طراحی الگوریتم

حسین بیاتی

تمرین ۳۰

۸۱۰۱۹۸۳۶۶

۱ـ Bellman-Ford : Detecting a ~~neg~~ cycle

1 >

فقط اینجا به دنبال بیشینه کردن
ضرب یال ها هستیم و دور
نباید از ۱ بزرگتر باشد

instead of "dist"

تفاوت ها
فاصله

```
Relax(u, v, w){
    if product[v] < product[u] * w(u,v){
        product[v] = product[u] * w(u,v)
        parent[v] = u
    }
}
```

then:

```
for each edges (u,v) ∈ E
    if product[v] < product[u] * w(u,v)
        return null
```

۲- در constructing a shortest path به کمک الگوریتم Floyd-warshall

کافیست به جای $\pi[i,j]$ ، $PC[i,j]$ را ذخیره کنیم که تعداد

کوتاه ترین مسیرها از i به j را نشان می دهد.

در هر حالت تساوی : $A[i,j] = A[i,k] + A[k,j]$

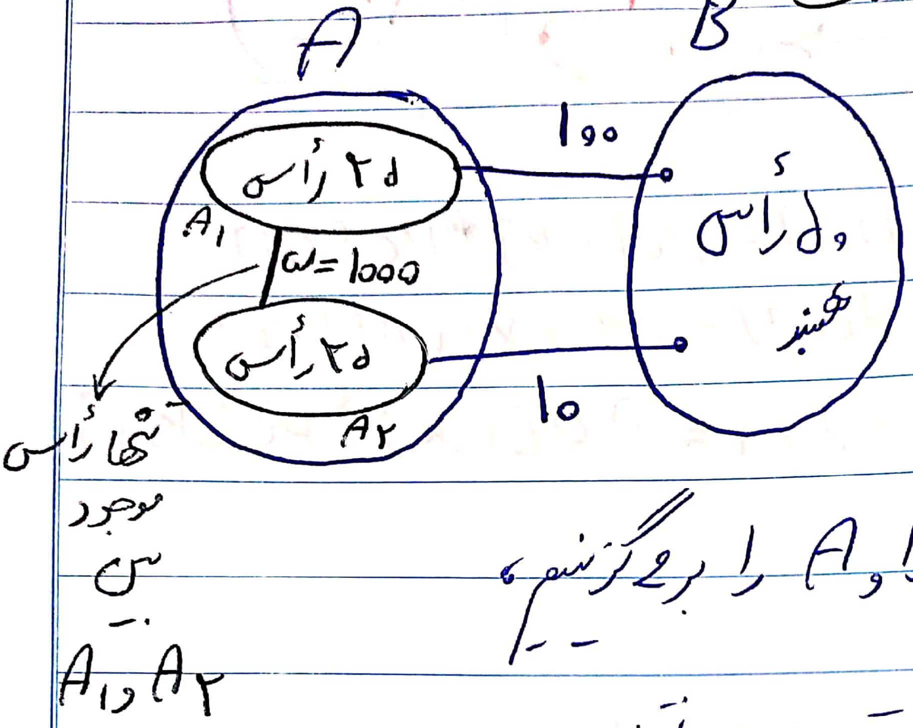
قرار می دهیم : $PC[i,j] += PC[i,k] * PC[k,j]$

شبه کد :

```
Floyd_warshall(W) {
    n = W. rows
    for i=1 to n
        for j=1 to n
            A[i,j] = W[i,j]
            if i==j OR W[i,j]==∞
                PC[i,j] = Ø
            else
                PC[i,j] = 1
    for k=1 to n
        for i=1 to n
            for j=1 to n
                if A[i,j] > A[i,k] + A[k,j]
                    A[i,j] = A[i,k] + A[k,j]
                    PC[i,j] = PC[i,k]*PC[k,j]
                else if A[i,j] == A[i,k] + A[k,j]
                    PC[i,j] += PC[i,k]*PC[k,j]
    return A, PC
```

