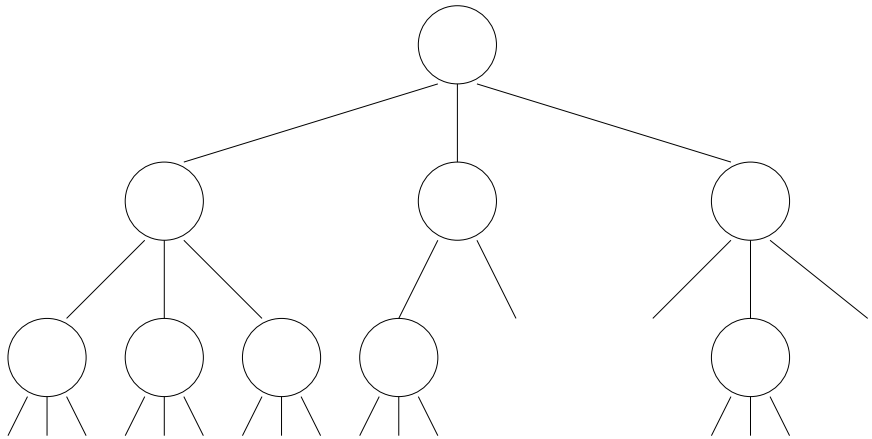
Tree Parallelization of Ary on a Cluster

Jean Méhat, Tristan Cazenave

Université Paris 8 & Université Paris-Dauphine

Giga 2011, July, 2011

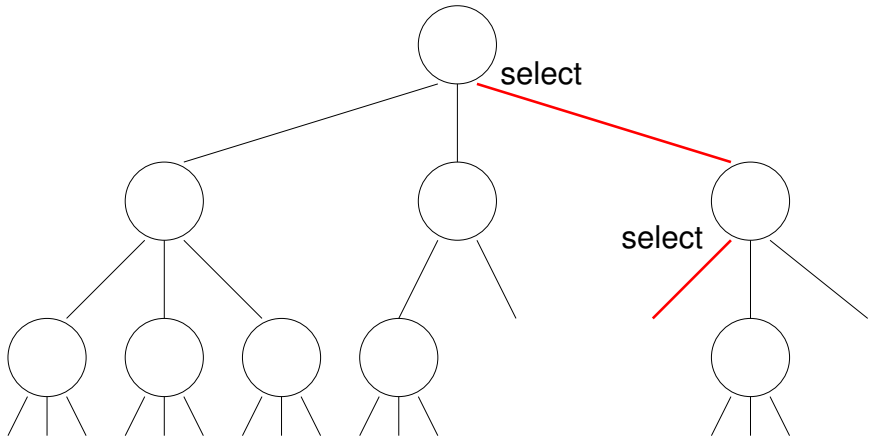
Monte-Carlo Tree Search (MCTS)



- Select a path to a leaf
- Create a new node
- Perform a playout
- Update move estimations on the path

All these can be parallelized

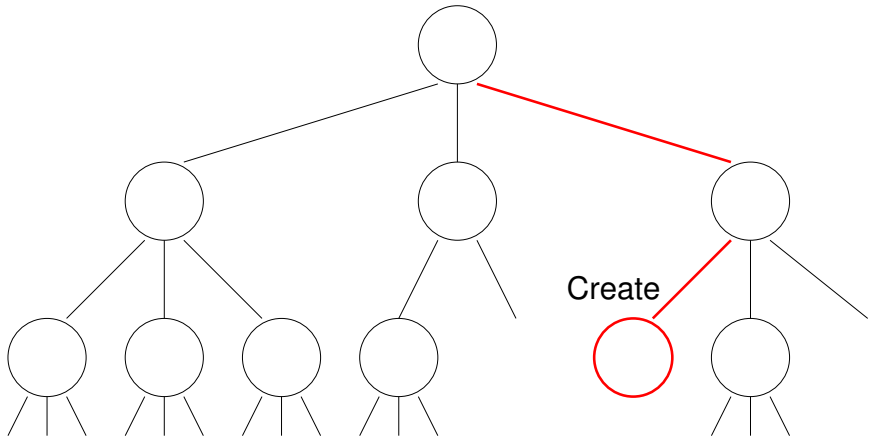
Monte-Carlo Tree Search (MCTS)



- Select a path to a leaf
- Create a new node
- Perform a playout
- Update move estimations on the path

All these can be parallelized

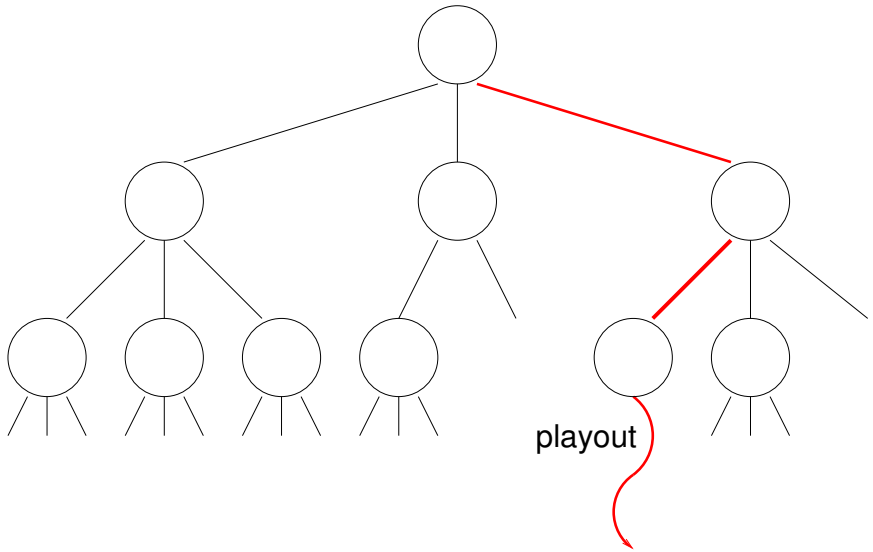
Monte-Carlo Tree Search (MCTS)



- Select a path to a leaf
- Create a new node
- Perform a playout
- Update move estimations on the path

All these can be parallelized

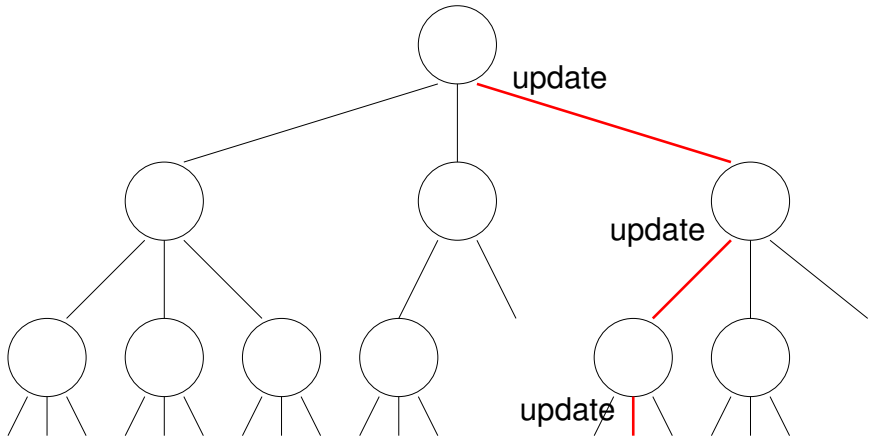
Monte-Carlo Tree Search (MCTS)



- Select a path to a leaf
- Create a new node
- Perform a playout
- Update move estimations on the path

