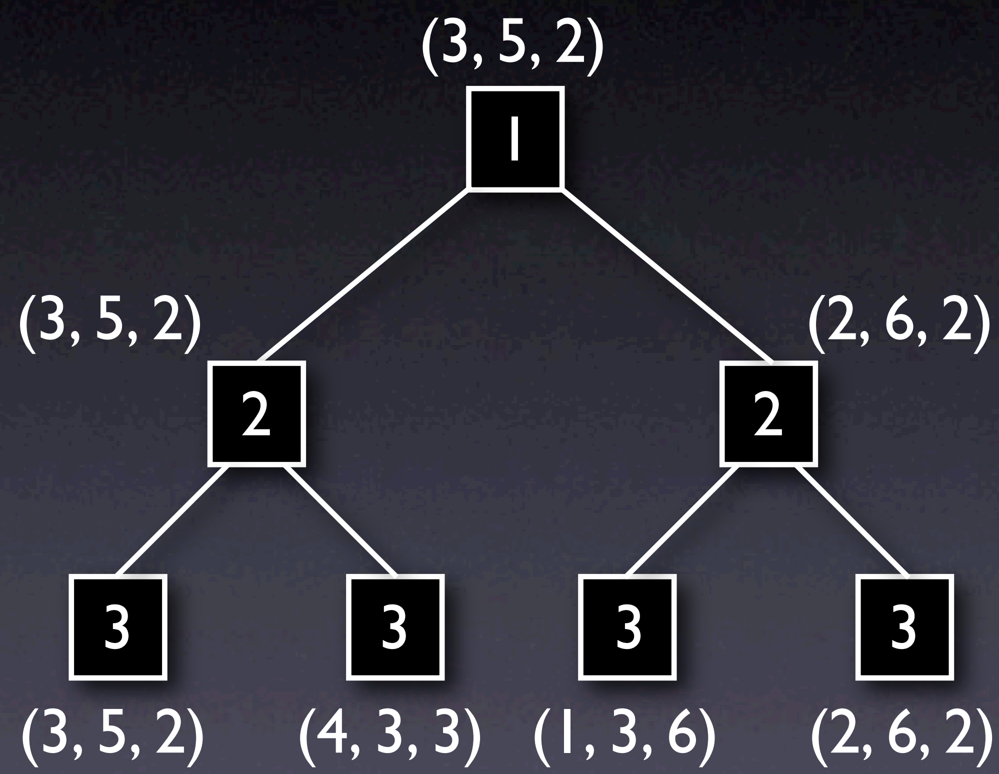
Max^n Decision Rule



Max^n Decision Rule



Max^n Decision Rule



Outline

- Max^n Decision Rule
- Max^n Pruning Techniques
- Experimental Results
- Conclusions

Previous Max^n Pruning

Previous Max^n Pruning

- Shallow Pruning (Korf, 1991)

Previous Max^n Pruning

- Shallow Pruning (Korf, 1991)
- Alpha-Beta Branch and Bound Pruning (Sturtevant and Korf, 2000)

Previous Max^n Pruning

- Shallow Pruning (Korf, 1991)
- Alpha-Beta Branch and Bound Pruning (Sturtevant and Korf, 2000)
- Are not always effective and/or applicable

7

Previous Max^n Pruning

- Shallow Pruning (Korf, 1991)
- Alpha-Beta Branch and Bound Pruning (Sturtevant and Korf, 2000)
- Are not always effective and/or applicable
  - Effectiveness depends on *both* node ordering and static evaluation function

Max^n Pruning

Max^n Pruning

- Assume at least:

Max^n Pruning

- Assume at least:
  - Lower bound on each score (0)

Max^n Pruning

- Assume at least:
  - Lower bound on each score (0)
  - Upper bound on sum of all scores (*maxsum*)

Max^n Pruning

- Assume at least:
  - Lower bound on each score (0)
  - Upper bound on sum of all scores (*maxsum*)
- If the sum of scores always equals *maxsum*, the game is *constant-sum*

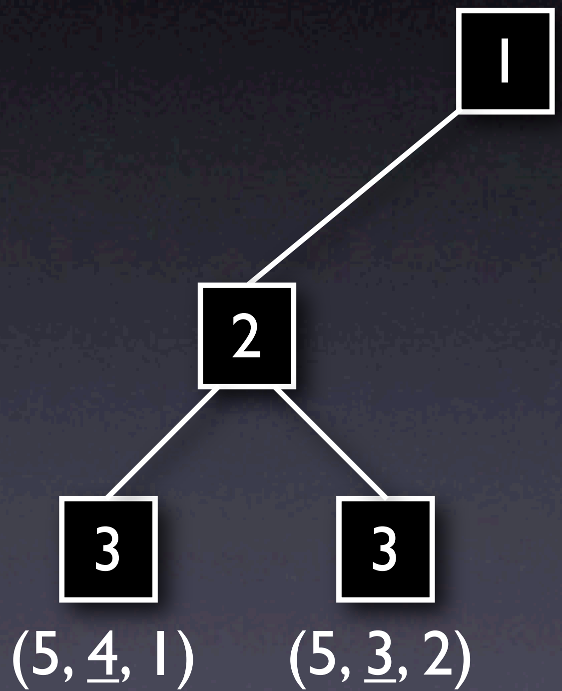
Shallow Pruning

maxsum = 10

1

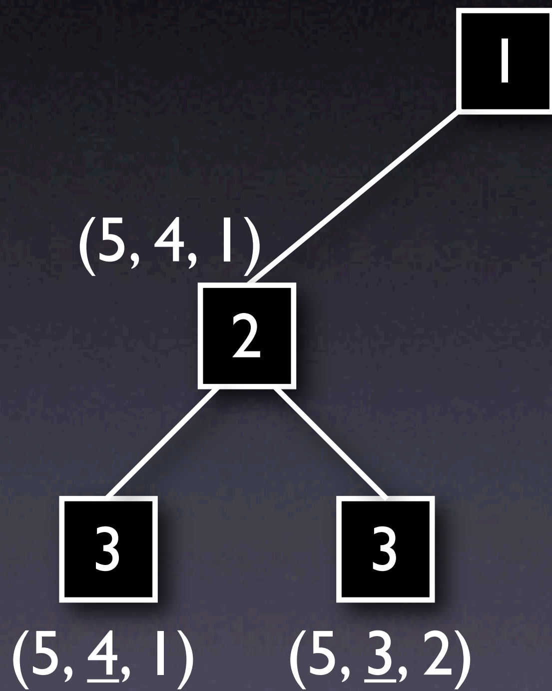
Shallow Pruning

maxsum = 10



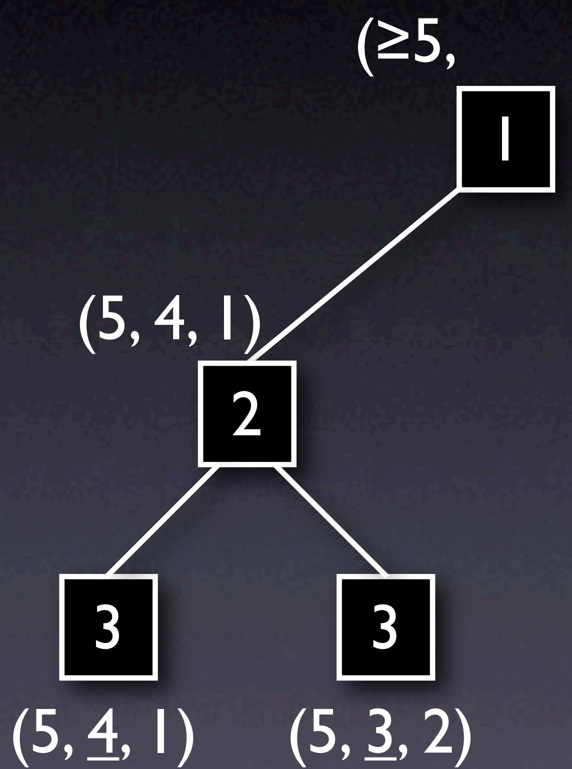
Shallow Pruning

maxsum = 10



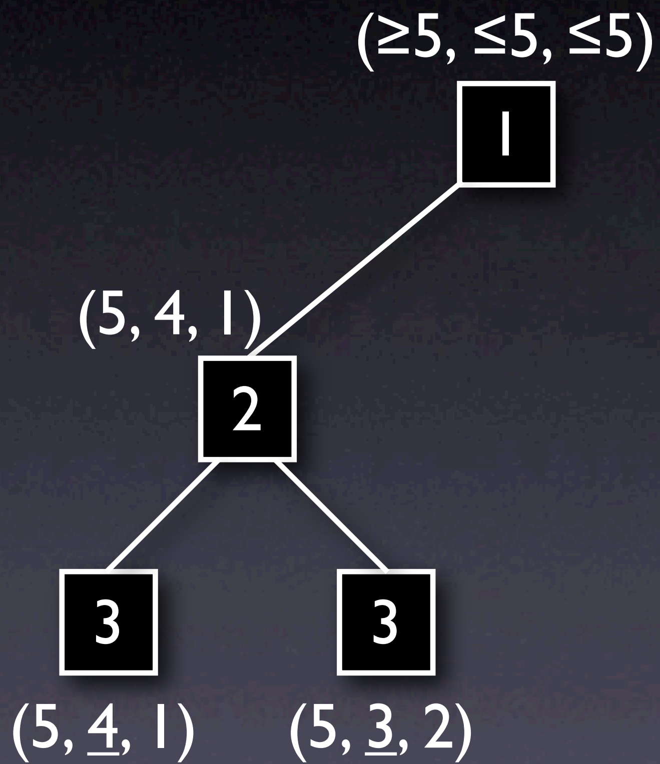
Shallow Pruning

maxsum = 10



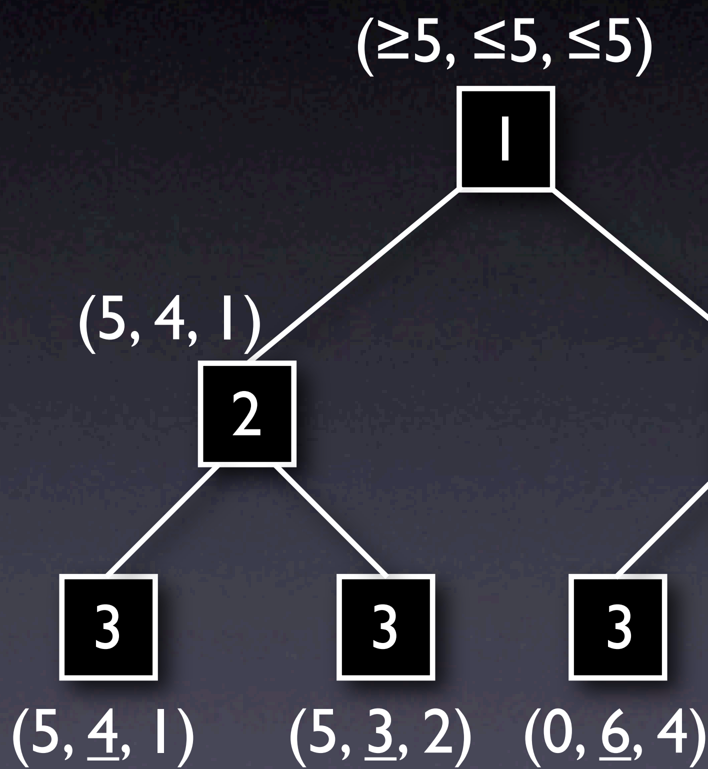
Shallow Pruning

maxsum = 10



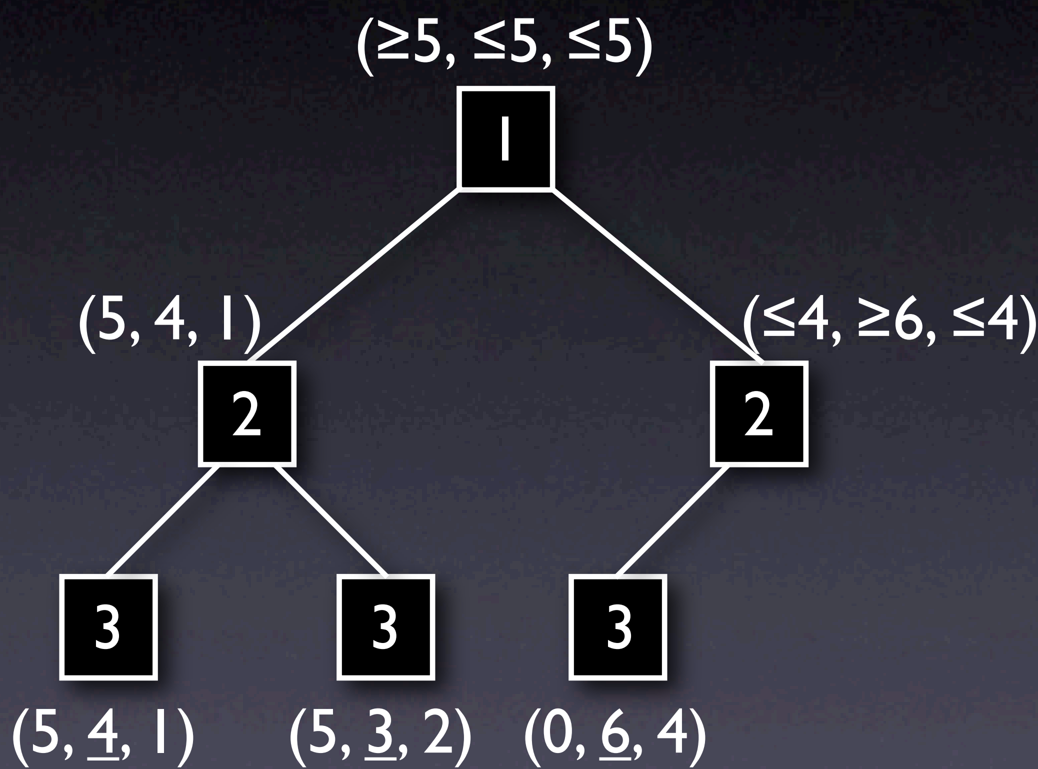
Shallow Pruning

maxsum = 10



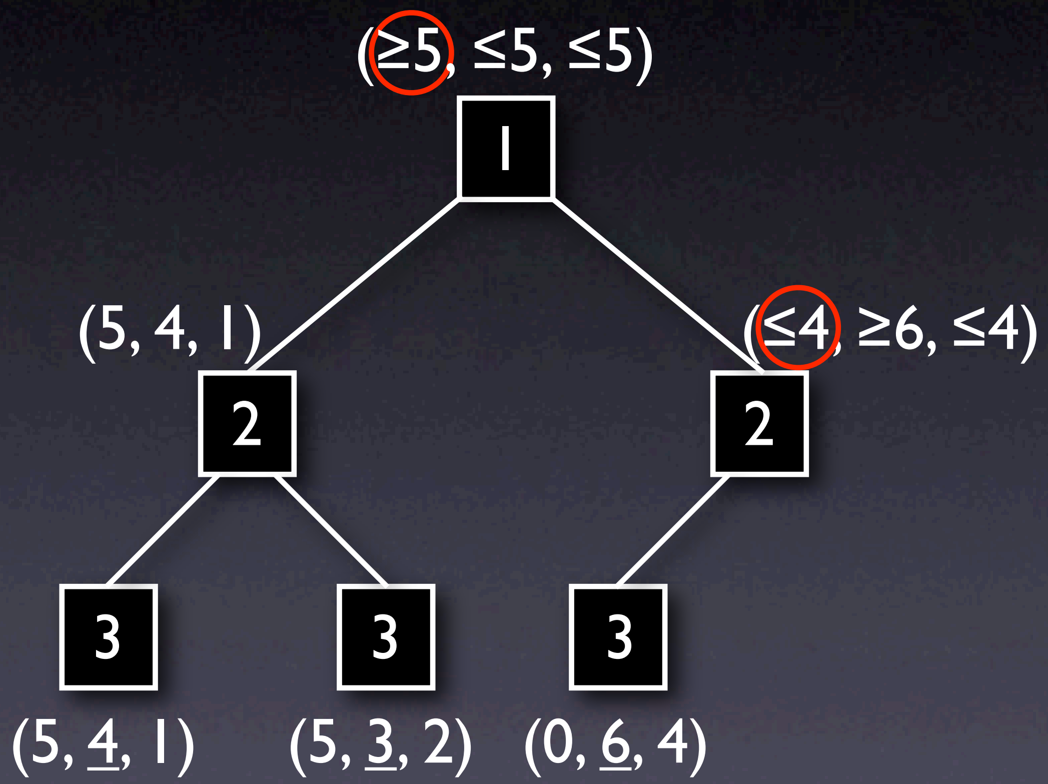
Shallow Pruning

maxsum = 10



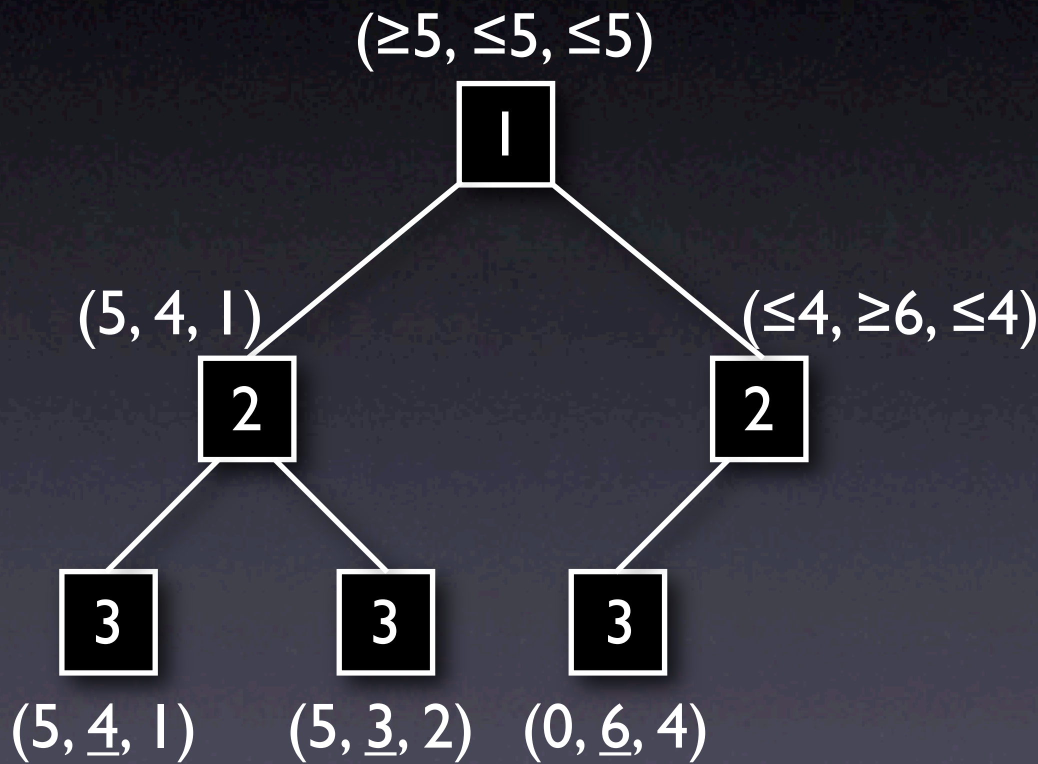
Shallow Pruning

maxsum = 10



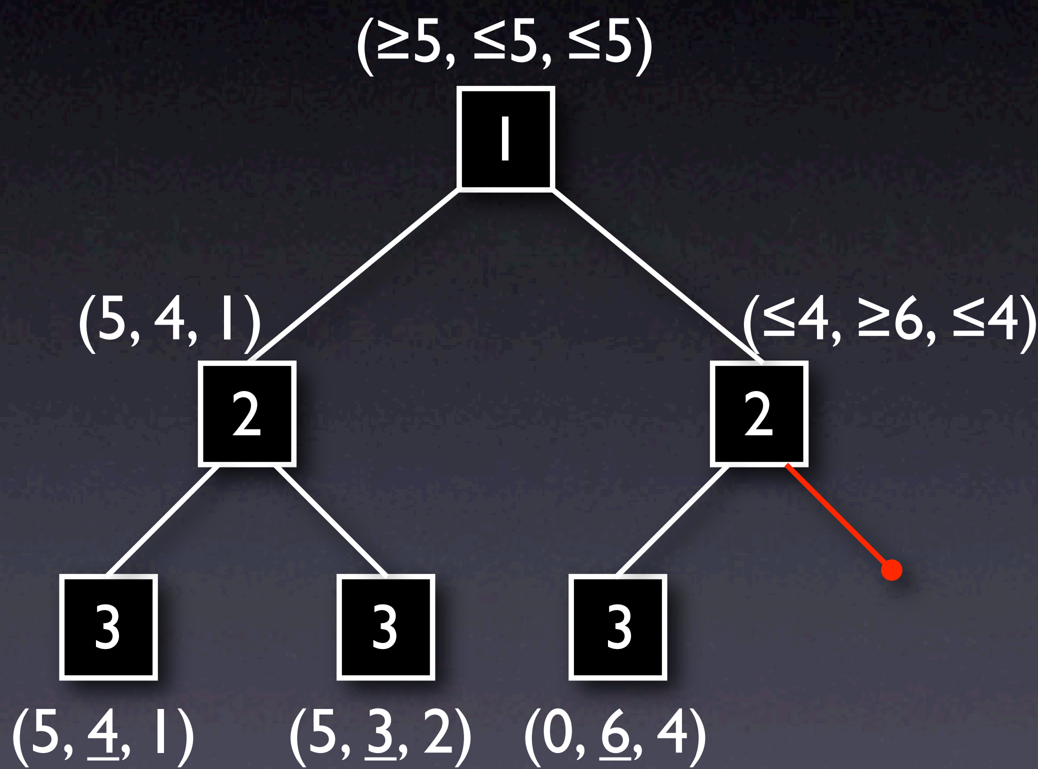
Shallow Pruning

maxsum = 10



Shallow Pruning

maxsum = 10



Shallow Pruning

Shallow Pruning

- Average case model predicts no asymptotic gain (Korf, 1991)

10

Shallow Pruning

- Average case model predicts no asymptotic gain (Korf, 1991)
- Dependent on properties of evaluation function

10

Shallow Pruning

- Average case model predicts no asymptotic gain (Korf, 1991)
- Dependent on properties of evaluation function
  - For some games, natural evaluation functions allow no pruning

Deep Pruning

Deep Pruning

- Can we prune deeper in the tree based on the same bound?

