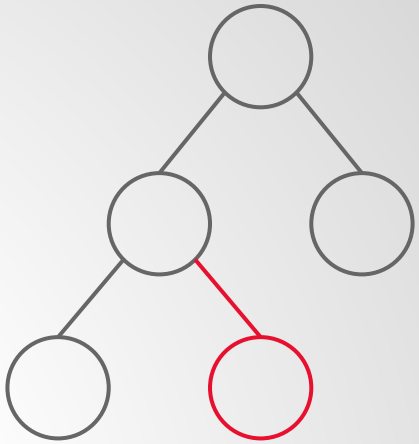


SimpleAI

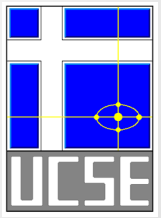
Artificial Intelligence Python lib

Juan Pedro Fisanotti

"fisa"

work with python, django, AI, ... at

teach web engineering and AI at



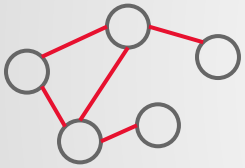
member (mail list, events, projects) of

SimpleAI

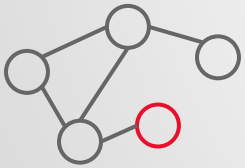
- implement algorithms from AIMA book
- alternative to aima-python
- focus on **simple API** and **docs**
- modern, "pythonic"
- tests!