All these can be parallelized

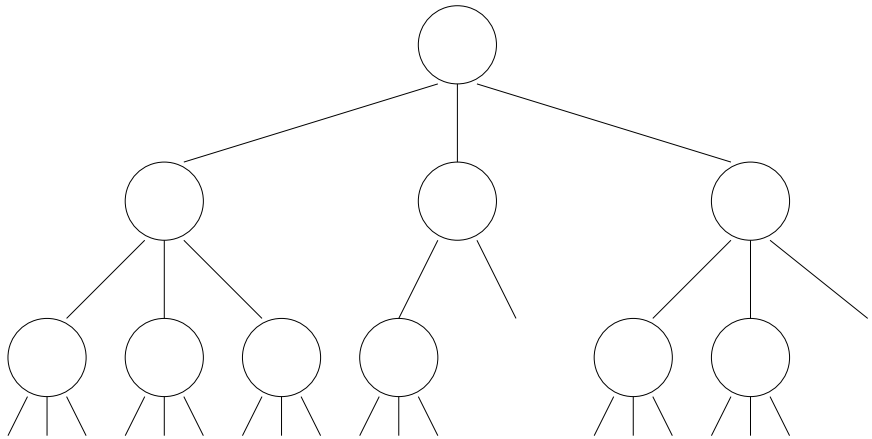
Monte-Carlo Tree Search (MCTS)



- Select a path to a leaf
- Create a new node
- Perform a playout
- Update move estimations on the path

All these can be parallelized

Monte-Carlo Tree Search (MCTS)



- Select a path to a leaf
- Create a new node
- Perform a playout
- Update move estimations on the path

All these can be parallelized

MCTS Parallelization

Many ways to parallelize.

- Path selection and node creation (full Tree Parallelism). For multicore machines.
- Playouts (restricted Tree parallelism). Interesting when playouts are slow.
- Many parallel playouts for each new node (Leaf Parallelism). Does not seem to give good results.
- Many trees developped independently (Root Parallelism). Merge evaluations at end of thinking time.

+ Mixed Schemes (e. g. Root Parallelism + periodic exchanges of move estimations near the root)

Available hardware and software

Not used:

- Big clusters are expensive and/or complicated to exploit and maintain.
- Middleware (e.g. MPI) is complicated and/or does not show astounding performances. Difficult to debug.

What we use:

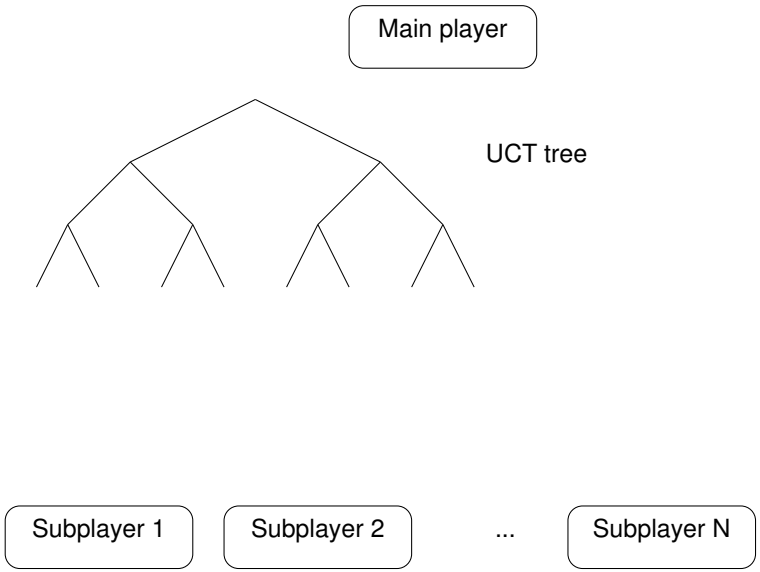
- Many (relatively) low end PCs in Student Lab.
- TCP channels: simple to use, good performances, well understood. Can be used for communication and synchronisation.

GGP characteristics:

- slow GDL interpretation
- large memory usage

Tree Parallelism

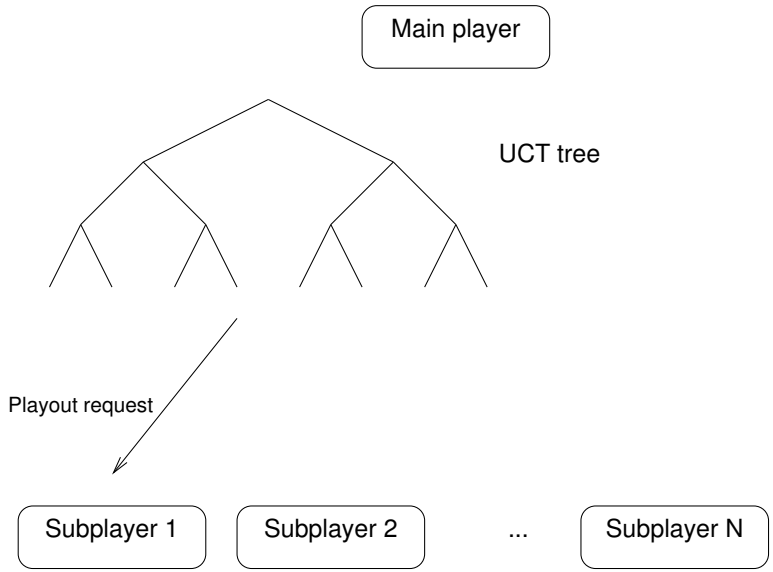
Delegate playouts to Subplayers.



Node expansion is the only GDL interpretation in the Main Player.

Tree Parallelism

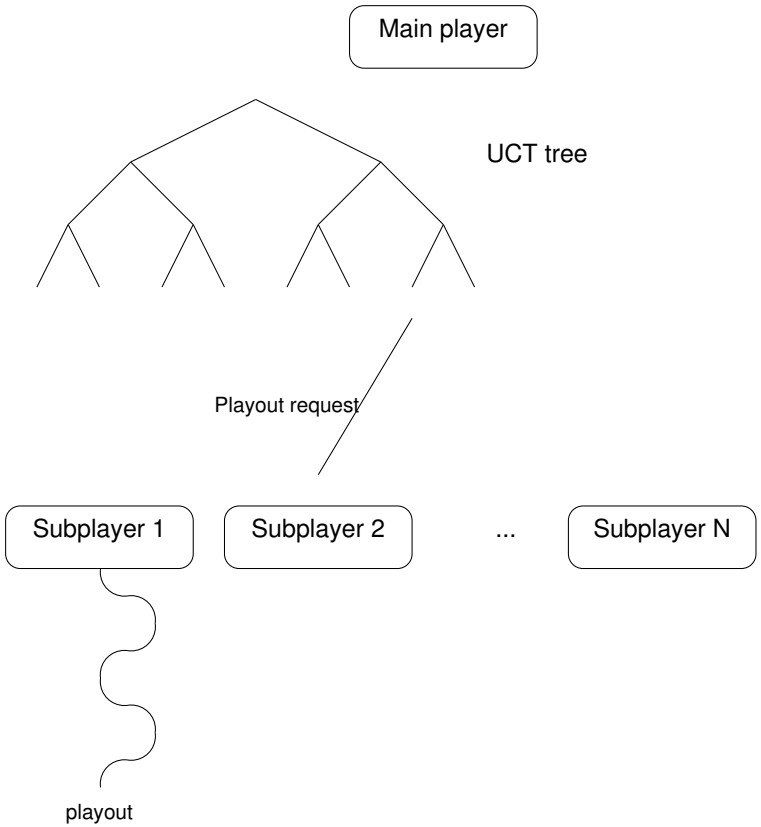
Delegate playouts to Subplayers.



Node expansion is the only GDL interpretation in the Main Player.

