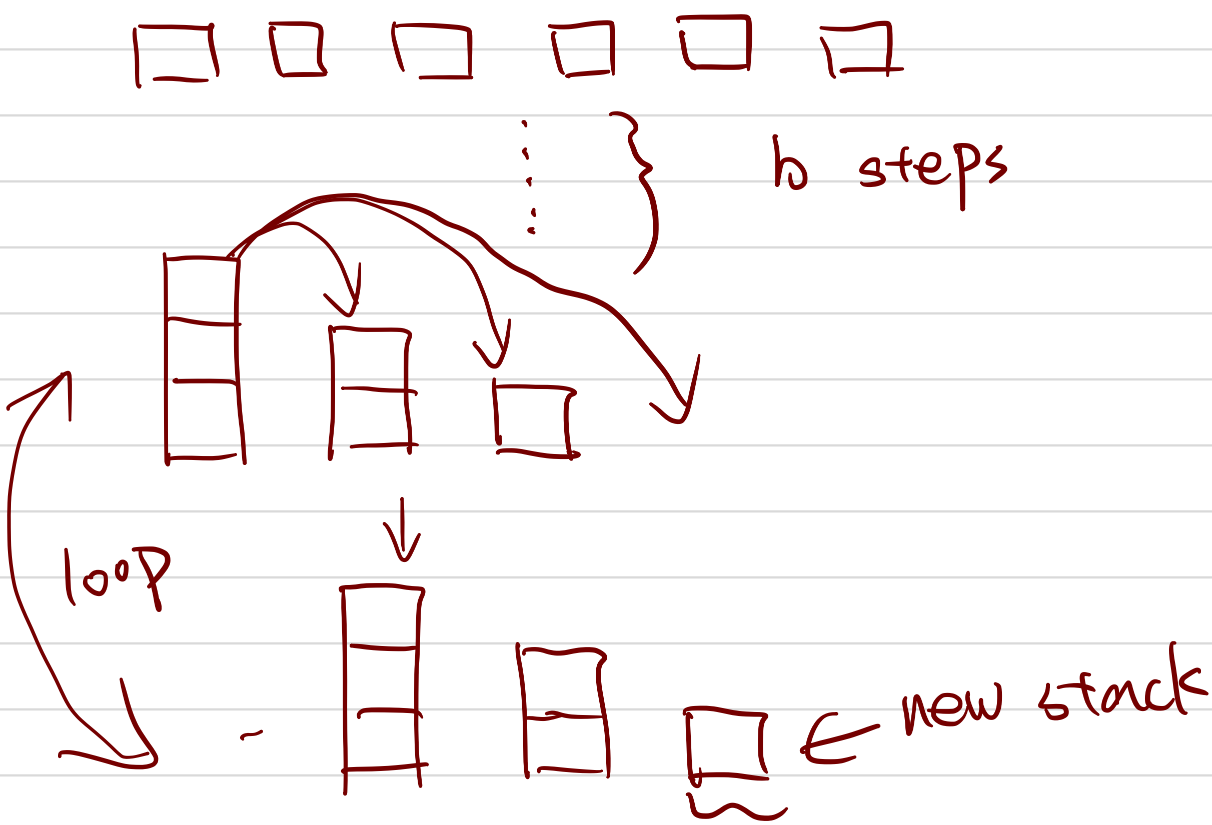
As a result, the average reward per move will equal to :

$$\frac{\text{avrage reward in loop} \cdot (n-b) + \text{average reward out loop} \cdot b}{n}$$

What is average reward in loop?

take $k=6$ as an example



① observation : b is less than k . b is the number of step create a reward stack without recreating a stack .

(2)

In the loop the rewards relationship to k is

$$\Rightarrow \text{In total} = k$$

$$\Rightarrow \frac{1}{2}\left(r \cdot (r+1)\right) = k$$

$$\Rightarrow k = \theta(r^2)$$

$$\Rightarrow r = \theta(\sqrt{k})$$

As a result, we do exaggerate and simplify:

$$\frac{\overset{c_1 \cdot \sqrt{k}}{\text{avrage reward in loop}} \cdot \overset{\text{exaggerate to } n}{(n-b)} + \overset{\text{exaggerate to } c_2 \cdot \sqrt{k}}{\text{average reward out loop}} \cdot \overset{\text{exaggerate to } n}{b}}{n}$$

$$\frac{c_1 \cdot \sqrt{k} \cdot n + c_2 \sqrt{k} \cdot n}{n} \Rightarrow \frac{\sqrt{k} \cdot n (c_1 + c_2)}{n}$$

$$\Rightarrow (c_1 + c_2)\sqrt{k} \Rightarrow O(\sqrt{k})$$

(b)