Search problems

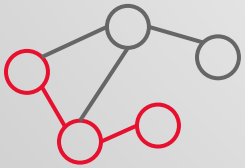
set of states



actions connect states

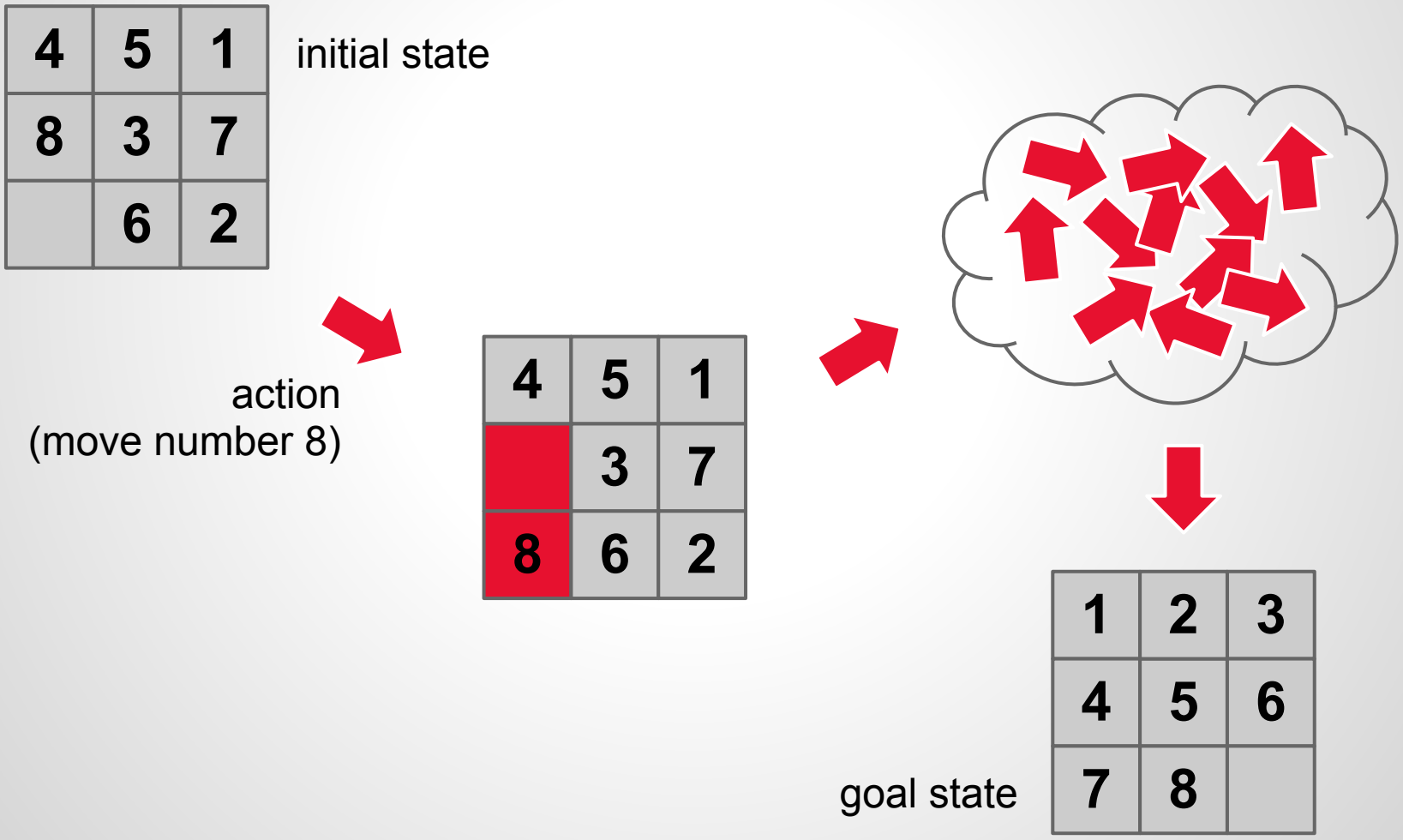


find goal states, or



find paths to goal states

