RUTGERS

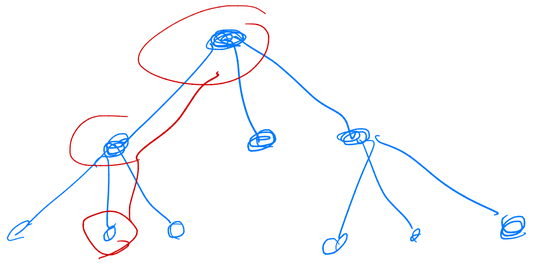
THE STATE UNIVERSITY
OF NEW JERSEY

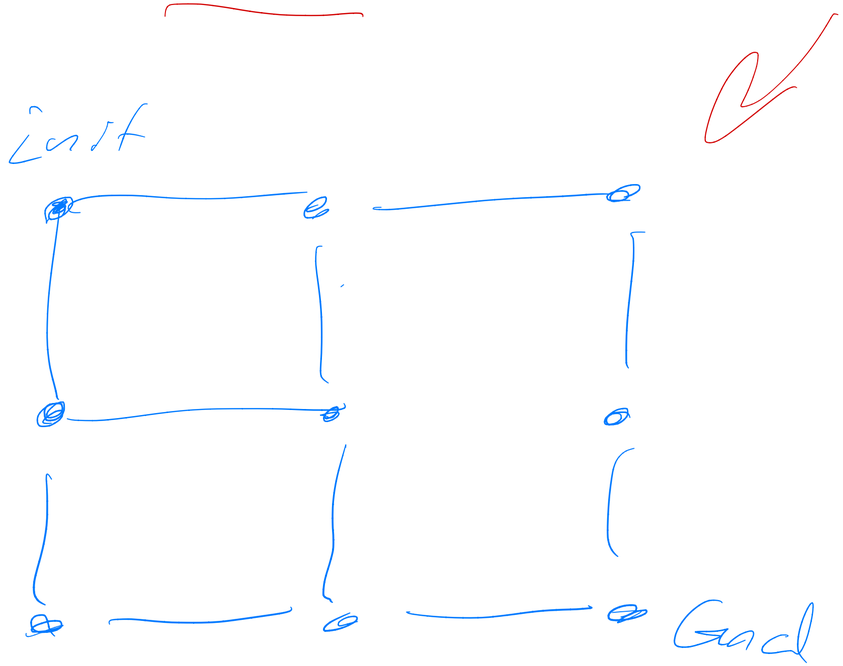
Problem Solving by Searching

Edgar Granados

Search Algorithms

- **Problem**: (States, Initial State, Goal State, Actions, Transition Model, Cost)
- **Solution**: Sequence of actions from *initial state* to *goal state*