Deep Pruning

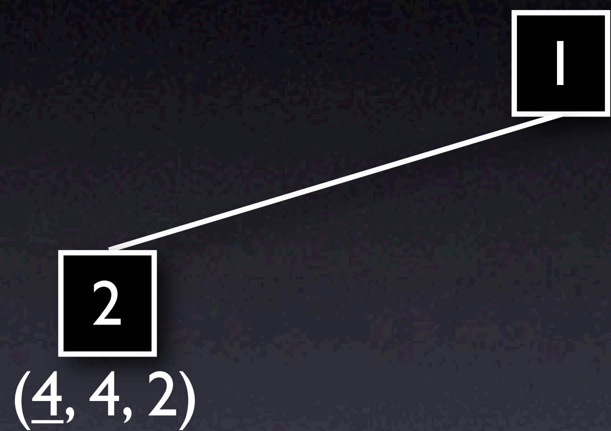
- Can we prune deeper in the tree based on the same bound?
  - Not directly valid in multi-player games

Deep Pruning

maxsum = 10

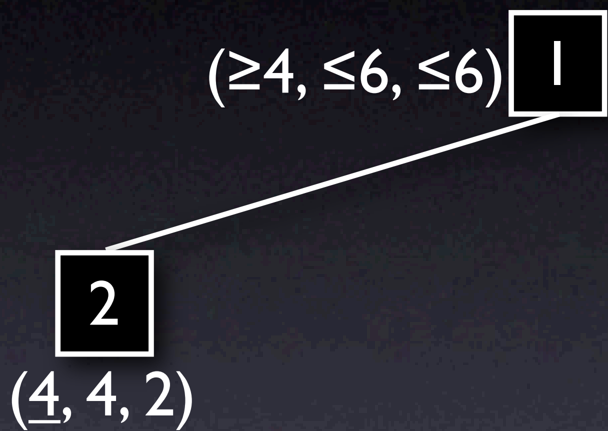
Deep Pruning

maxsum = 10



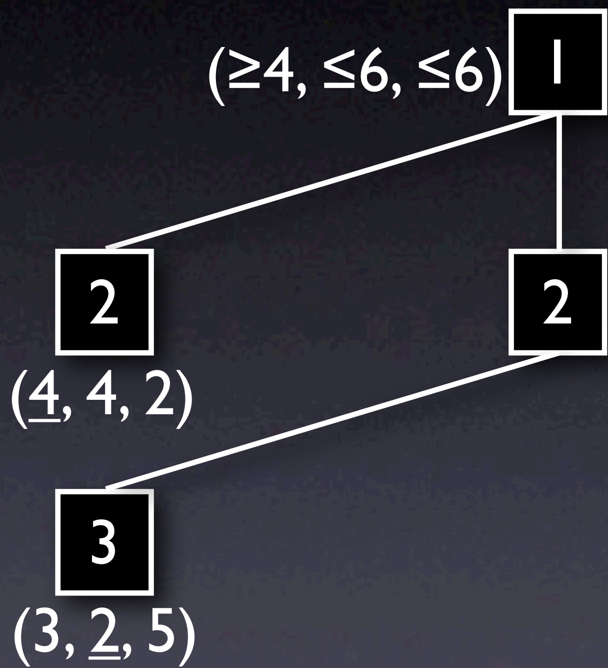
Deep Pruning

maxsum = 10



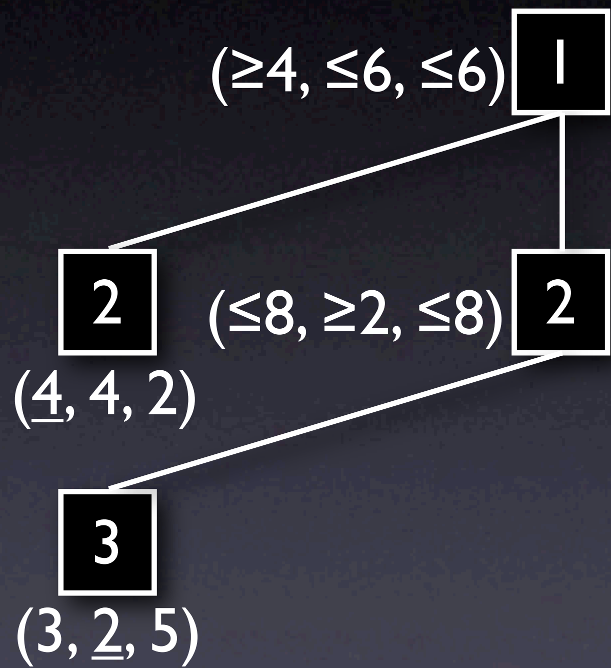
Deep Pruning

maxsum = 10



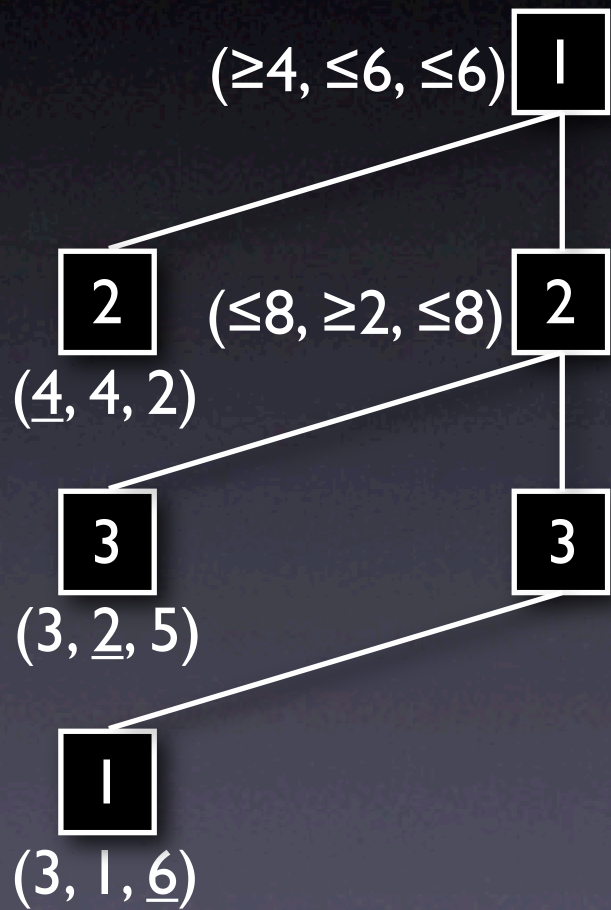
Deep Pruning

maxsum = 10



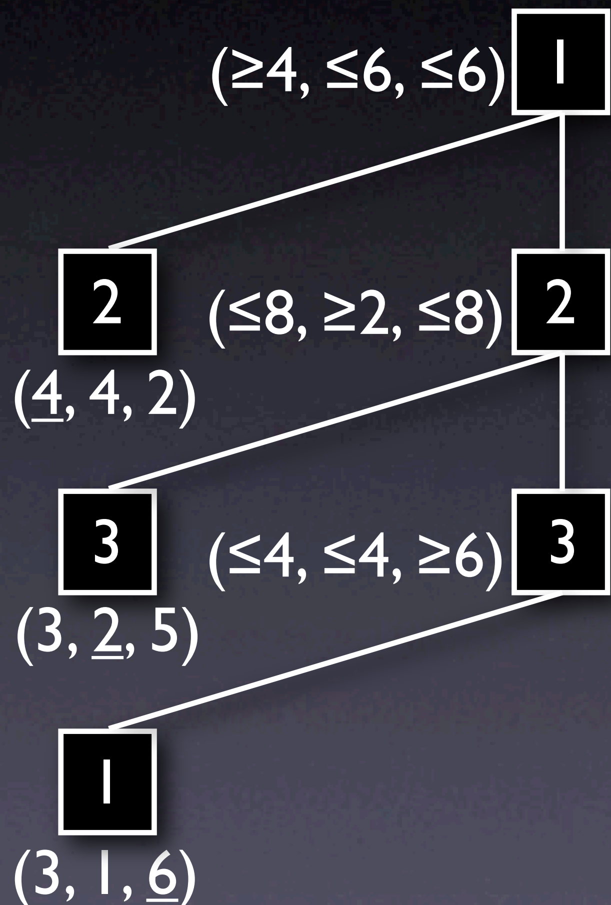
Deep Pruning

maxsum = 10



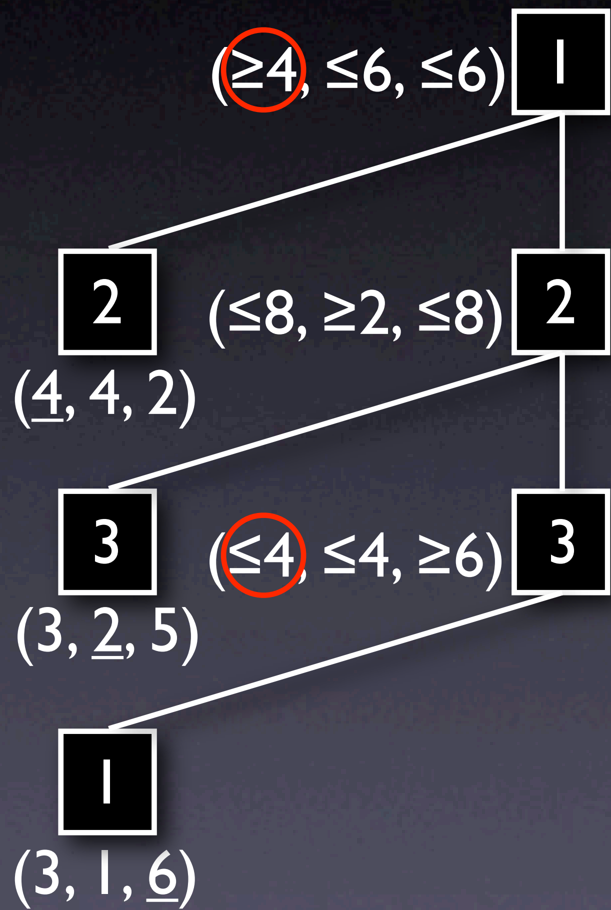
Deep Pruning

maxsum = 10



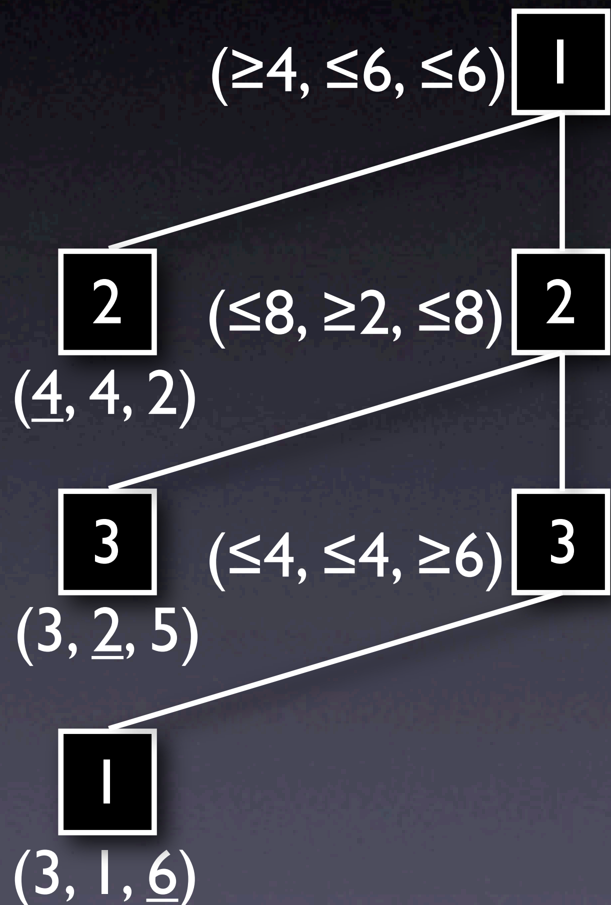
Deep Pruning

maxsum = 10



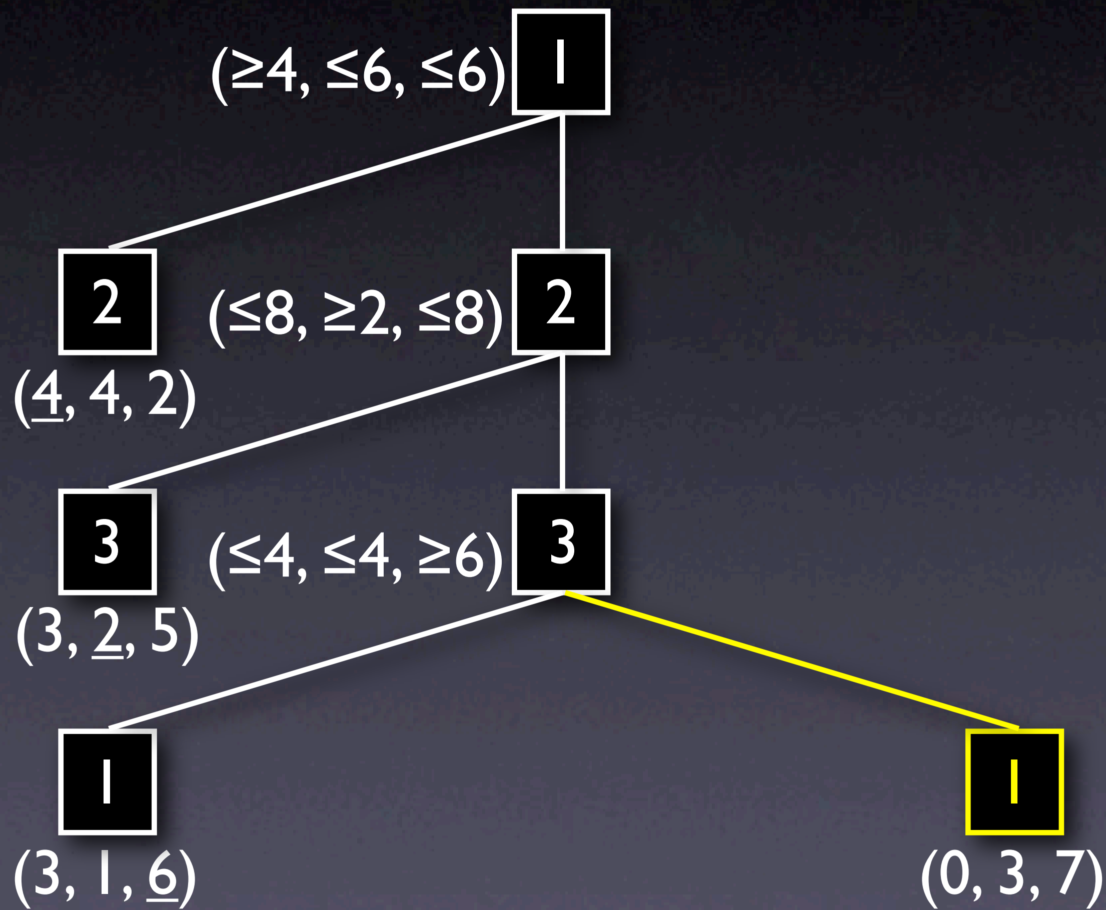
Deep Pruning

