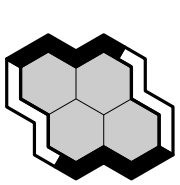
## mcts example

```
def monte_carlo_tree_search(self):
        if self.winning_move is not None:
            return self.winning_move

        end_time = time.time() + MCTS_TIME
        while time.time() < end_time:

            for child in self.root_node.children:  # winner found?
                if child.results == float('inf'):
                    self.root_node.show_data()
                    return child.move

            leaf = self.traverse_and_expand(self.root_node)  # traverse
            result = leaf.rollout()  # rollout
            leaf.backpropagate(result)  # backpropagate

        self.root_node.show_data()
        return self.get_best_move()
```

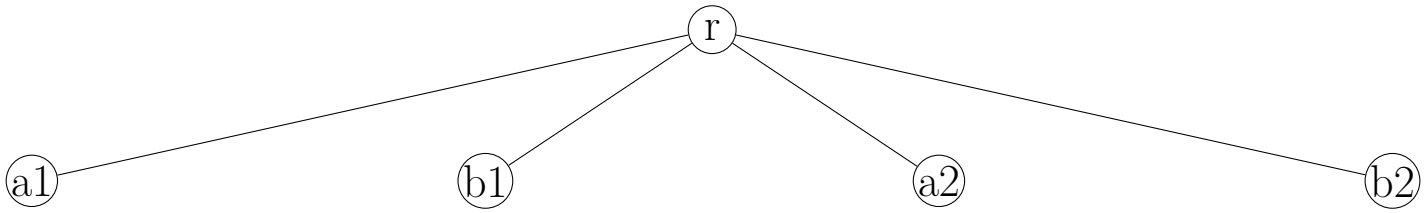
```
def traverse_and_expand(self, node: TreeNode0):
        while not node.is_leaf:
            node = self.best_uct(node)
        if len(node.moves) > 0 and node.sims > 0:
            node.expand_node()
            node = random.choice(node.children)
        return node
```



```
github repo mcts/main.py

= size 2          node names     5   6         a1  b1
= mcts x                           9  10           a2  b2

root node init * > 5, 6, 9, 10, done
```

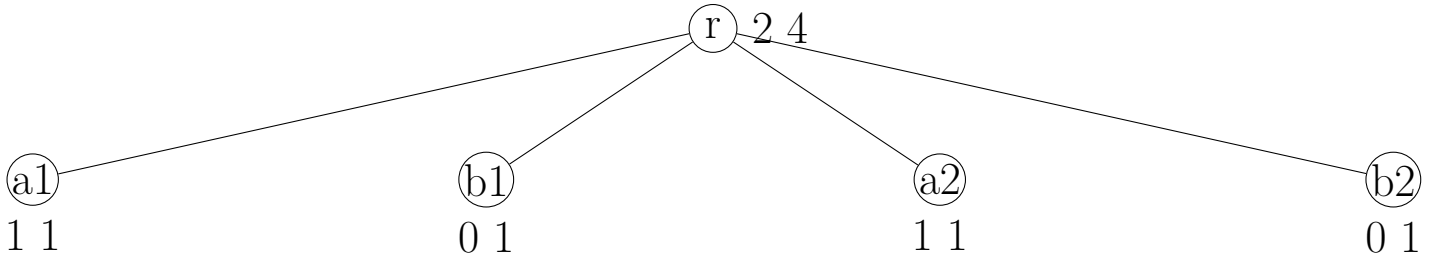
- **init tree, traverse**


- **expand**



```
trv_xpnd bu * 5 no-sims child
sim 1. * 5 roll 6 9 parent win

trv_xpnd bu * 1.0 6 no-sims child
sim 2. * 6 roll 9 5 10 parent loss

trv_xpnd bu * 1.2 .2 9 no-sims child
sim 3. * 9 roll 10 5 parent win

trv_xpnd bu * 1.3 .3 1.3 10 no-sims child
sim 4. * 10 roll 6 5 9 parent loss
```