Example: the 8-puzzle



Why the 8-puzzle?

- toy problem

- brute force is **really** bad

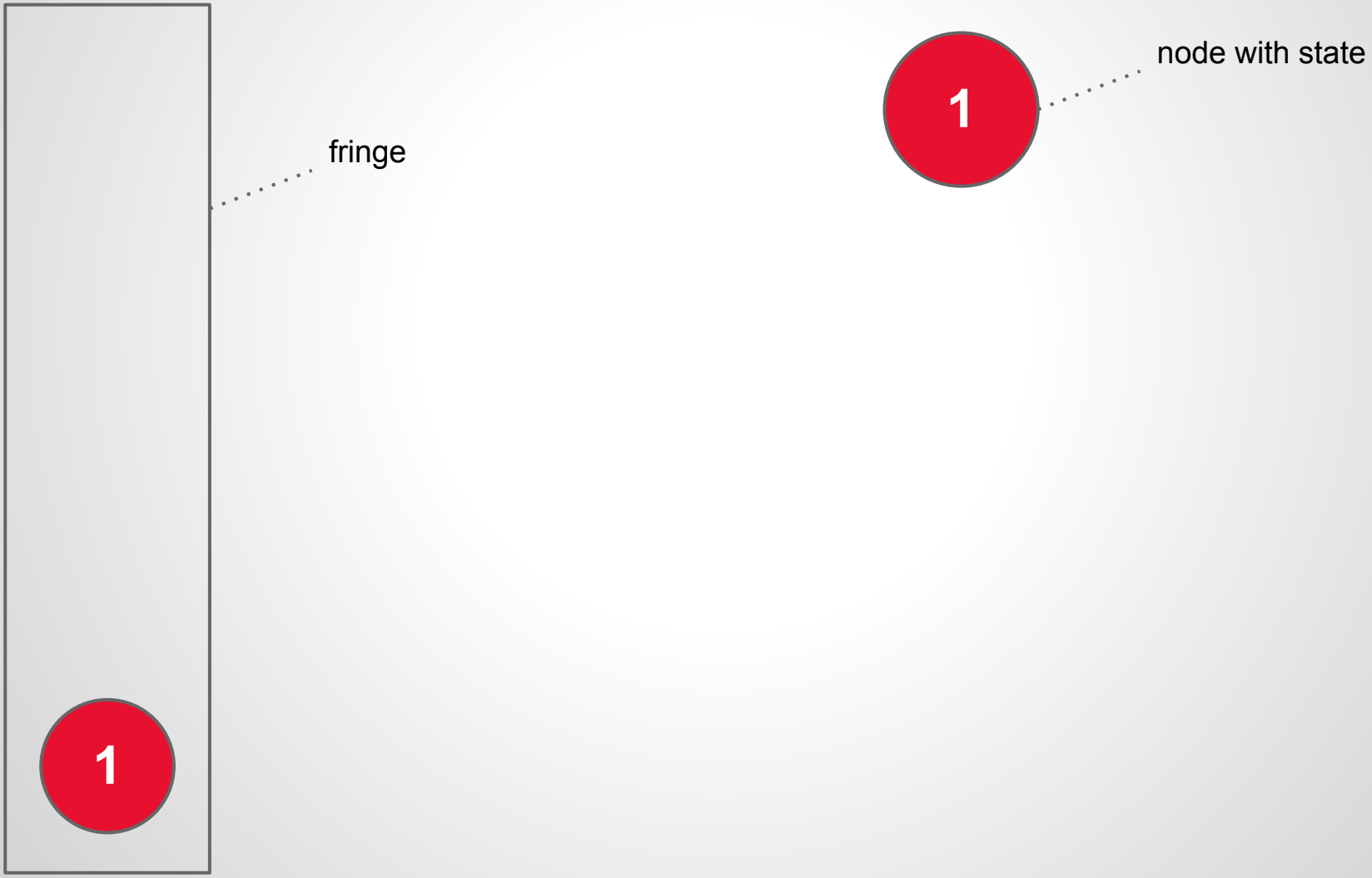
  (hope you have enough terabytes of RAM)