Search Algorithms

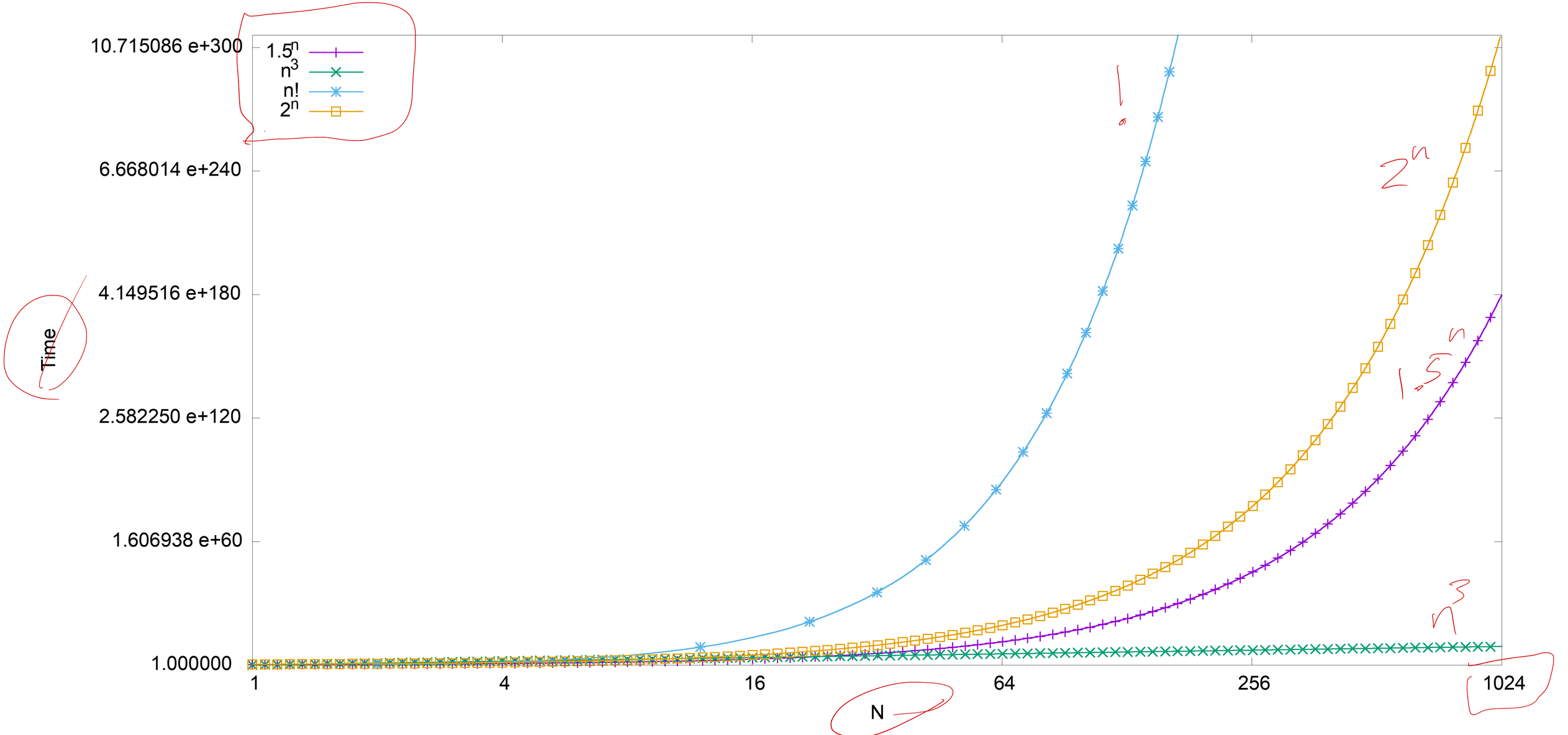
Comparison Criteria

- **Completeness**: find a solution when one exists
- **Optimality**: find the optimal solution
- **Time complexity**: Time to find a solution
- **Space complexity**: Memory used for the search

Parameters:

- **b** - max branching factor
- **d** - depth of the shallowest goal
- **m** - max length of any path

Search Algorithms: Complexity

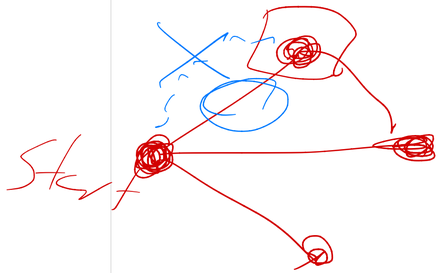


Generic Tree Search

```
1  def generic-tree-search(problem):
2   frontier <- problem.start_state
3   loop do:
4    if frontier.empty():
5     FAIL
6    node <- frontier.next()
7    if goal-function(node):
8     return solution
9    frontier += expand(node)
10
```

Generic Graph Search

```
1  def generic-graph-search(problem):
2   frontier <- problem.start_state
3   closed <- []
4   loop do:
5    if frontier.empty():
6     FAIL
7    node <- frontier.next()
8    if goal-function(node):
9     return solution
10   closed += node
11   candidates <- expand(node)
12   for c in candidates:
13    if c not in closed and c not in frontier:
14     frontier += c
```



Search Algorithms

- Breadth-first search
- Uniform-cost search
- Depth-first search
- Depth-limited search
- Iterative deepening depth-first search
- Bidirectional search

