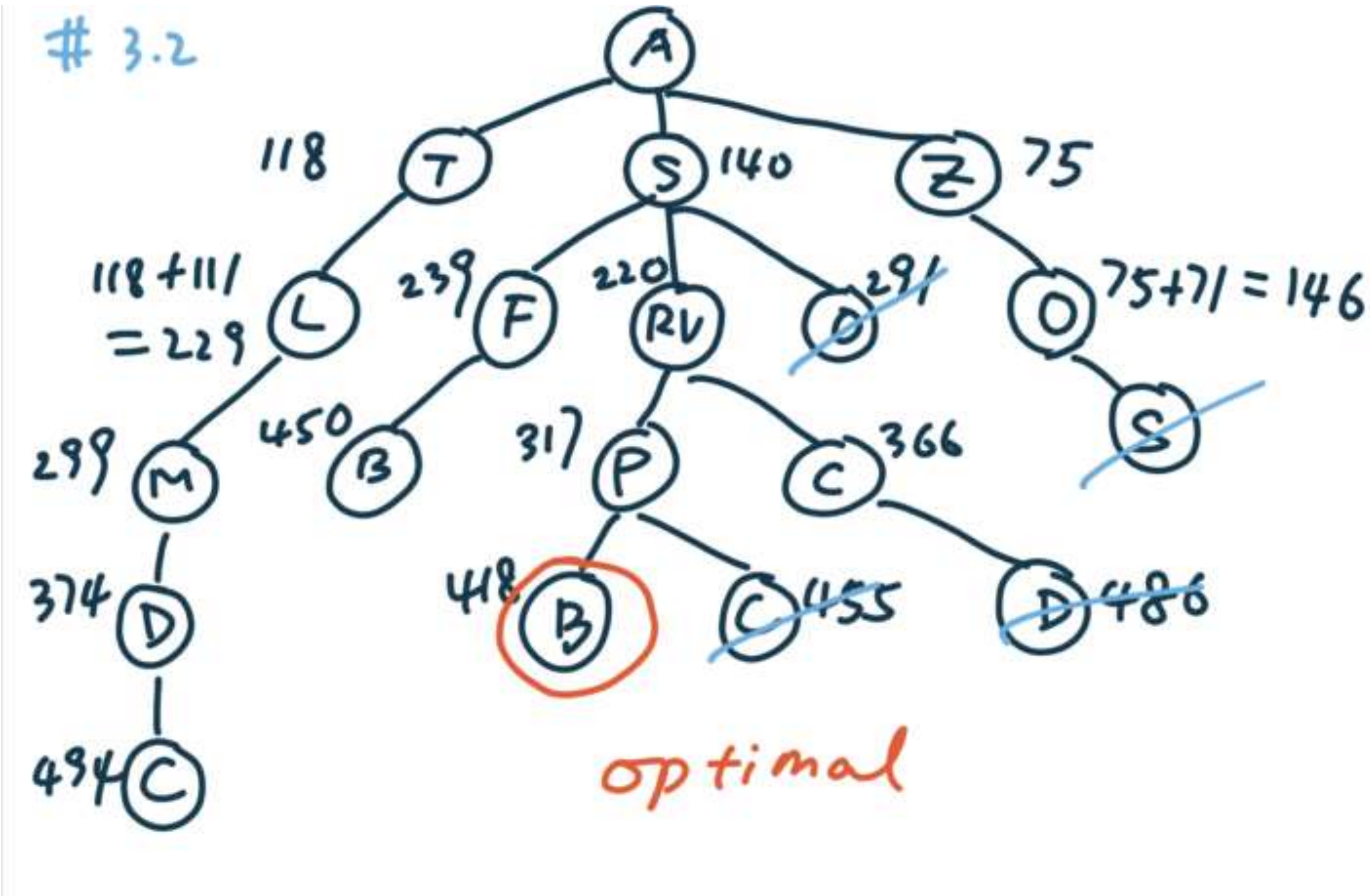
#3.1 Color a planar map with 4 colors

- States: a planar map with states as nodes and state connectivity as edges
- Action: pick a state to color
- Transition model: a state will have a color
- Goal test: all states colored with no adjacent states with the same color.
- Step cost: 1 for each step

#3.1 Monkey and Banana

- States: just as described in the problem.
- Actions: get_banana, move_crate, stack_crate, climb_crate, etc.
- Transition mode: e.g., once monkey executes get_banana, it will have banana
- Goal test: monkey has banana
- Path cost: 1 for each action

