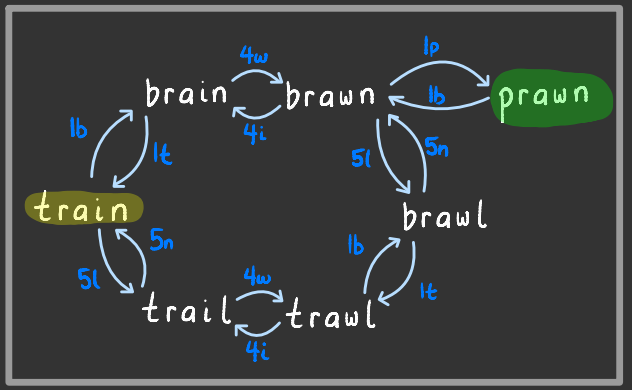
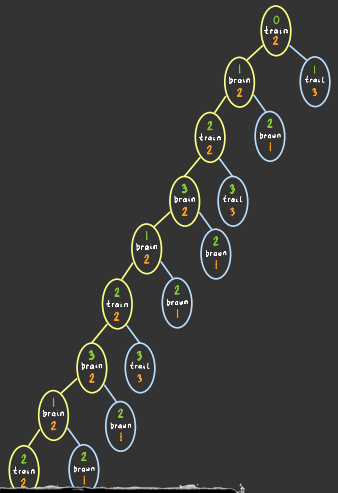
memoized search

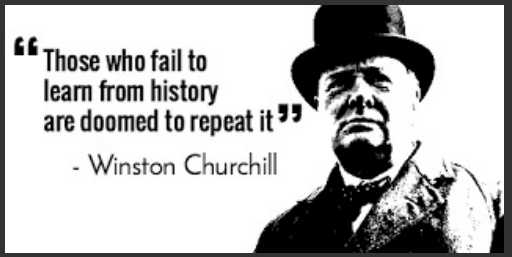
16 sept 2022

CSCI 373

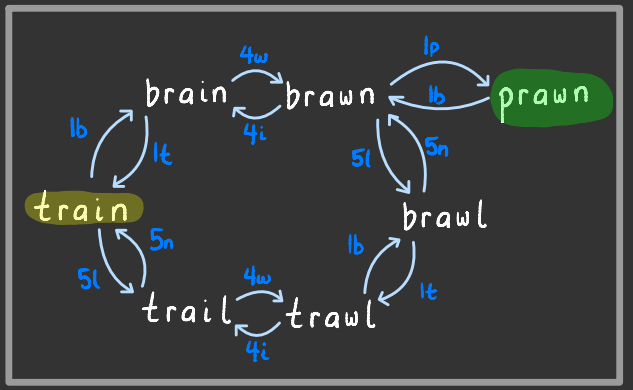
Can't we avoid revisiting the same state over and over?

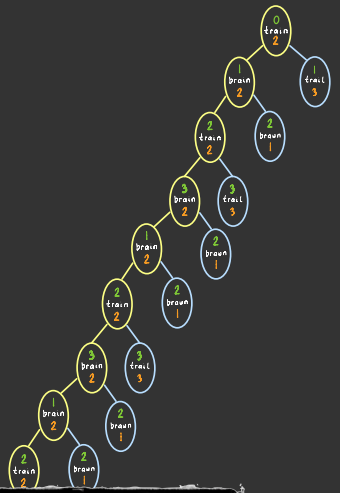


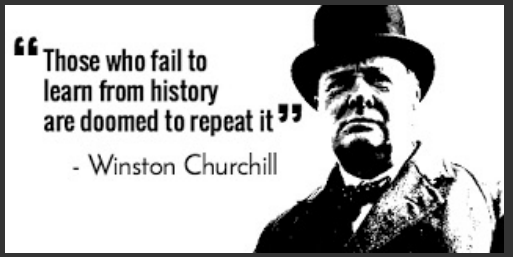

0 train 2
1 brain 2
trail 3
2 train 2
2 brawn 1
3 brain 2
3 trail 3
brain 2
2 brawn 1
2 train 2
2 brawn 1
3 brain 2
3 trail 3
1 brain 2
2 brawn 1
2 train 2
2 brawn 1

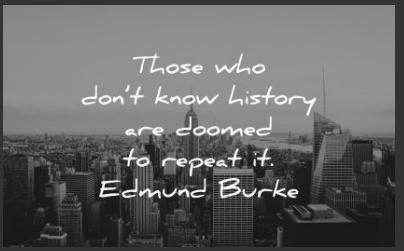

“Those who fail to learn from history are doomed to repeat it”
- Winston Churchill

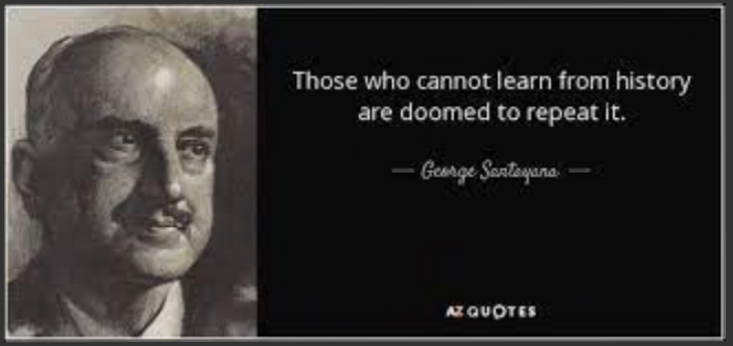

hieronymus bosch


brain
brawn
prawn
train
brawl
trail
trawl
4w
4i
1p
1b
1b
1t
5l
5n
5n
5l
1b
1t
4w
4i