RUTGERS

THE STATE UNIVERSITY
OF NEW JERSEY

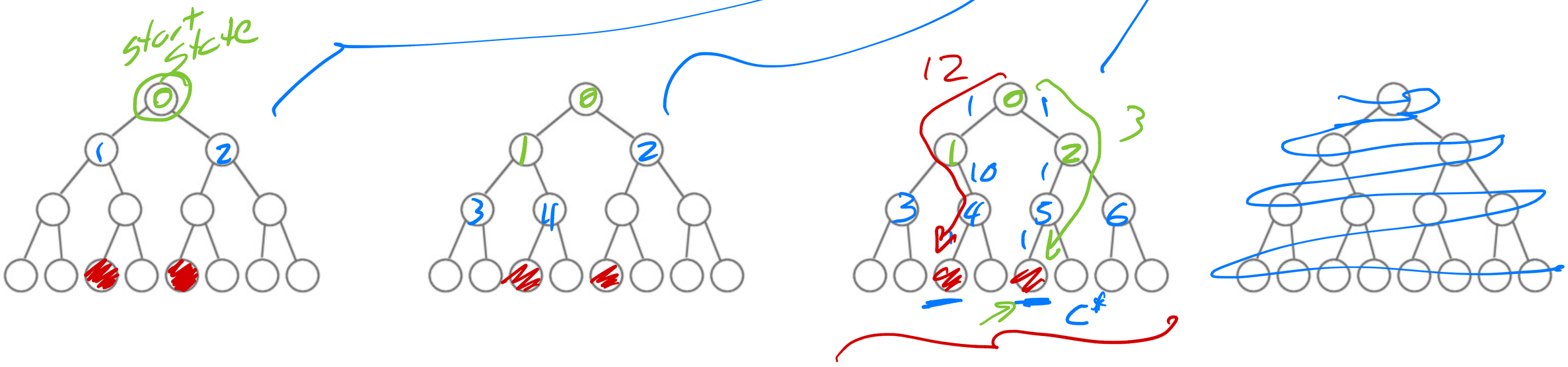
Problem Solving by Searching

BFS & Uniform search

Edgar Granados

Breadth-First Search

- Expand nodes by level



Breadth-First Search

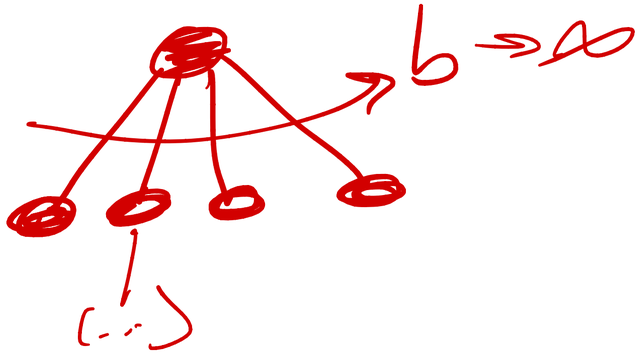
BFS

```
1  def generic-graph-search(problem):
2   frontier <- problem.start_state
3   closed <- []
4   loop do:
5    if frontier.empty():
6     FAIL
7    node <- frontier.next()
8    if goal-function(node):
9     return solution
10   closed += node
11   candidates <- expand(node)
12   for c in candidates:
13    if c not in closed and c not in frontier:
14     frontier += c
```