- **traverse-expand, sims 1-4.**

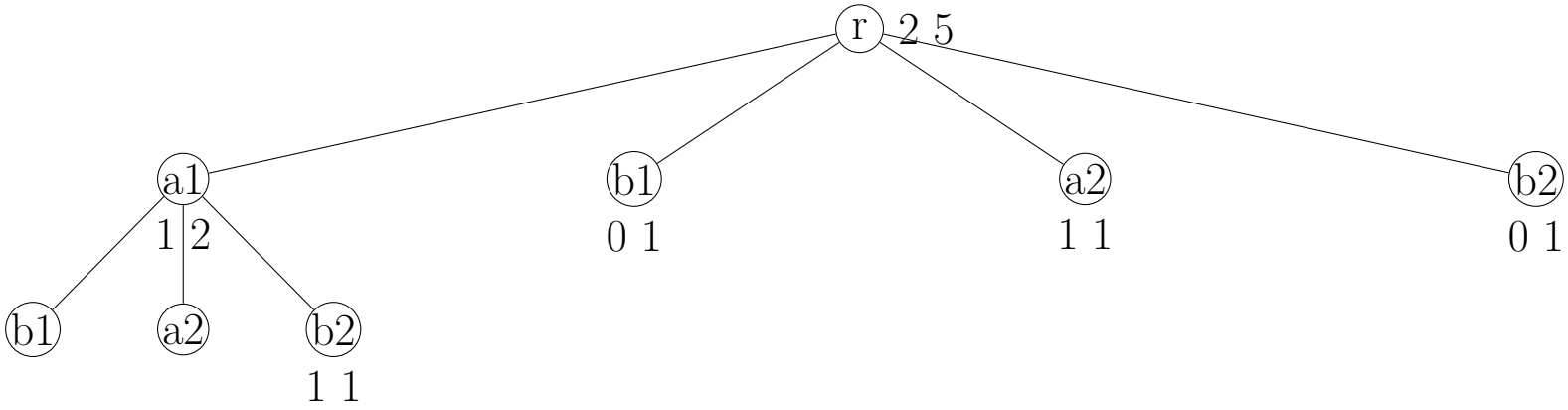



root node: win, visit(sim) counts: sum, over children

non-root: win, visit(sim) counts: for player-who-moved

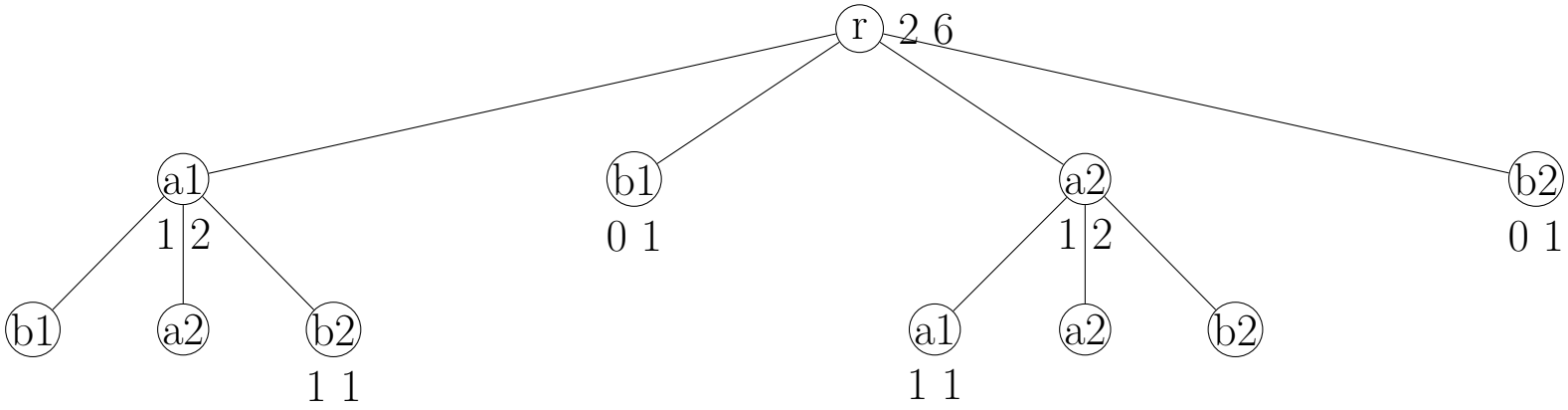
```
trv_xpnd bu * 1.4 .4 1.4 .4 5
    xpnd_nd * 5 > 6
    xpnd_nd * 5 > 9
    xpnd_nd * 5 > 10
  sim 5. * 5 10 roll 6 9 parent win
```