maxsum = 10



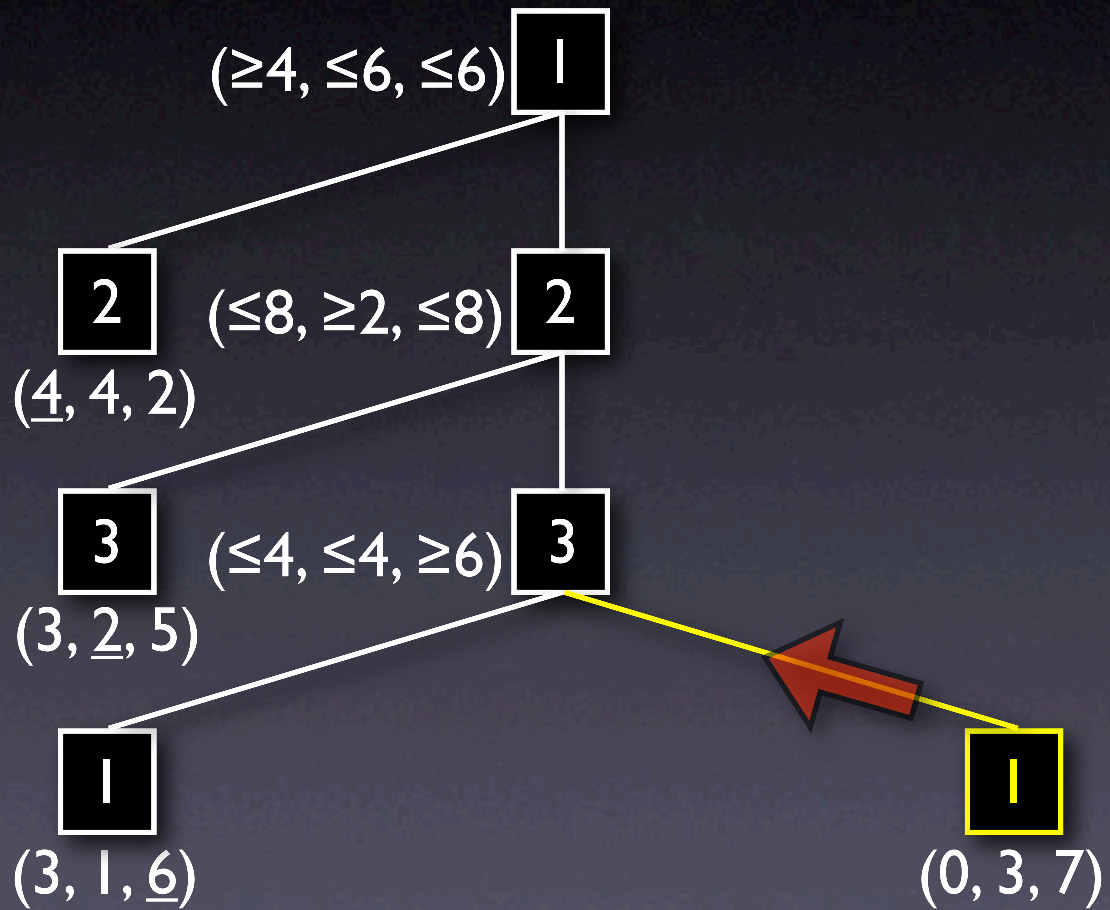
Deep Pruning

maxsum = 10



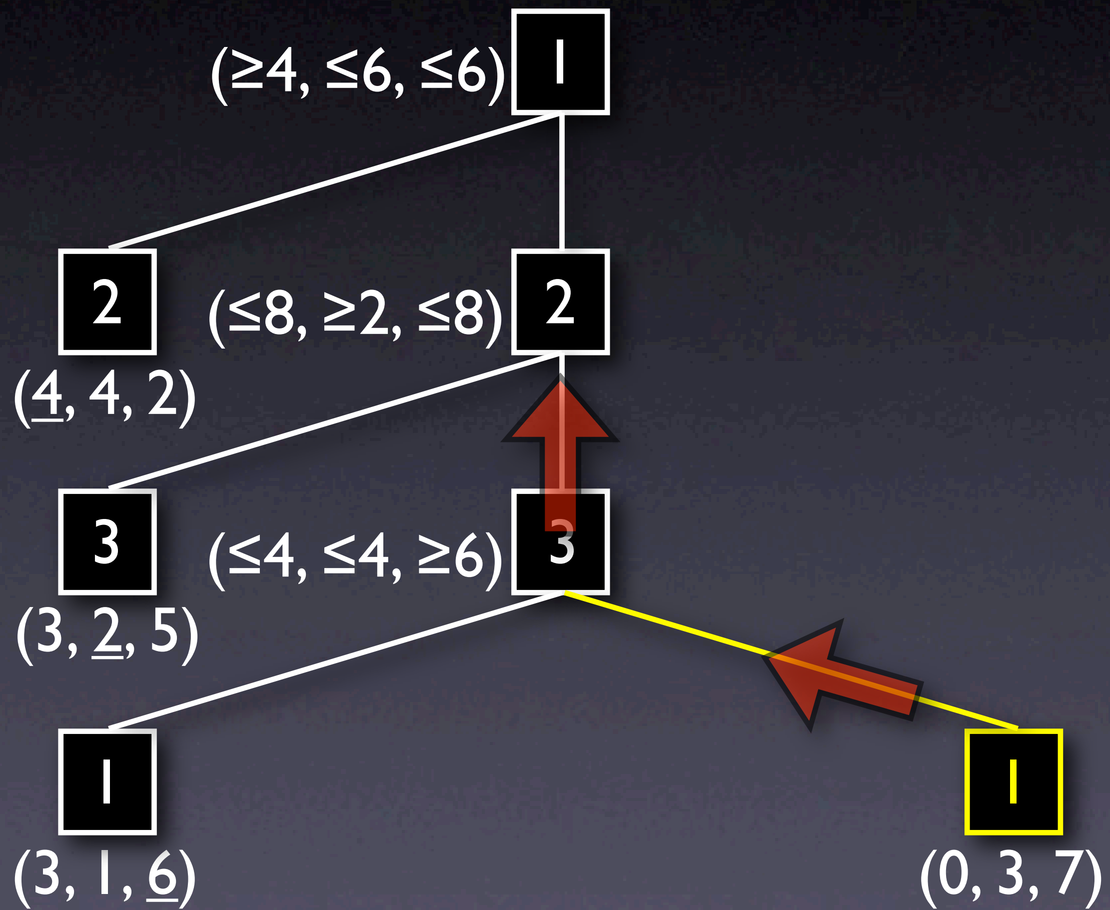
Deep Pruning

maxsum = 10



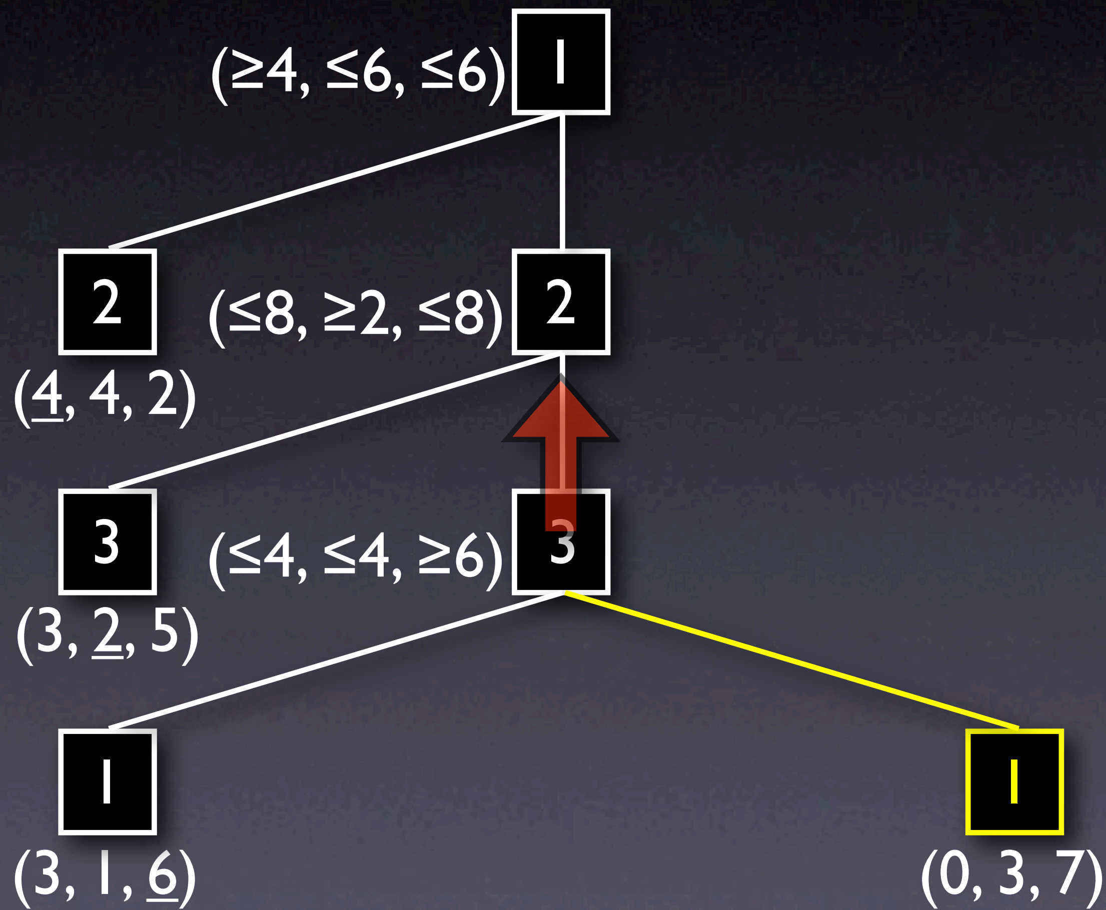
Deep Pruning

maxsum = 10



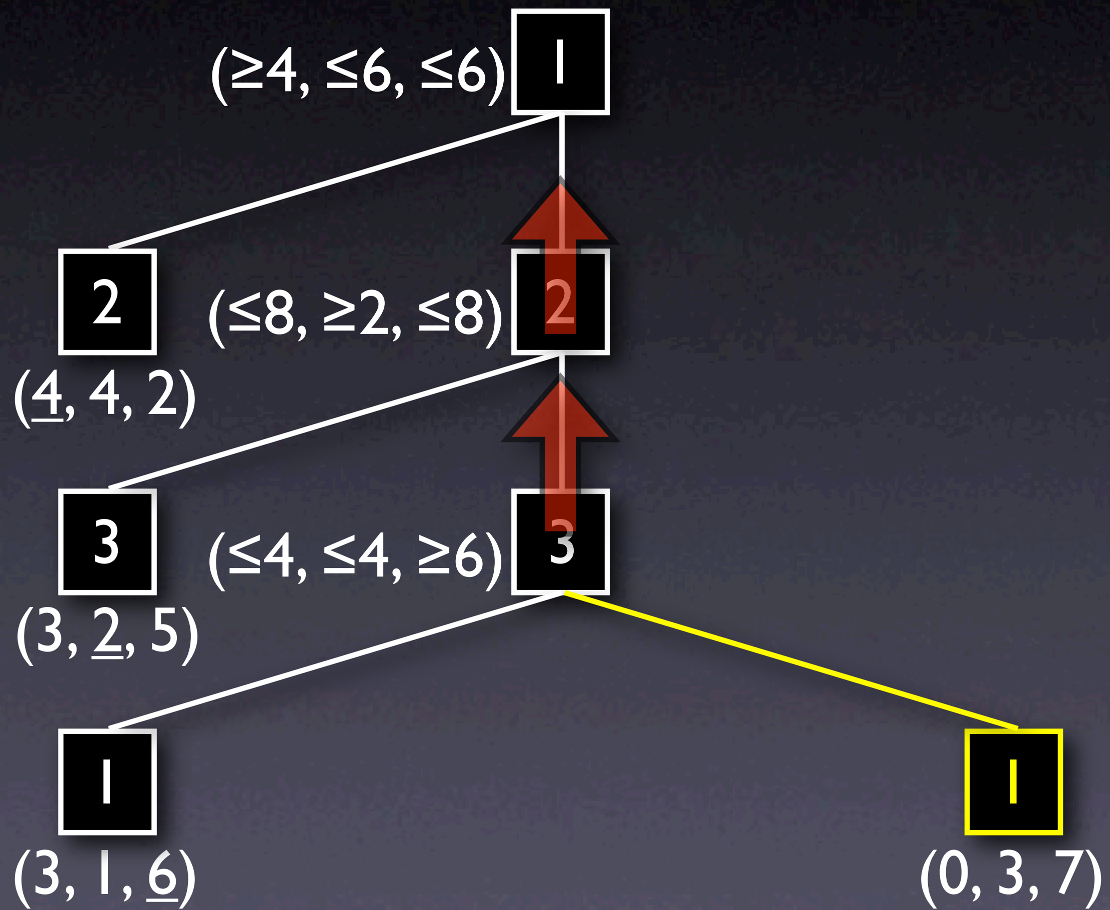
Deep Pruning

maxsum = 10



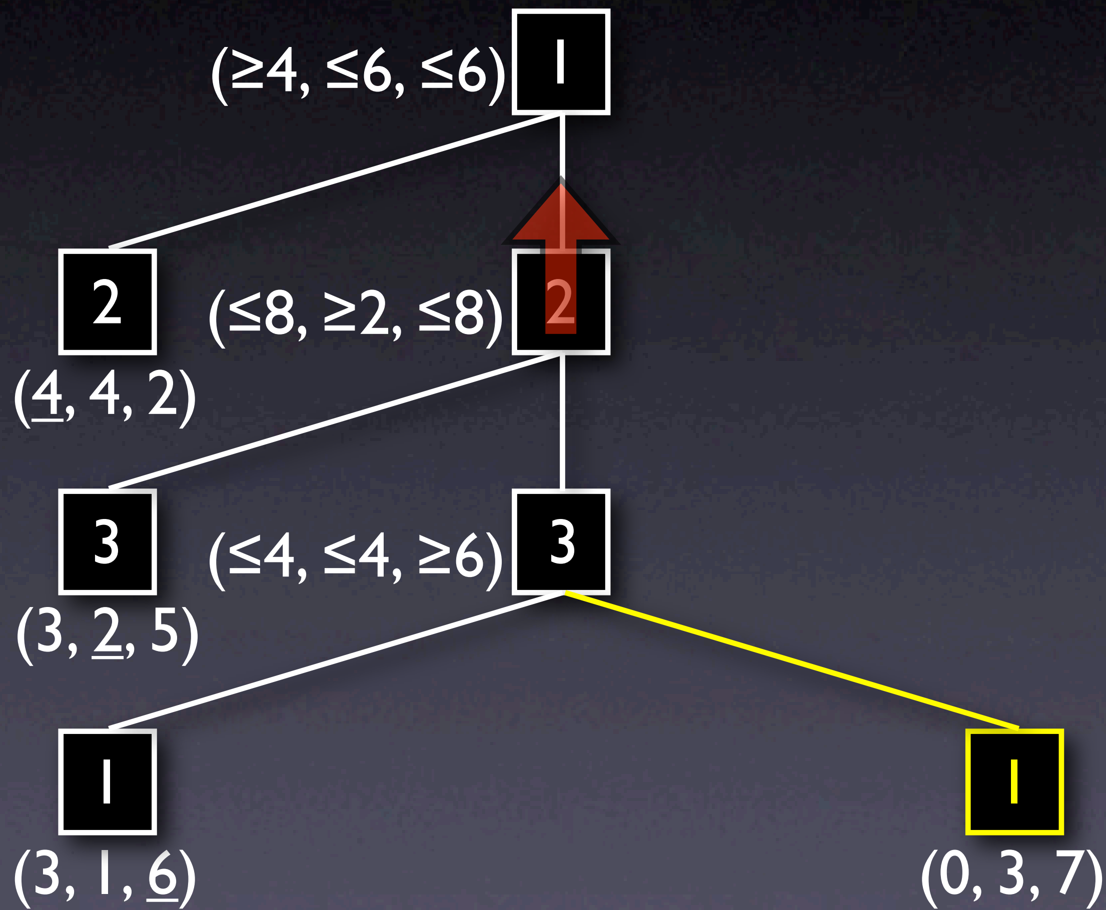
Deep Pruning

maxsum = 10



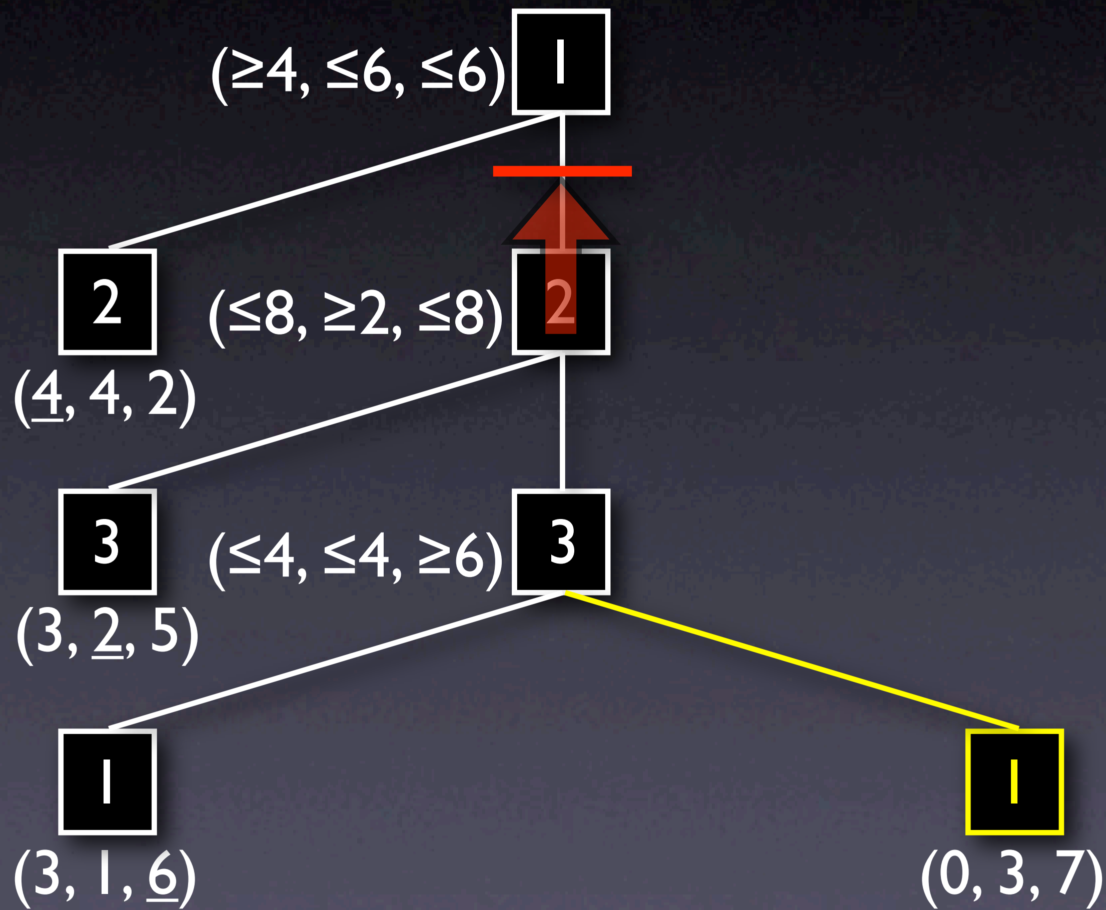
Deep Pruning

maxsum = 10



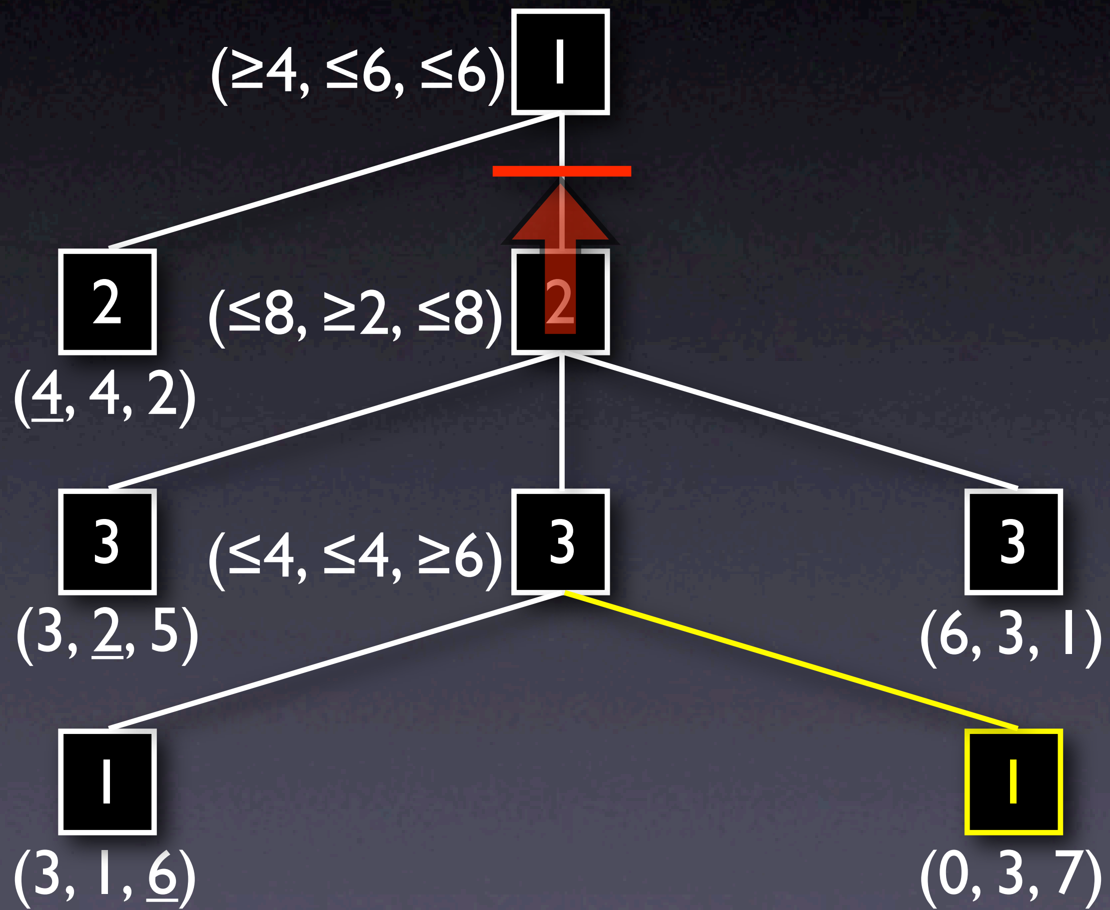
Deep Pruning

maxsum = 10



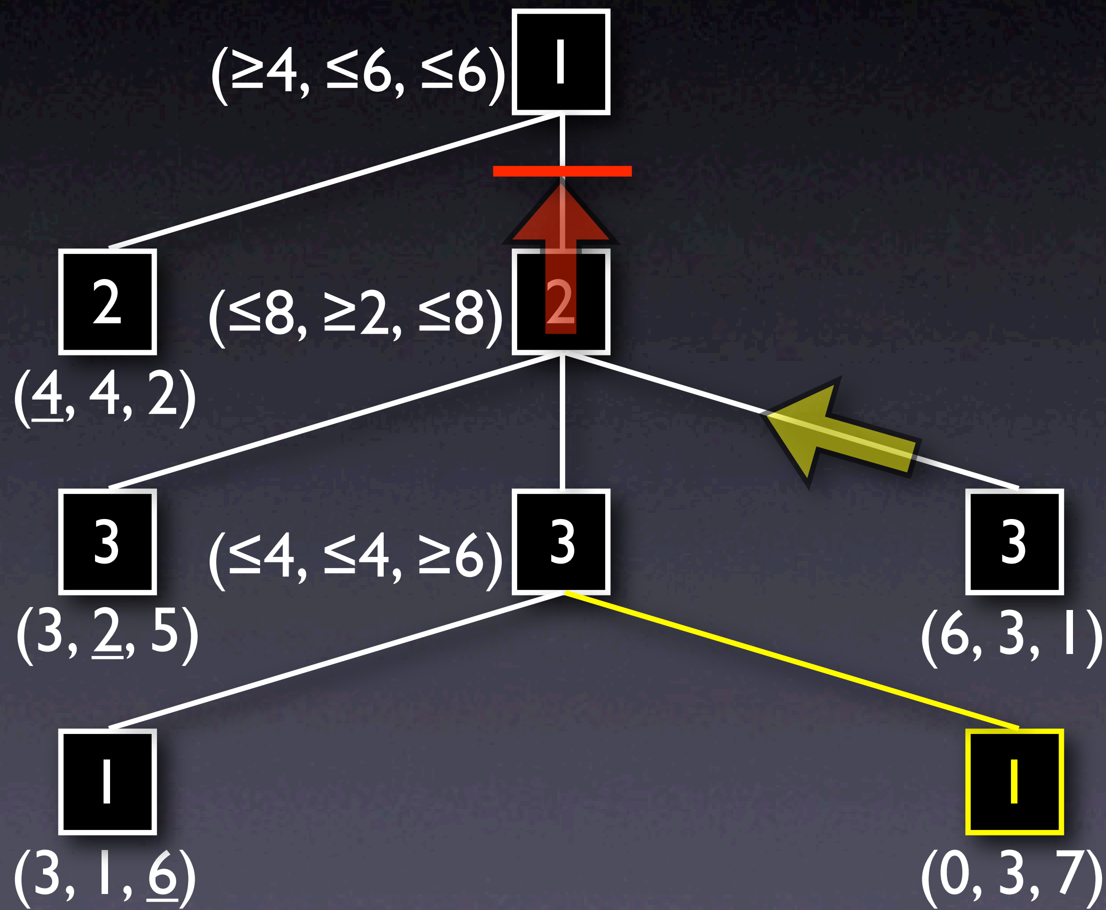
Deep Pruning

maxsum = 10



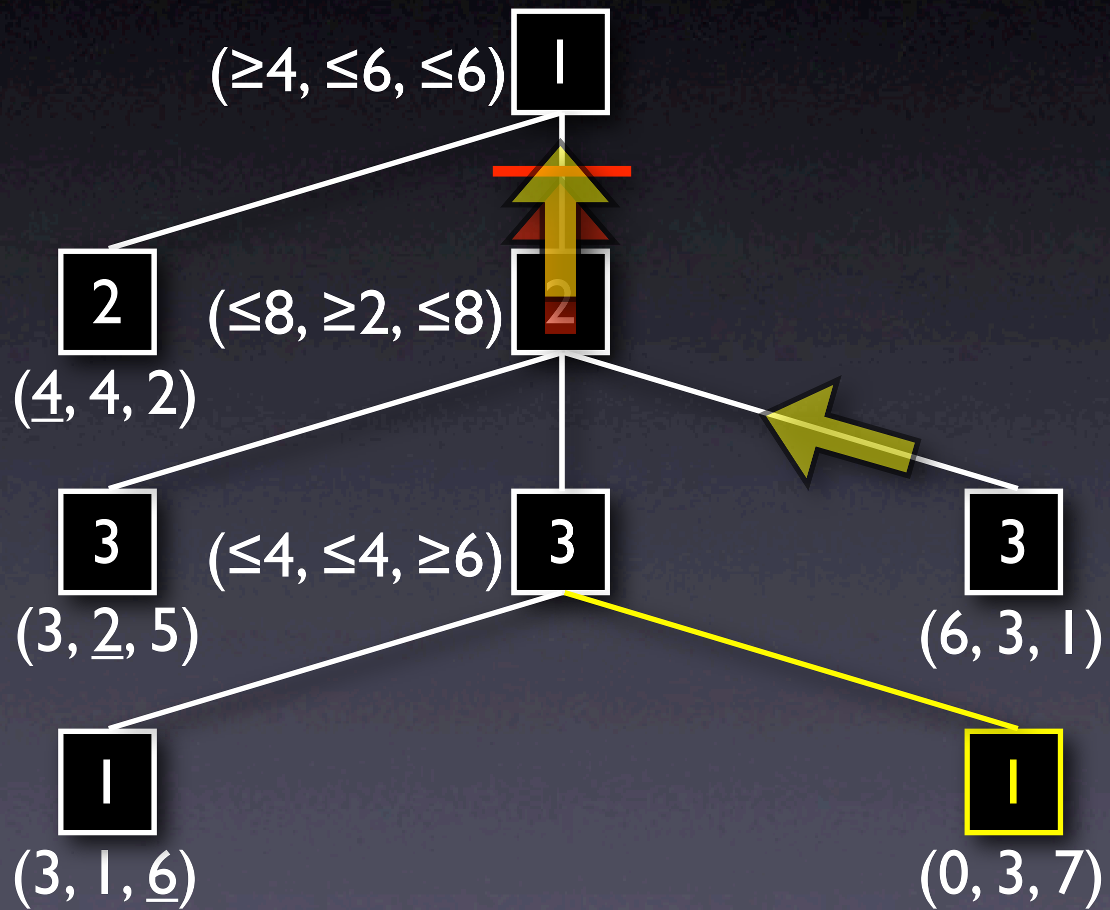
Deep Pruning

maxsum = 10



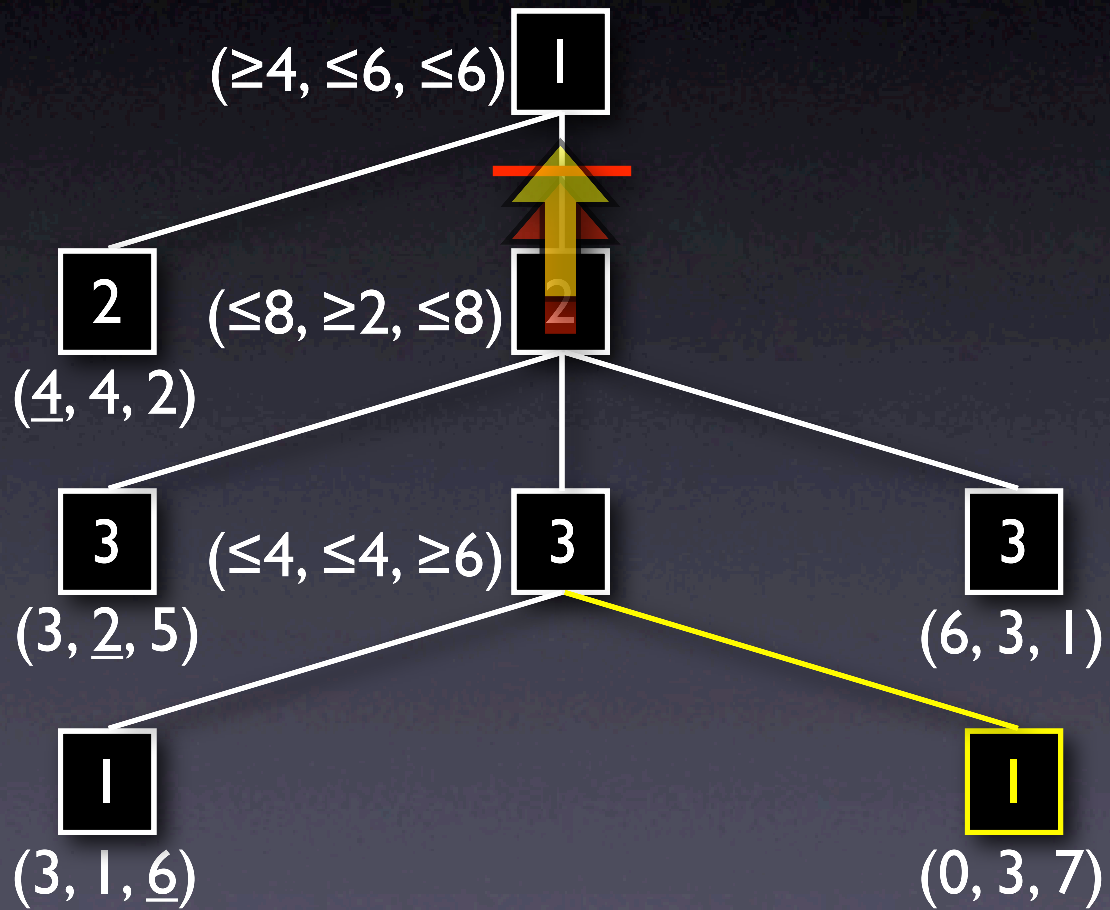
Deep Pruning

maxsum = 10