FIFO

dequeue

Breadth-First Search

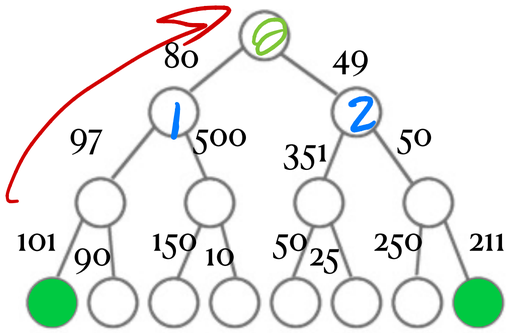


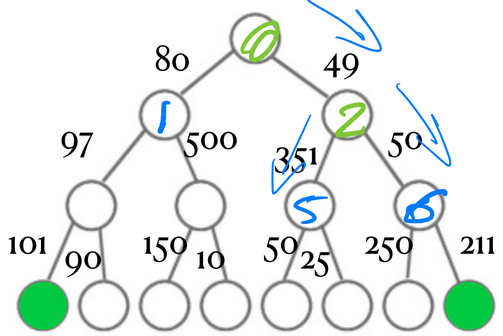
- Frontier: FIFO

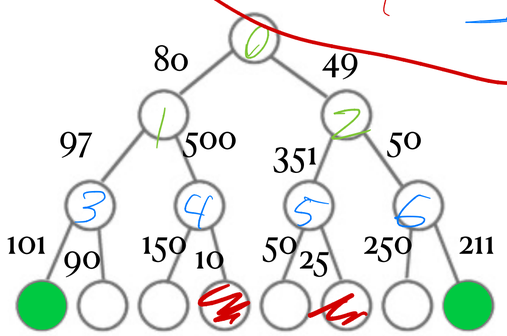
- **Completeness:** Complete if branching factor is finite
- **Optimality:** Optimal if all costs are the same (equal)
- **Time complexity:** $\mathcal{O}\left(\sum_{i=0}^{d-1} b^i\right) = \mathcal{O}(b^d)$
- **Space complexity:**

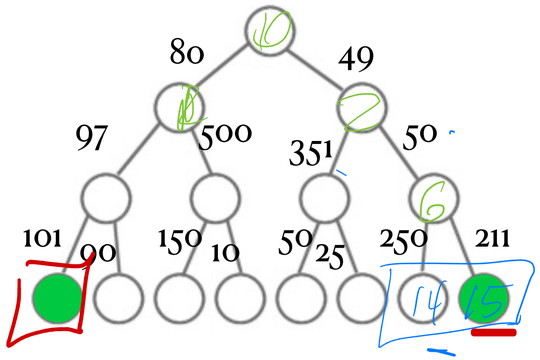
Uniform-cost search

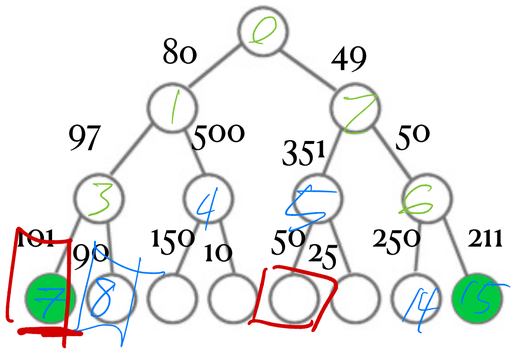
- Expand lowest path cost - g(n)

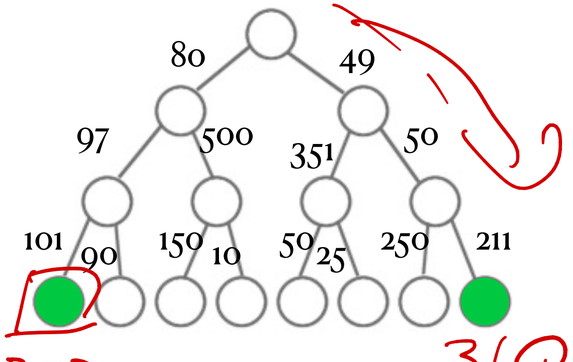








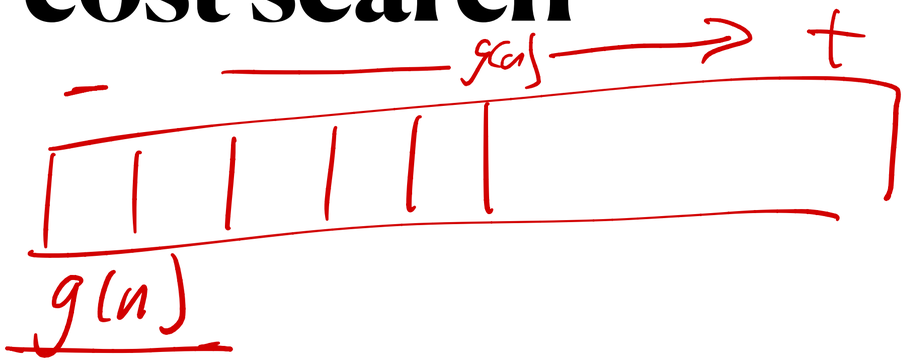




Uniform-cost search

```
1  def generic-graph-search(problem):
2   frontier <- problem.start_state
3   closed <- []
4   loop do:
5    if frontier.empty():
6     FAIL
7    node <- frontier.next()
8    if goal-function(node):
9    return solution
10   closed += node
11   candidates <- expand(node)
12   for c in candidates:
13    if c not in closed and c not in frontier:
14     frontier += c
```

