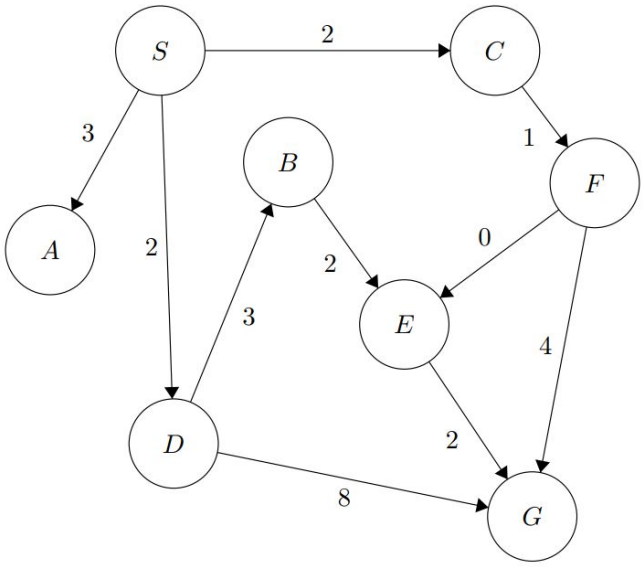
WH #0: Search

- با استفاده از الگوریتم های جستجوی گراف UCS، DFS، BFS و IDS، گراف شکل زیر را با راس شروع S و راس هدف G پیمایش کرده و نودهای بررسی شده (explore)، لیست بررسی مسیر (frontier)، مسیر نهایی مربوط به خروجی الگوریتم و هزینه مسیر را مشخص کنید (Explored set, Frontier, Path, Path cost).



(a) Breadth-first
Explored set: S, A, C, D, F, B
Frontier: S, S-A, S-C, S-D, S-C-F, S-D-B, S-D-G, S-C-F-E
Path: S-D-G
Path cost: 10

(b) Depth-first
Explored set: S, A, C, F, E
Frontier: S, S-D, S-C, S-A, S-C-F, S-C-F-G, S-C-F-E
Note: S-C-F-E-G is never added to the frontier because G is already on the frontier.
Path: S-C-F-G
Path cost: 7

(c) Uniform cost
Explored set: S, C, D, A, F, E
Frontier:
(S, 0), (S-A, 3), (S-C, 2), (S-D, 2), (S-C-F, 3), (S-D-B, 5), (S-D-G, 10), (S-C-F-E, 3), (S-C-F-G, 7), (S-C-F-E-G, 5)
Note: Horizontal strike-through means the node was replaced on the frontier. Specifically, (S-D-G, 10) was replaced by (S-C-F-G, 7), which was, in turn, replaced by (S-C-F-E-G, 5)
Path: S-C-F-E-G
Path cost: 5

(d) Iterative deepening
depth = 0:
Explored set: S
Frontier: S

depth = 1:
Explored set: S, A, C, D
Frontier: S, S-D, S-C, S-A

depth = 2:
Explored set: S, A, C, F, D, B
Frontier: S, S-D, S-C, S-A, S-C-F, S-D-B, S-D-G
Path: S-D-G
Path cost: 10

- برج هانوی یک پازل متعارف است که در حل مشکلات و فرمول بندی مطالعه میشود. بازی با n دیسک با اندازه های مختلف که به ترتیب اندازه در یک میله جمع شده اند به همراه دو میله خالی دیگر شروع می شود. ما دیسک ها را آزادانه بین میله ها حرکت می دهیم، اما با توجه به این شرط که دیسک های بزرگتر را نباید در بالای دیسک های کوچک تر قرار دهیم. هدف این است که همه دیسک ها را به سومین میله منتقل کنیم.
  - چطور میتوانیم این بازی را به شکل یک مسئله جسنجو مدل کنیم، درواقع stateها چه چیزهایی خواهند بود؟

Possible answer: store 3 lists tracking the disks stacked on each peg. Enumerate the disks 1 to n

  - با توجه به اندازه n، اندازه فضای state چقدر خواهد بود؟ (با فرض اینکه از مدلسازی stateای که در سوال قبل مدل کردید، استفاده کنید)

Assume that a disk number can appear on any of the three lists, independently of where the other numbers are. This means that there are 3n possible combinations of the three lists. We don't need to factor in the order of those lists, because the only legal order is sorted from smallest to largest. For example, with only two disks, the number of combinations is 32.

- State شروع چگونه خواهد بود؟

([1, 2, ..., n], [], [])

- عمل هایی (actionهایی) که در هر state می توانید انجام دهید چه خواهد بود؟

We can pop the first integer from any list and push it to the front of any other list, given that this integer is less than the current first integer of the target list (or that the target list is empty).