Deep Pruning

maxsum = 10



Deep Pruning

Deep Pruning

- The max^n value of the last child cannot be the max^n value of the tree

Deep Pruning

- The max^n value of the last child cannot be the max^n value of the tree
- It can still affect the max^n value of the tree (Korf, 1991)

13

Deep Pruning

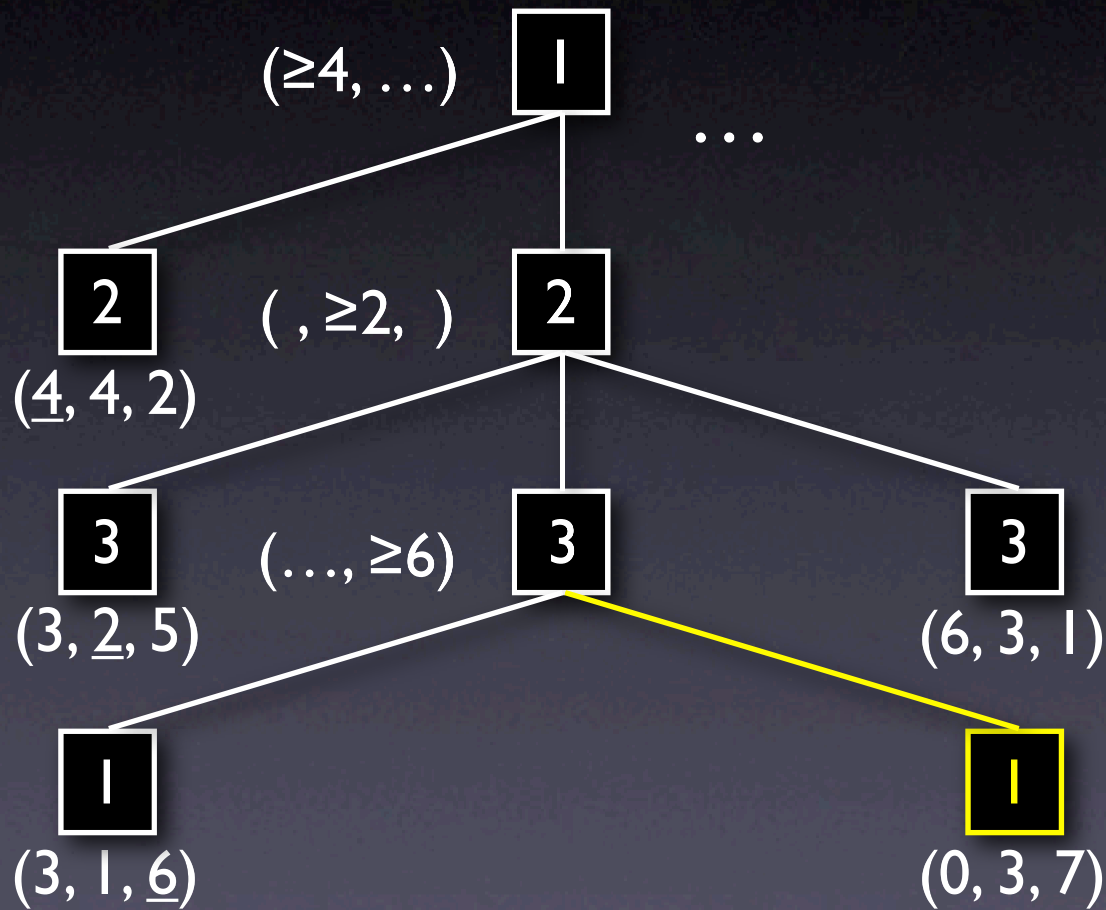
- The max^n value of the last child cannot be the max^n value of the tree
- It can still affect the max^n value of the tree (Korf, 1991)
- Is there a valid way to do this pruning?

13

Last-Branch Pruning

(Part 1)

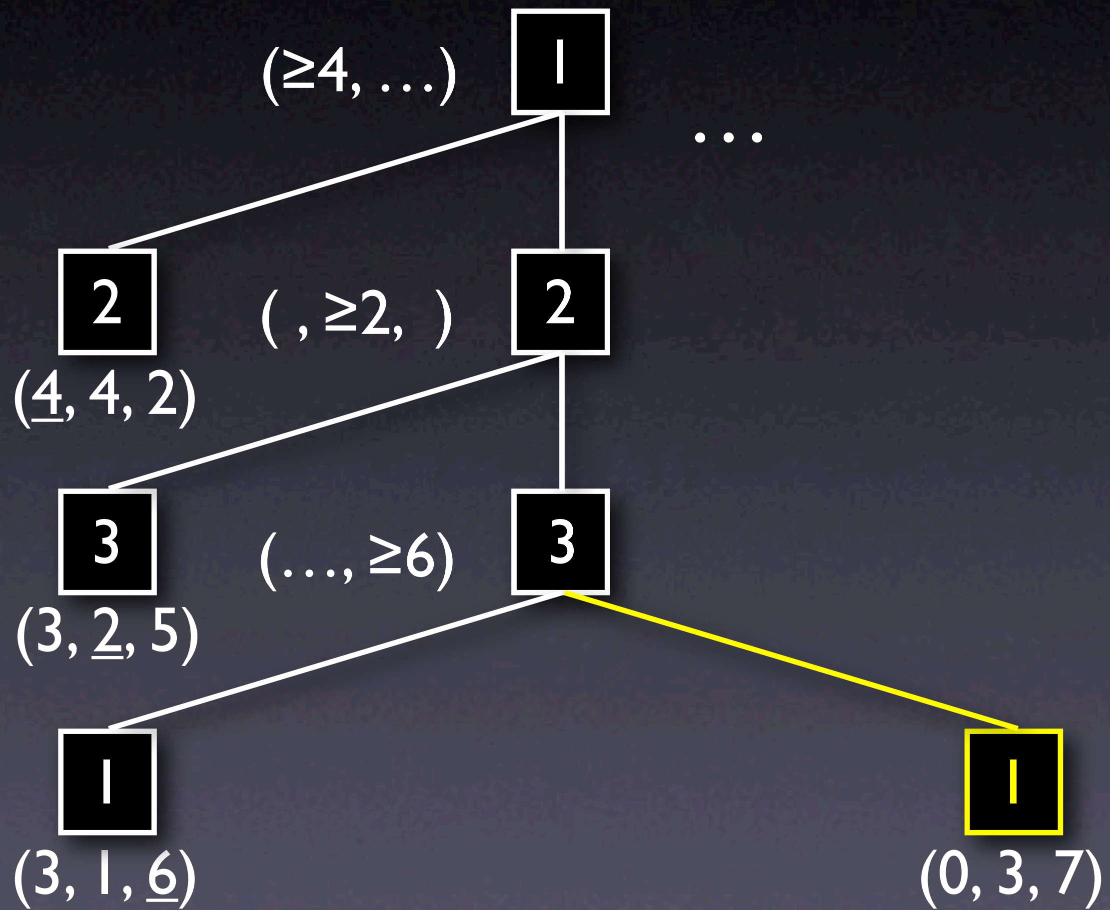
maxsum = 10



Last-Branch Pruning

(Part 1)

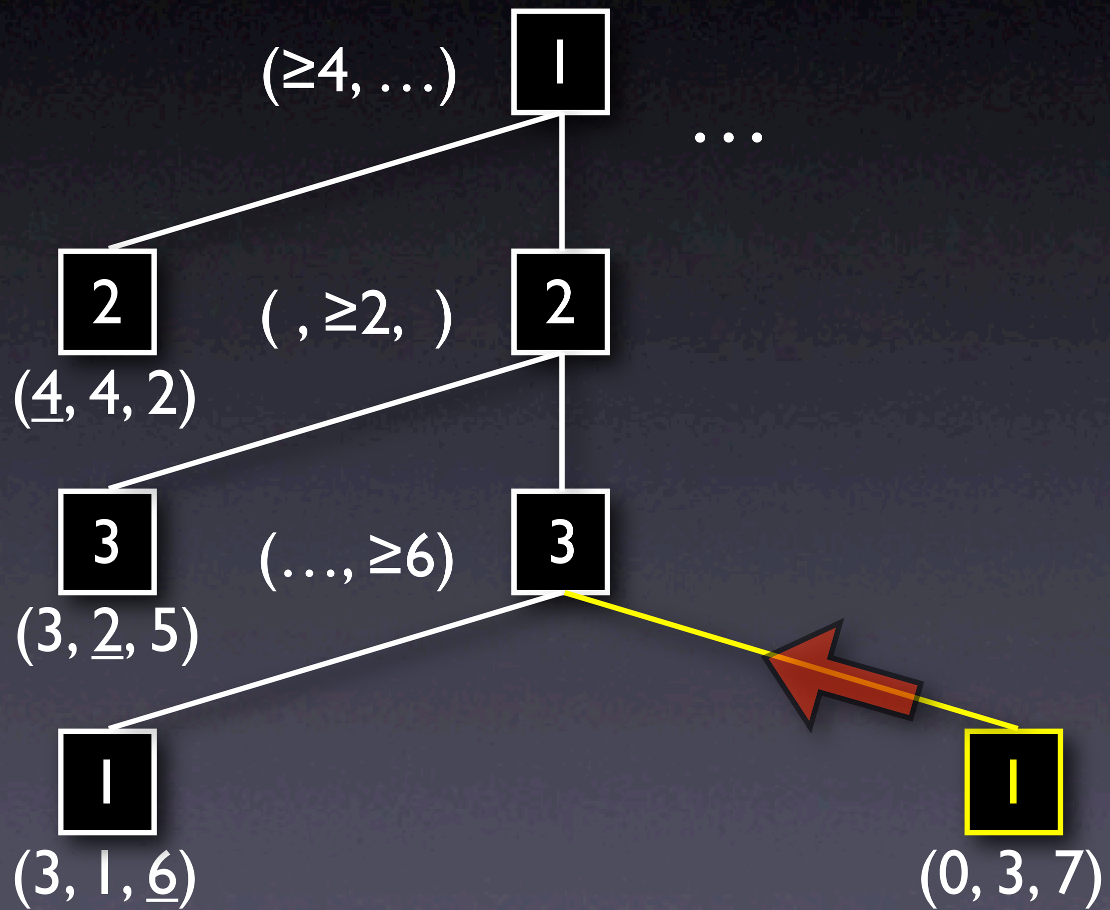
maxsum = 10



Last-Branch Pruning

(Part 1)

