CS225 #Assignment 10

Wangjie Solo

2022-5-12

Ex1: 有一个比内存还大的图，只能 go through once，边无向

⇒ 设计 空间复杂度为 O(|V|) 的 Algo 去算 MST (minimum spanning tree)

A: ① Initialize { empty partial set of edges E'
an array of length V (use it to represent connected componants)

② for (every incoming new edge e∈E) {
if (E'∪{e}.iscycle == true) {
E'.find (e' of cycle with maximal cost);
E'.replace (e' → e); }
else { continue; // Span the tree with minimal cost
} // Since |E'| ≤ |V|-1 edges, So. space complexity is O(|V|)

③ When there's only one connected componant in E' & V,
it's exactly the MST we are looking for.

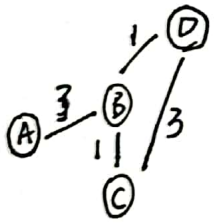


Fig2-1i)

Ex2:

(i): Verify a greedy TSP Algo with 重走 can be better.

A: Like Fig 2-(i), we start from A, when no allowance to repeat:

The cost is: A → B → C → D
3 + 1 + 3 = 7

But with repetition: A → B → C → B → D
cost = 3 + 1 + 1 + 1 = 6 < 7

It's clear (i) is true, Q.E.D.

(ii) Euclidean distance: $d(i,j) \le d(i,k) + d(k,j)$ for vertex i, j, k

Show in such a matrix, repeatition of crossing one city is meaningless for TSP

For a complete tour $\{v_1 \to v_2 \to v_3 \to \cdots \to v_n\}$, we assume repeating can lower the total cost $w_{total} = \{e_1 + e_2 + \cdots + e_{n-1}\}$ (反证法)

edge set
$e_i = \{v_i, v_{i+1}\}$
$i \le n$

Summary ⇒ So we can have edges $\begin{cases} e_\alpha = \{v, w_1\} \text{ replaced by } e_\gamma = \{w_1, w_2\} \\ e_\beta = \{v, w_2\} \end{cases}$

Since $d(w_1, w_2) \le d(w_1, v) + d(w_2, v)$ So they won't be $e_\alpha$ & $e_\beta$ in the TSP ⇒ There will be no repetition advantage

Ex2-(iii) Specifying a greedy Algo for TSP with a Euclidean distance matrix, let it produces tours $\leq$ 2×length of optimal one

Ex3 (i). Well, based on Edmonds-Karp Algorithm, we ~~store~~ add an if-statement when augment the path;

⇒ if ( lst outgoing_edge. next == augmenting path) { store it; }
// i.e. it connects sink vertex
else { search next vertex;
jump; }

(ii) ① Once there exist path between source and sink, we can always use BFS to find it

② The terminate condition for ford-fulkerson Algorithm is that there don't exist any path from source to sink i.e. sink is isolated.

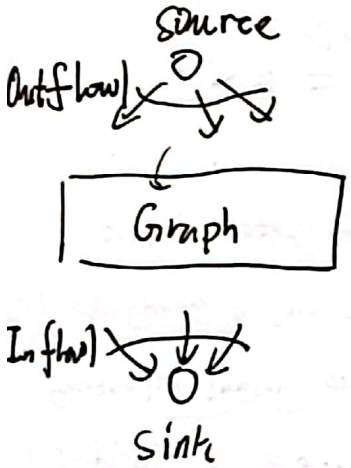


Figure 3-(ii)

③ Since the flow out of source is countable, i.e., like left figure shows, |outflow| ∈ R is limited

④ So the |inflow| = maxflow enters sink point is also bounded, |inflow| ≤ |outflow| ⑤ each augmenting path will consume some of the out flow, then the number of augmenting path is limited, finally sink will be isolated

⑥ ⇒ So we prove ford-fulkerson has a safe terminate.

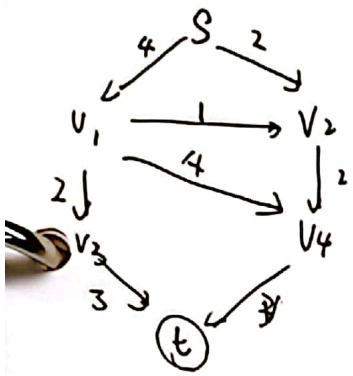
# Max Flow : Ford-Fulkerson Algorithm

1° Def: 用有向图表示资源的流动，在不 violate any capacity constraint 时求最大 rate

Source → Sink

$\frac{N_{sink}}{N_{source}}$

→ conduit n. 导管 有定 Capacity




Flow conservation 流动守恒 ≒ Kirchhoff's current law

A flow network $G=(V,E)$

- each edge capacity $c(u,v) \geq 0$

2° FF Algo 的核心：「迭代」

<1> 残留网络 "去掉饱和的边"

⇔

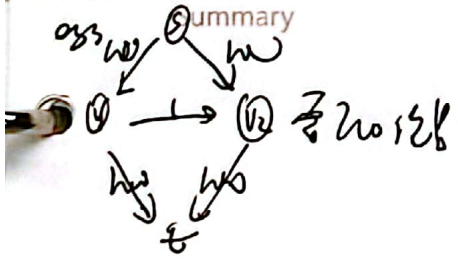
⇝ 增广路径：Augmenting path [ a path from source to sink that doesn't contain cycles ]

「添加反向路径」→ 可反悔

<3> 割：

把 Saturated edge remove from residual graph

3° Time Complexity：很慢




- Each iteration increases the amount of flow by at least 1
- Thus, # iterations ≤ Amount of max flow

∴ Worst $O(f \cdot m)$ ↑ f 为 max flow, m 为 # edges