Tree Parallelism

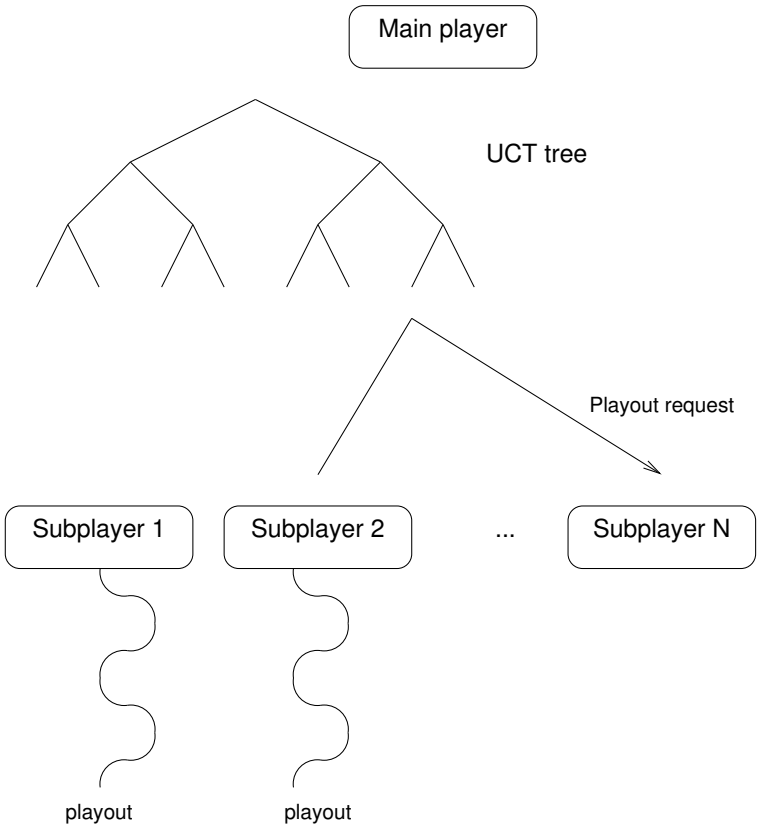
Delegate playouts to Subplayers.



Node expansion is the only GDL interpretation in the Main Player.

Tree Parallelism

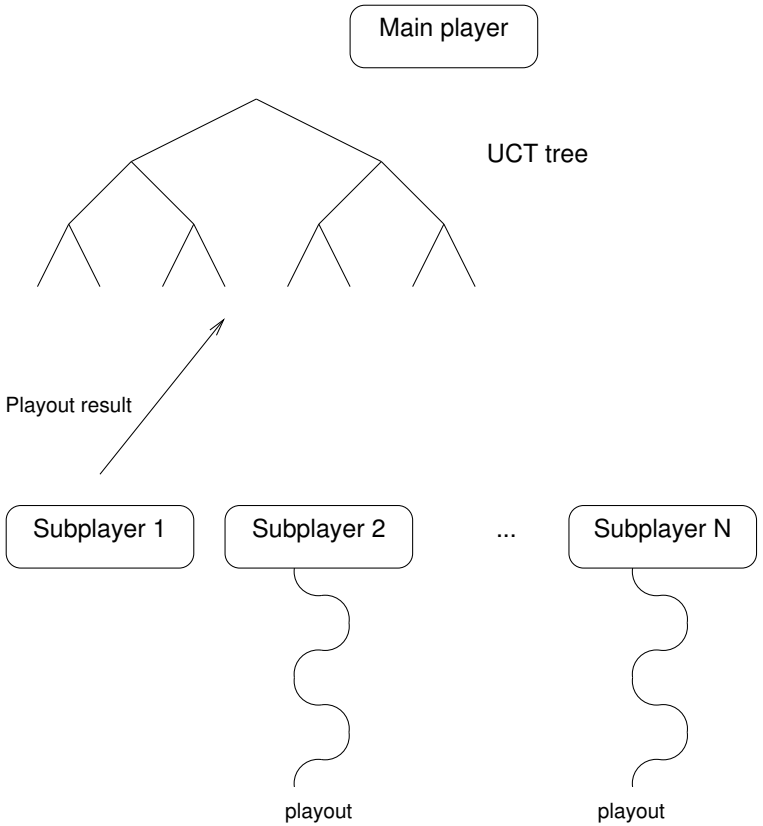
Delegate playouts to Subplayers.



Node expansion is the only GDL interpretation in the Main Player.

Tree Parallelism

Delegate playouts to Subplayers.



Node expansion is the only GDL interpretation in the Main Player.

Subplayer implementation

Subplayer implementation is straightforward:

```
receive Game Description
while(true){
    get a state description
    perform a playout
    send back result for all players
        (with moves played to enable RAVE etc.)
}
```

Communications as readable KIF strings over TCP channels.

One TCP connection between every Subplayer and the Main Player for the match.

Main Player implementation

Replace

```
  while(there is time){
      choose branch
      expand node
      run playout
      update tree
  }
```

with

```
  while(there is time){
      choose branch
      expand node
      send playout request
      collect available playout results
        and update tree
  }
```

Experimental setup

Varying the number of Subplayers.
100 matches with Tree Parallel as second player against a serial Ary.
A group of approx. 40 PC computers (student lab)
Switched 100 Mb Ethernet.
CPUs from 2GHz to 3GHz, only one core per machine used (avoid memory competition)
2 Gbytes of memory
Tested on *Breakthrough, Connect 4, Othello, Pawn whopping, Pentago* and *Skirmish*.
8 seconds initial time + 8 seconds per move

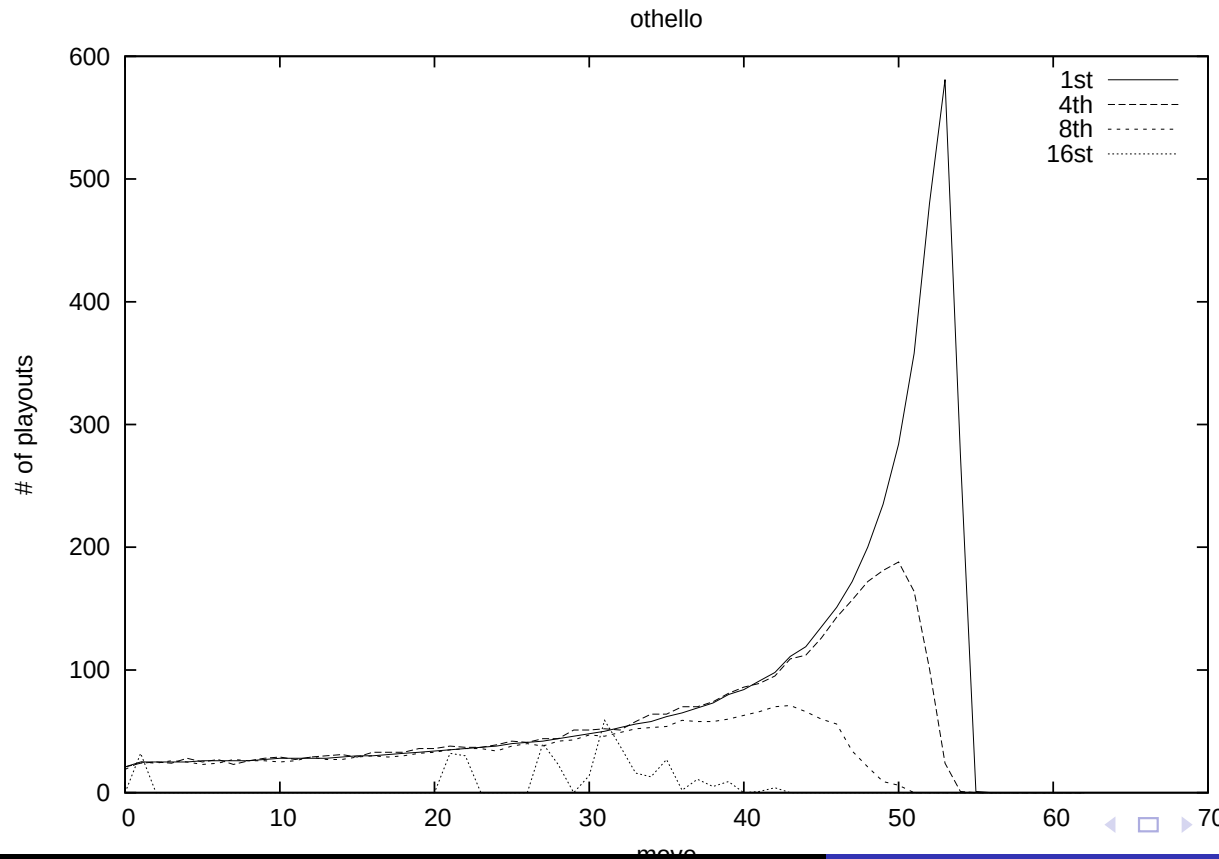
Results for Tree Parallelism

game	Number of Subplayers 1	2	4	8	16
Breakthrough	18	40	58	68	77
Connect 4	26	38	38	42	50
Othello	41	68	67	81	96
Pawn whopping	43	36	51	54	59
Pentago	38	40	55	73	87
Skirmish	73	76	76	78	79