Uniform-cost search



- Frontier: Priority Queue

- **Completeness:** Complete if the branching factor is finite
- **Optimality:** Optimal
- **Time complexity:** $O(5^{1+\lfloor C^*/\varepsilon \rfloor})$? $O(5^{d})$
- **Space complexity:**

ε

RUTGERS

THE STATE UNIVERSITY
OF NEW JERSEY

Problem Solving by Searching

More searching methods

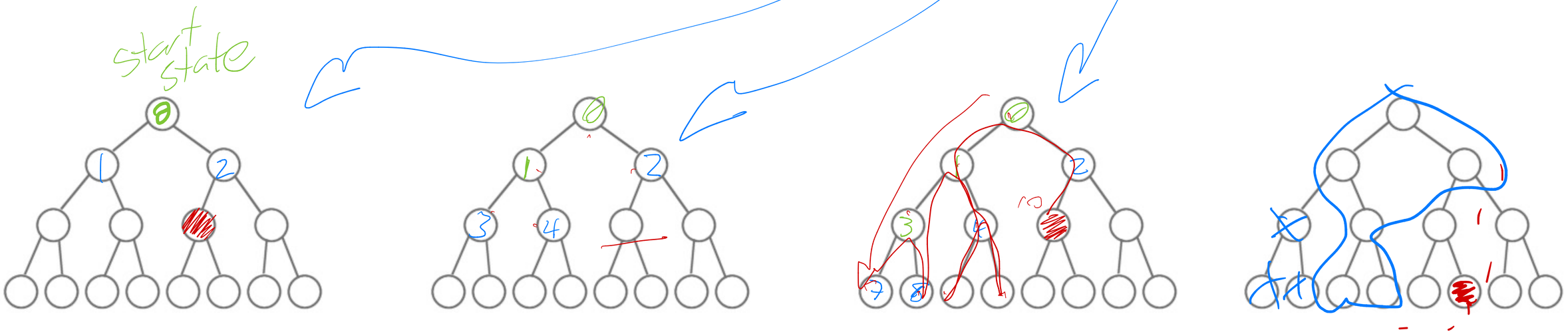
BFS U search

DFS

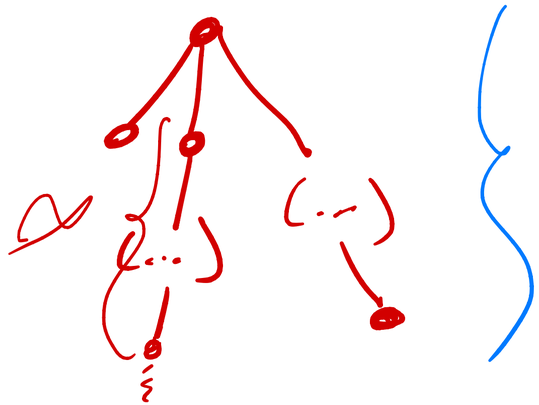
Edgar Granados

Depth-First Search

- Expand deepest node first



Depth-First Search



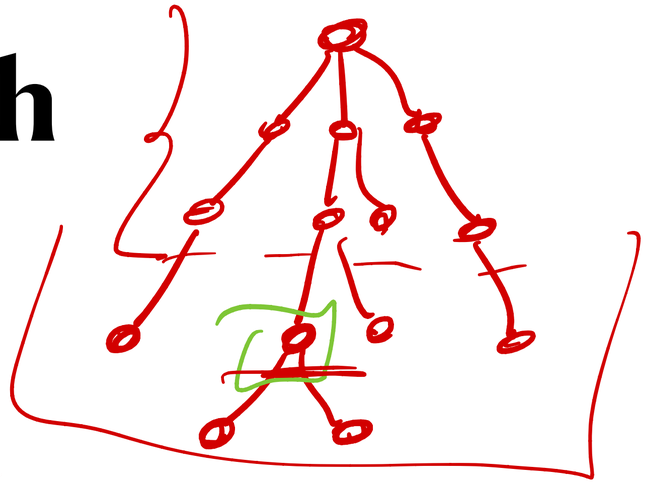
- Frontier: LIFO

- **Completeness:** Fails if paths are infinite
- **Optimality:** Not optimal
- **Time complexity:** $O(b^m)$
- **Space complexity:** $O(bm)$

