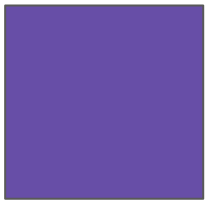
Simulating Extensive-Form Dilemmas

Justus Hibshman - 11/08/18

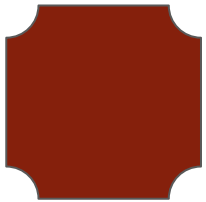
Dilemmas:

P1

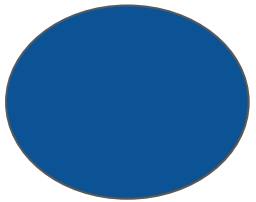
“Ok.”



“Wonderful!”



“Ugh.”



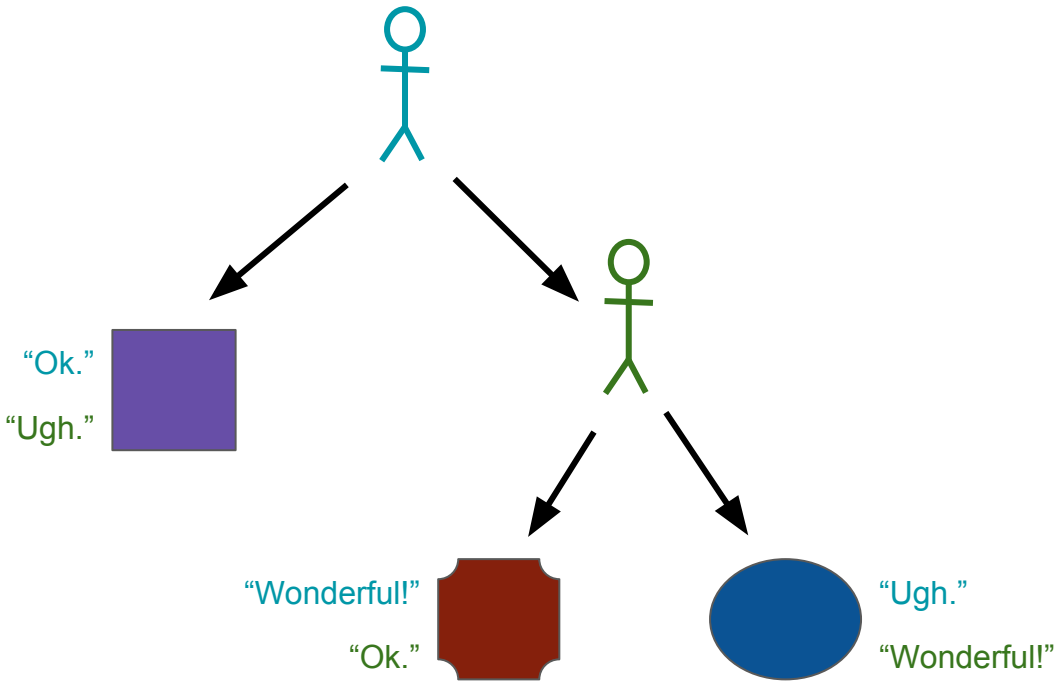
P2

“Ugh.”

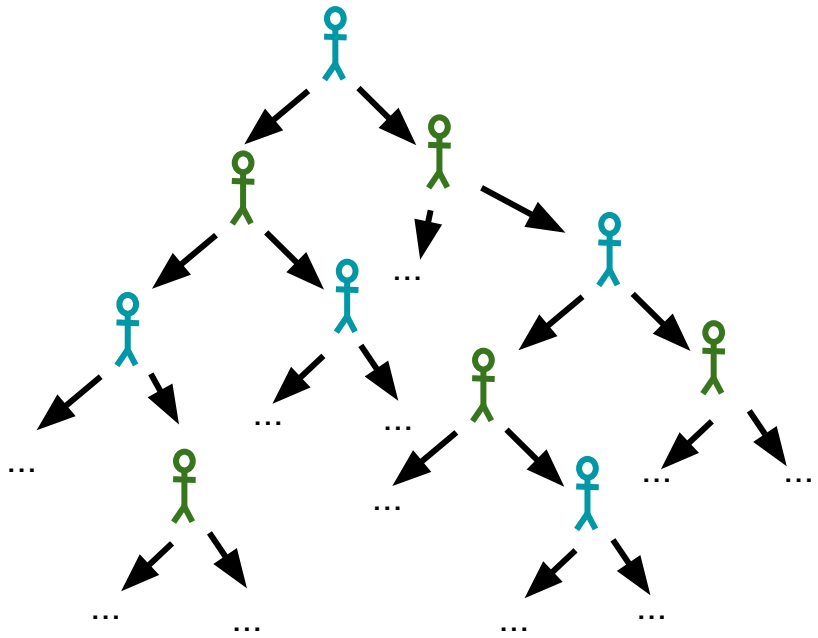
“Ok.”