Traditional search algorithm

(basic notion)

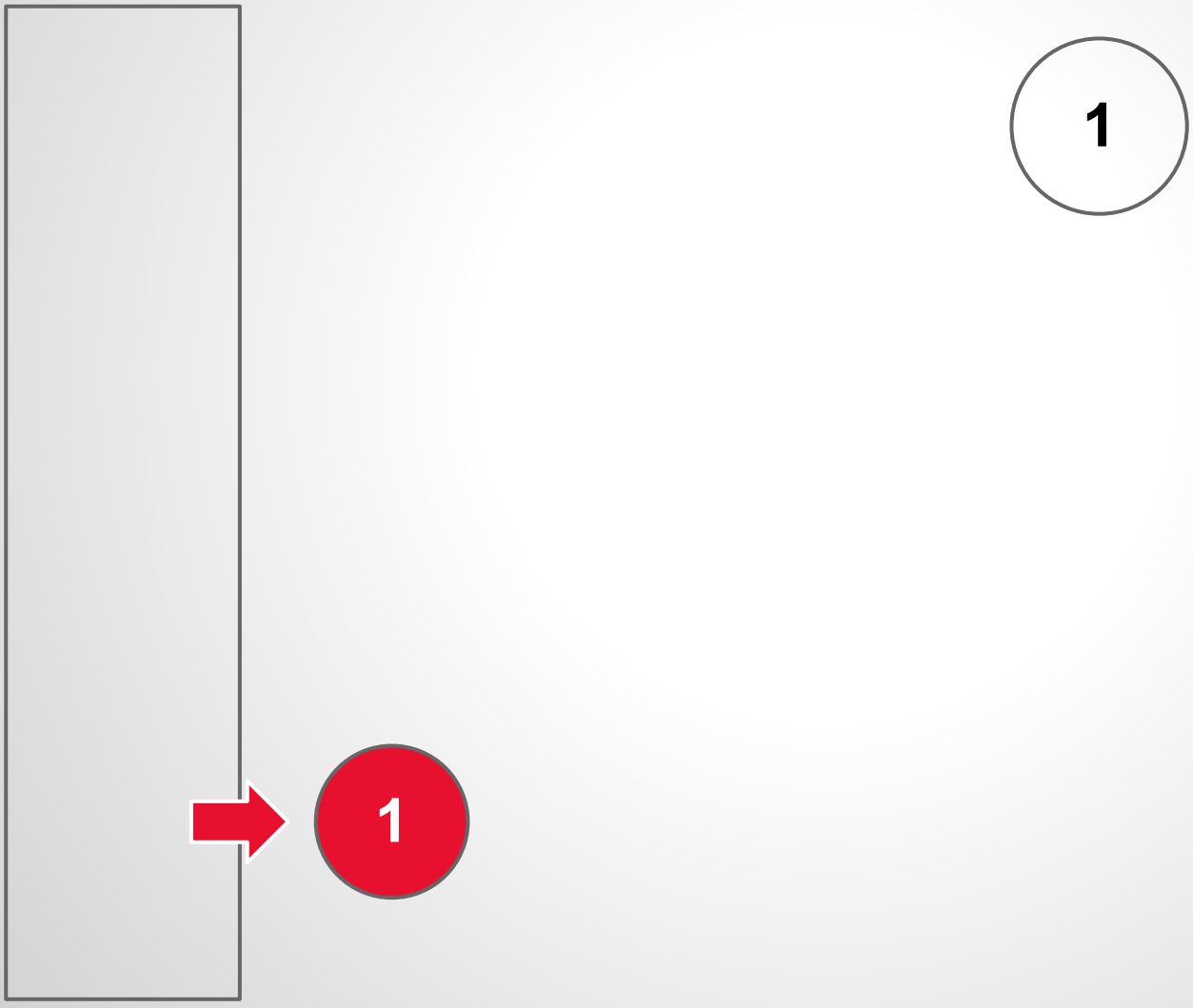
```
fringe = new bag for "pending" nodes
initial = new node with initial state
fringe.append(initial)

while fringe not empty:
    node = fringe.magick_pop()
    if node.state is goal:
        return node
    else:
        children = other nodes we can reach from node
        fringe.append(children)
```