$\hookrightarrow O(m)$

$m = l$

Depth-Limited Search



- Limit the depth to l $l=3$

$l < m$

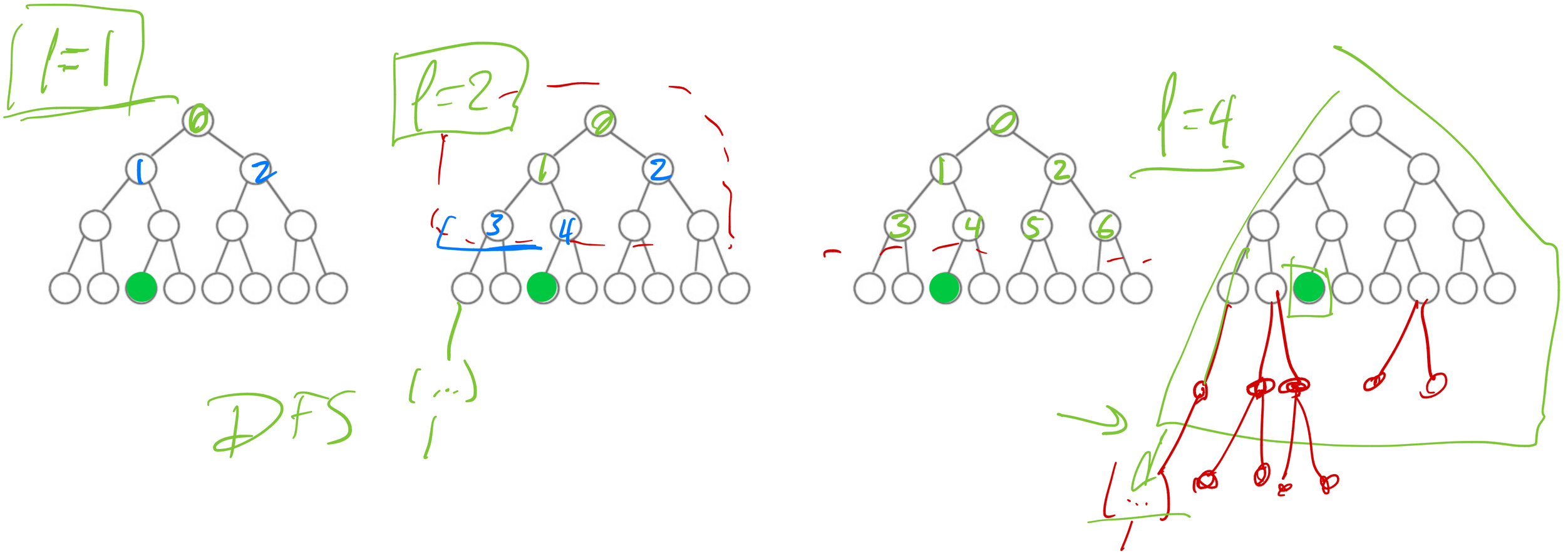
$l = m \Rightarrow$ DFS

- **Completeness:** if $l < d \Rightarrow$ Not complete
- **Optimality:** Non optimal.
- **Time complexity:** $O(b^l)$
- **Space complexity:** $O(bl)$