“Wonderful!”

Dilemmas:



Do Dilemmas Occur Naturally?



Kernel: MiniMax with DFS

```
Def MiniMax(root)
    If(root.children.size() == 0)
        Return root.preferences
    best_outcome = -inf
    result = Null
    For child in root.children
        child_result = MiniMax(child)
        If(child_result[root.player_id] > best_outcome)
            best_outcome = child_result[root.player_id]
            result = child_result
    Return result
```

Complexity

Number of players: **p**
Number of vertices: **v**
Number of edges: **e = v - 1** because its a tree.

MiniMax with DFS: O(**p**(**v** + **e**)) = O(**pv**)

Game Tree Generation: O(**pv**)

Checking Optimality of Result: O(**pv**)

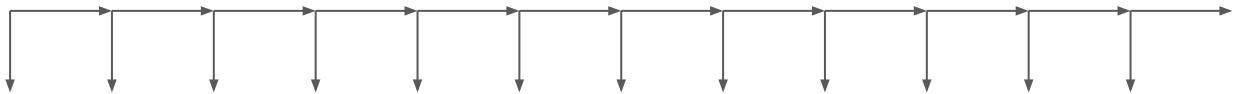
Implementation

- DFS: Boost Graph Library (C++)
  - Overload “DFS Visitor” class
- Game Generation
  - Custom C++ code
  - Boost graph format
  - Players’ preferences are strict orderings (all >, no >=).
  - “Player i prefers A over B” tells you nothing about Player j’s preferences.