Serious penalty when the number of Subplayers is small (one communication per playout).
Improvement for all the games when the number of Subplayers increases.

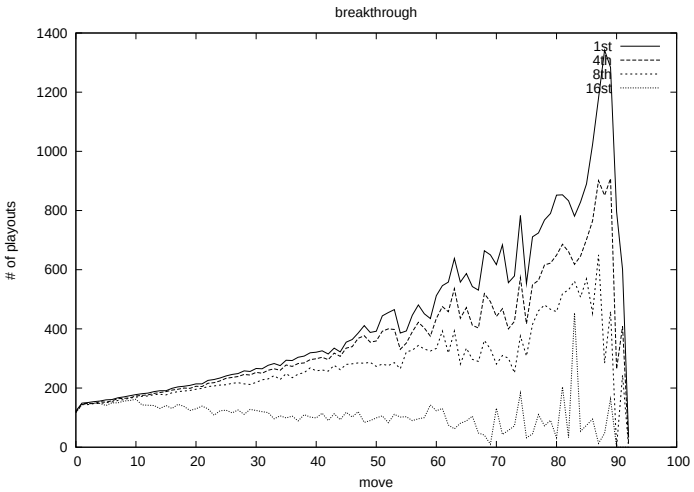
Use of the Subplayers (1/2)

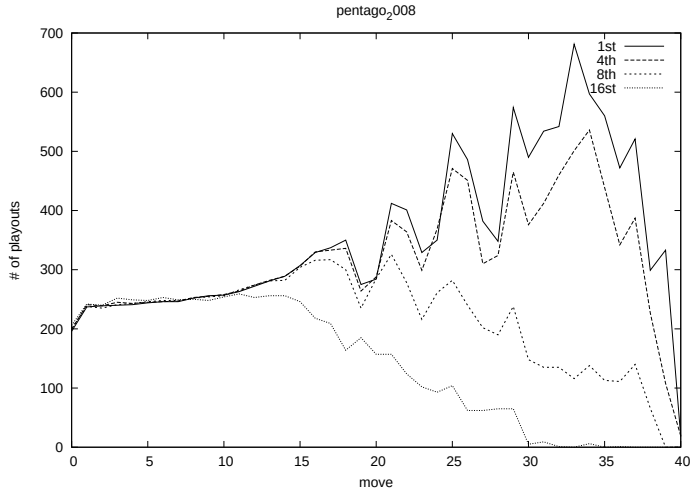
Two limiting factors: node creation (Main) and Subplayer availability.
At a match goes on, playouts usually get shorter.
Count the number of playouts performed by a Subplayer
$\rightarrow$ is there enough Subplayers?

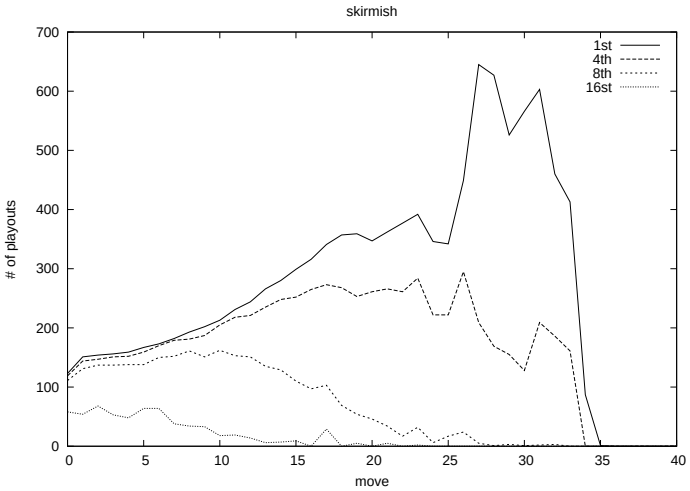


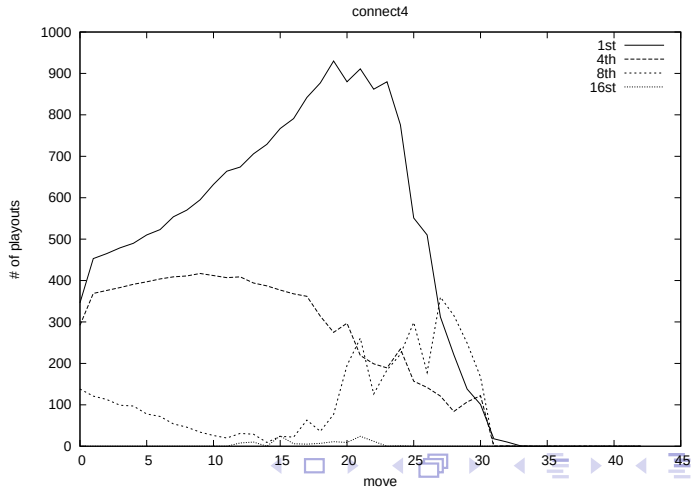
Use of the Subplayers (2/2)

The curve for all the studied games have the same shape.