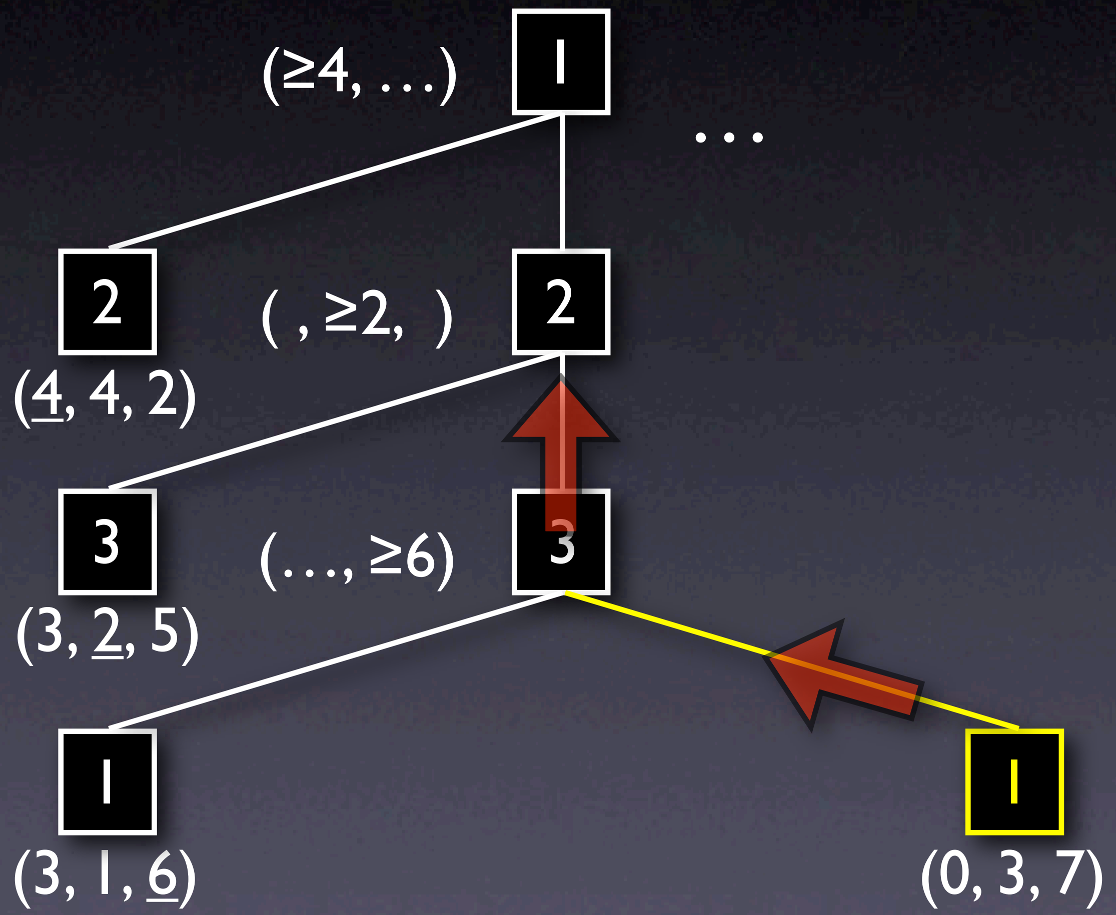
maxsum = 10



Last-Branch Pruning

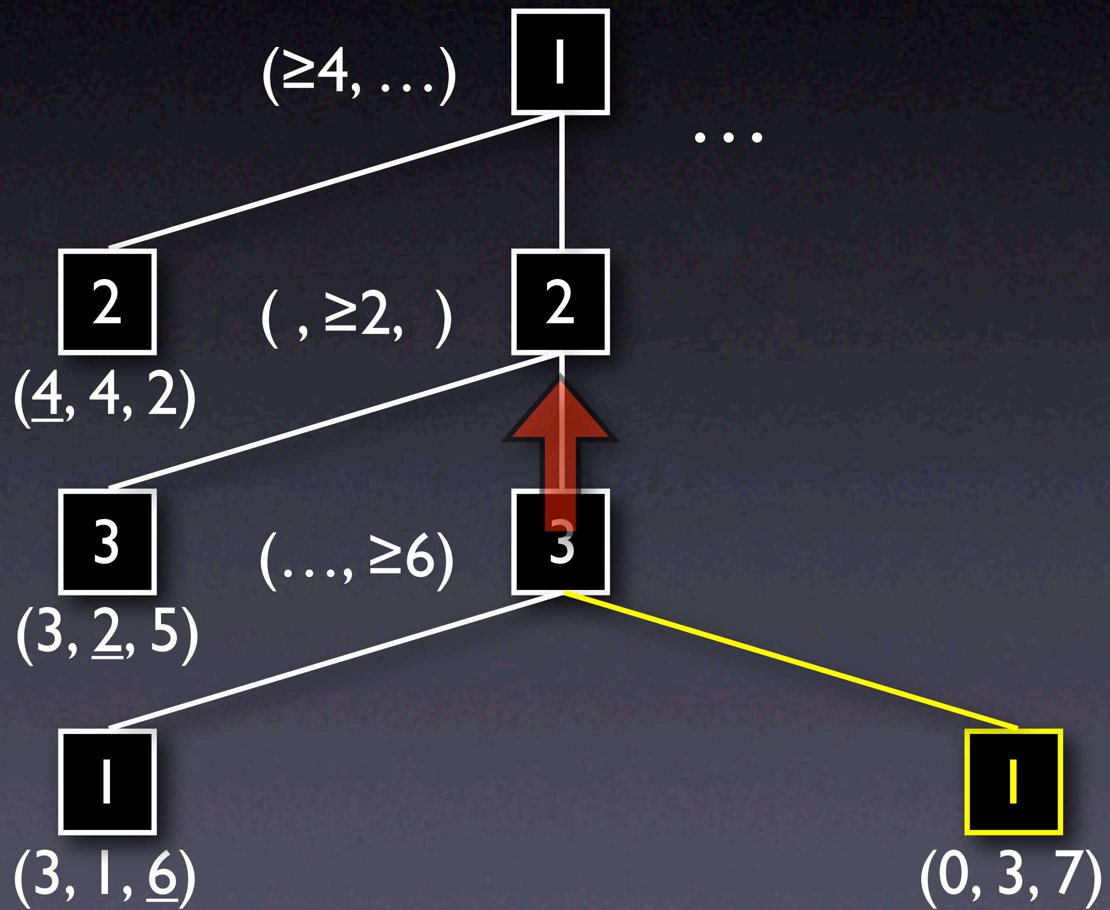
(Part 1)

maxsum = 10



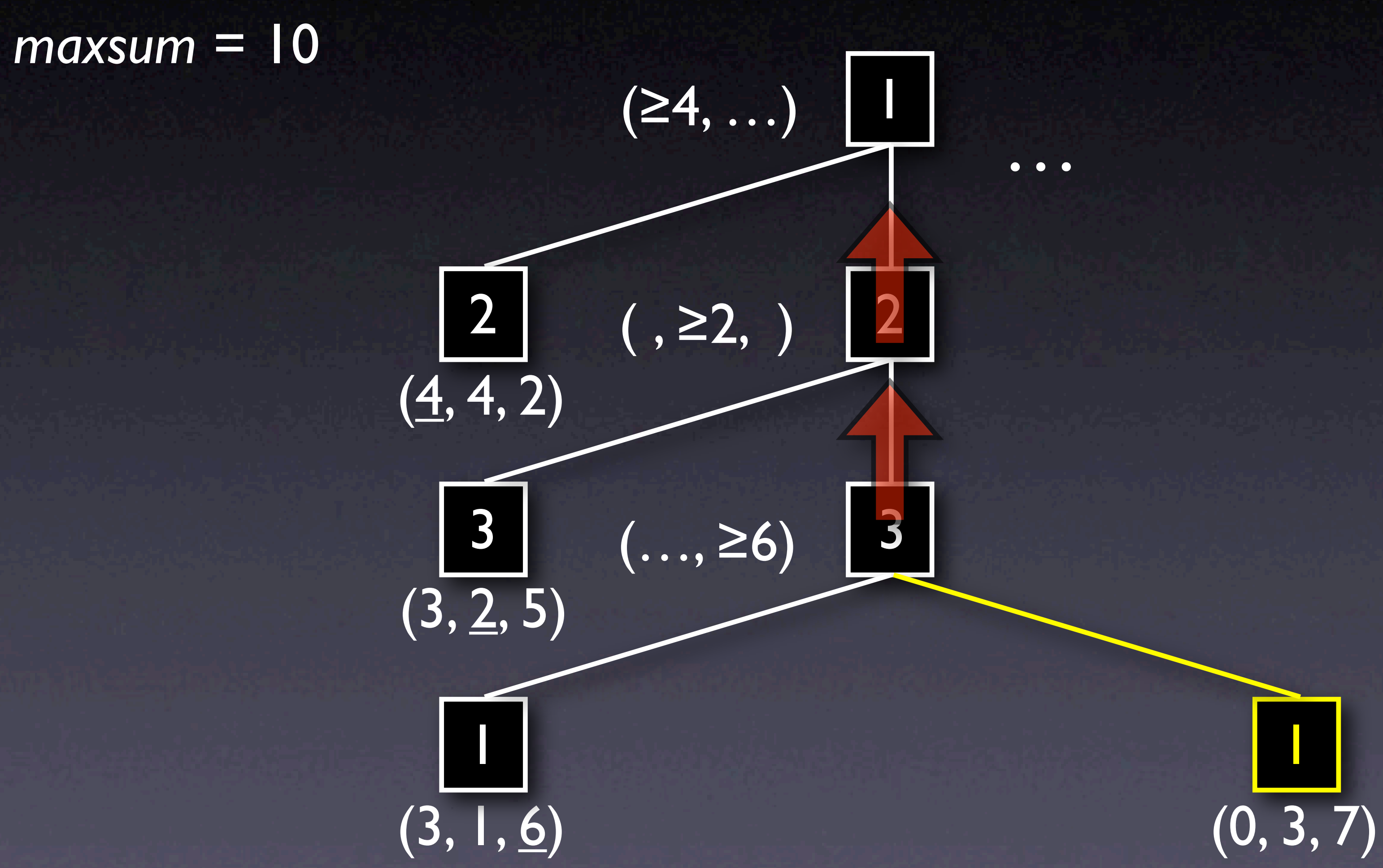
Last-Branch Pruning
(Part 1)

maxsum = 10



Last-Branch Pruning

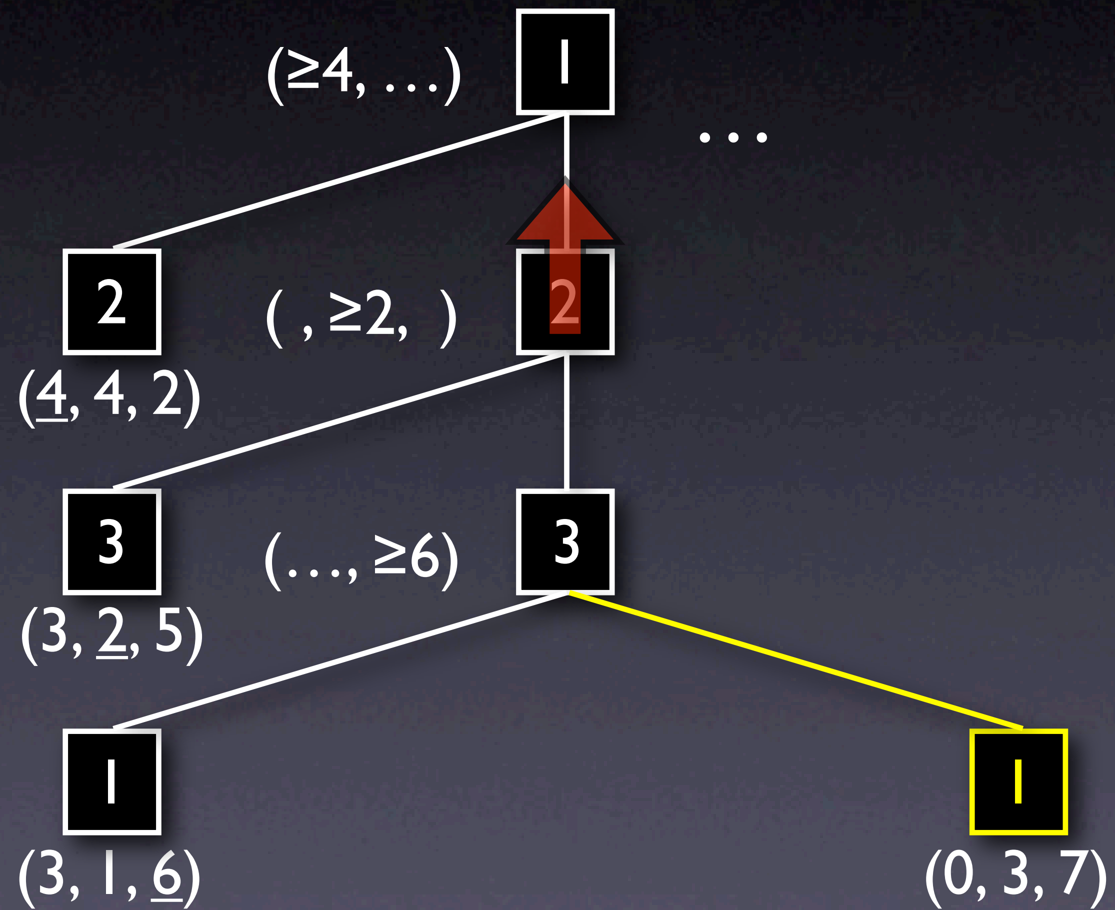
(Part 1)



Last-Branch Pruning

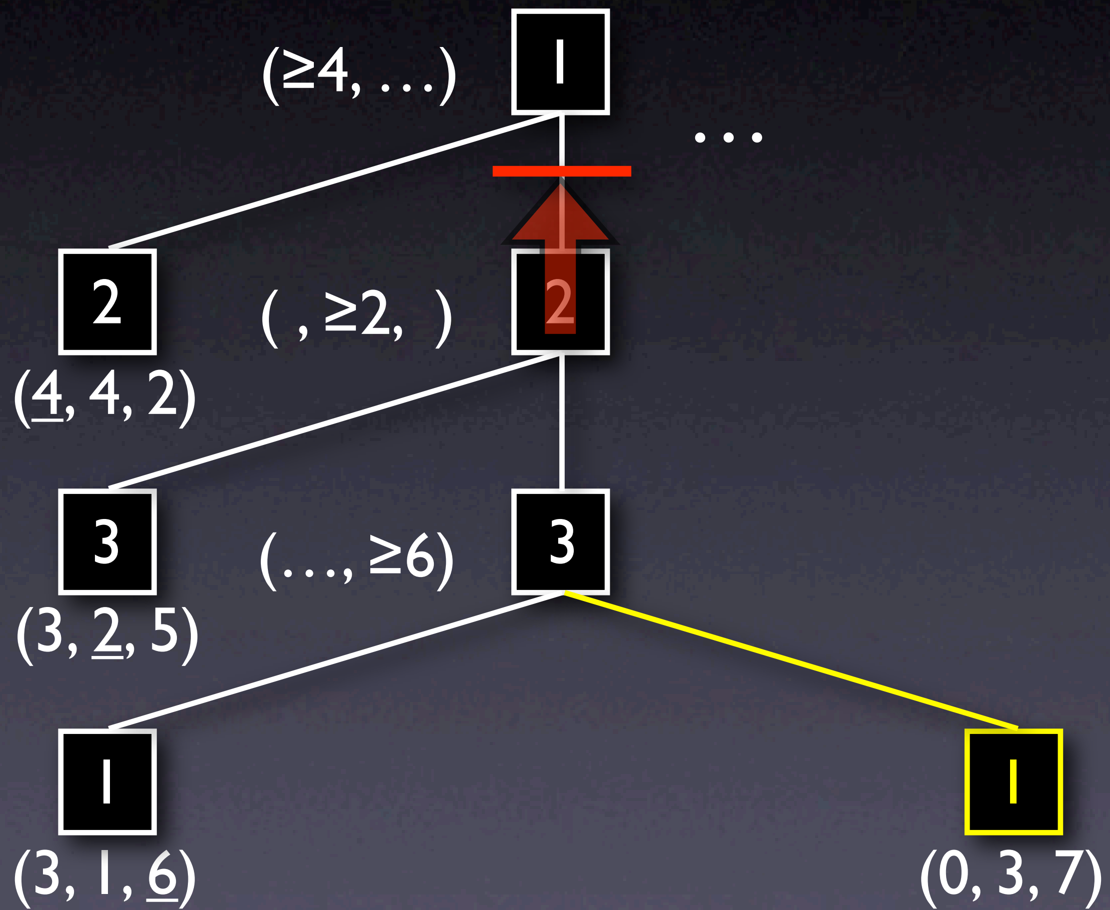
(Part 1)

maxsum = 10



Last-Branch Pruning
(Part 1)

maxsum = 10



Last-Branch Pruning I

Last-Branch Pruning I

- We can now deep prune when:

15

Last-Branch Pruning I

- We can now deep prune when:
  - Scores sum to *maxsum* or greater

15

Last-Branch Pruning I

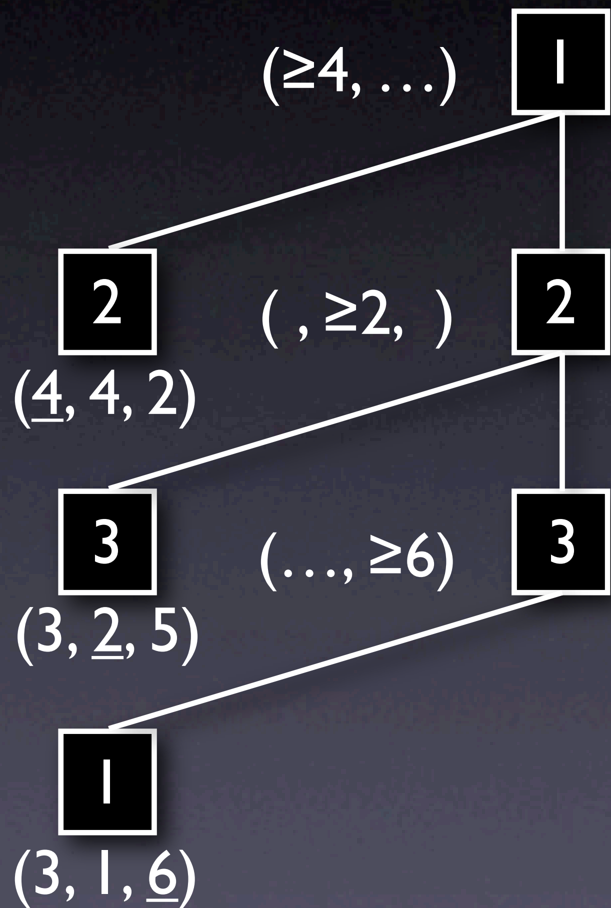
- We can now deep prune when:
  - Scores sum to *maxsum* or greater
  - Second player on their last branch

Last-Branch Pruning I

- We can now deep prune when:
  - Scores sum to *maxsum* or greater
  - Second player on their last branch
- Can we still do better?

Last-Branch Pruning
(Part 2)

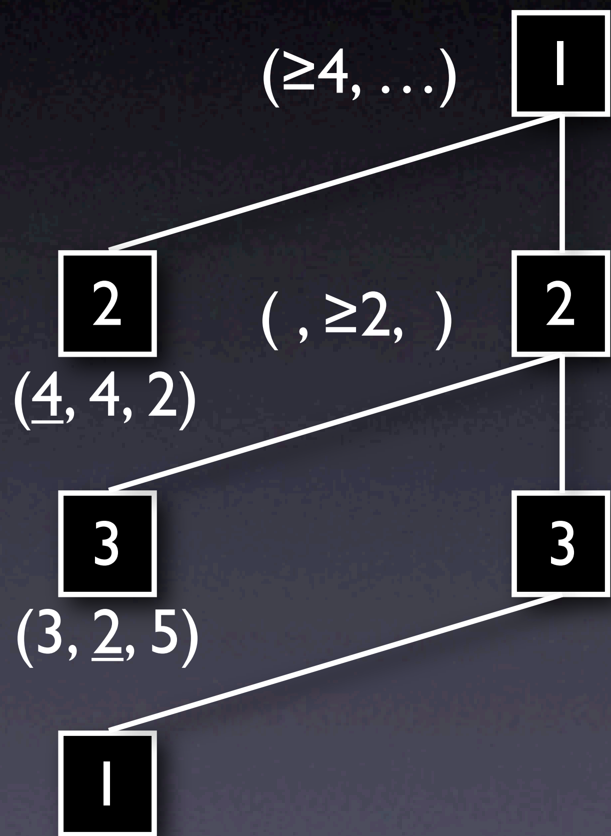
maxsum = 10



Last-Branch Pruning

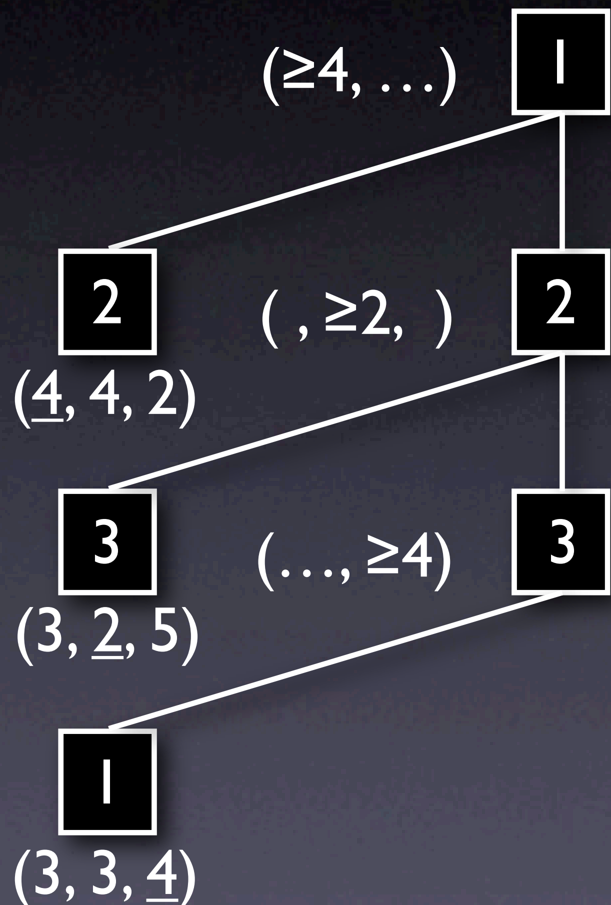
(Part 2)

maxsum = 10



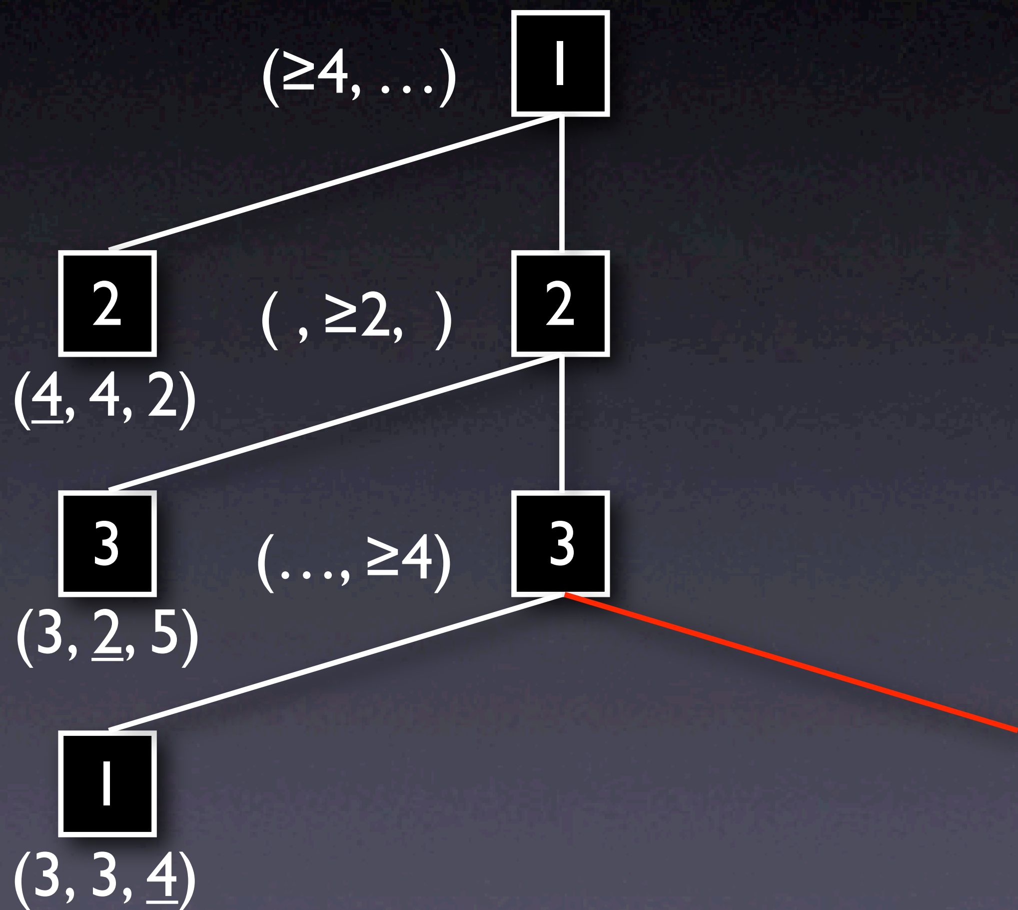
Last-Branch Pruning
(Part 2)

maxsum = 10



Last-Branch Pruning
(Part 2)

maxsum = 10



Last-Branch Pruning

17

Last-Branch Pruning

- Directional Algorithm

17

Last-Branch Pruning

- Directional Algorithm
- Prunes when:

Last-Branch Pruning

- Directional Algorithm
- Prunes when:
  - Sum of players' bounds ≥ *maxsum*

17

Last-Branch Pruning

- Directional Algorithm
- Prunes when:
  - Sum of players' bounds ≥ *maxsum*
  - Second player on their last branch

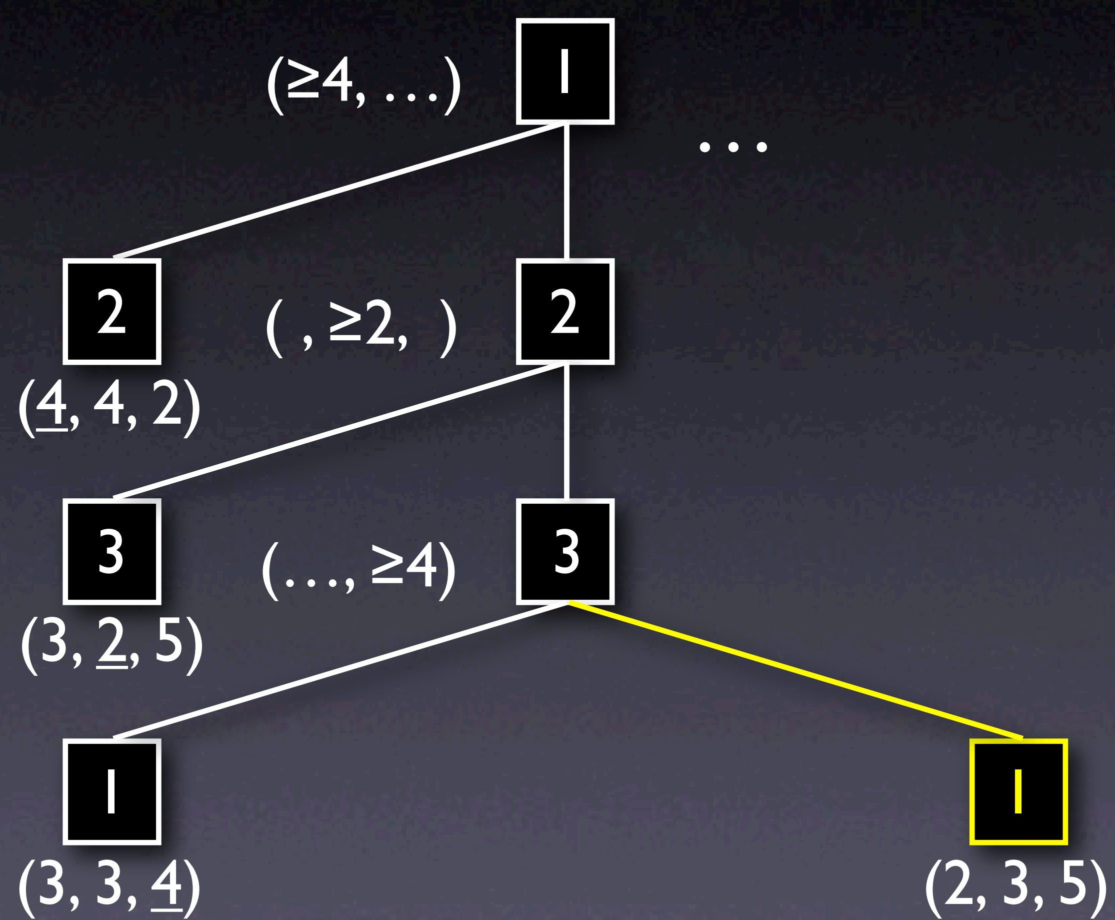
17

Last-Branch Pruning

- Directional Algorithm
- Prunes when:
  - Sum of players' bounds ≥ *maxsum*
  - Second player on their last branch
- Limited to last branch! Can we do better?