diameter of the state space

Iterative Deepening Depth-First Search

- Start with l=0 and increase it iteratively

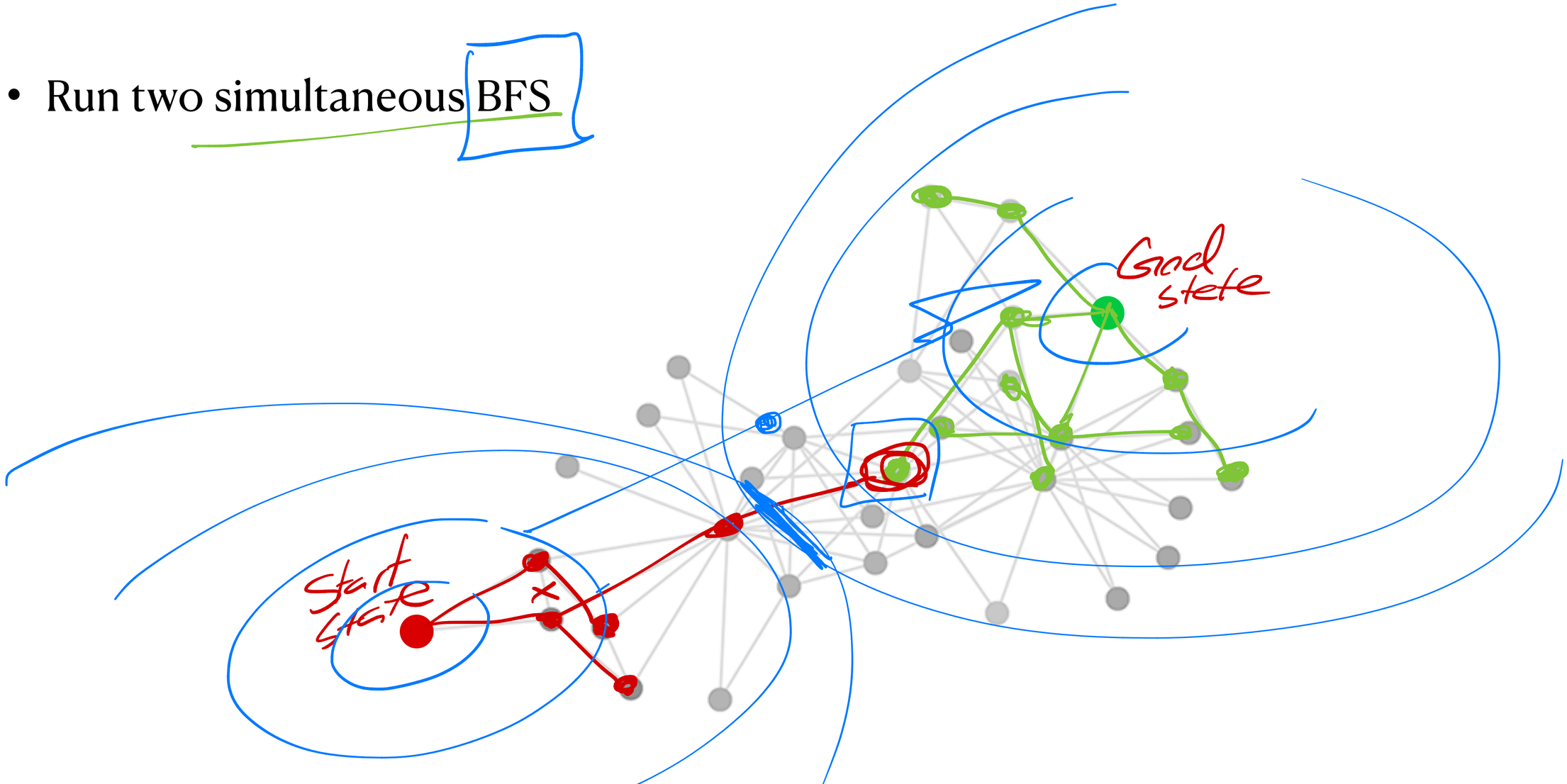


Iterative Deepening Depth-First Search

- **Completeness:** Yes, is complete
- **Optimality:** Only if all costs are equal
- **Time complexity:** $O(b^d) \sim$ BFS
- **Space complexity:** $O(bd) \sim$ DFS

Bidirectional search

- Run two simultaneous BFS



~~Iterative Deepening Depth-First Search~~ Bidirectional

- **Completeness:** Same as BFS
- **Optimality:** Same as BFS
- **Time complexity:** $O(b^{d/2})$
- **Space complexity:**