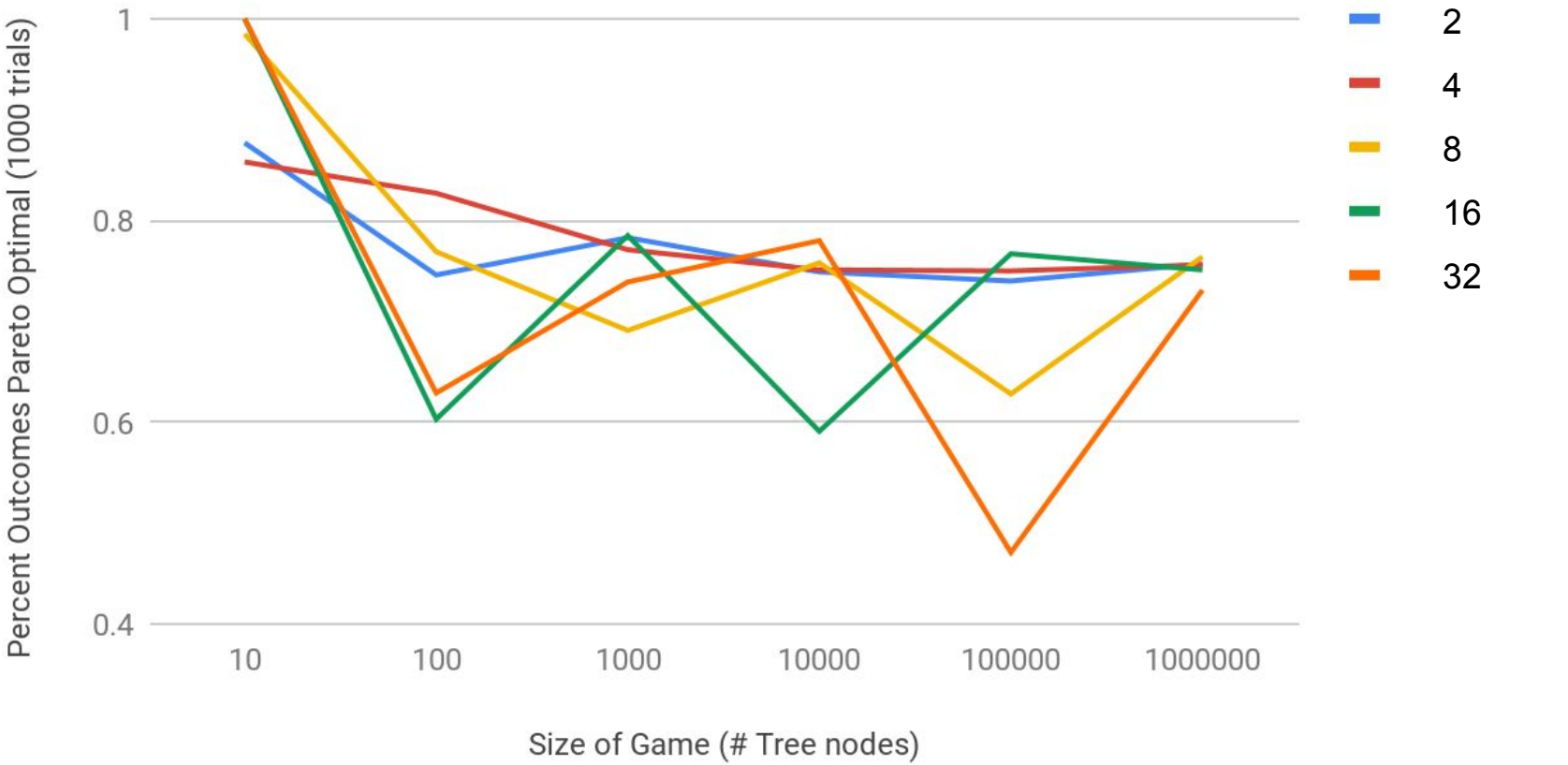
Experiment Setup

Run 1000 trials for each parametrization:

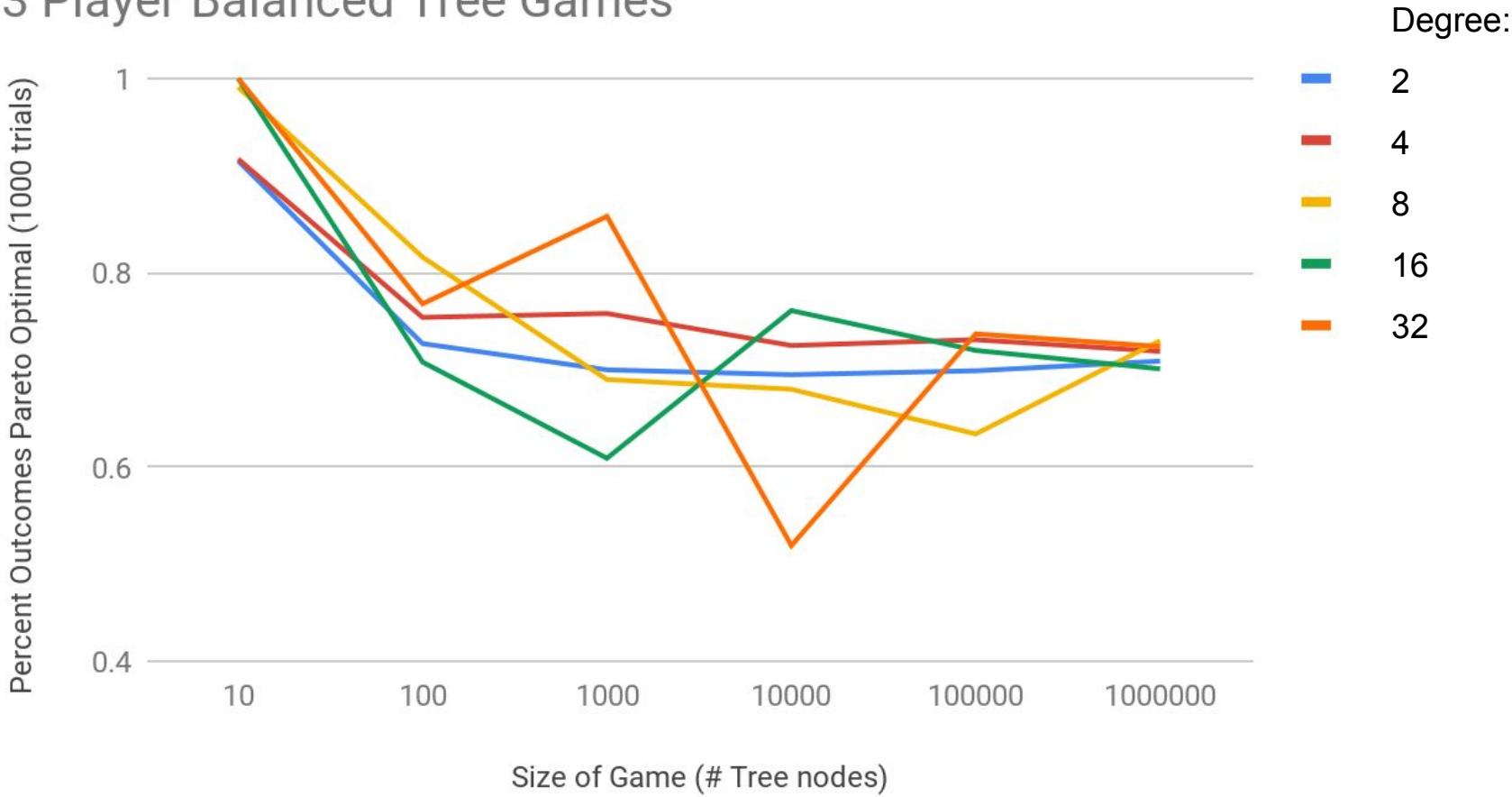
- Vary number of players from 2 to 5
- Vary number of game tree nodes from 10 to 1,000,000
- Balanced Trees
  - Vary degree from 2 to 32
- “Chain” Trees
  - Every decision node has one stop-edge leading to a final outcome and one continue-edge.
  - Example:



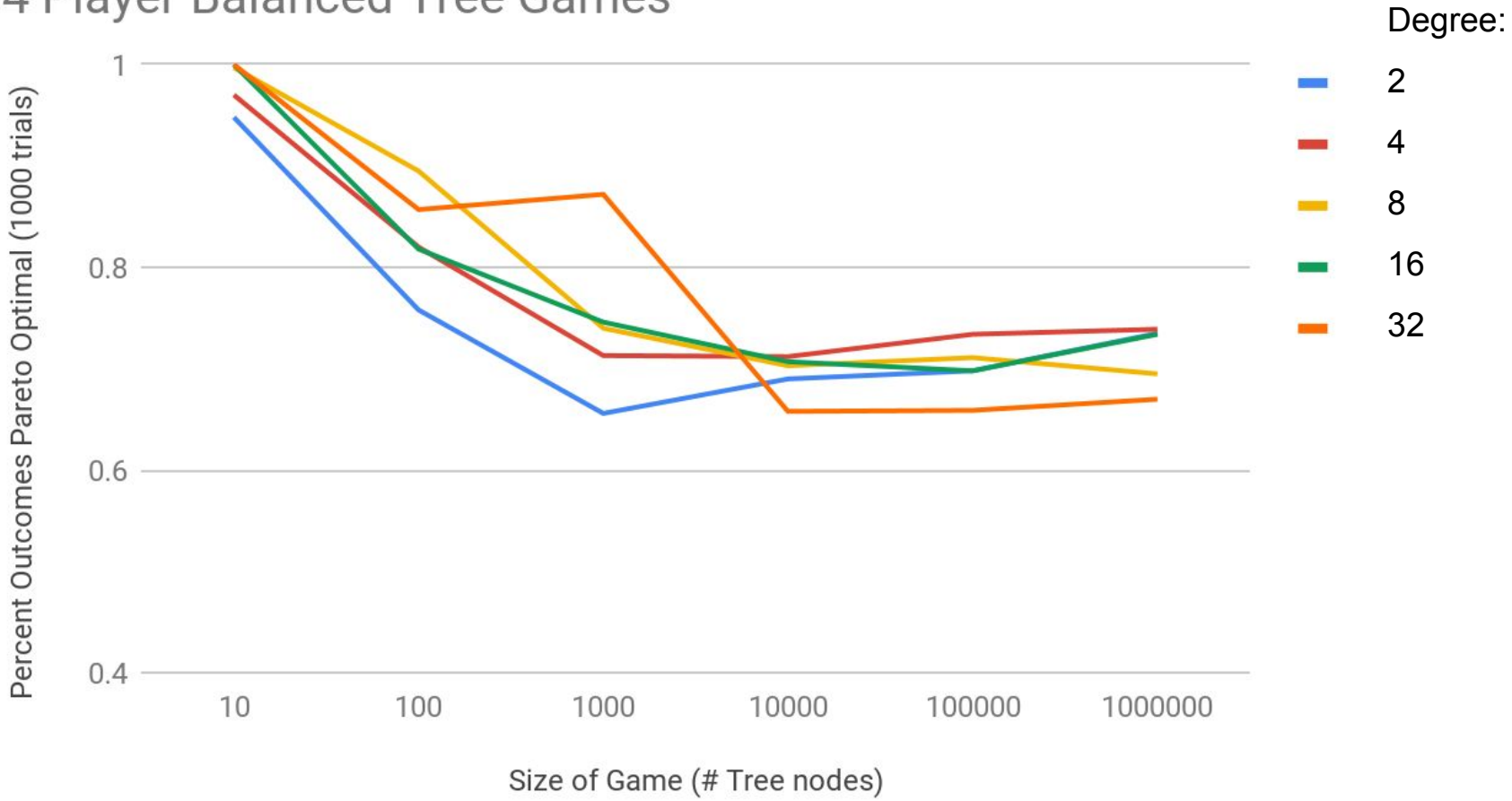
2 Player Balanced Tree Games



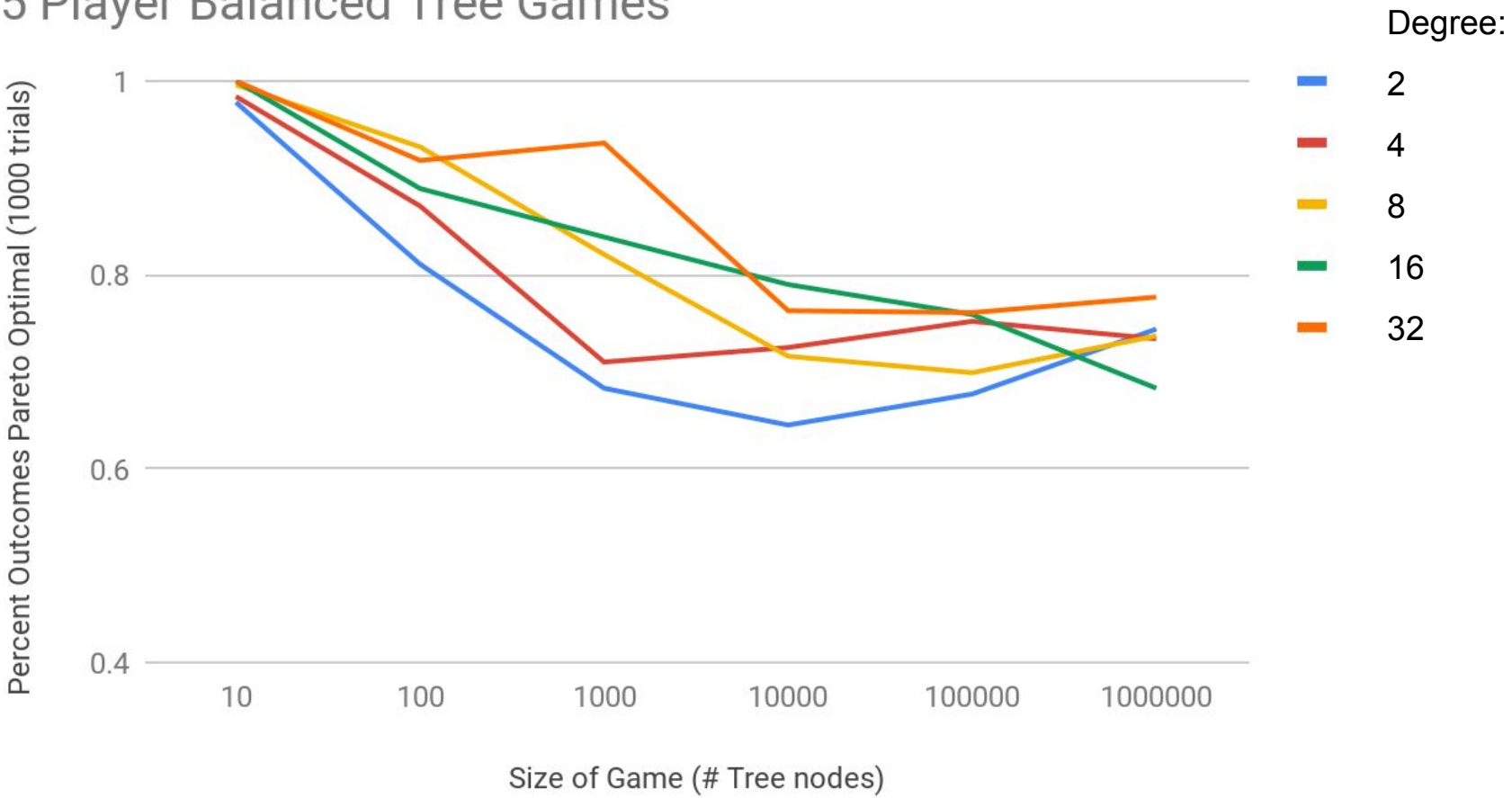
3 Player Balanced Tree Games



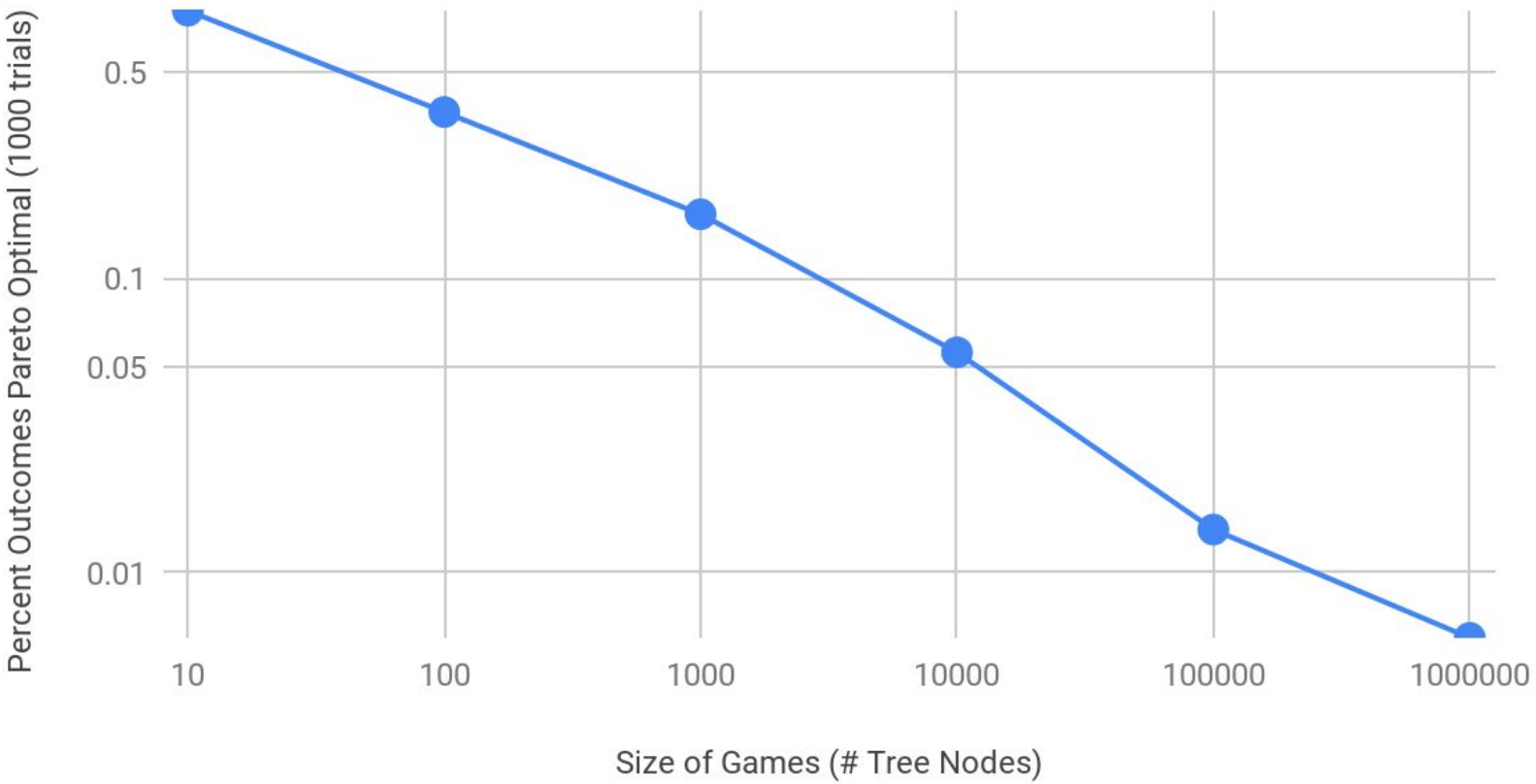