We define inexpensive cost as the operation cost less than $\sqrt{k}$, expensive is the operation cost equal or higher than $\sqrt{k}$

Φ: total # of bricks move than $\sqrt{k}$ in every stack

Inexpensive operation, $0 < \Delta\Phi < \sqrt{k}$, because inexpensive operation means distribute less than $\sqrt{k}$ bricks

$$\hat{c} = \underset{\sqrt{k}}{c} + \underset{0<\Phi<\sqrt{k}}{\Phi} \Rightarrow \underbrace{\sqrt{k} < \hat{c} < 2\sqrt{k}}$$

$\therefore$ It will cost $O(\sqrt{k})$

expensive operation,

true cost is $\sqrt{k} + c$ ← the cost more than $\sqrt{k}$

define b = total number of bricks above $\sqrt{k}$ in every stack

c = the number of bricks above current stack that more than $\sqrt{k}$

$\phi_i = b \qquad \phi_{i+1} = b - c$

$\Delta\phi = b - c - b = -c$

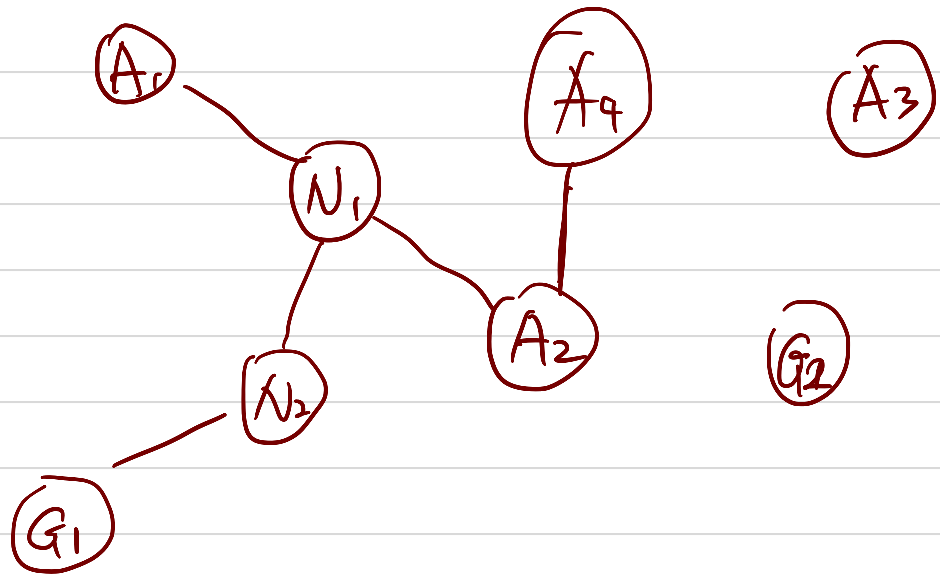
$\hat{c} = c + \Delta\phi = \sqrt{k} + c - c = \sqrt{k}$

Expensive cost $O\sqrt{k}$

As a result, the upper bound is $O(\sqrt{k})$

(c) For the accounting method, every time we have a operation cost less than $\sqrt{k}$, we put 1 in the bank. so every time we have a stack more than $\sqrt{k}$, it means we at least stores $\sqrt{k}$ times to the bank. So when we operate on high stack, we have enough saving. $O\sqrt{k}$ for operation

(2)



We are going to do a DFS on every awful city, that is, if the neighbour node of a_1 is nutral and unvisited, go on explore on n_1 as long as it hit a good city or another awful city.

1 If meet an awful city, remain it unvisited, at the same time, backtrack go to the previous node (e.g. n_1 again) to see if there is any other path unvisited to explore. (go to N2, for example)

② If we meet a good city eventually in some path, backtrack and mark those Nutral city on the way as (can lead to G city)

Resume, a dFS on A_2 based on the marked graph. If it encounter a good city, then mark it as finished with a vacation result. If it encounter a awful city, mark it as finished and no vacation. If it encounter a nutrual city marked as lead to good city, finished with vacation.

Since there are probably isolated city like A_3, the total cost will still be cost of DFS, $O(V+E)$