- **traverse-expand, sim 5.**



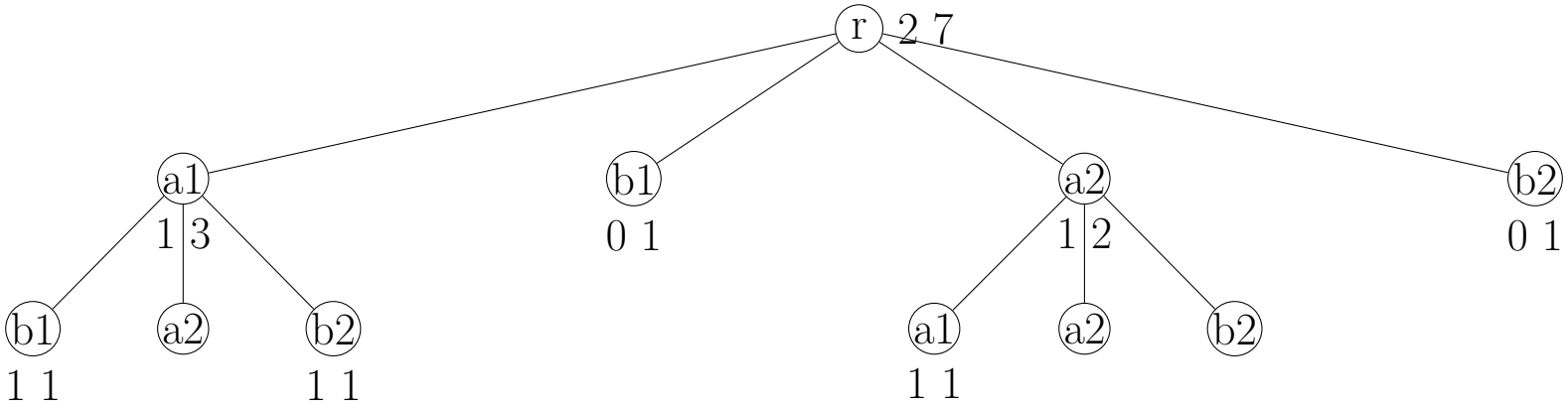
```
trv_xpnd bu * .8 .4 1.4 .4 9
   xpnd_nd * 9 > 5
   xpnd_nd * 9 > 6
   xpnd_nd * 9 > 10
 sim 6. * 9 5 roll 10 6 parent win
```