Start with initial in fringe

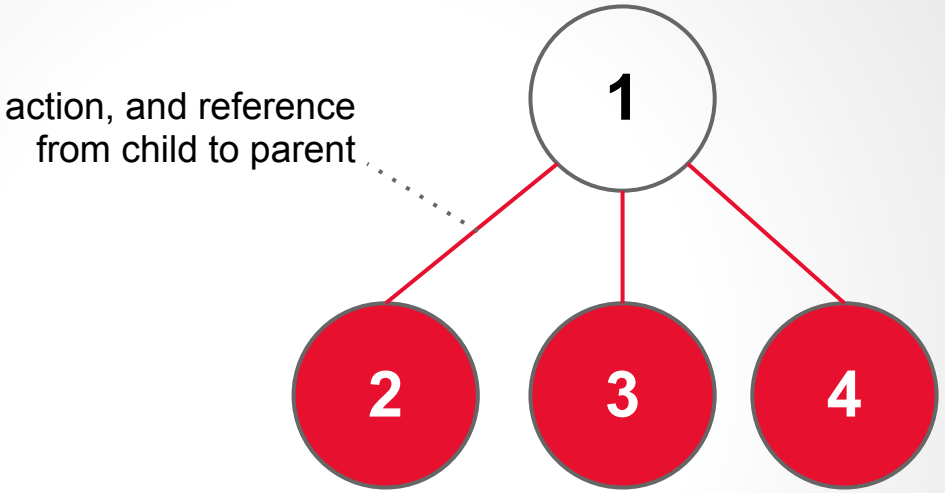


Loop: new iteration

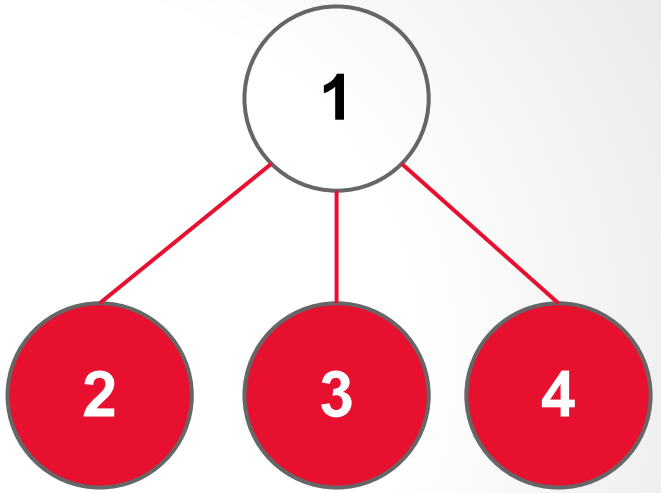
Loop: pop node from fringe



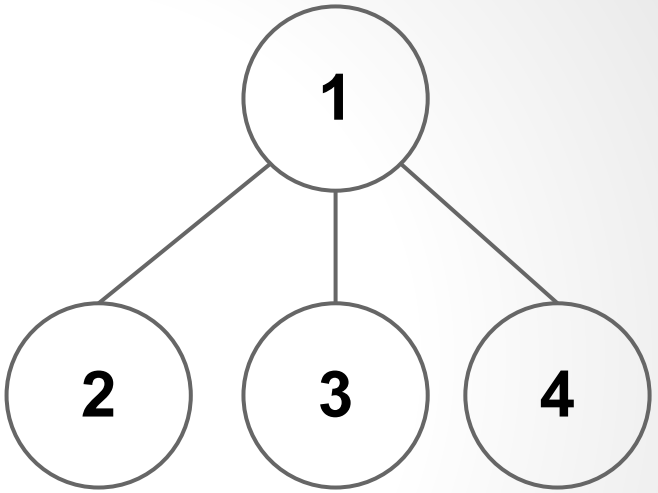
Loop: not goal, generate children



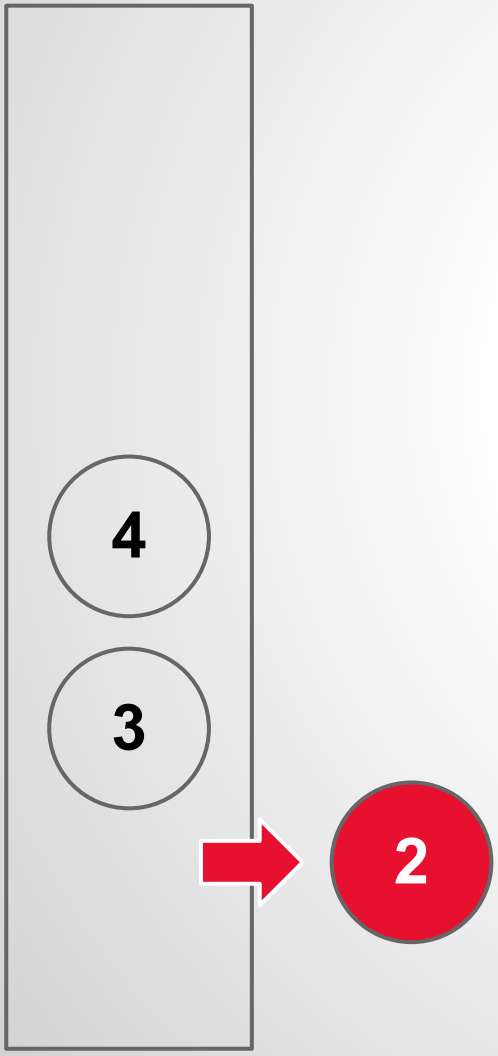
Loop: append children to fringe

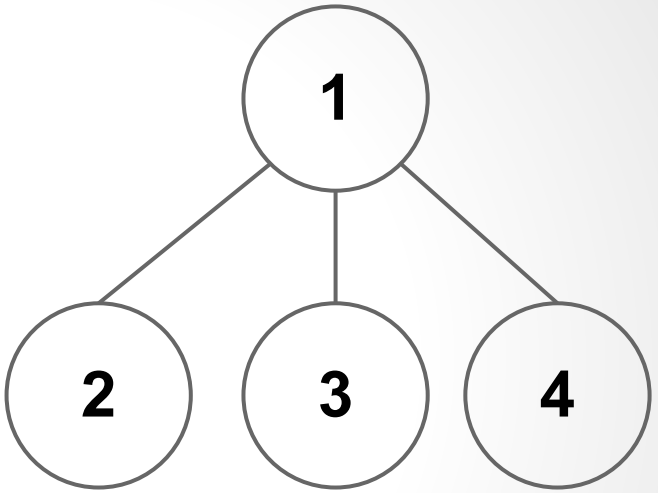


Loop: new iteration

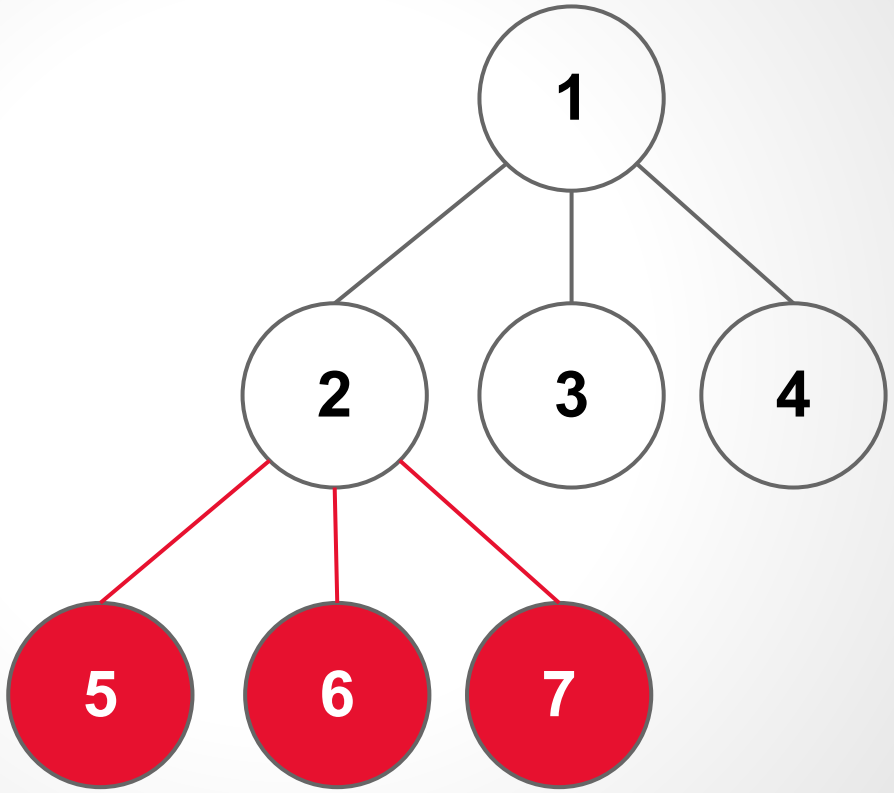


Loop: pop node from fringe

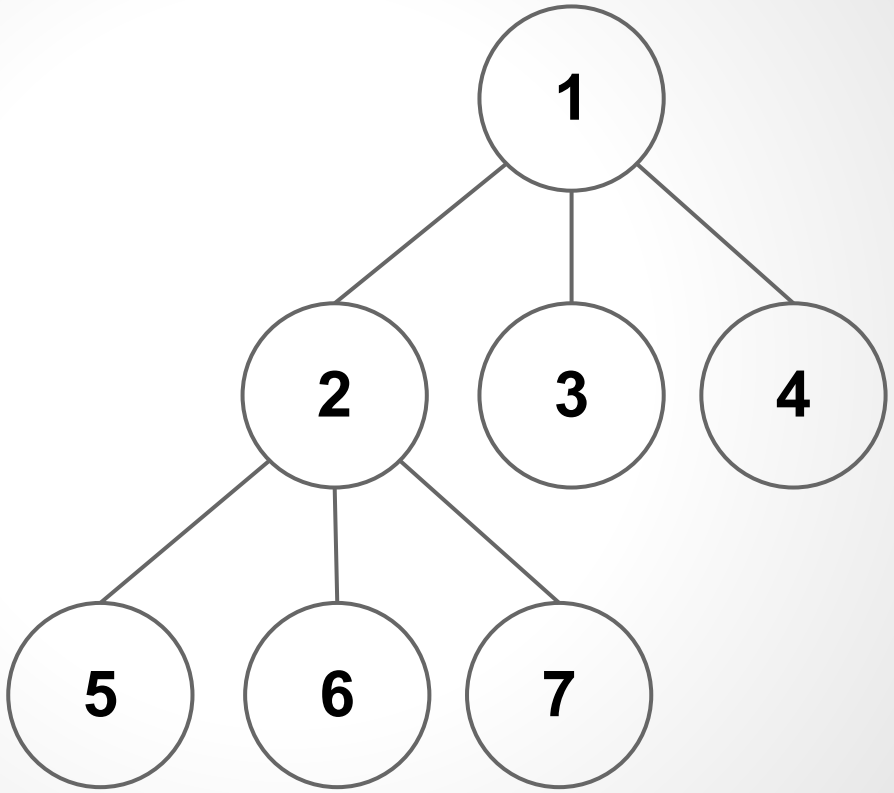




Loop: not goal, append children

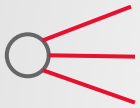


Loop: new iteration

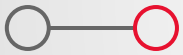


Implementation using SimpleAI

class SearchProblem:



actions(state)



result(state, action)

cost(state1, action, state2)

heuristic(state)

is_goal(state)

Conclusions

- easy to use
- broad field of application
- graphical debuggers are really useful

More info

- AIMA book:
  http://aima.cs.berkeley.edu/
- SimpleAI docs:
  http://simpleai.readthedocs.org
- SimpleAI discussion:
  http://groups.google.com/group/simpleai

- Can I help?
  fisadev@gmail.com / @fisadev

?

questions