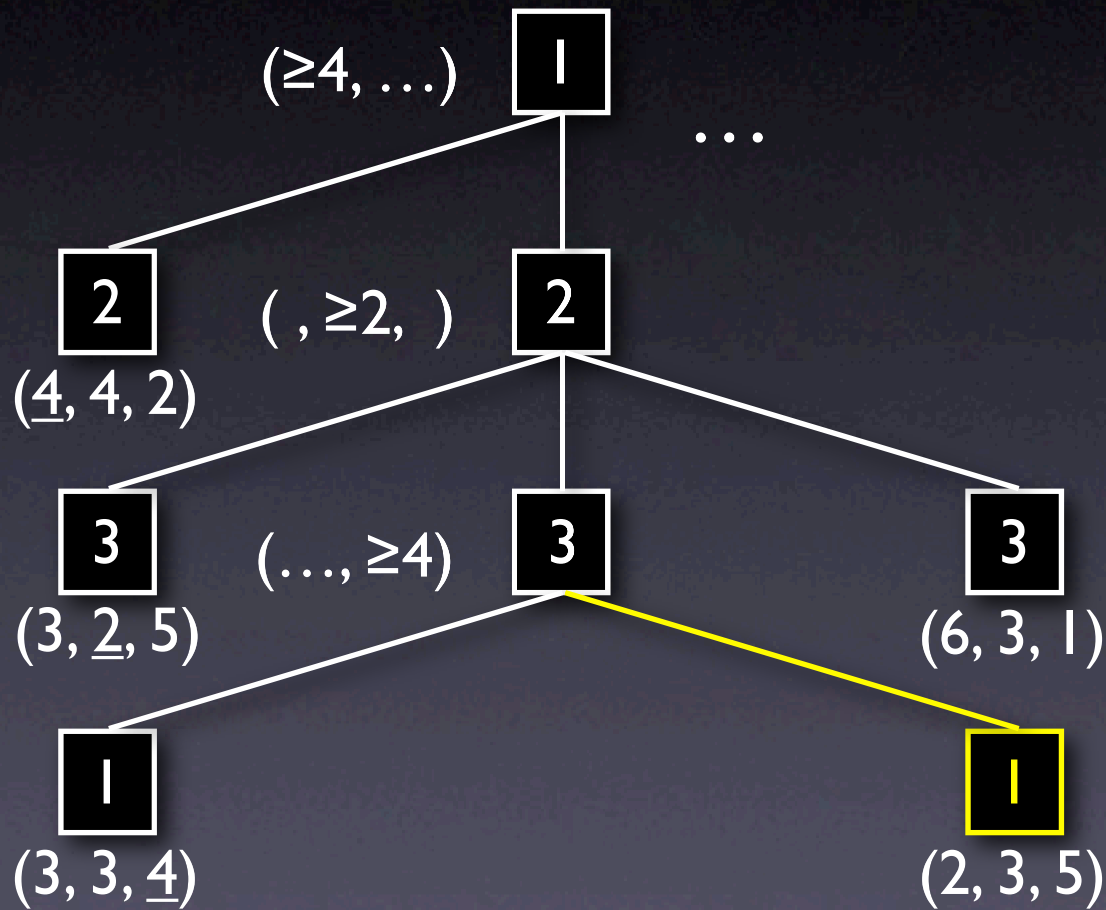
Speculative Pruning

maxsum = 10



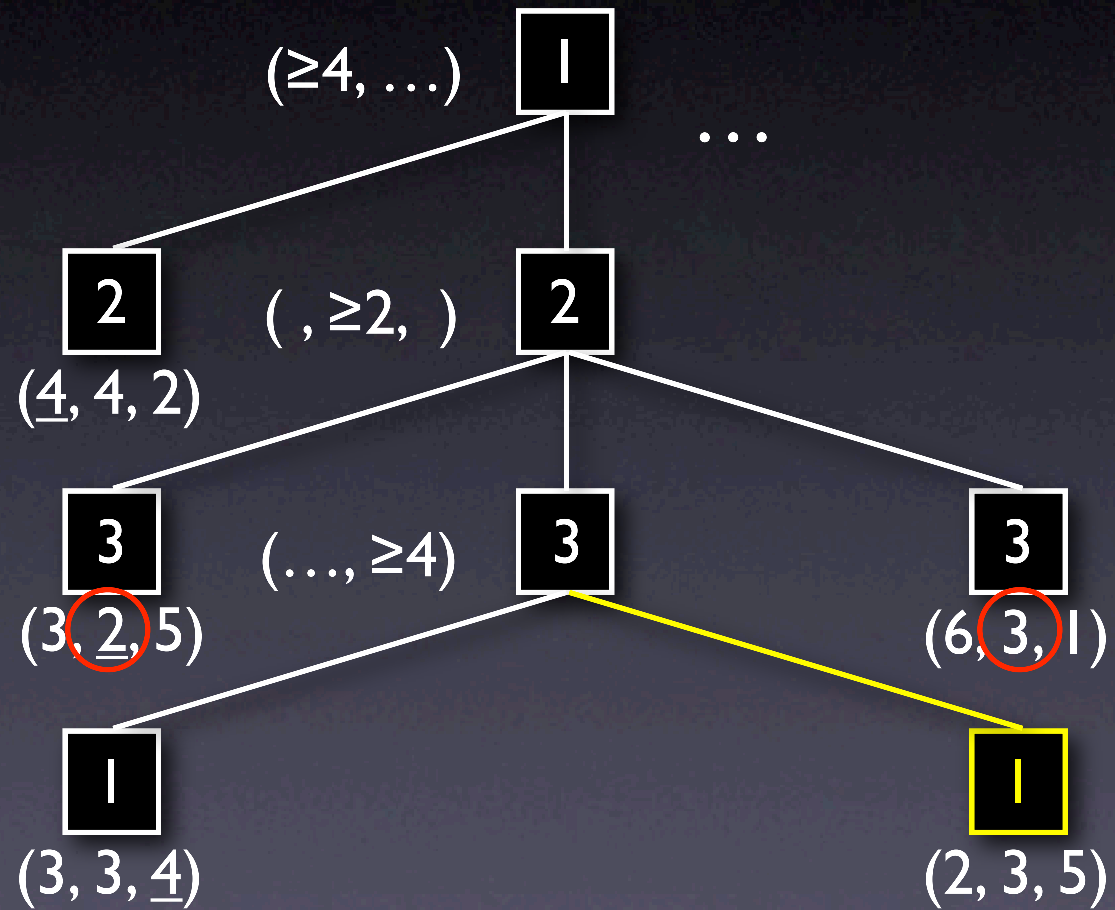
Speculative Pruning

maxsum = 10



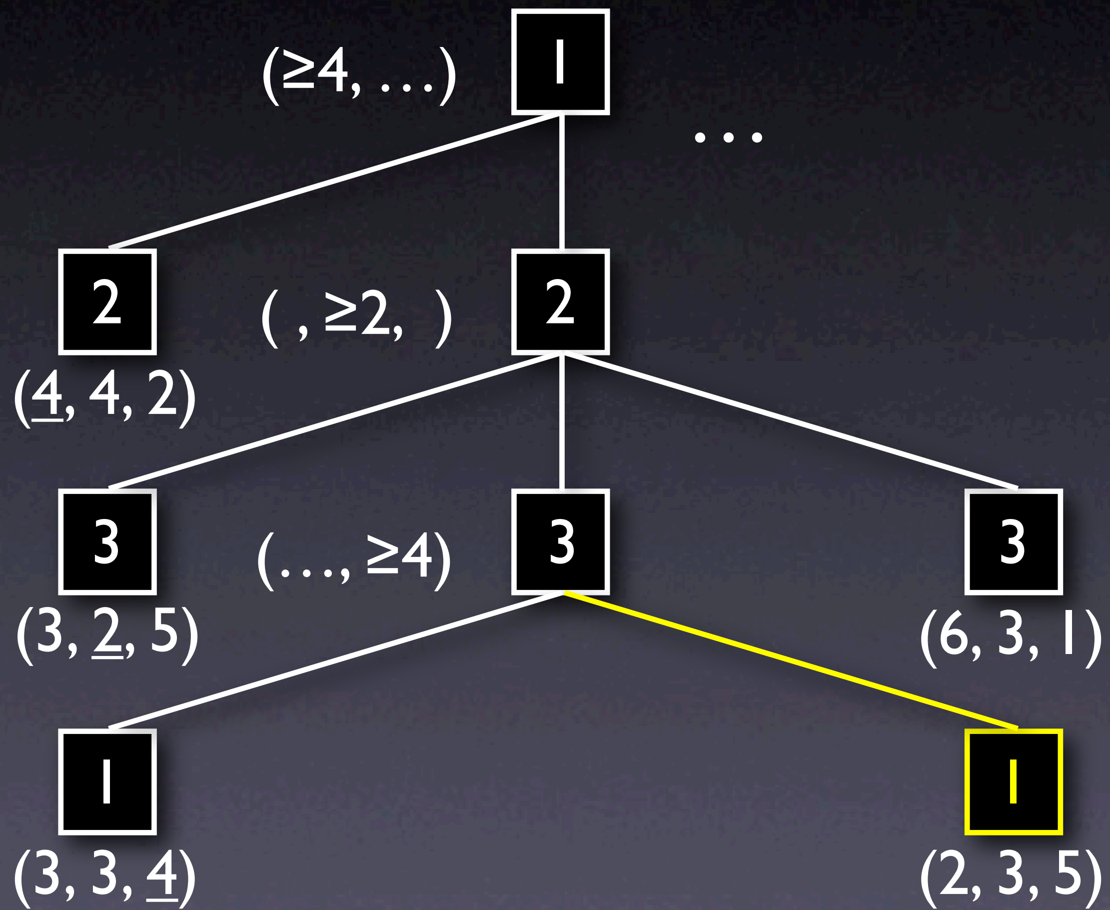
Speculative Pruning

maxsum = 10



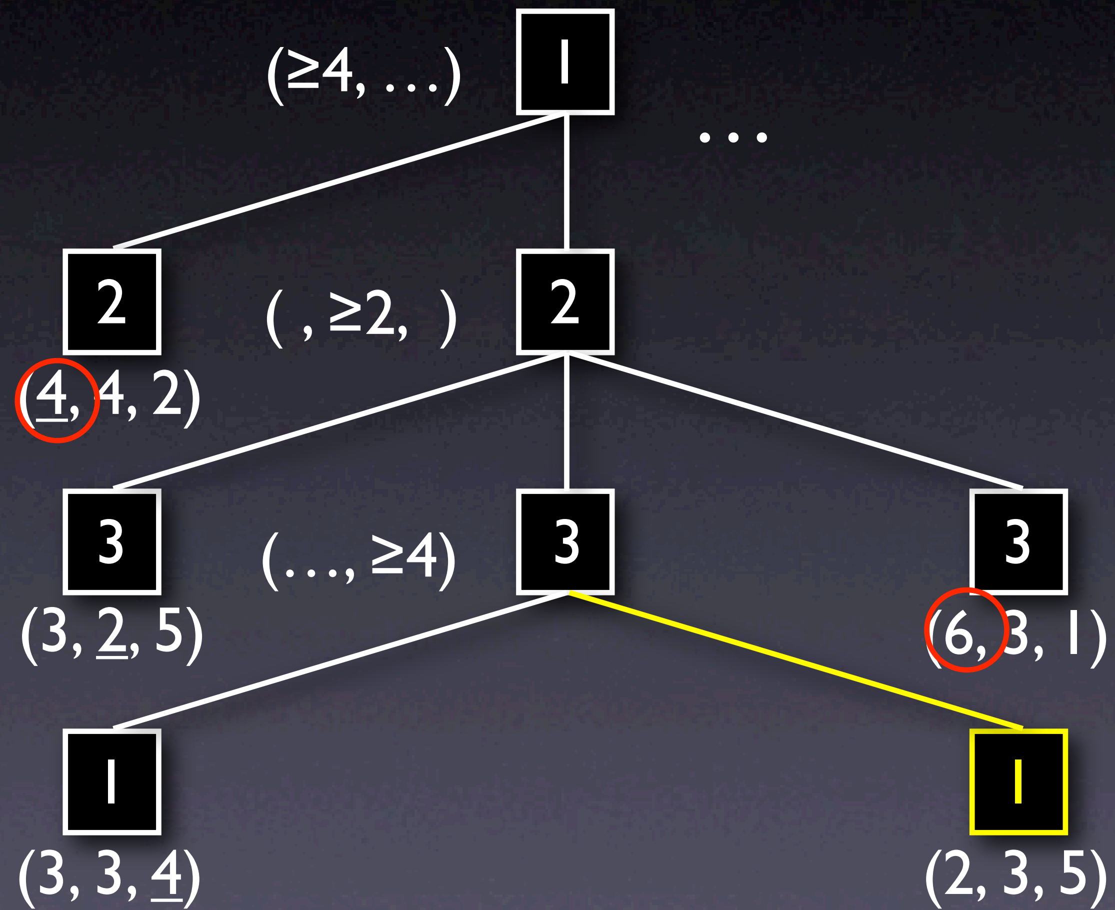
Speculative Pruning

maxsum = 10



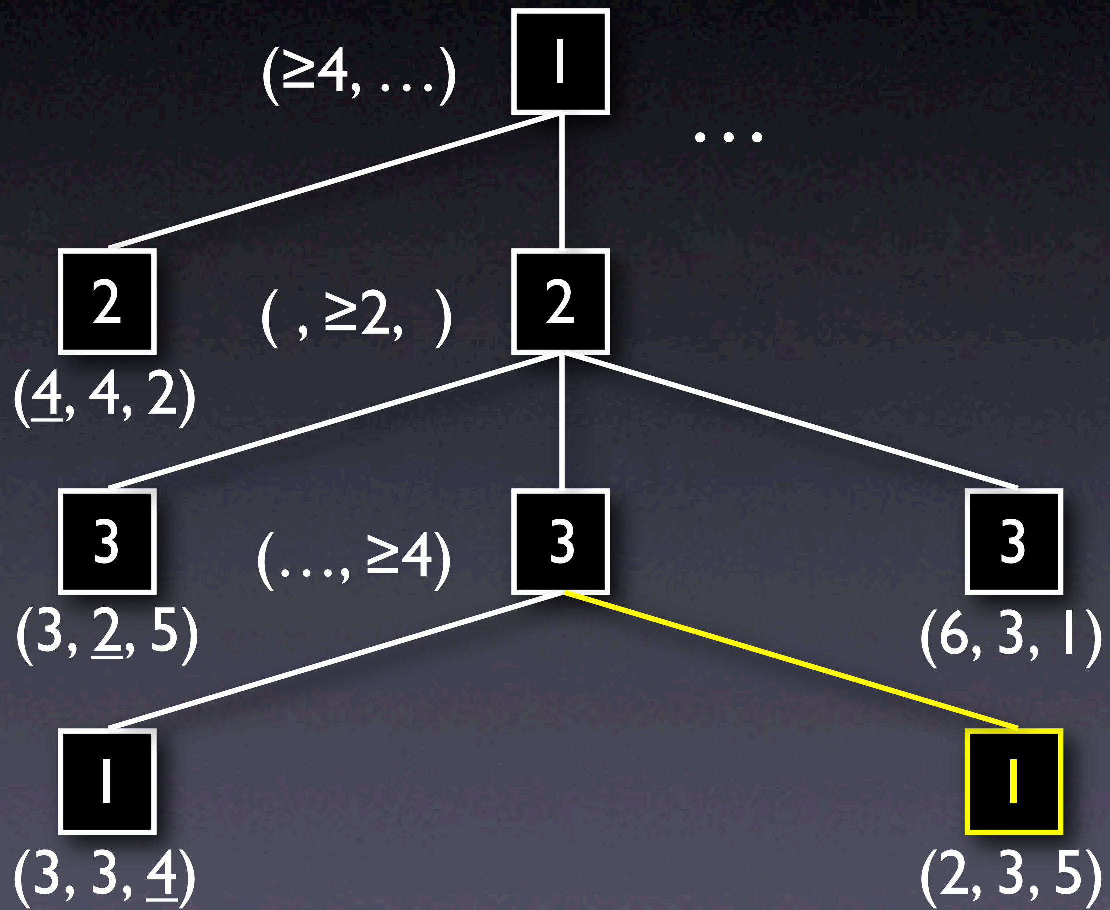
Speculative Pruning

maxsum = 10



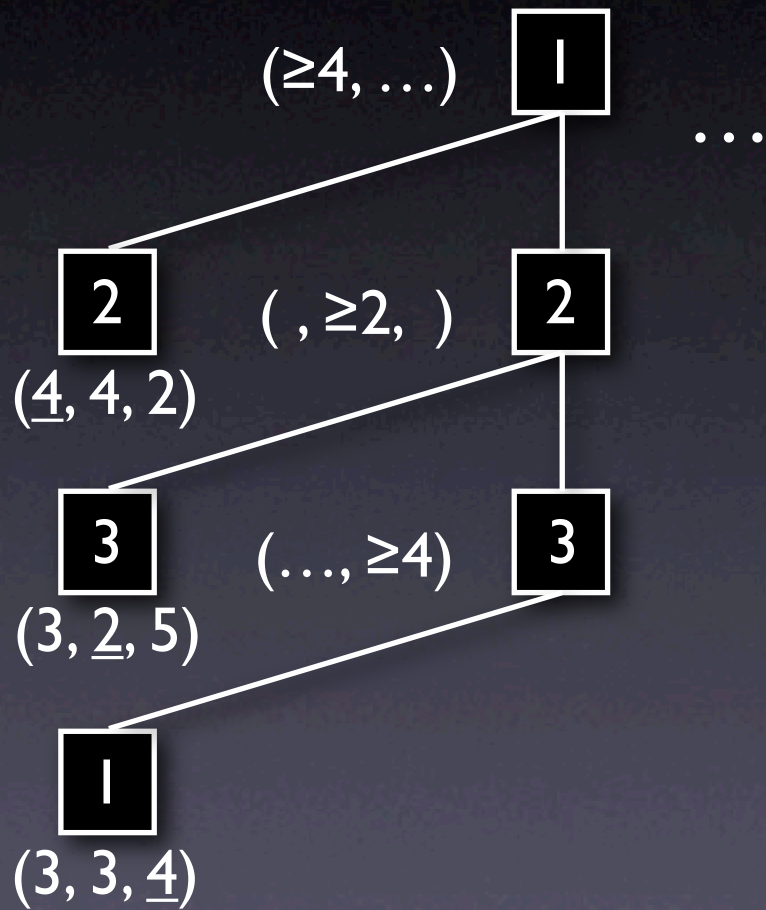
Speculative Pruning

maxsum = 10



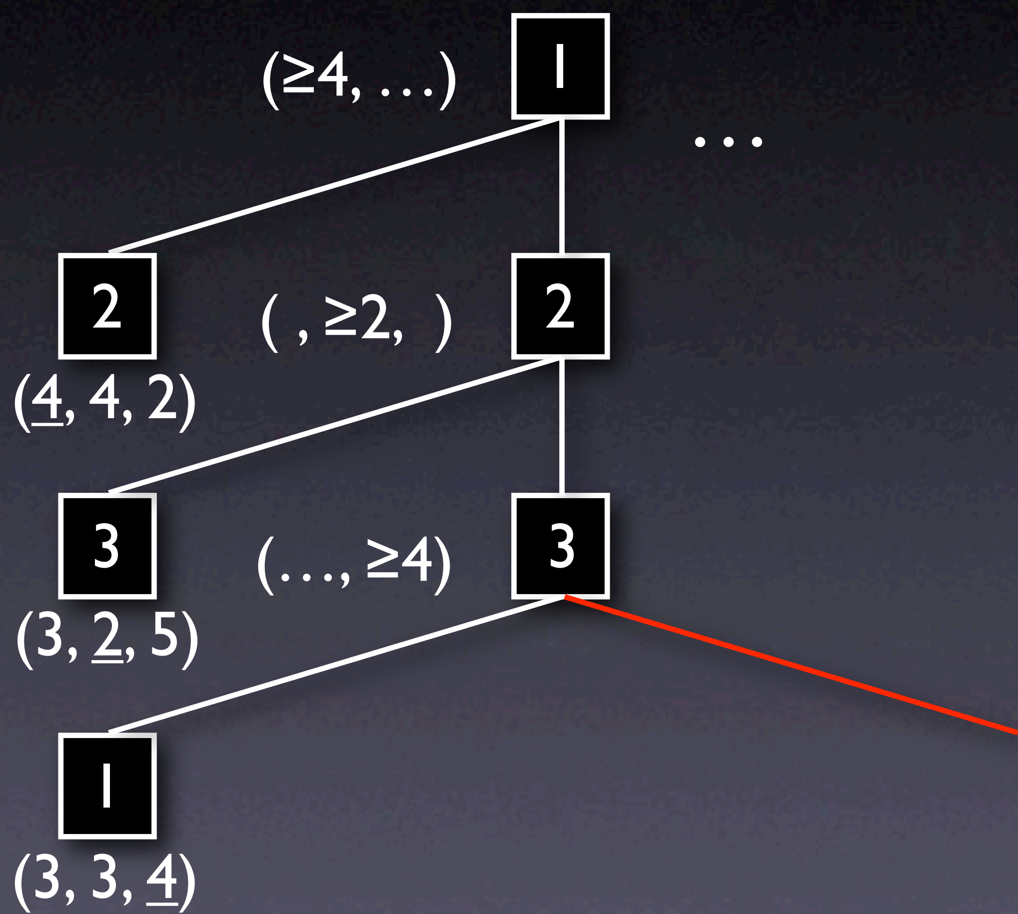
Speculative Pruning

maxsum = 10



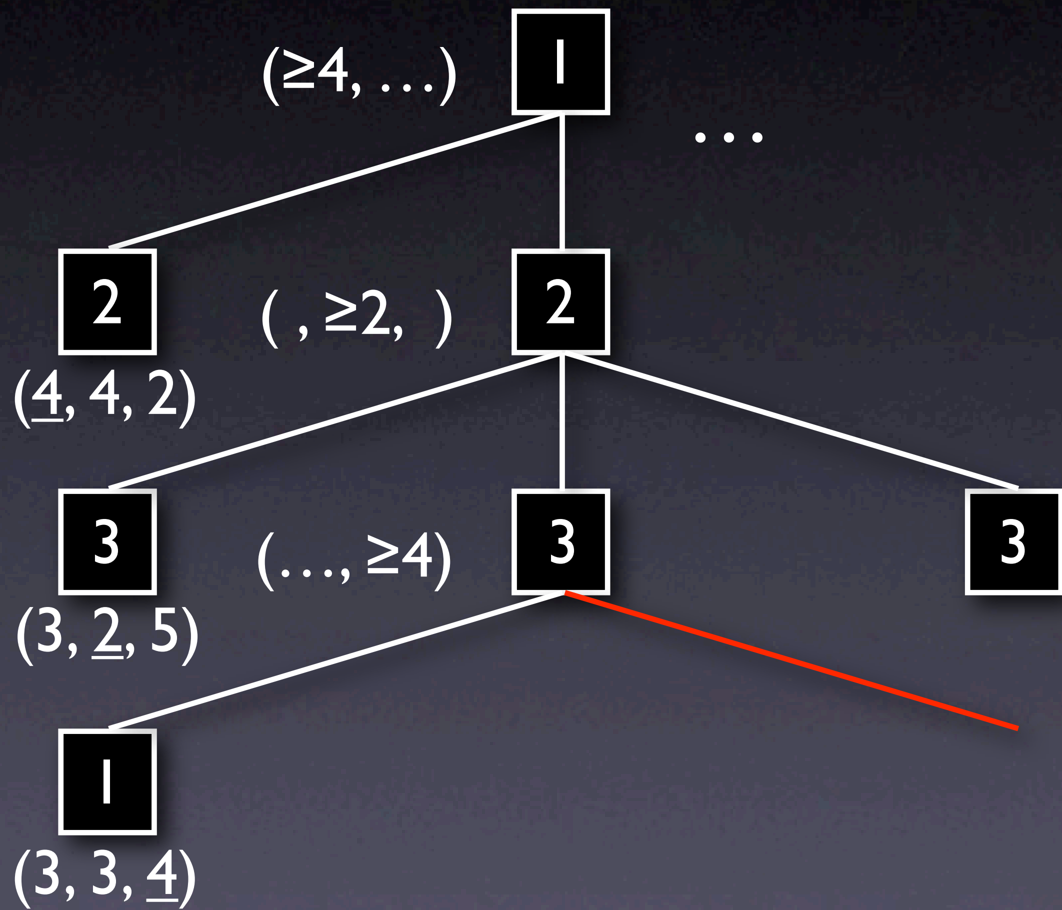
Speculative Pruning

maxsum = 10



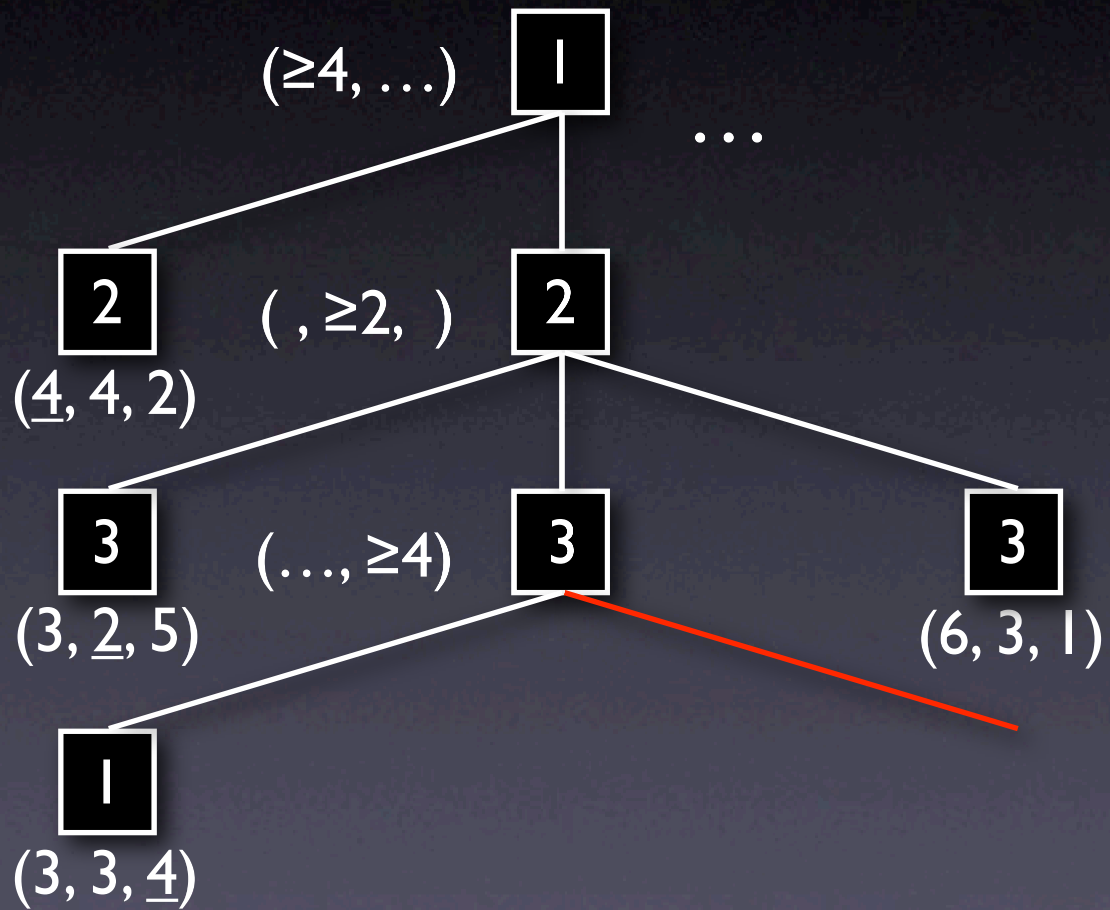
Speculative Pruning

maxsum = 10



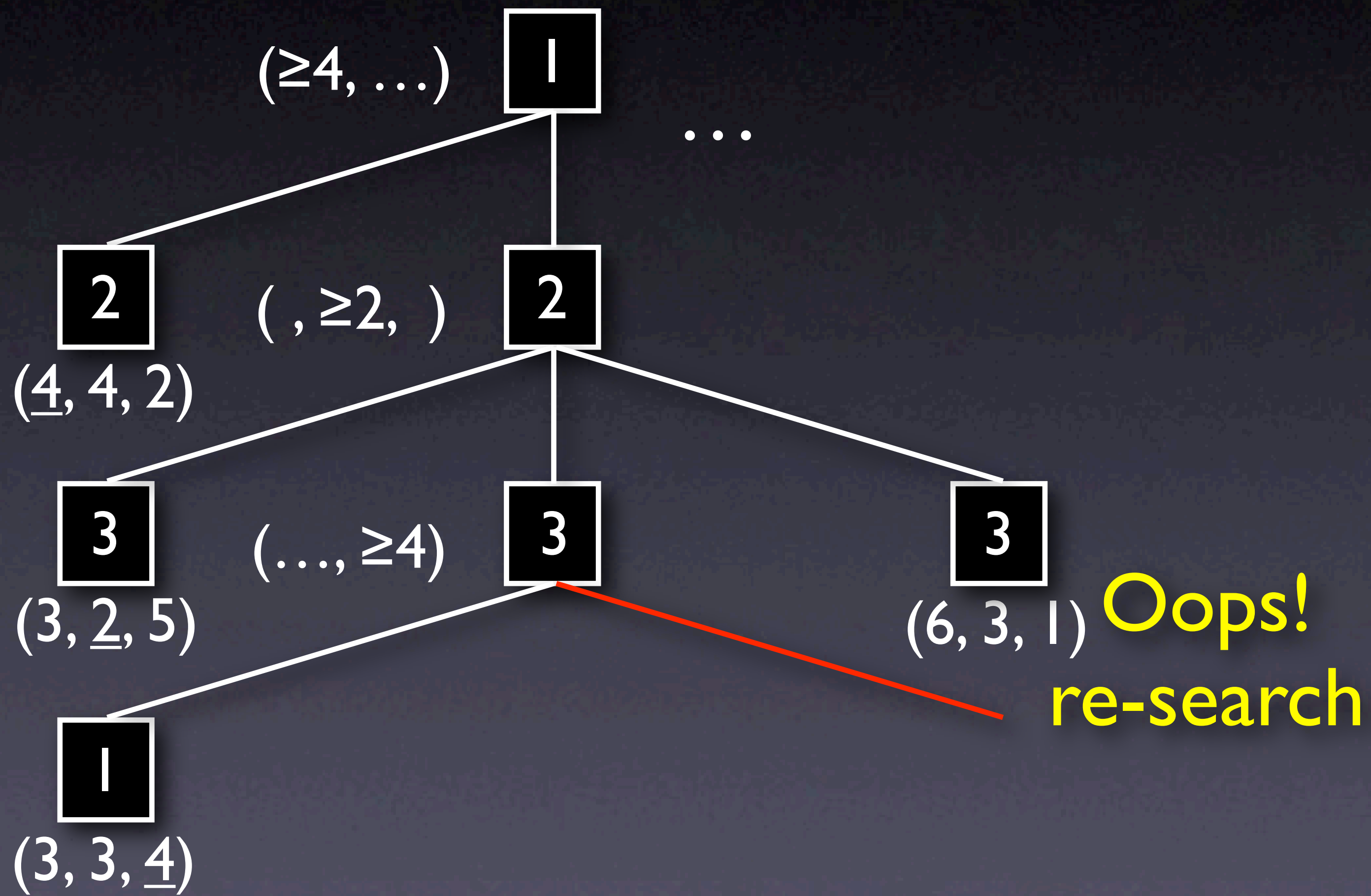
Speculative Pruning

maxsum = 10



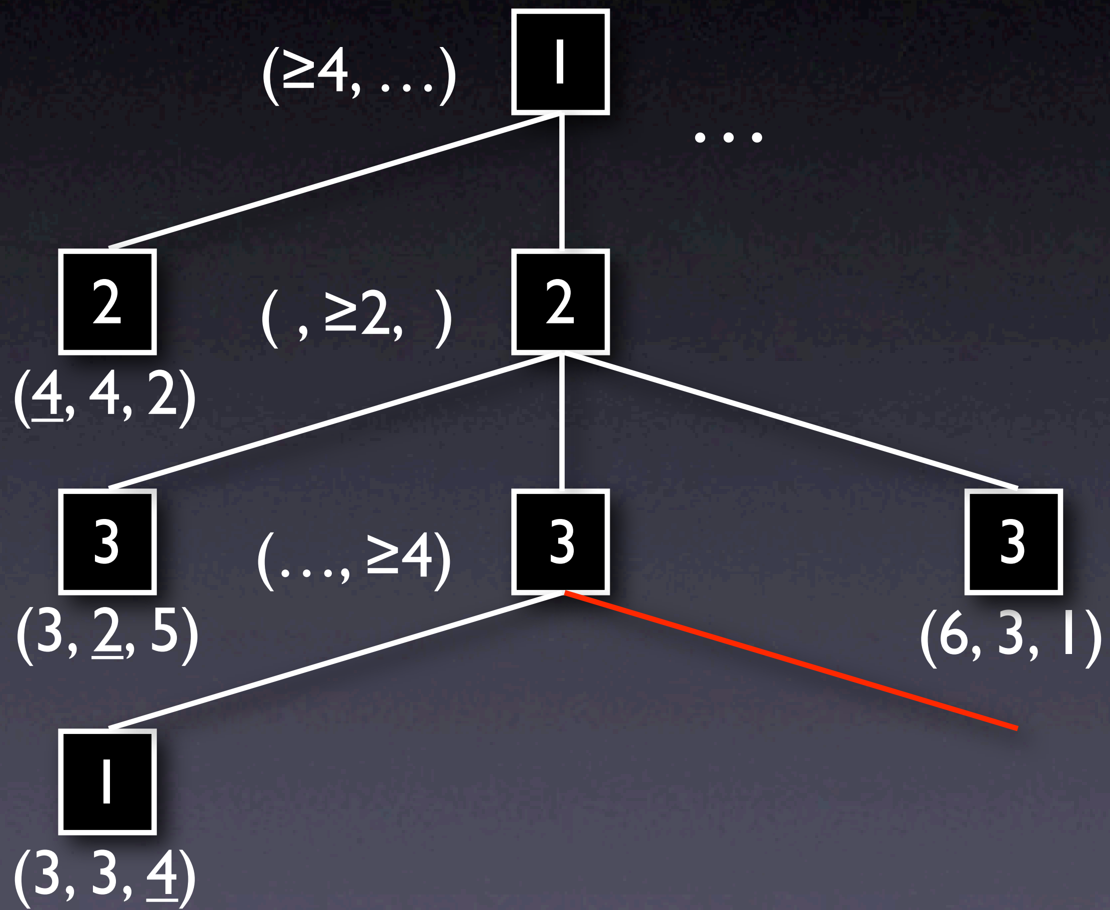
Speculative Pruning

maxsum = 10



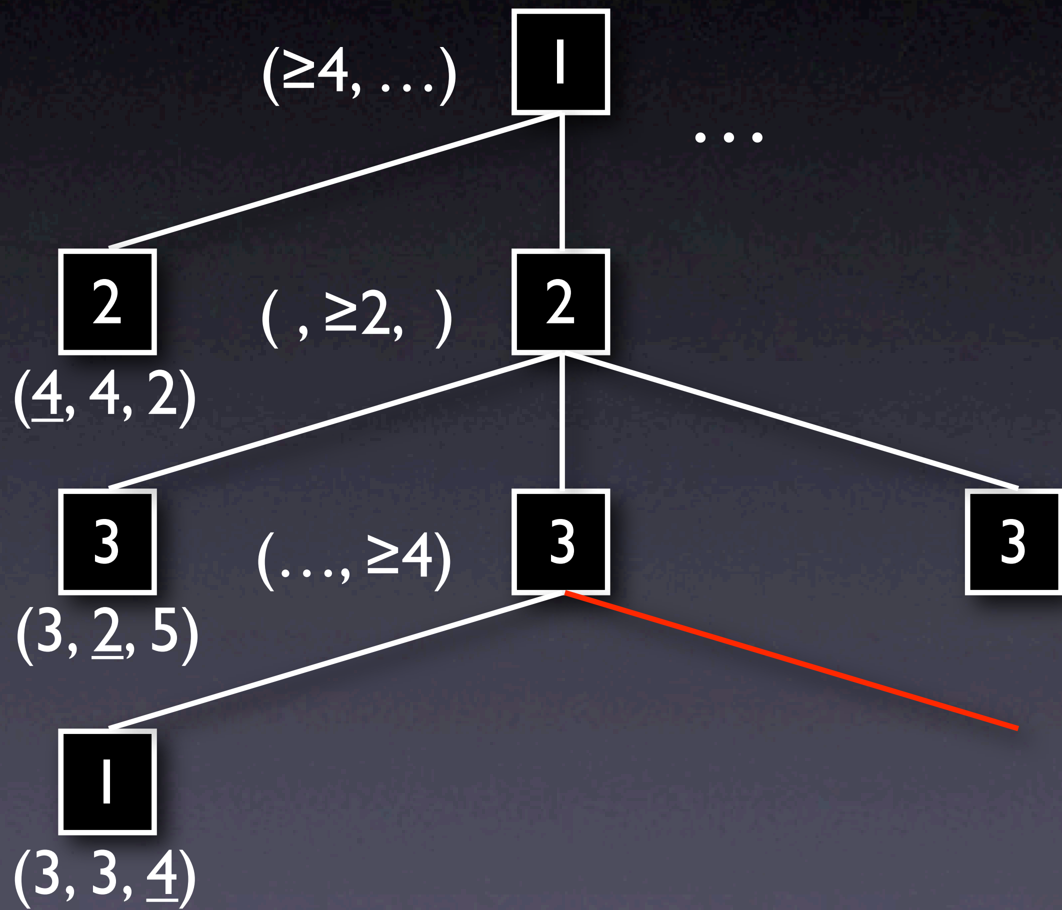
Speculative Pruning

maxsum = 10



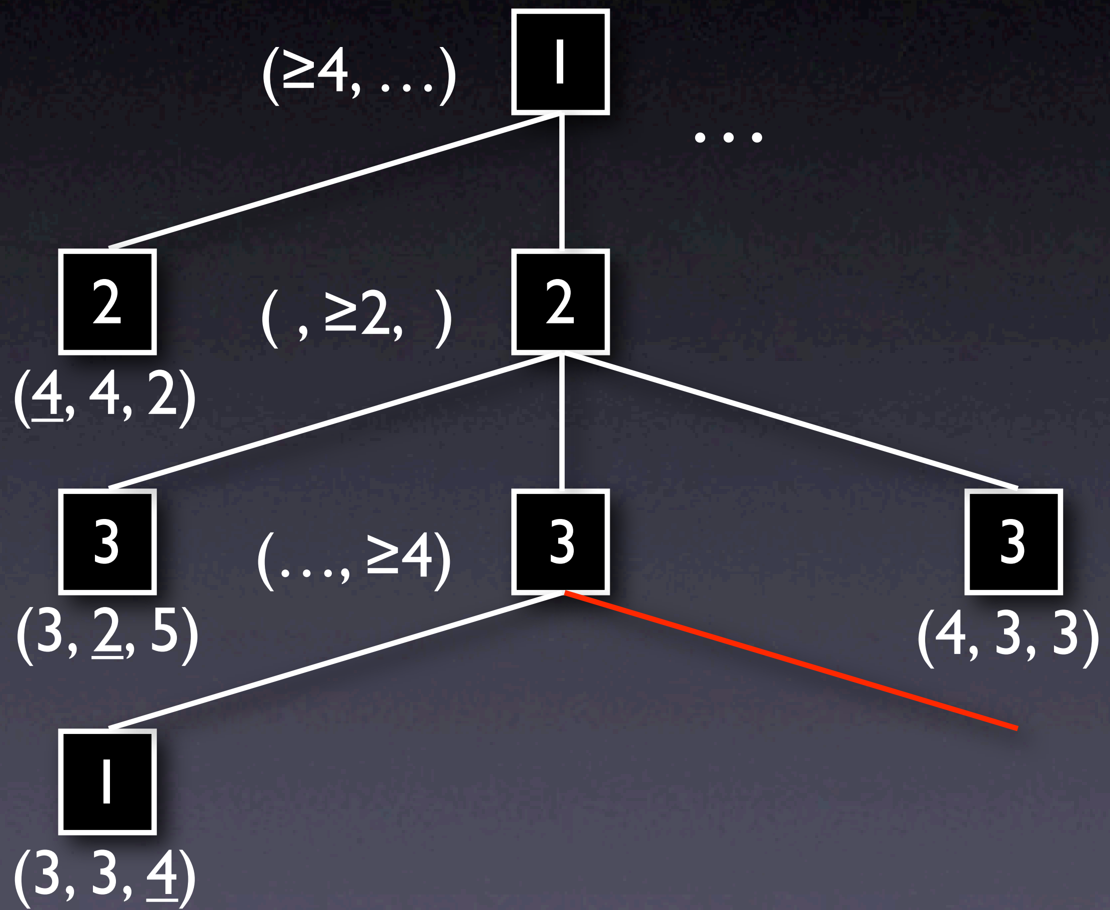
Speculative Pruning

maxsum = 10



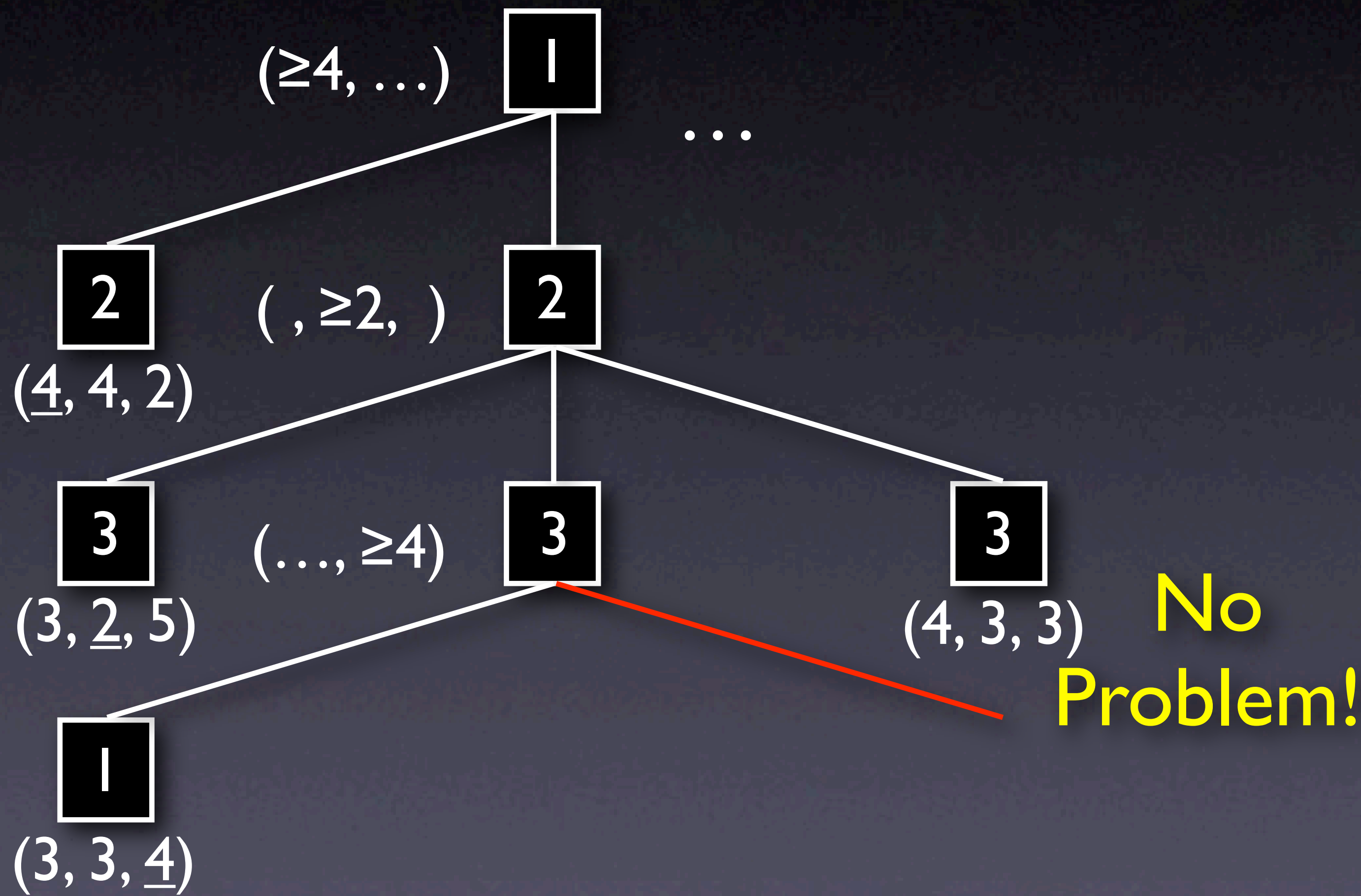
Speculative Pruning

maxsum = 10



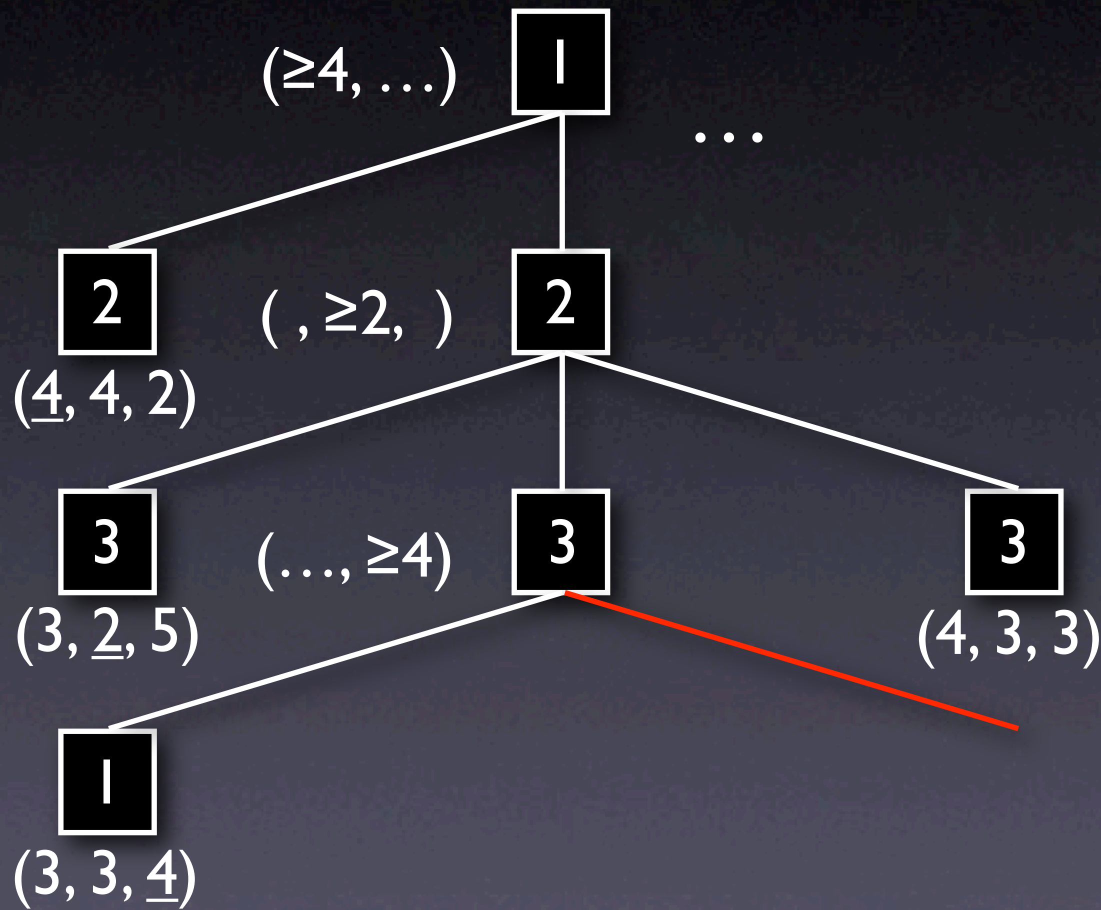
Speculative Pruning

maxsum = 10



Speculative Pruning

maxsum = 10



Speculative Pruning

Speculative Pruning

- Non-directional algorithm

20

Speculative Pruning

- Non-directional algorithm
  - Prunes are not guaranteed to be correct

Speculative Pruning

- Non-directional algorithm
  - Prunes are not guaranteed to be correct
  - Can re-search branches if needed

Speculative Pruning

- Non-directional algorithm
  - Prunes are not guaranteed to be correct
  - Can re-search branches if needed
- Works on any constant-sum game tree

Speculative Pruning

- Non-directional algorithm
  - Prunes are not guaranteed to be correct
  - Can re-search branches if needed
- Works on any constant-sum game tree
  - Effectiveness depends only on node ordering

20

Speculative Pruning

21

Speculative Pruning

- Reduces branching factor:

Speculative Pruning

- Reduces branching factor:
  - In the best case, as *b* gets large

Speculative Pruning

- Reduces branching factor:
  - In the best case, as *b* gets large
  - $b^{(n-1)/n}$

Speculative Pruning

- Reduces branching factor:
  - In the best case, as *b* gets large
  - $b^{(n-1)/n}$
- Effective even in the average case

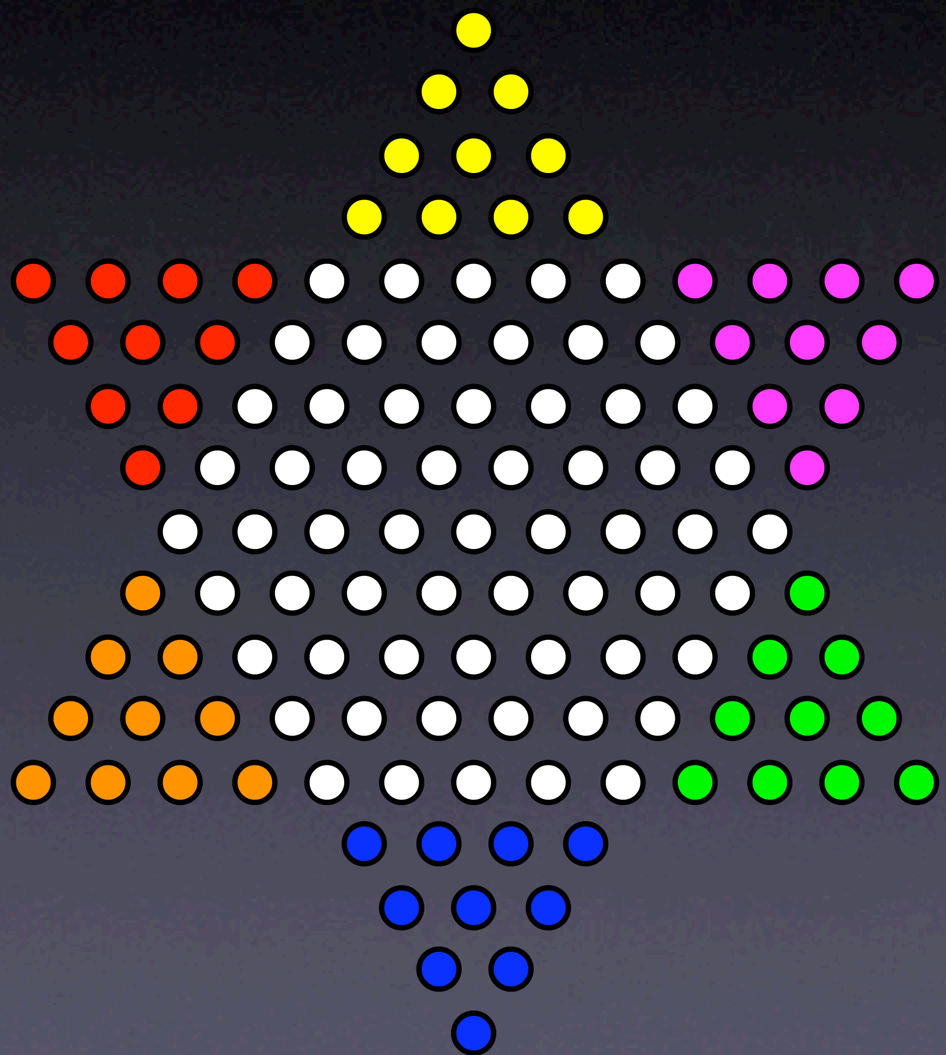
Speculative Pruning
(3-Players)

b	$b^{2/3}$	*actual b*
2	1.59	1.84
3	2.08	2.47
4	2.52	3.00
10	4.64	5.42
1000	100.0	103.6

Outline

- Max^n Decision Rule
- Max^n Pruning Techniques