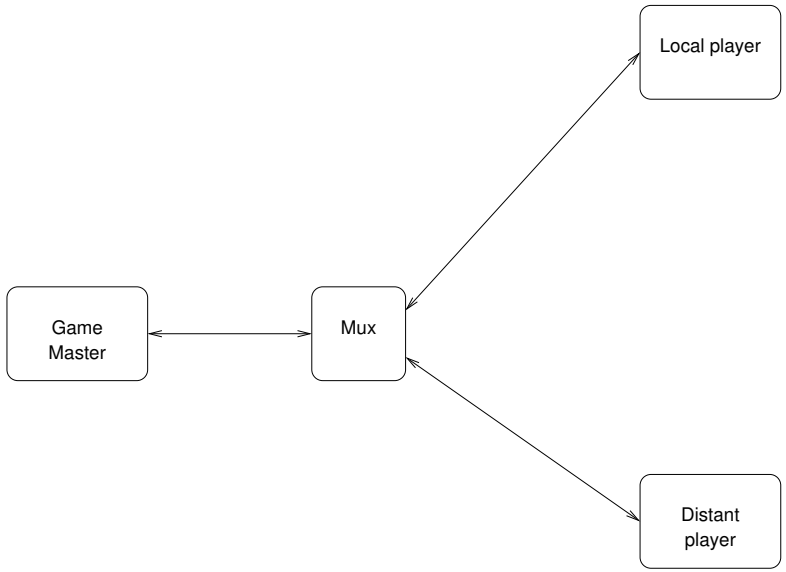






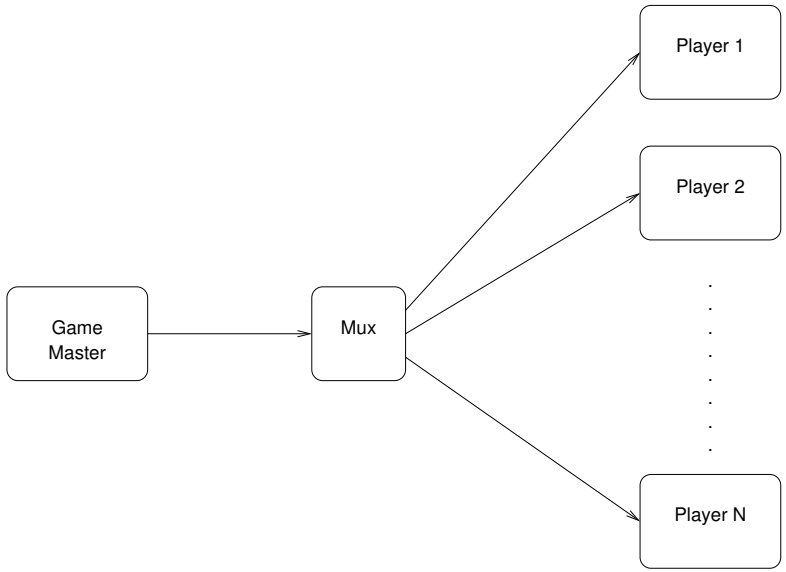
Comparison with Root Parallelism 1/3

Start from a multiplexer to meet deadlines with network lag.



Comparison with Root Parallelism 1/3

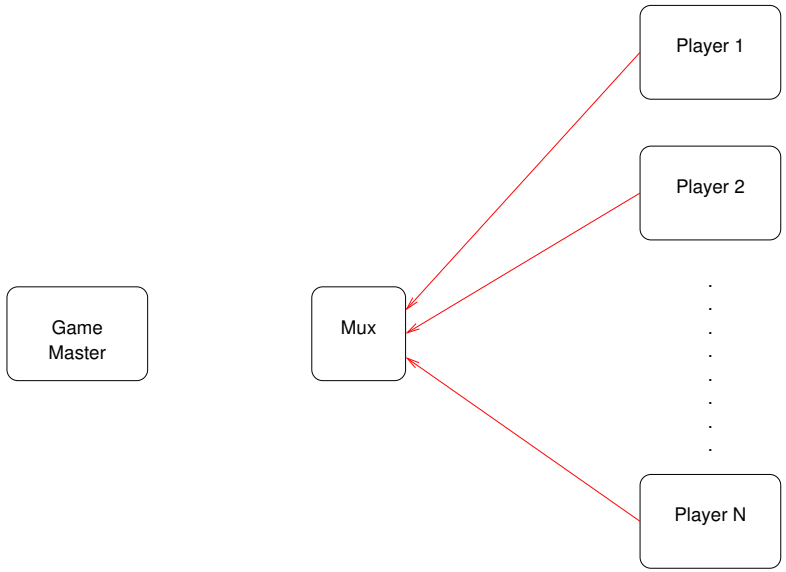
Start from a multiplexer to meet deadlines with network lag.



Send every message from the Game Master to all the players.

Comparison with Root Parallelism 1/3

Start from a multiplexer to meet deadlines with network lag.



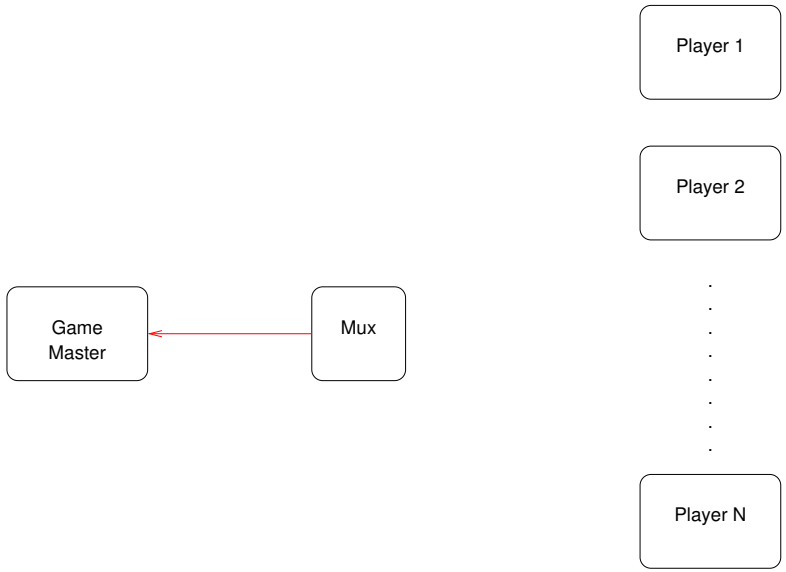
Send every message from the Game Master to all the players.

Players explore the situation as if standalone.

Players add move qualifications to their answer.

Comparison with Root Parallelism 1/3

Start from a multiplexer to meet deadlines with network lag.



Send every message from the Game Master to all the players.

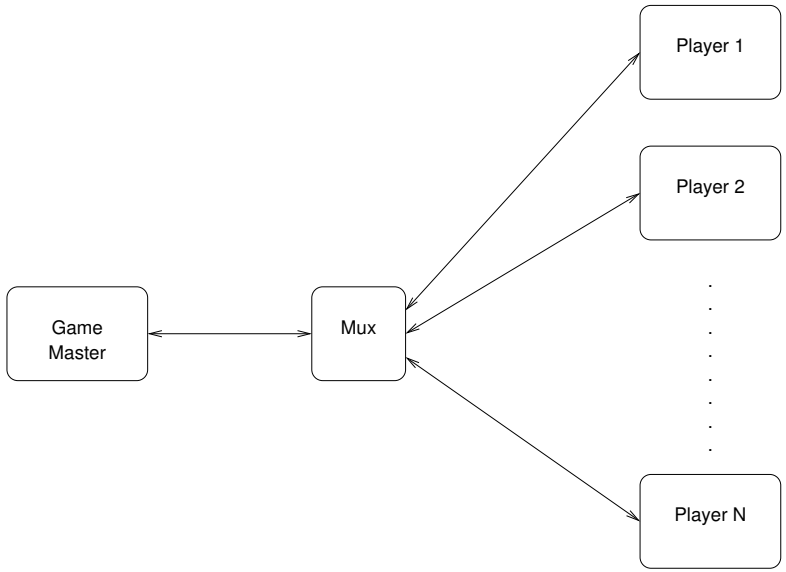
Players explore the situation as if standalone.

Players add move qualifications to their answer.

MUX aggregates answers.

Comparison with Root Parallelism 1/3

Start from a multiplexer to meet deadlines with network lag.



Send every message from the Game Master to all the players.

Players explore the situation as if standalone.

Players add move qualifications to their answer.

MUX aggregates answers.

- Ary in 2010 competition used 35 PCs in this way.

Comparison with Root Parallelism 2/3

+ The communication penalty is negligible (only one exchange per move played).
– The same nodes have to be expanded in every Player.
– Information gathered in one Player can not be used by other Players to orient search...
+ ... may introduce variety.

Comparaison with Root Parallelism 3/3

Same settings, except 10 seconds per move.

Tree parallelism

	Number of Subplayers				
game	1	2	4	8	16
Breakthrough	18	40	58	68	77
Connect 4	26	38	38	42	50
Othello	41	68	67	81	96
Pawn whopping	43	36	51	54	59
Pentago	38	40	55	73	87
Skirmish	73	76	76	78	79

Root parallelism

	Number of Subplayers				
game	1	2	4	8	16
Breakthrough	44	65	60	67	65
Connect 4	28	44	63	66	75
Othello	59	60	72	84	83
Pawn whopping	44	45	43	46	35
Pentago	35	54	68	64	68
Skirmish	71	71	74	76	71

Comparaison with Root Parallelism 3/3

Same settings, except 10 seconds per move.