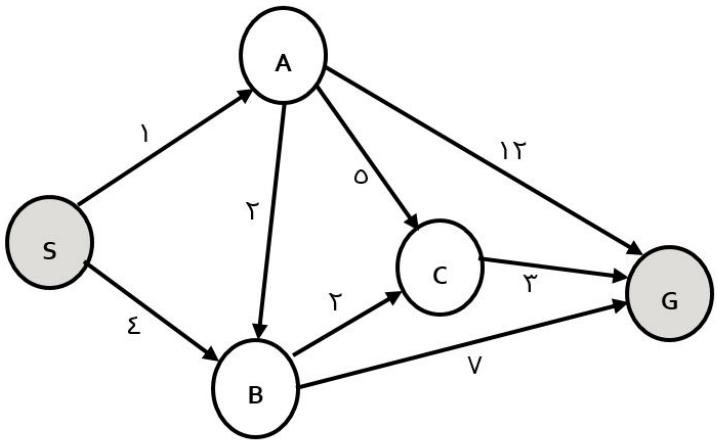
- تست هدف (goal test) را چگونه تعریف می کنید؟ (منظور تابعی است که یک state را میگیرد و تعیین میکند state هدف است یا خیر).

goalTest should check that state == ([], [], [1, 2, ..., n]).

- با توجه به شکل زیر به سوالات پاسخ دهید (S راس شروع و G راس هدف می باشد).
  - نشان دهید تابع هیوریستیک قابل قبول است.
  - درخت جستجوی مربوط به A* را رسم کرده و برای هر نود درخت مقادیر f را به دست آورید.

$$f(n) = g(n) + h(n)$$

state	h
S	۷
A	۶
B	۲
C	۱
G	۰

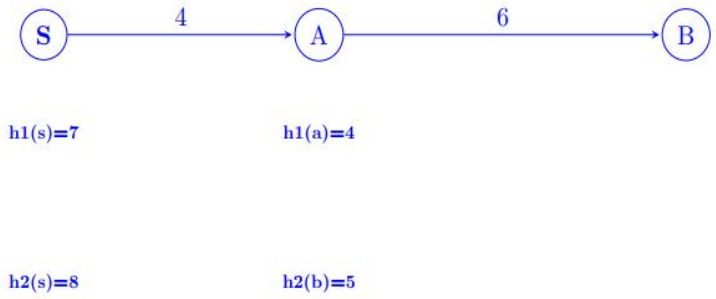


- درستی یا نادرستی گزاره های زیر را با دلیل شرح دهید:
  - تابع اقلیدسی (euclidean distance) یک هیوریستیک قابل پذیرش (admissible) برای مسئله Pacman path-planning است.

True: Euclidean distance will be the minimum cost to travel the path. Thus, it will always be lesser than or equal to the actual cost, making it admissible (0 ≤ h(n) ≤ h*(n)).

  - مجموع چند تابع هیوریستیک قابل پذیرش، قابل پذیرش است.

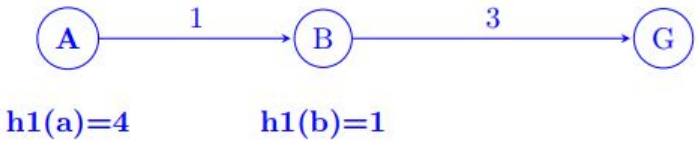
False:



Both of these heuristics (h1 and h2) are admissible, but if we sum them, we find that $h3(s) = 15$ and $h3(a) = 9$. However, this is not admissible.

  - یک تابع هیوریستیک قابل پذیرش برای A^*، سازگار (consistent)نیز می باشد.

False:



$h1(a) > c(a, b) + h1(b)$, which violates the triangle inequality.

- اثبات کنید که اگر یک تابع هیوریستیک سازگار باشد، قابل پذیرش نیز می باشد.

We can prove that consistency implies admissibility through induction.

Recall that consistency is defined such that h(n) ≤ c(n; n + 1) + h(n + 1).

Base Case: We begin by considering the *n - 1th* node in any path where *n* denotes the goal state.

$$h(n - 1) \leq c(n - 1; n) + h(n)$$

Because *n* is the goal state, by definition, $h(n) = h^*(n)$. Therefore, we can rewrite the above as

$$h(n - 1) \leq c(n - 1; n) + h^*(n)$$

and given that $c(n - 1; n) + h^*(n) = h^*(n - 1)$, we can see:

$$h(n - 1) \leq h^*(n - 1)$$

which is the definition of admissibility!

Inductive Step: To see if this is always the case, we consider the *n - 2nd* node in any of the paths we considered above (e.g. where there is precisely one node between it and the goal state). The cost to get from this node to the goal state can be written as

$$h(n - 2) \leq c(n - 2; n - 1) + h(n - 1)$$

From our base case above, we know that

$$h(n - 2) \leq c(n - 2; n - 1) + h(n - 1) \leq c(n - 2; n - 1) + h^*(n - 1)$$
$$h(n - 2) \leq c(n - 2; n - 1) + h^*(n - 1)$$

And again, we know that $c(n - 2; n - 1) + h^*(n - 1) = h^*(n - 2)$, so we can see:

$$h(n - 2) \leq h^*(n - 2)$$

By the inductive hypothesis, this holds for all nodes, proving that consistency does imply admissibility!