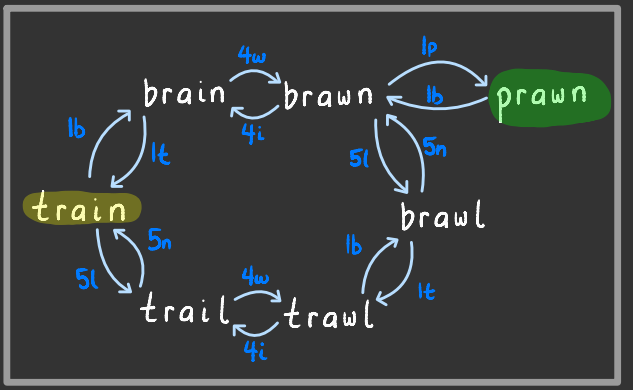

0 train 2
1 brain 2
1 trail 3
2 train 2
2 brawn 1
3 brain 2
3 trail 3
2 brawn 1
brain 2
2 brawn 1
2 train 2
2 brawn 1
3 brain 2
3 trail 3
1 brain 2
2 brawn 1
2 train 2
2 brawn 1


“Those who fail to learn from history are doomed to repeat it”
- Winston Churchill


Those who don't know history are doomed to repeat it.
Edmund Burke


Those who cannot learn from history are doomed to repeat it.
— George Santayana —
AZ QUOTES


hieronymus bosch


brain
brawn
prawn
train
brawl
trail
trawl
4w
4i
1p
1b
1t
5n
5l

SEARCH$(M = (Q, \Sigma, \Delta, q_0, F, w), H)$:

- container = new Container()
- container.put($\langle q_0, 0, H(q_0) \rangle$)
- repeat:
  - if container.empty() then return ∞
  - n = container.get()
  - if $q(n) \in F$ then return $g(n)$
  - let $\text{successors}_{M,H}(n) = \{ \langle q', g(n) + w(q(n), \sigma, q'), H(q') \rangle \mid (q(n), \sigma, q') \in \Delta \}$
  - for $n' \in \text{successors}_{M,H}(n)$:
    container.put(n')

how can we adapt this so that we don't revisit states?

SEARCH$(M = (Q, \Sigma, \Delta, q_0, F, w), H)$:

- your addition here
- container = new Container()
- container.put($\langle q_0, 0, H(q_0) \rangle$)
- repeat:
  - if container.empty() then return ∞
  - n = container.get()
  - your addition here
    - your addition here
    - if $q(n) \in F$ then return $g(n)$
    - let $\text{successors}_{M,H}(n) = \{ \langle q', g(n) + w(q(n), \sigma, q'), H(q') \rangle \mid (q(n), \sigma, q') \in \Delta \}$
    - for $n' \in \text{successors}_{M,H}(n)$:
      container.put(n')

how can we adapt this so that we don't revisit states?

Search$(M = (Q, \Sigma, \Delta, q_0, F, w), H)$:

- visited = {}
- container = new Container()
- container.put($\langle q_0, 0, H(q_0) \rangle$)
- repeat:
  - if container.empty() then return ∞
  - n = container.get()
  - if $q(n) \notin$ visited:
    - visited.add($q(n)$)
    - if $q(n) \in F$ then return $g(n)$
    - let $\text{successors}_{M,H}(n) = \{ \langle q', g(n) + w(q(n), \sigma, q'), H(q') \rangle \mid (q(n), \sigma, q') \in \Delta \}$
    - for $n' \in \text{successors}_{M,H}(n)$:
      container.put(n')

how can we adapt this so that we don't revisit states?

MemoSearch($M = (Q, \Sigma, \Delta, q_0, F, w)$, H):

- visited = {}
- container = new Container()
- container.put($\langle q_0, 0, H(q_0) \rangle$)
- repeat:
  - if container.empty() then return ∞
  - n = container.get()
  - if $q(n) \notin$ visited:
    - visited.add($q(n)$)
    - if $q(n) \in F$ then return $g(n)$
    - let $\text{successors}_{M,H}(n) = \{ \langle q', g(n) + w(q(n), \sigma, q'), H(q') \rangle \mid (q(n), \sigma, q') \in \Delta \}$
    - for $n' \in \text{successors}_{M,H}(n)$:
      container.put(n')

MemoSearch($M = (Q, \Sigma, \Delta, q_0, F, w)$, H):

- visited = {}
- container = new Container()
- container.put($\langle q_0, 0, H(q_0) \rangle$)
- repeat:
  - if container.empty() then return ∞
  - n = container.get()
  - if $q(n) \notin$ visited:
    - visited.add($q(n)$)
    - if $q(n) \in F$ then return $g(n)$
    - let $\text{successors}_{M,H}(n) = \{ \langle q', g(n) + w(q(n), \sigma, q'), H(q') \rangle \mid (q(n), \sigma, q') \in \Delta \}$
    - for $n' \in \text{successors}_{M,H}(n)$:
      container.put(n')

what are the implications in terms of performance?

MEMO SEARCH$(M = (Q, \Sigma, \Delta, q_0, F, w), H)$:

- visited = {}
- container = new Container()
- container.put($\langle q_0, 0, H(q_0) \rangle$)
- repeat:
  - if container.empty() then return ∞
  - n = container.get()
  - if $q(n) \notin$ visited:
    - visited.add($q(n)$)
    - if $q(n) \in F$ then return $g(n)$
    - let $\text{successors}_{M,H}(n) = \{ \langle q', g(n) + w(q(n), \sigma, q'), H(q') \rangle \mid (q(n), \sigma, q') \in \Delta \}$
    - for $n' \in \text{successors}_{M,H}(n)$:
      container.put(n')

what are the implications in terms of performance?

we trade space for time