Legrövidebb utak gráfokban. Negatív súlyú élek. A Dijkstra és a Bellman-Ford algoritmus

Széltében keresés

Adott egy súlyozatlan, egyszerű, irányítatlan v. irányított gráf. Adott egy s kezdőcsúcs. Minden csúcsra határozzuk meg a legrövidebb távolságát s-től.
Megoldás: a szélességi keresés.

- Egy adott kezdőpontból kiindulva bejárja a többi csúcsot
- A kezdőcsúcsból vagy az adott rétegből egy lépésben elérhető csúcsok alkotják az első/a következő réteget
- Rátalál minden elérhető csúcsra

– Minden egyes lépésben feljegyezzük az s-től mért távolságot, valamint azt, hogy mely pontból értük el (ős).
– Azoknak a pontoknak nem lesz őse, amelyek nem elérhetőek s-ből, valamint magának s-nek.
– Több ősjelölt esetén elég egyet megjegyezni. Ezt kihasználhatjuk egy legrövidebb útvonal kinyomtatására.
– Ha mindegyik csúcsra ráírjuk az elérési réteg sorszámát, akkor megkapjuk a csúcsok távolságát a kezdőponttól.

- Kiszámítja a legrövidebb (a legkevesebb élből álló) utat
- Létrehoz egy ún. szélességi fát: amelynek a gyökere a kezdőcsúcs, ágai pedig a keresés lépései
- A szélességi fát is rétegenként hozza létre.

Legrövidebb utak irányított gráfokban

- G=(V,E) irányított gráf + w:E→R súlyfüggvény.
- Egy $p=(v_0, v_1, \ldots, v_k)$ út súlya: $w(p)= \sum_{i=0}^{k} w(v_i, v_{i+1})$
- Az u-ból v-be vivő legrövidebb út definíciója:
- $\delta(u,v)=\Sigma \min\{w(p), p: u \to\to v\}$, ha létezik p, egyébként ∞
- Figyelem!! δ(u,v) NEM egyértelmű!!
- Legrövidebb utak feszítőfája←→minimális súlyú feszítőfa
- Problématípusok:
- Alapprobléma: egy adott s∈V kezdőcsúcsból az összes v csúcsba vivő legrövidebb utak problémája. Következmények:

  1. Adott csúcsba mindenhonnan beérkező legrövidebb utak problémája. (megoldás az élek megfordításával)

  2. Adott csúcspár közötti legrövidebb utak problémája (nem ismert aszimptotikusan gyorsabb megoldás, mint az Alapprobléma)

  3. Összes csúcspár közötti legrövidebb utak problémája (Alapprobléma minden csúcsra – ennél gyorsabb megoldás is létezik)

Negatív súlyú élek

- Negatív súlyú élek, ill. negatív összsúlyú körök problematikusak. Ha ilyet tartalmaz egy u-ból v-be vivő út, akkor δ(u,v)=- ∞. Ha v nem elérhető, akkor δ(u,v)=∞

- Algoritmuseredmények: 1. Súly, 2. Út (utak)

Algoritmusok általános jellemzése

- Dijkstra algoritmusa: csak nemnegatív élekre!
- Bellman-Ford algoritmus: negatív élekre is működik, megtalálja a negatív köröket is
- Fokozatos közelítés: az egyes csúcsok elérési súlyainak közelítése a felső korlát fokozatos finomítása (csökkentése) útján
- Alapelv: az optimális részstruktúrák elve. Egy optimális utakat tartalmazó feszítőfa részei szintén optimális utakat tartalmaznak.

Dijkstra algoritmusa

```
Dijkstra(s)                  // s: forrás
for ∀v∈V do                  // minden élre a gráfban
      Táv[v]=∞               // s-v távolság egyelőre ismeretlen (∞)
      Szülő[v]=NIL           // szülő definiálatlan
Táv[s]=0                     // s-s közti távolság 0
Bejáratlan=V                 // Bejáratlan élek
while Bejáratlan<>{}         // amíg van bejáratlan
      do u=Bejáratlan.KiveszMin
                // kivesszük a legkisebb bejáratlan élt
      for ∀v∈Szomszéd(u)  // ennek minden szomszédos élére
      do if Táv[v]>Táv[u]+Súly[u,v]
                then Táv[v]=Táv[u]+Súly[u,v]
                Szülő[v]=u
```

(A súly(u,v) a szomszédos u és v pontok közötti távolságot számolja ki. A Táv[u]+Súly[u,v] a gyökércsomópontból a v csomópontba vezető út hossza, abban az esetben, amikor ez az út az u ponton keresztül megy. Ha az így számolt úthossz rövidebb, mint aktuálisan ismert legrövidebb út a v pontra, akkor az aktuális utat ezzel a rövidebb úttal helyettesítjük.)