۳. خیر راه نادرستی است

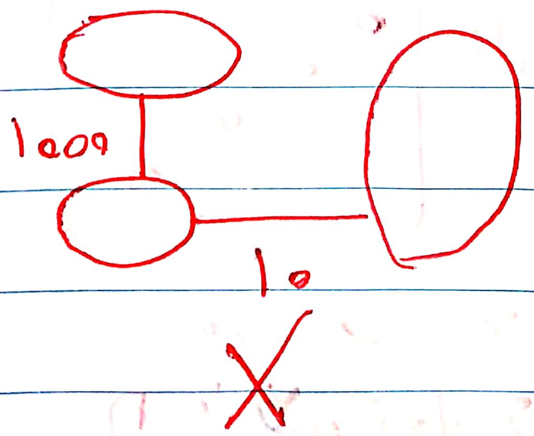
مثال نقض:

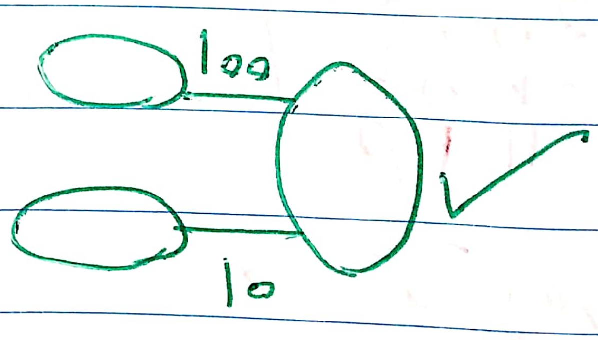


به طور دلخواه B و A را بر می‌گزینیم.

سپس طبق الگوریتم، ۱۰ انتخاب و ۱۰۰ حذف و به ناچار برای همبند شدن سمت چپ ۱۰۰۰ انتخاب می‌شود.

در حالی که می‌توان ۱۰ و ۱۰۰ را انتخاب و ۱۰۰۰ را حذف کرد.





۴. با استفاده از الگوریتم Dijkstra

```
Dijkstra(G, w, s) {
    S = {}
    for each node v ∈ V
        dist[v] = ∅
        Parent[v] = null
    dist[s] = ∅
    max_possible_dist = ∅    mpd_node = s
    while |S| ≠ |V| {
        u = argmax_{v ∈ V-S} {dist[v]}
        S = S + {u}
        for each neighbor of u in v ∈ V-S
            if (dist[v] < dist[u] + w(u,v)) & (w(u,v) != ∞)
                dist[v] = dist[u] + w(u,v)
                Parent[v] = u
            }   if dist[v] > max_Possible_dist
                    max_Possible_dist = dist[v]
                    mpd_node = v
    }
}
```

۵.

$O(n \times (n+m) \log n)$

راس‌ها — یال‌ها

از Dijkstra استفاده می‌کنیم، که هر راس به جا کوچک‌ترین مسیر، دو مسیر کوتاه را نگه می‌داریم و در هر گسترش، از میان dist1[v], dist2[v], dist1[u]+w(u,v) و dist2[u]+w(u,v) دو تا کوتاه‌ترین مسیرها را انتخاب کرده و در dist1[v] و dist2[v] قرار می‌دهیم.

```
Dijkstra (G, w, s) {
    S = {}
    for each node v ∈ V
        dist1[v] = ∞ , dist2[v] = ∞
        Parent[v] = null
    dist[s] = 0
    for each node v ∈ V
        Add v to Q with priority dist[v] **
    while Q ≠ {} {
        u = ExtractMin (Q) **
        S = S + {u}
        for each neighbors of u in V - S {
            if dist1 [v] >= dist1[u] + w(u,v) {
Pdist1=dist1[v]
                dist1 [v] = dist1[u] + w(u,v)
                dist2 [v] = min(Pdist1, dist2[u] + w(u,v))
            }else{
                dist2 [v] = min(dist2[v], dist1[u]+w(u,v))
            }
        }
    }
}
```