Tree parallelism

	Number of Subplayers				
game	1	2	4	8	16
Breakthrough	18	40	58	68	77
Connect 4	26	38	38	42	50
Othello	41	68	67	81	96
Pawn whopping	43	36	51	54	59
Pentago	38	40	55	73	87
Skirmish	73	76	76	78	79

Root parallelism

	Number of Subplayers				
game	1	2	4	8	16
Breakthrough	44	65	60	67	65
Connect 4	28	44	63	66	75
Othello	59	60	72	84	83
Pawn whopping	44	45	43	46	35
Pentago	35	54	68	64	68
Skirmish	71	71	74	76	71

Tree // is globally better that Root //.

Comparaison with Root Parallelism 3/3

Same settings, except 10 seconds per move.

Tree parallelism

	Number of Subplayers				
game	1	2	4	8	16
Breakthrough	18	40	58	68	77
Connect 4	26	38	38	42	50
Othello	41	68	67	81	96
Pawn whopping	43	36	51	54	59
Pentago	38	40	55	73	87
Skirmish	73	76	76	78	79

Root parallelism

	Number of Subplayers				
game	1	2	4	8	16
Breakthrough	44	65	60	67	65
Connect 4	28	44	63	66	75
Othello	59	60	72	84	83
Pawn whopping	44	45	43	46	35
Pentago	35	54	68	64	68
Skirmish	71	71	74	76	71

Tree // is globally better that Root //.

Root // better than Tree // for small number of players.

Comparaison with Root Parallelism 3/3

Same settings, except 10 seconds per move.

Tree parallelism

	Number of Subplayers				
game	1	2	4	8	16
Breakthrough	18	40	58	68	77
Connect 4	26	38	38	42	50
Othello	41	68	67	81	96
Pawn whopping	43	36	51	54	59
Pentago	38	40	55	73	87
Skirmish	73	76	76	78	79

Root parallelism

	Number of Subplayers				
game	1	2	4	8	16
Breakthrough	44	65	60	67	65
Connect 4	28	44	63	66	75
Othello	59	60	72	84	83
Pawn whopping	44	45	43	46	35
Pentago	35	54	68	64	68
Skirmish	71	71	74	76	71

Tree // is globally better that Root //.

Root // better than Tree // for small number of players.

Tree // scales better Root // (until 16).

Comparaison with Root Parallelism 3/3

Same settings, except 10 seconds per move.

Tree parallelism

	Number of Subplayers				
game	1	2	4	8	16
Breakthrough	18	40	58	68	77
Connect 4	26	38	38	42	50
Othello	41	68	67	81	96
Pawn whopping	43	36	51	54	59
Pentago	38	40	55	73	87
Skirmish	73	76	76	78	79

Root parallelism

	Number of Subplayers				
game	1	2	4	8	16
Breakthrough	44	65	60	67	65
Connect 4	28	44	63	66	75
Othello	59	60	72	84	83
Pawn whopping	44	45	43	46	35
Pentago	35	54	68	64	68
Skirmish	71	71	74	76	71

Tree // is globally better that Root //.

Root // better than Tree // for small number of players.

Tree // scales better Root // (until 16).

Root // does not work at all for *Skirmish* and *Pawn Whopping*.

Comparaison with Root Parallelism 3/3

Same settings, except 10 seconds per move.

Tree parallelism

	Number of Subplayers				
game	1	2	4	8	16
Breakthrough	18	40	58	68	77
Connect 4	26	38	38	42	50
Othello	41	68	67	81	96
Pawn whopping	43	36	51	54	59
Pentago	38	40	55	73	87
Skirmish	73	76	76	78	79

Root parallelism

	Number of Subplayers				
game	1	2	4	8	16
Breakthrough	44	65	60	67	65
Connect 4	28	44	63	66	75
Othello	59	60	72	84	83
Pawn whopping	44	45	43	46	35
Pentago	35	54	68	64	68
Skirmish	71	71	74	76	71

Tree // is globally better that Root //.

Root // better than Tree // for small number of players.

Tree // scales better Root // (until 16).

Root // does not work at all for *Skirmish* and *Pawn Whopping*.

Root // better than Tree // for *Connect 4*.

Conclusion

Tree Parallelism :

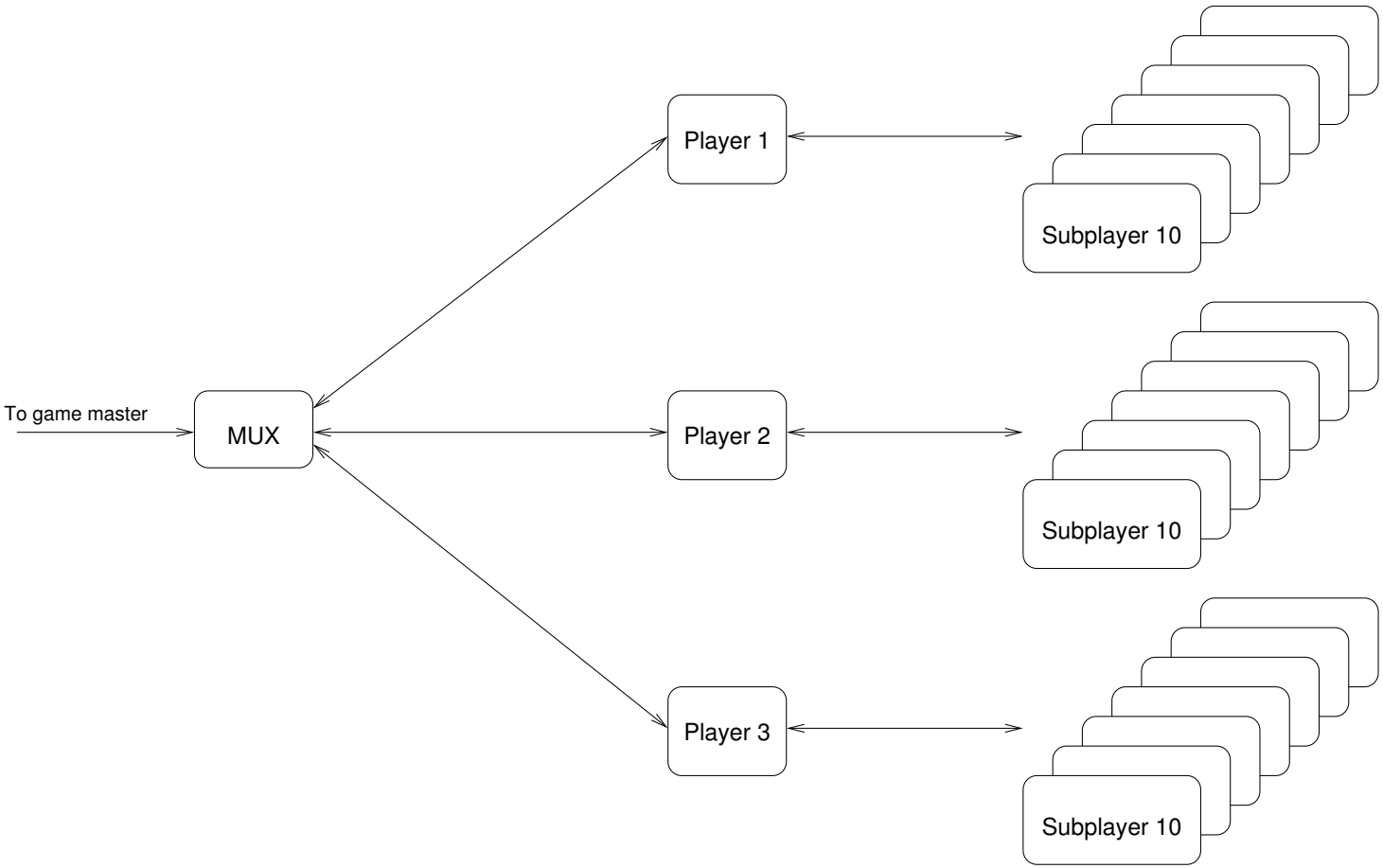
- brings a benefit with 4-16 Subplayers.
- penalty when number of Subplayers is small.
- not useful when number of Subplayer reaches 16.