- Experimental Results
- Conclusions

Chinese Checkers



Chinese Checkers

Chinese Checkers

- 3 Players

25

Chinese Checkers

- 3 Players
- Played 30 games, ~1500 searches

Chinese Checkers

- 3 Players
- Played 30 games, ~1500 searches
  - Iterative deepening search

25

Chinese Checkers

- 3 Players
- Played 30 games, ~1500 searches
    - Iterative deepening search
    - Limited branching factor to 10 moves

25

Chinese Checkers

- 3 Players
- Played 30 games, ~1500 searches
  - Iterative deepening search
  - Limited branching factor to 10 moves
  - Measured average expansions at depth 6

Chinese Checkers

- 3 Players
- Played 30 games, ~1500 searches
  - Iterative deepening search
  - Limited branching factor to 10 moves
  - Measured average expansions at depth 6
- Previous algorithms could not prune tree

25

Chinese Checkers

Algorithm	Node expansions (depth 6)
Max^n	1.2 million
Spec. Max^n	100k

Hearts and Spades

Hearts and Spades

- Trick-based card games

Hearts and Spades

- Trick-based card games
  - Have monotonic heuristics

Hearts and Spades

- Trick-based card games
  - Have monotonic heuristics
- Shallow pruning does not occur in Hearts

Hearts and Spades

- Trick-based card games
  - Have monotonic heuristics
- Shallow pruning does not occur in Hearts
- Measured average search depth

Hearts and Spades

- Trick-based card games
  - Have monotonic heuristics
- Shallow pruning does not occur in Hearts
- Measured average search depth
  - Iterative deepening search

Hearts and Spades

- Trick-based card games
  - Have monotonic heuristics
- Shallow pruning does not occur in Hearts
- Measured average search depth
  - Iterative deepening search
  - 500k node search limit

Average Search Depth

	Hearts	Spades
Best Prev. Max^n	20.9	21.7
Spec. Max^n	22.1	24.3

Conclusions

29

Conclusions

- New pruning algorithms for max^n

Conclusions

- New pruning algorithms for max^n
- Last-Branch Pruning

Conclusions

- New pruning algorithms for max^n
- Last-Branch Pruning
    - Directional algorithm

29

Conclusions

- New pruning algorithms for max^n
- Last-Branch Pruning
  - Directional algorithm
- Speculative Pruning

29

Conclusions

- New pruning algorithms for max^n
- Last-Branch Pruning
  - Directional algorithm
- Speculative Pruning
  - Non-directional algorithm

Conclusions

- New pruning algorithms for max^n
- Last-Branch Pruning
  - Directional algorithm
- Speculative Pruning
  - Non-directional algorithm
- First algorithms effective in pruning any constant-sum game