#3.2



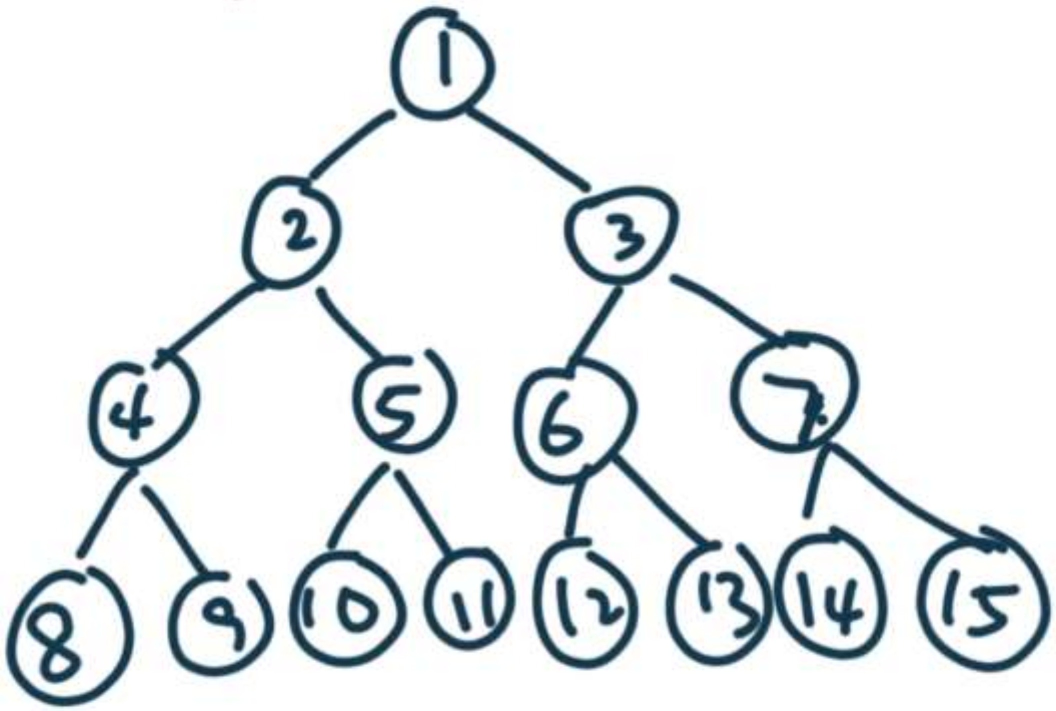
3.2
A
T
118
S
140
Z
75
118+111
=229
L
239
F
220
RV
~~291~~
O
75+71=146
~~S~~
299
M
450
B
317
P
C
366
374
D
418
B
~~C 455~~
~~D 486~~
494
C
optimal

#3.3

- State: a representation of a physical configuration of the problem.
- State space: how states are connected with each other through actions.
- Search tree: start from some initial state, apply all possible actions; select the next state and apply all possible actions; continue the process until goal is found or no new nodes can be generated.
- A finite state space doesn’t always lead to a finite search tree because there could be loops or repeated states.
- A tree or a finite directed acyclic graph will always lead to a finite search tree since there’s no cycle/loop.