4 Player Balanced Tree Games



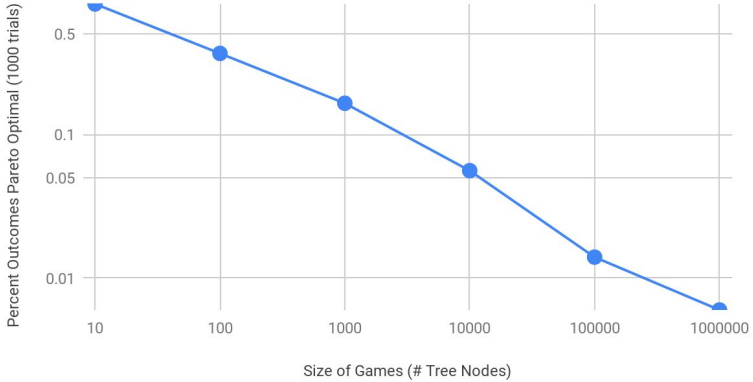
5 Player Balanced Tree Games



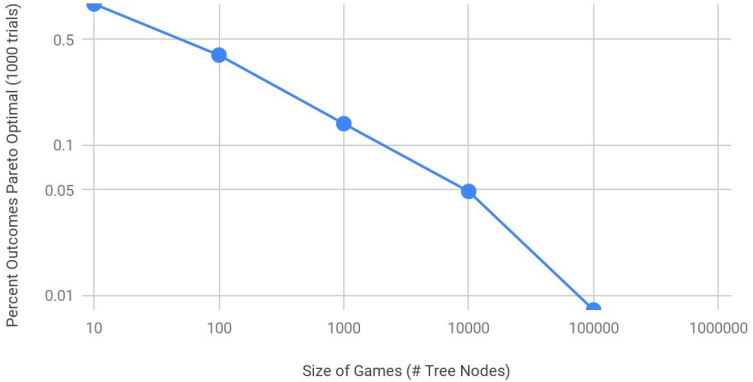
2 Player Chain Games



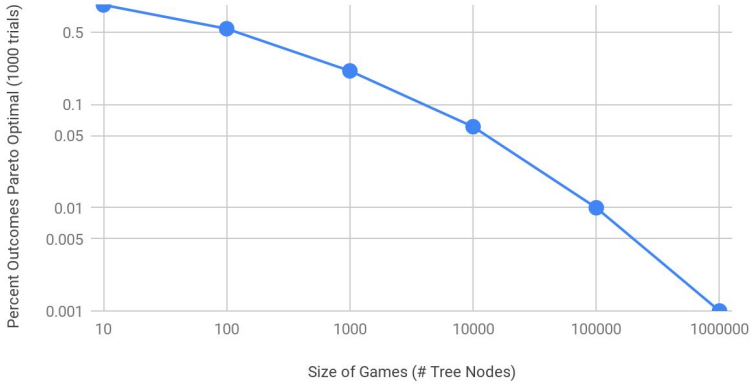