Optimális feszítőfa←→Legrövidebb utak

- Az egyes élekhez vezető utak KÜLÖN-KÜLÖN minimálisak, a fa teljes súlya nem feltétlenül.
- Minden lépésben a nyitott vágat legrövidebb úthosszúságú élét vesszük hozzá a halmazhoz, ez nem feltétlenül a legkönnyebb is.
- A fa TELJES súlya minimális, az egyes élekhez vezető utak nem feltétlenül
- Minden lépésben a nyitott vágat legkönnyebb élét vesszük hozzá a halmazhoz. Ehhez nem feltétlenül vezet a legrövidebb út.

A Dijkstra algoritmus elemzése

1. A „Bejáratlan" tömb lineáris vektor . ➔ KiveszMin O(V), és |V| ilyen művelet van ➔ KiveszMin össz. $O(V^2)$ A Szomszéd(u) a gráf minden élét 1szer vizsgálja meg. ➔ $O(V^2+E)$

2. A „Bejáratlan" tömb bináris kupac , ➔ KiveszMin ideje O(lgV), és |V| ilyen művelet van. Kupacfelépítés: O(V) KulcsotCsökkent: O(lgV), és legfeljebb |E| ilyen művelet van. ➔ össz.O((V+E)lgV) = O(ElgV)

3. Fibonacci kupacokkal : O(VlogV+E)

A Bellman-Ford algoritmus + elemzése

- Negatív súlyú élek is lehetségesek
- Negatív kör van a gráfban → nincs legrövidebb út
- Futási ideje: O(V*E)
- Buborékrendezéshez hasonló működési elv
- Ha egy csúcs nem elérhető, akkor Táv[u]=∞

```
BellmanFord(s)
for ∀v∈V
      do Táv[v]=∞
      Szülő[v]=NIL
Táv[s]=0
for i=1 to |V|-1
      do for ∀u,v∈E
                 if Táv[v]>Táv[u]+Súly[u,v]
                  then Táv[v]=Táv[u]+Súly[u,v]
                           Szülő[v]=u
for ∀u,v∈E
      do if Táv[v]>Táv[u]+Súly[u,v]
      then    return HAMIS
return IGAZ
```

Legrövidebb utak Irányított Körmentes Gráfokban (KIG)

- KIG: A legrövidebb utak jól definiáltak, mert nincs negatív kör
- Algoritmus: az éleket a csúcsok topologikus rendezésének sorrendjében tekinti
- Nincs ciklus, nincs visszamutató él
- Futási idő: Topologikus rendezés: Θ(V+E) – Minden élt egyszer vizsgálunk – Közelítés konstans idő → Θ(V+E)
- Sejtés: elég lenne csak a kiindulási csúcstól?

A legrövidebb utak meghatározása

1. egyetlen belső élt sem tartalmazó utakra $Szülő^{0}_{ij}$ =NIL, ha i=j vagy $w_{ij}=\infty$ $Szülő^{0}_{ij}$ =i, ha i<>j és $w_{ij}<\infty$

2. az i→→ k→→ j úton a j-t k belső csúccsal megelőző csomópont ugyanaz, mint a k→→ j úton, k- 1 belső csúccsal $Szülő^{k}_{ij}=Szülő^{k-1}_{kj}$, ha $d^{k-1}_{ij} > d^{k-1}_{ik} + d^{k-1}_{kj}$

3. ha az i→→ j út k-t nem tartalmazza, akkor megegyezik az i-ből j-be k-1 csúcsot tartalmazó úton a megelőző csúccsal $Szülő^{k}_{ij}=Szülő^{k-1}_{ij}$, ha $d^{k-1}_{ij} <= d^{k-1}_{ik} + d^{k-1}_{kj}$