- **traverse-expand, sim 6.**



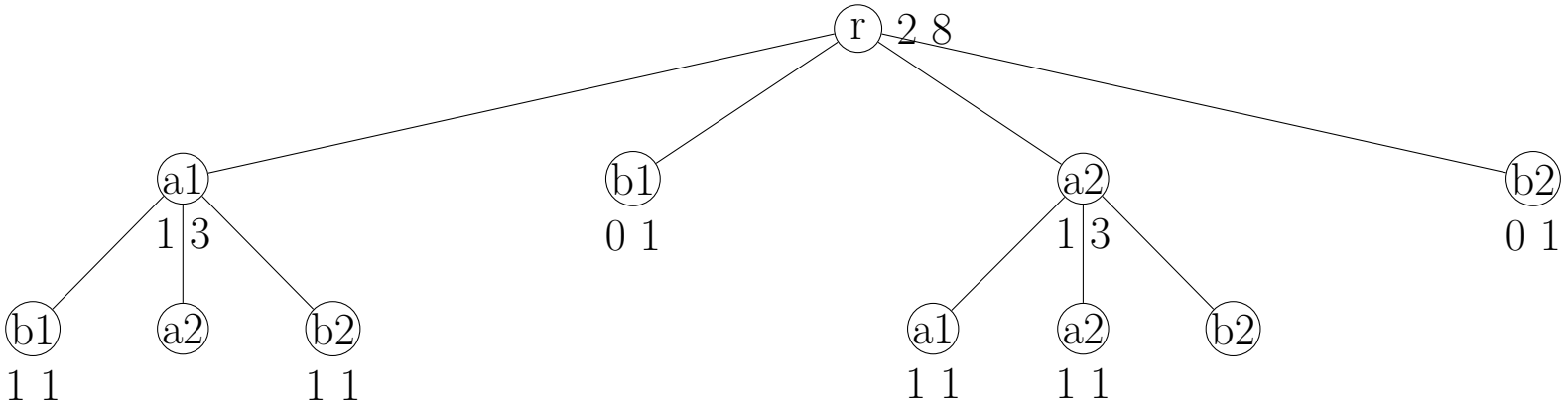
```
trv_xpnd bu * .8 .4 .8 .4 5 bu 5 6 no-sims child
  sim 7. * 5 6 roll 10 9 parent win
```

- **traverse-expand, sim 7.**



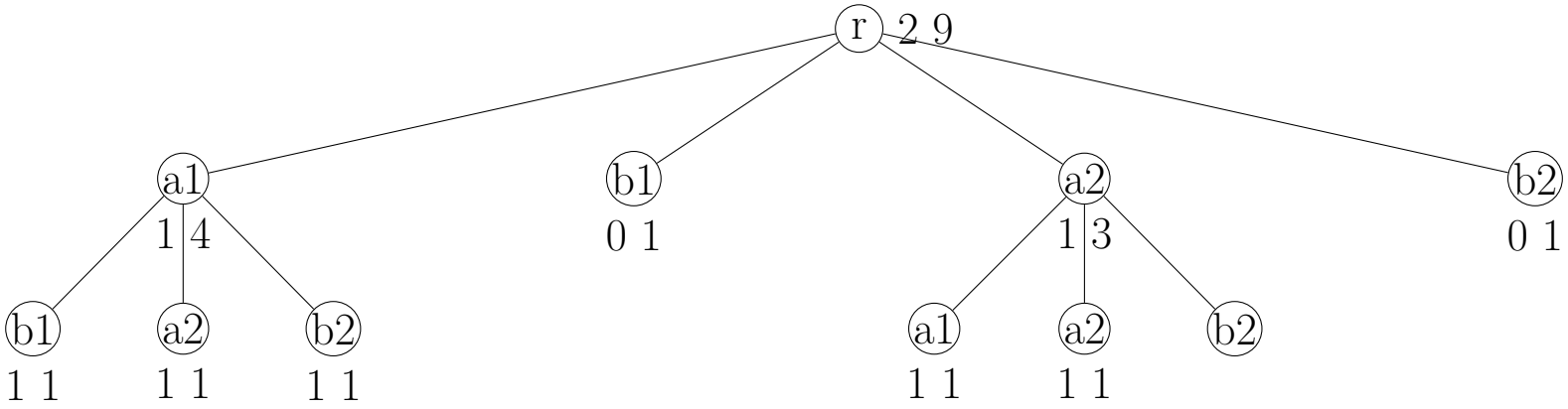
```
trv_xpnd bu * .6 .4 .8 .4 9 bu 9 1.4 6 no-sims child
  sim 8. * 9 6 roll 10 5 parent win
```