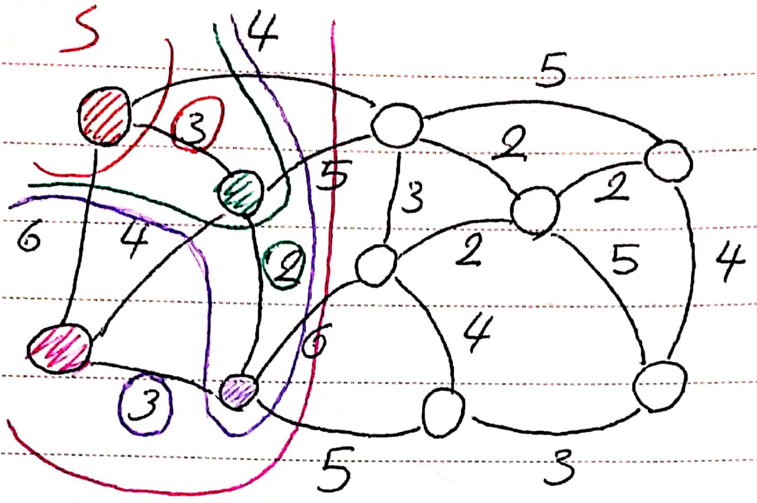
در $O(m \log n)$ (مثل دایکسترا)