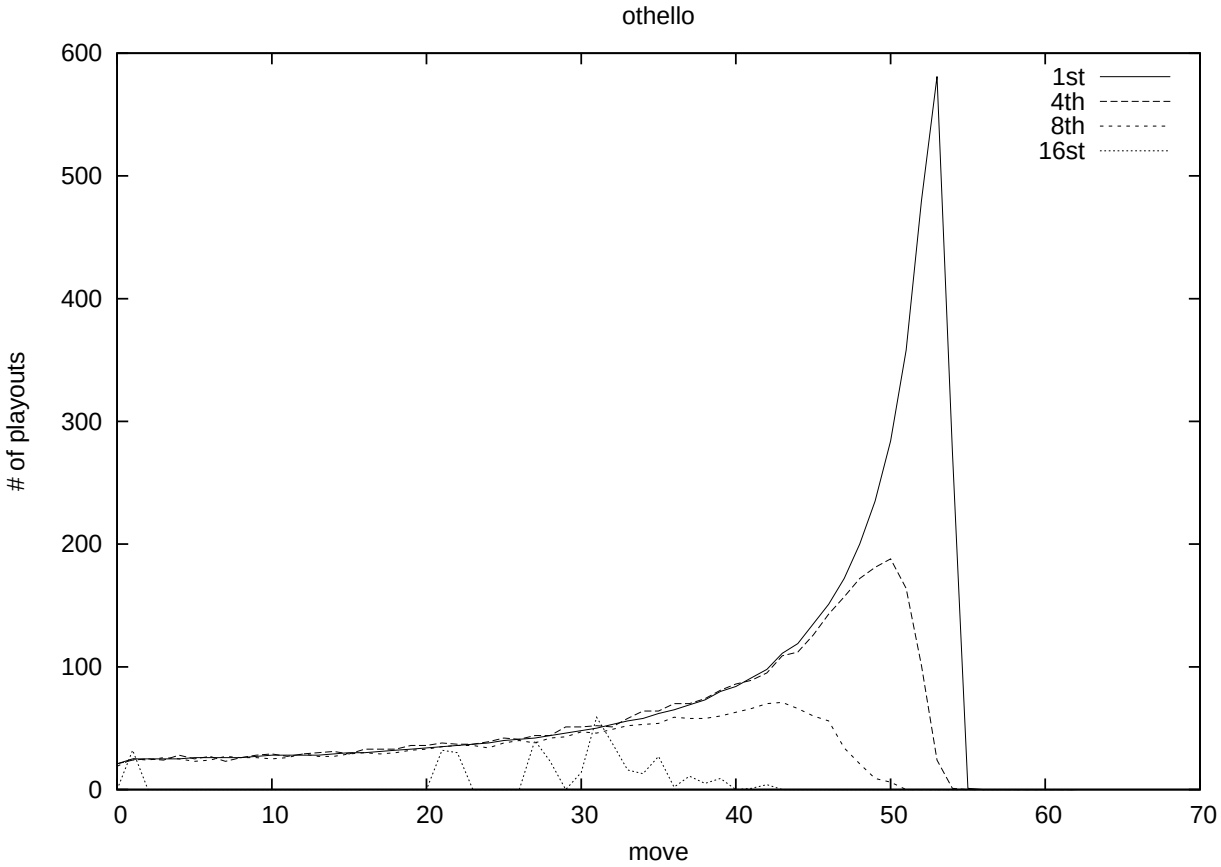
Will try to combine Root Parallelism and Tree Parallelism.

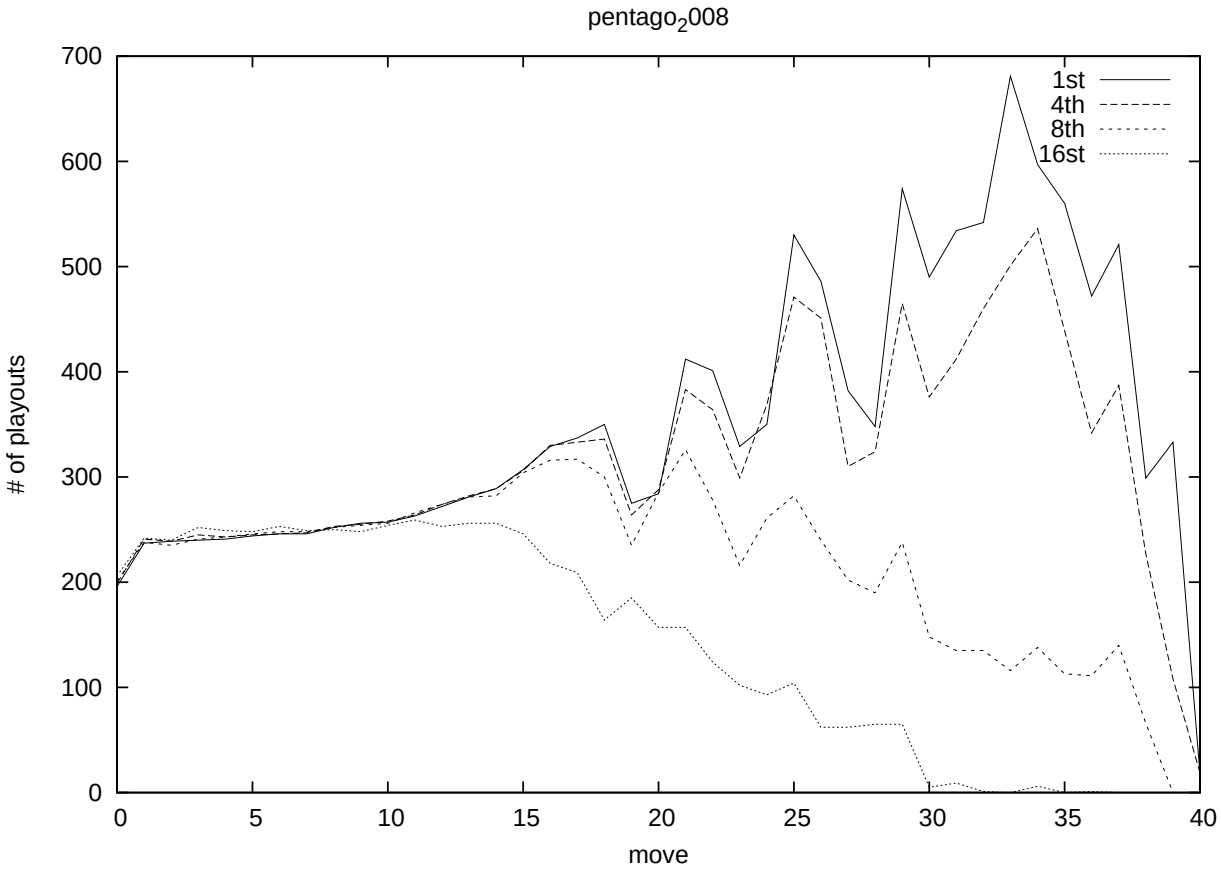
To do: full Tree Parallelism (Fuego's way?).