- **traverse-expand, sim 8.**



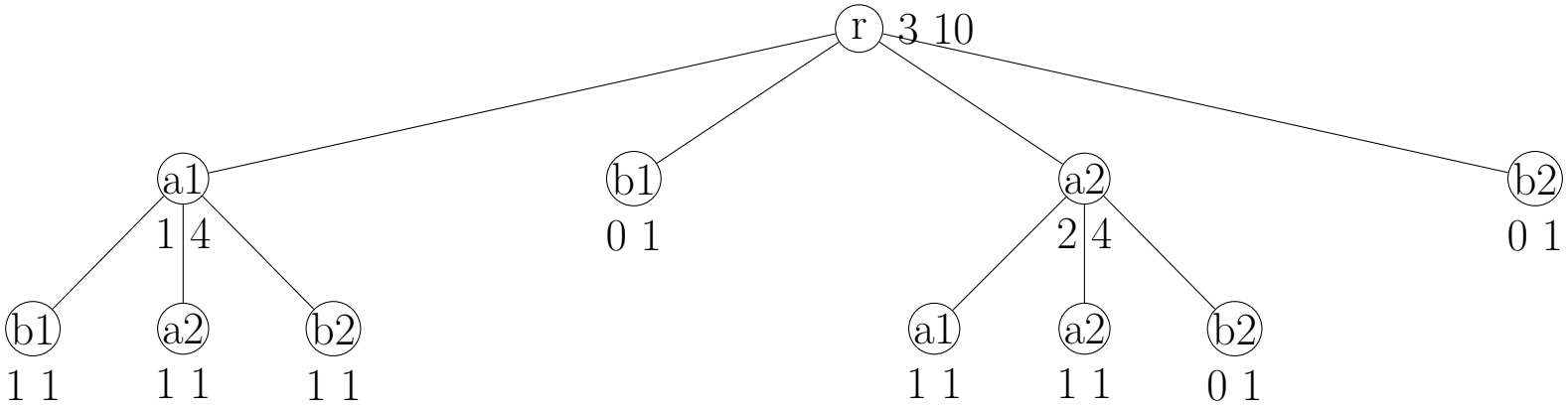
```
trv_xpnd bu * .6 .4 .6 .4 5 bu 5 1.4 9 no-sims child
  sim 9. * 5 9 roll 6 10 parent win
```

- **traverse-expand, sim 9.**



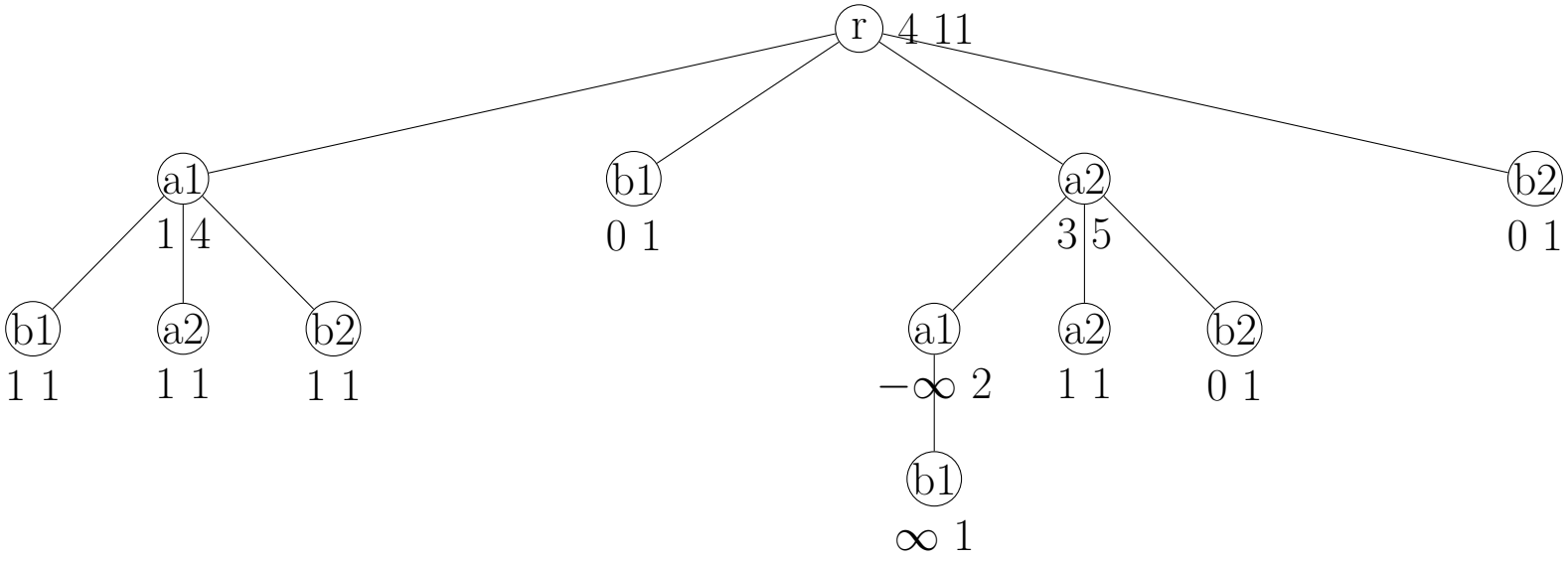
```
trv_xpnd bu * .5 .4 .6 .4 9 bu 9 1.4 1.4 10 no-sims child
  sim 10. * 9 10 roll 5 parent loss
```