2 Player Chain Games



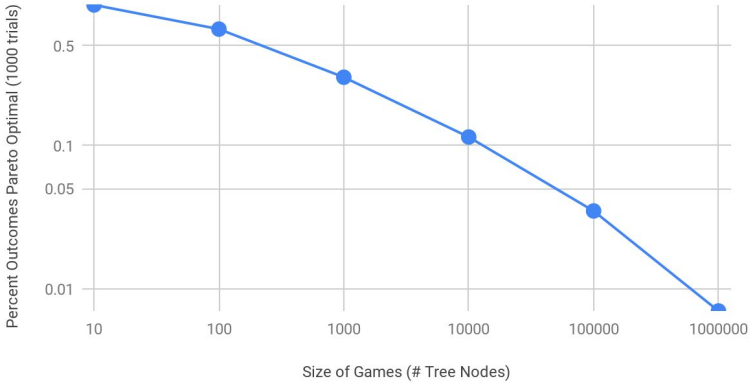
3 Player Chain Games



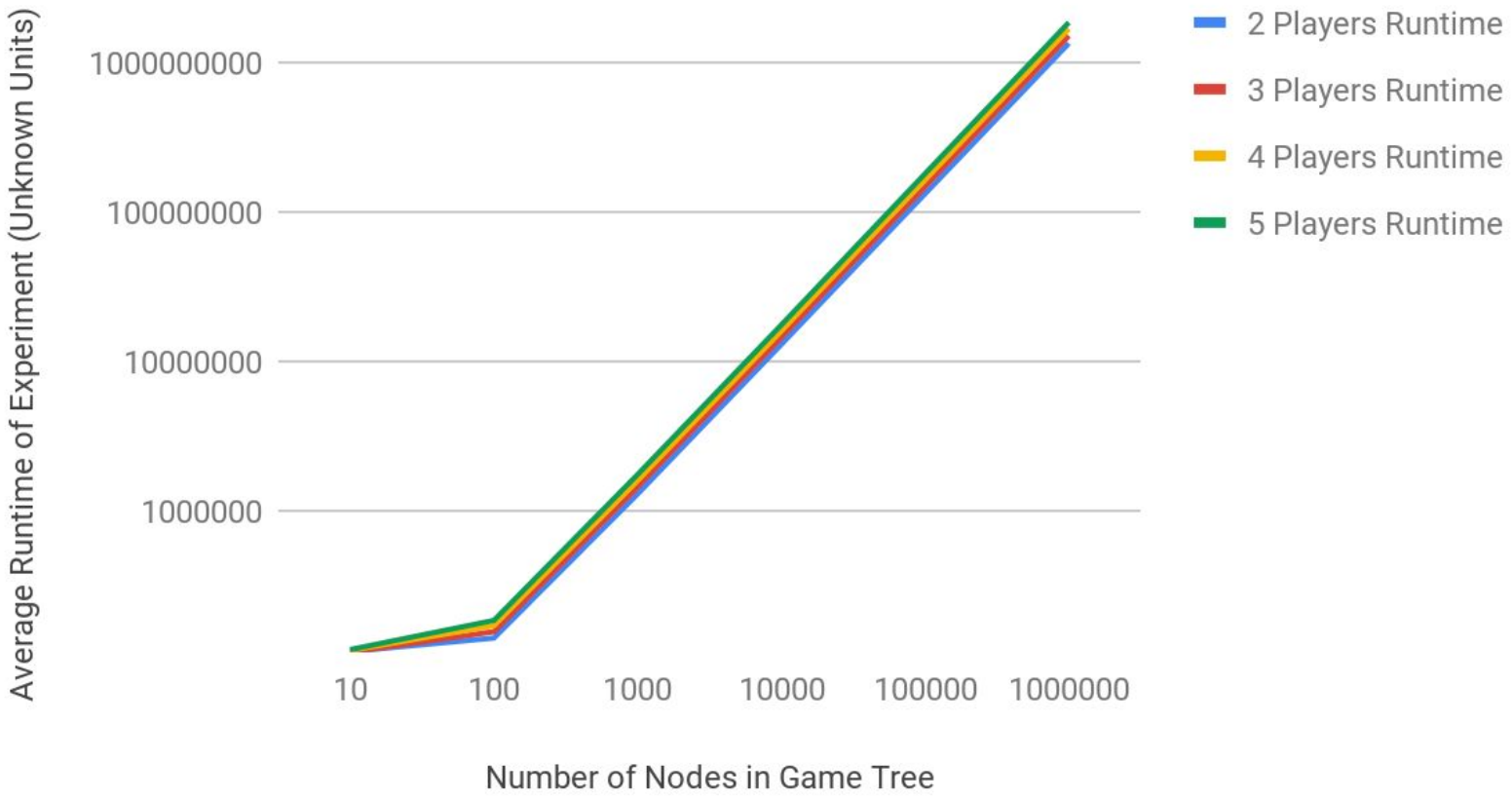
4 Player Chain Games