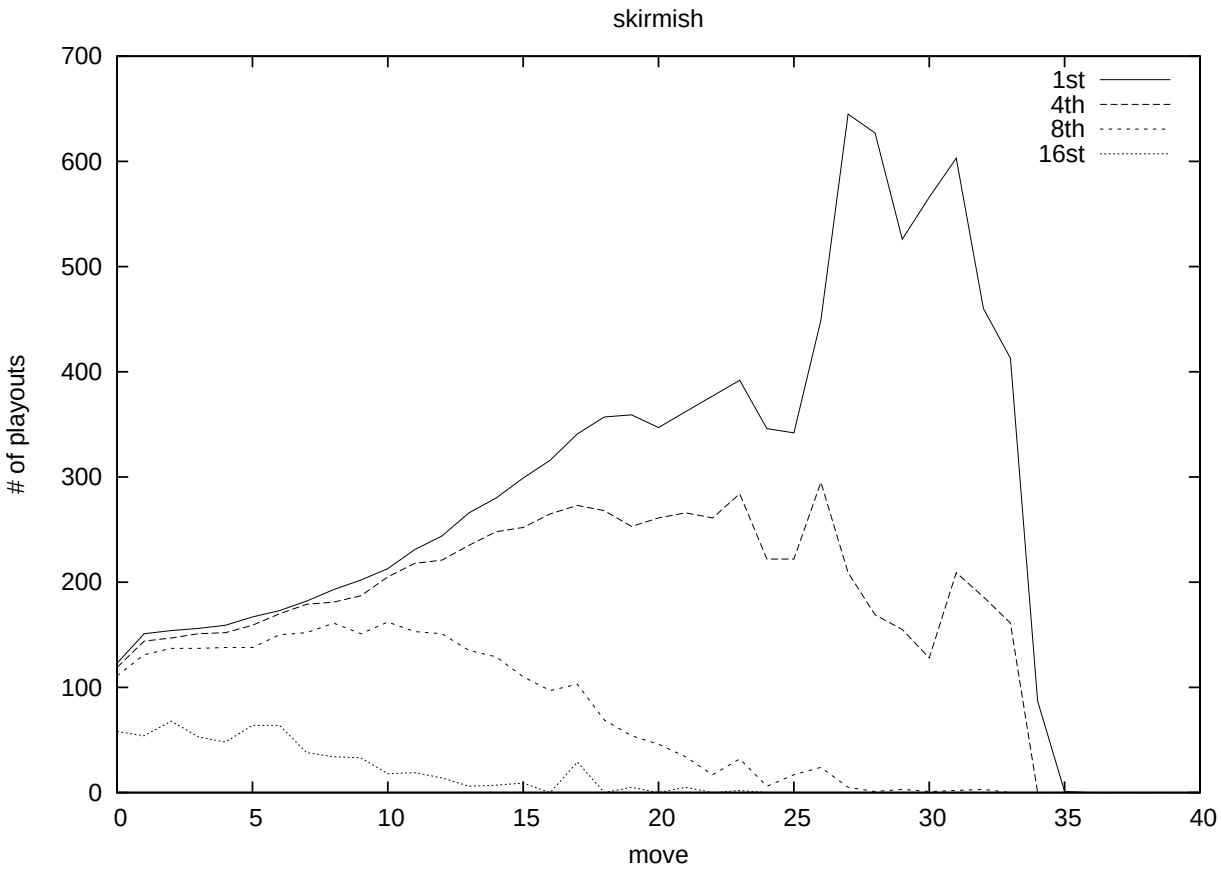
Competition 2011 Configuration

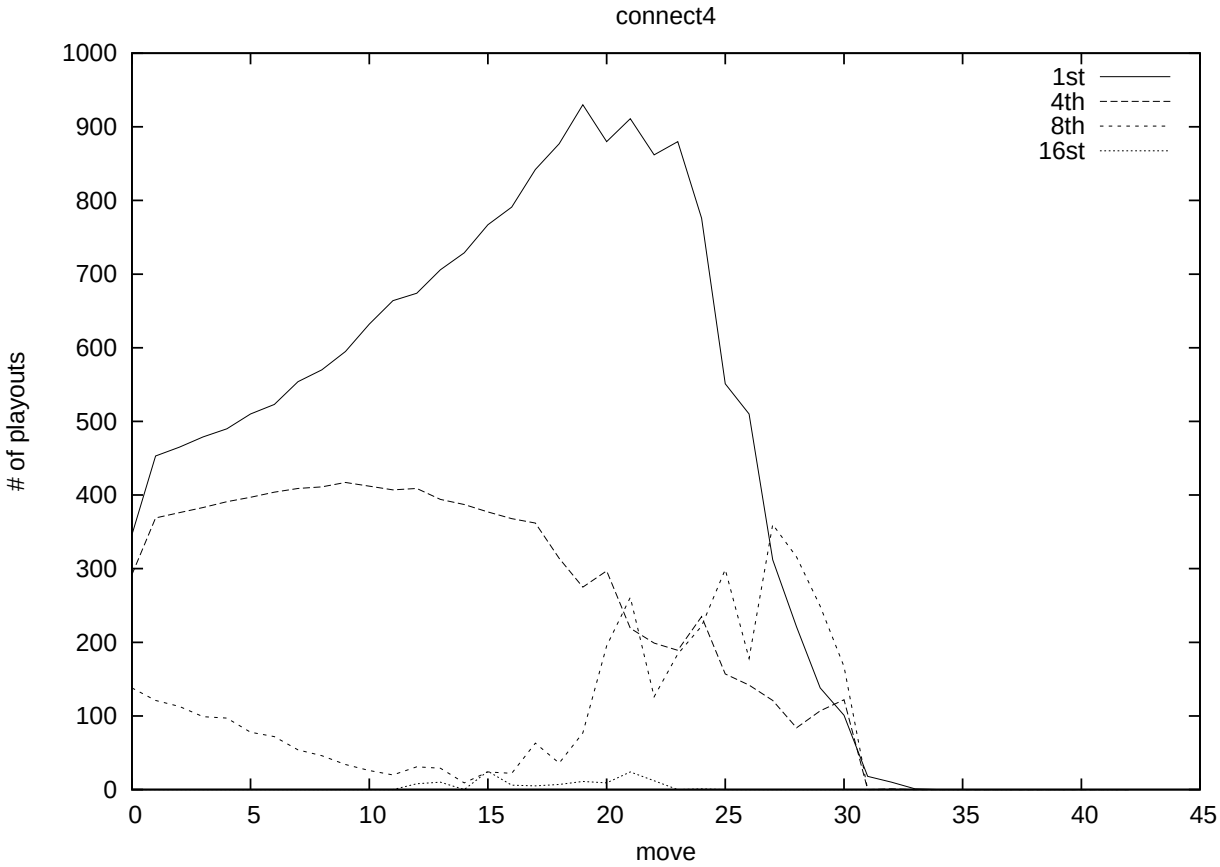
Groups of Tree Parallel Players, merged via Root Parallelisn (hopefully).




othello
# of playouts
move
1st
4th
8th
16st
0
100
200
300
400
500
600
0
10
20
30
40
50
60
70


pentago$_2$008
# of playouts
move
1st
4th
8th
16st
0
100
200
300
400
500
600
700
0
5
10
15
20
25
30
35
40


skirmish
# of playouts
move
1st
4th
8th
16st
0
100
200
300
400
500
600
700
0
5
10
15
20
25
30
35
40


connect4
# of playouts
move
1st
4th
8th
16st
0
100
200
300
400
500
600
700
800
900
1000
0
5
10
15
20
25
30
35
40
45

Root parallelism vs. Tree parallelism

Tree parallelism

game	Number of Subplayers 1	2	4	8	16
Breakthrough	18	40	58	68	77
Connect 4	26	38	38	42	50
Othello	41	68	67	81	96
Pawn whopping	43	36	51	54	59
Pentago	38	40	55	73	87
Skirmish	73	76	76	78	79

Root parallelism

game	Number of Subplayers 1	2	4	8	16
Breakthrough	44	65	60	67	65
Connect 4	28	44	63	66	75
Othello	59	60	72	84	83
Pawn whopping	44	45	43	46	35
Pentago	35	54	68	64	68
Skirmish	71	71	74	76	71