- **traverse-expand, sim 10.**



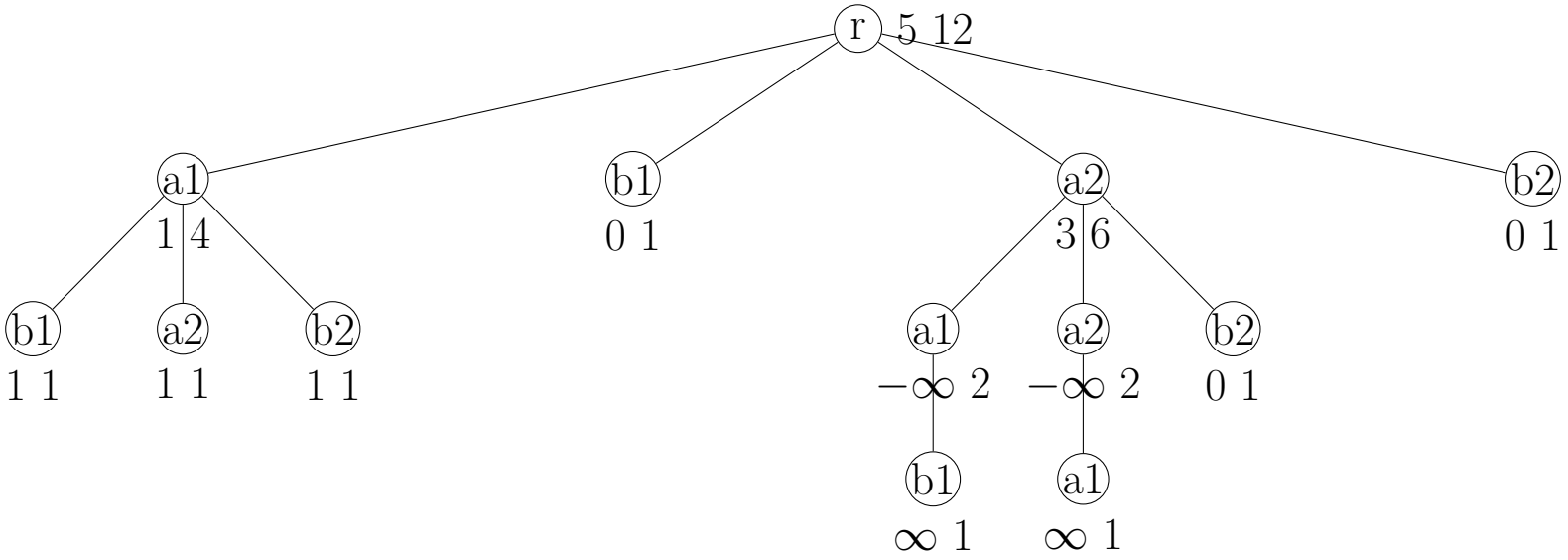
```
trv_xpnd bu * .5 .5 .7 .5 9 bu 9 1.5 1.5 .5 5
   xpnd_nd * 9 5 > 6
   sim 11. * 9 5 6 win, no more sibs
```