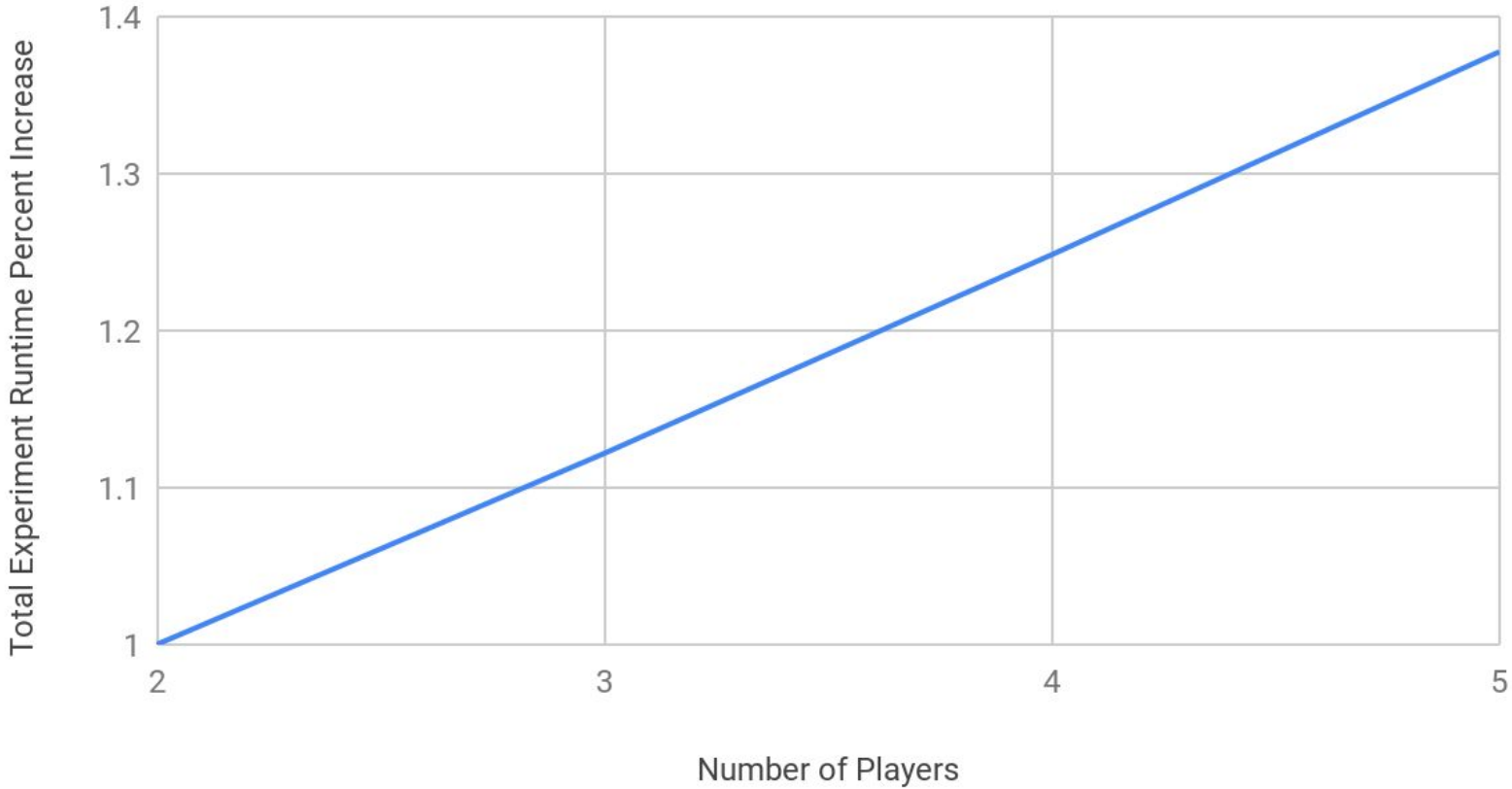
5 Player Chain Games



Runtime Scaling with Game Size



Runtime Scaling with Number of Players



Future Work

- Parallelize DFS
  - May require locks in DFS visitor?
- Parallelize Game Generation
  - Need to learn MPI?
- Rather than running a single test in parallel, run different tests on different processes? (Less interesting. Might waste memory.)