#3.4

#3.4



BFS : 1, 2, 3, ⋯ , 10, 11

DLS : 1, 2, 4, 8, 9, 5, 10, 11

IDS : 1, 1, 2, 3,

1, 2, 4, 5, 3, 6, 7

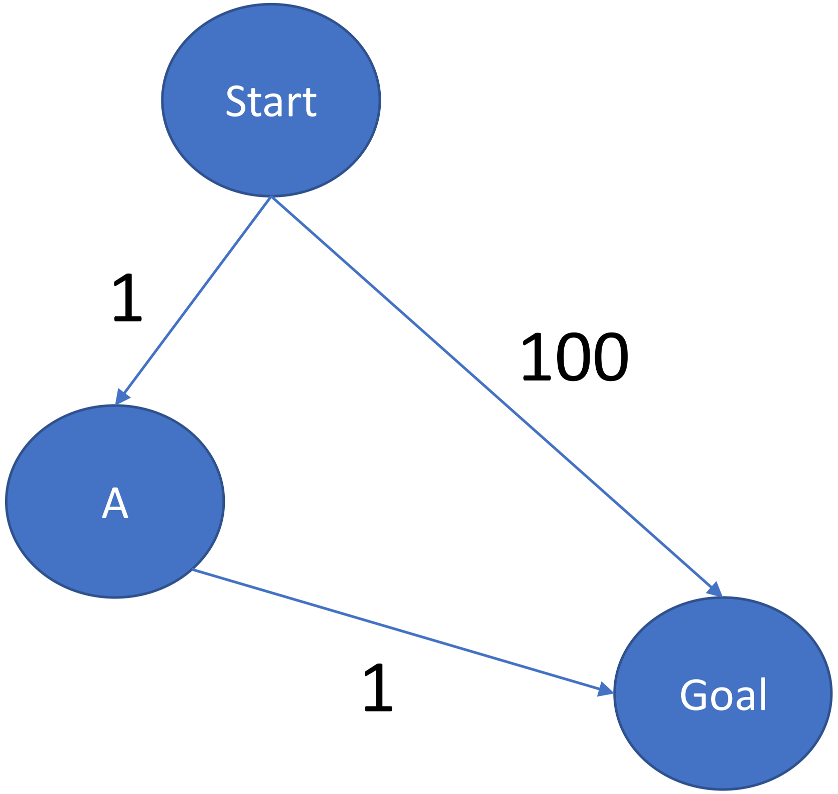
1, 2, 4, 8, 9, 5, 10, 11

#3.5

- Uniform-cost search is optimal when implemented with graph-search algorithm. The search is strictly expanding nodes with increasing path cost g(n). The very first time when the search selects a node for expansion, we find the the lowest cost for that node. It’s not possible to expand to a node later (like in tree search) with a lower cost.
- BFS with constant step costs is optimal with graph-search. The shallowest solution is the cheapest cost node with constant step costs.

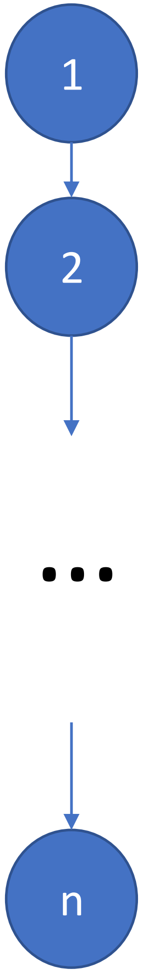
#3.6

- IDS will generate the shallowest solution, which will be {start, goal} with a path cost of 100.
- The optimal solution should be {start, A, Goal} with a cost of 2.