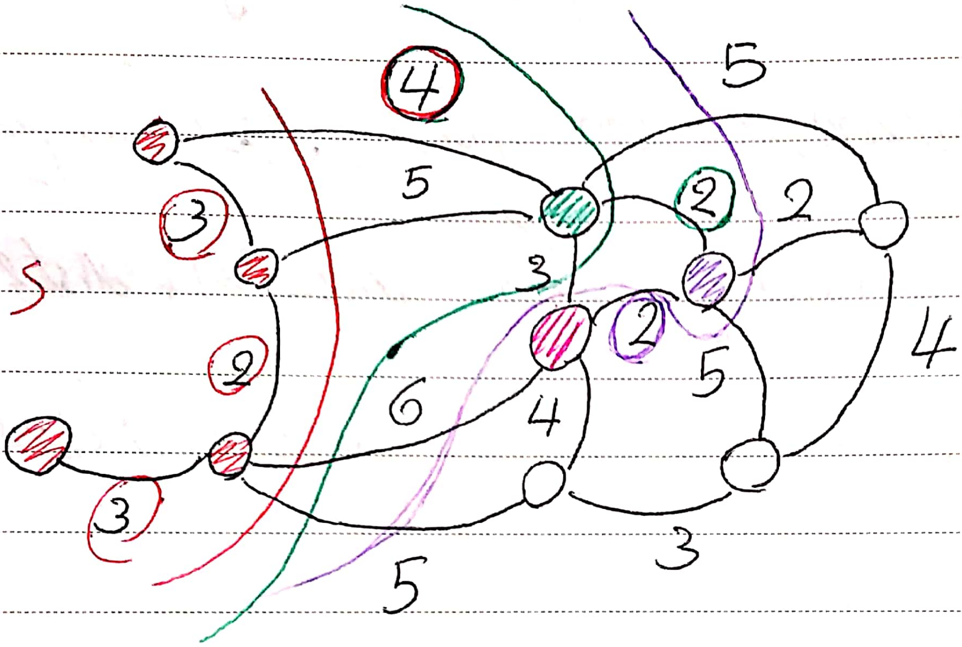
قابل حل است

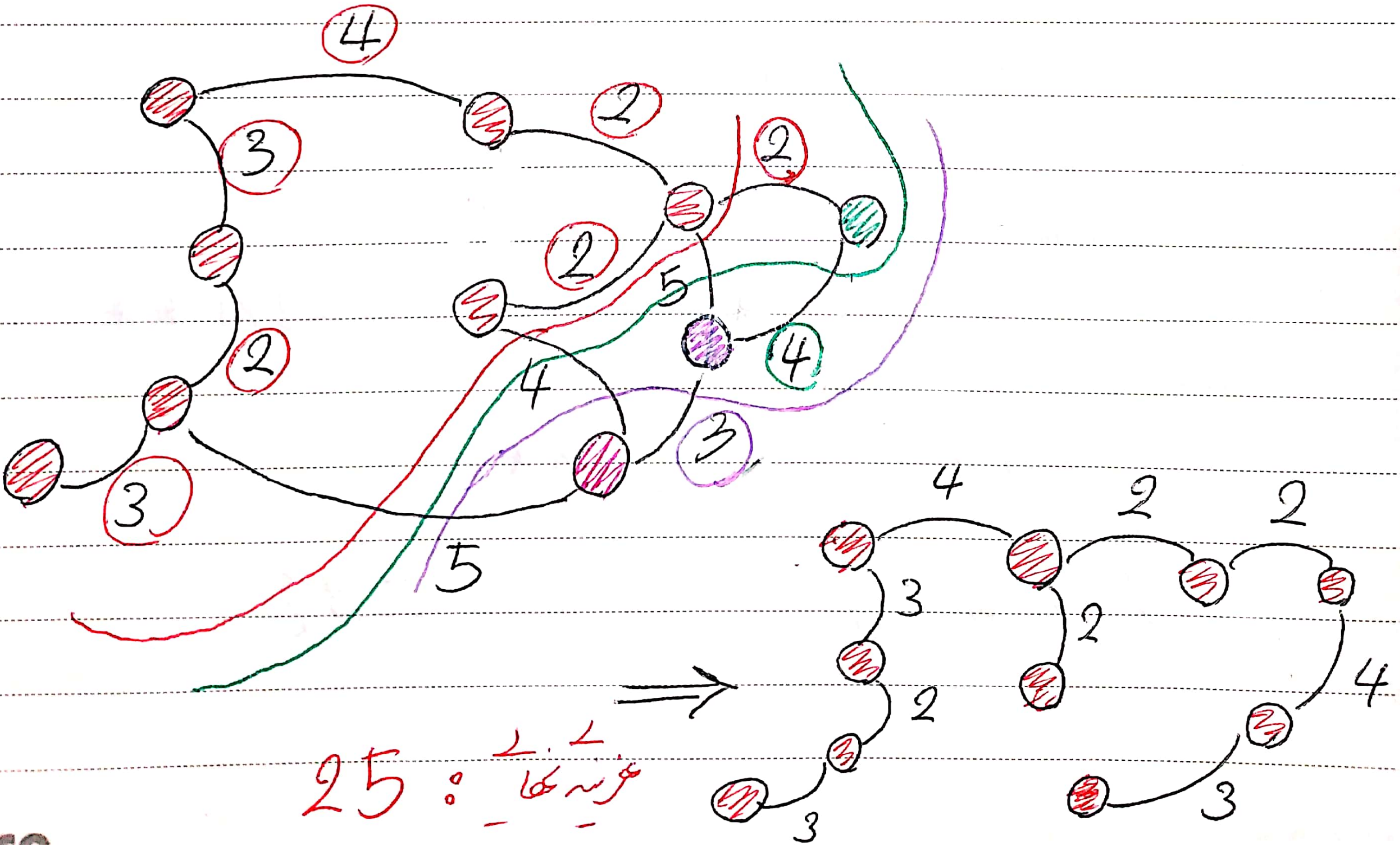
* (برای حل با $O(n(m+n) \log n)$ می‌توانیم هر بار یکی از یال‌های کوتاه‌ترین مسیر را
حذف کرده، دوباره کوتاه‌ترین مسیر را بدست می‌آوریم و طول آن را ذخیره می‌کنیم. بار بعد این یال را
بازگردانده و یال دیگری از کوتاه‌ترین مسیر را حذف کرده و کوتاه‌ترین را حساب می‌کنیم و —
و از بین این مسیرها کوتاه‌ترین را انتخاب می‌کنیم

$n \times m \log n$

$m+n$ از همان $O(m)$ است


5
4
5
3
2
5
3
2
6
4
2
5
4
2
6
4
3
5
3


4
5
5
3
5
2
2
3
5
2
6
4
5
4
2
3
3
5


4
2
3
2
2
5
2
4
4
3
3
5
4
2
2
3
2
2
4
3
3

هزینه کل : 25