- **traverse-expand, sim 11.**



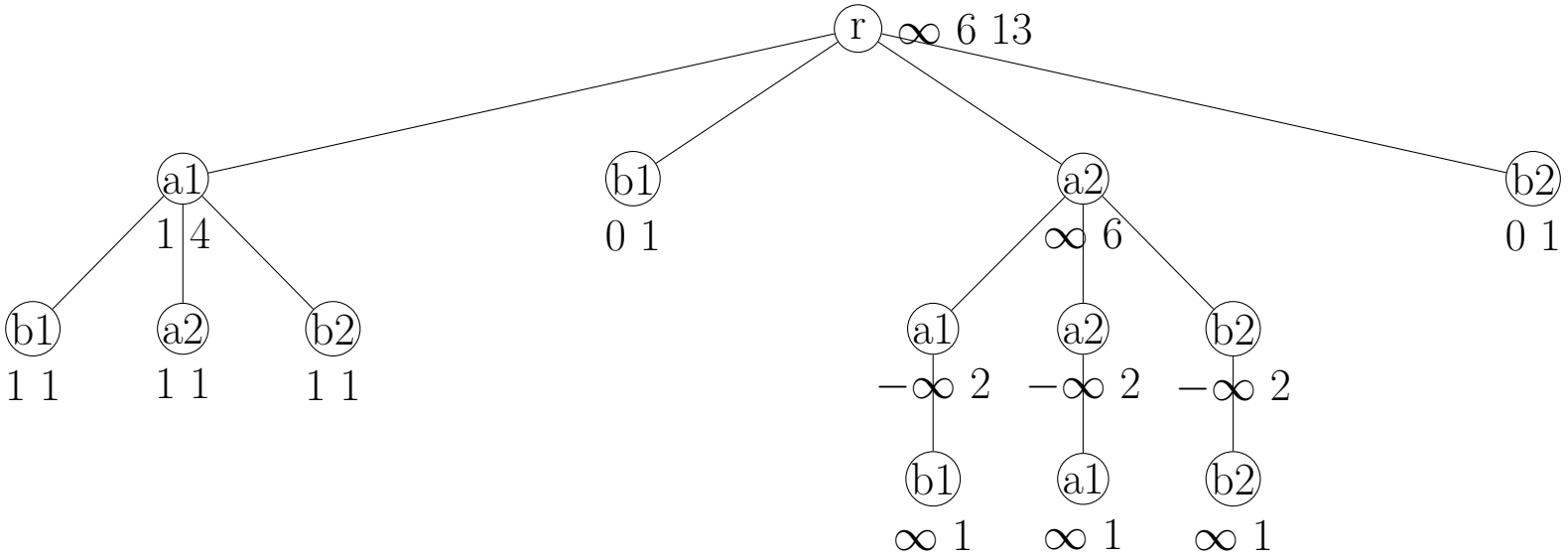
```
trv_xpnd bu * .5 .5 .8 .5 9 bu 9 -inf 1.5 .5 6
   xpnd_nd * 9 6 > 5
   sim 12. * 9 6 5 win, no more sibs
```

- **traverse-expand, sim 12.**



```
trv_xpnd bu * .5 .5 .9 .5 9 bu 9 -inf -inf .5 10
    xpnd_nd * 9 10 > 5
    sim 13. * 9 10 5 win, no more sibs
```

- **traverse-expand, sim 13.**



- **results after all simulations**

```
  move  sims  wins
    5     4     1
    6     1     0
    9     7   inf
   10     1     0
total    13   inf

    a b
  1  . . o
   2  x . o
       x x
```