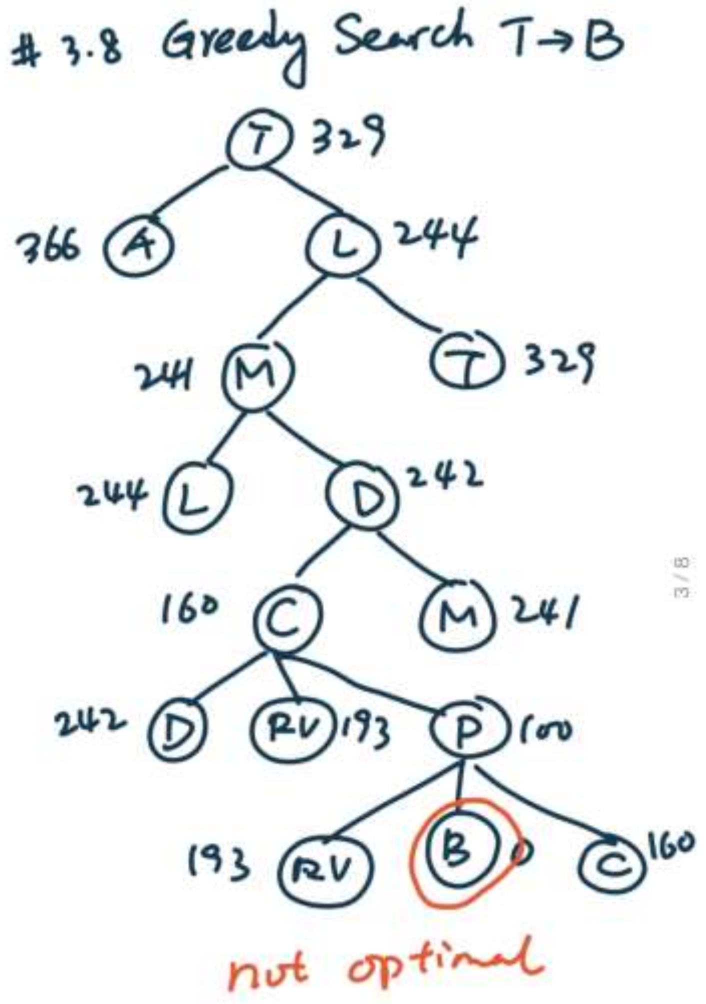


#3.7

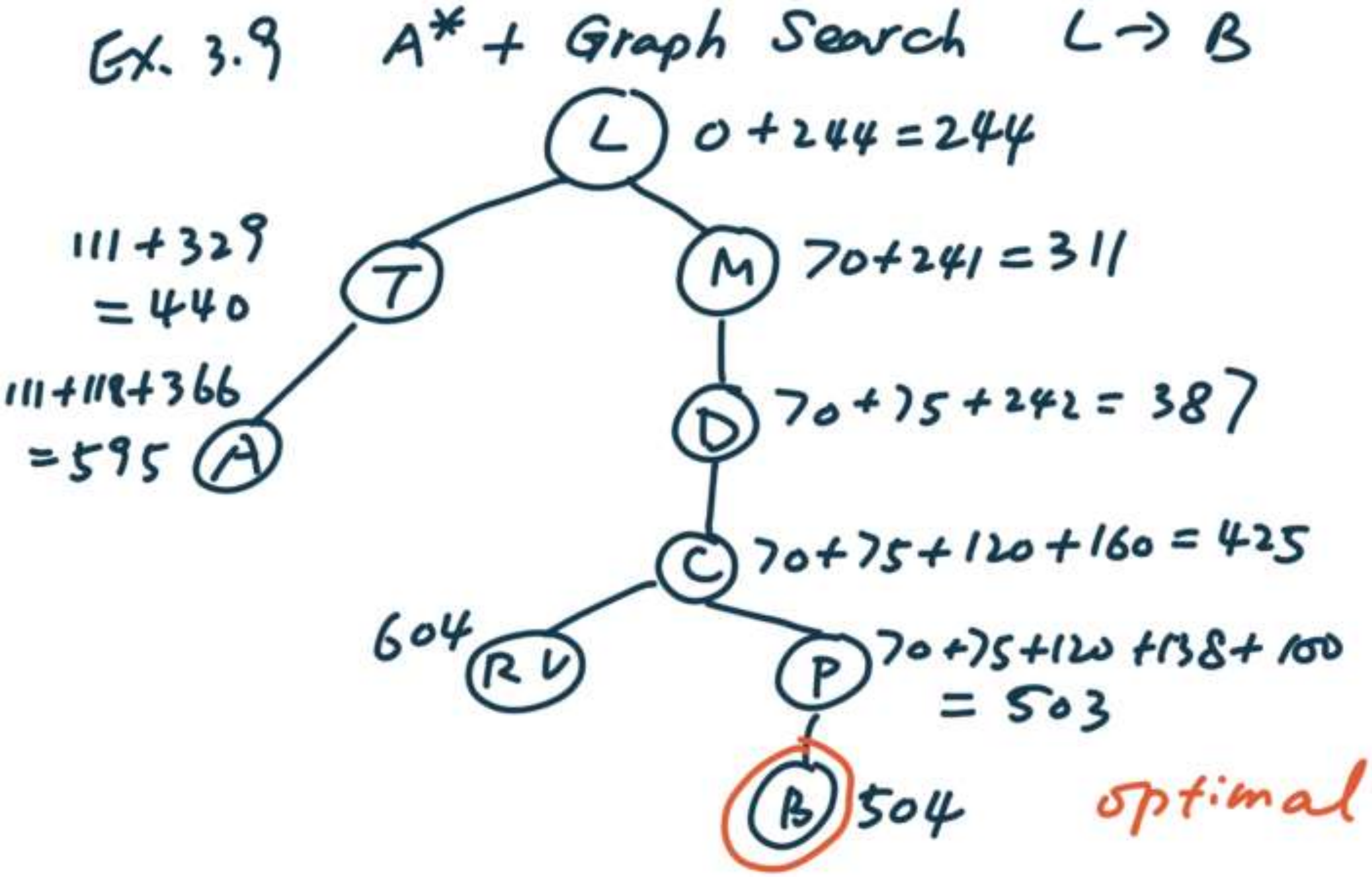
- Imagine the state space is a linked list
- Depth first search cost will be O(n)
- IDS will be 1 + 2+ … + n = O(n^2)



#3.8 Greedy Search + Tree Search



#3.9 A* + Graph Search



#3.10

\# 3.10

1) $f(n) = (2-w) \cdot g(n) + w \cdot h(n)$

$= (2-w) \cdot \left[g(n) + \frac{w}{2-w} \cdot h(n) \right]$

when $\frac{w}{2-w} \leq 1$, $\frac{w}{2-w} \cdot h(n) \leq h(n) \leq h^*(n)$

so $w \leq 2-w \Rightarrow 2w \leq 2 \Rightarrow w \leq 1$

2) $w = 0$: $f(n) = 2g(n) \Rightarrow$ UCS

$w = 1$: $f(n) = g(n) + h(n) \Rightarrow A^*$

$w = 2$: $f(n) = 2h(n) \Rightarrow$ Greedy

#3.11

- BFS is a special case of uniform-cost search when steps costs are identical or non-decreasing based on the depth of a node (both search will find the shallowest solution, which is the optimal solution)
- BFS is a special case of best-first search when $f(n) = depth(n)$
- DFS is a special case of best-first search when $f(n) = 1/depth(n)$
- Uniform-cost search is a special case of best-first search when $f(n) = g(n)$
- Uniform-cost search is a special case of A* when $h(n) = 0$. (hint: when $h(n) = 0$, $f(n) = h(n) + g(n) = g(n)$, which is exactly UCS.)