Algoritmusok és adatszerkezetek I.
9. Előadás

Legrövidebb utak minden csúcspárra. D és Pi mátrixok, Floyd-Warshall algoritmus; gráf tranzitív lezártja.

Tartalom

- Utak minden csúcspárra
- Legrövidebb utak minden csúcspárra: a Floyd-Warshall algo…
- Floyd-Warshall algoritmus (FW)
- A legrövidebb utak minden csúcspárra probléma visszavezet…
- Gráf tranzitív lezártja (TC)
- A tranzitív lezárt algoritmusa
- A tranzitív lezárt kiszámításának visszavezetése szélessé…
- Köszönöm a figyelmet!

Utak minden csúcspárra

- Két algoritmus hasonlósága
  - A gráfok csúcsmátrixos reprezentációjára épülnek
  - Műveletigényük $\Theta(n^3)$
  - Hasonló elven is működnek
    - Dinamikus programozás
  - Az irányítatlan gráfokat mindkét algoritmus olyan irányított gráfnak tekinti
    - minden (u, v) élnek megvan az ellentétes irányú (v, u) élpárja
    - $w(u, v) = w(v, u)$.

Utak minden csúcspárra

- Két algoritmus különbözősége
  - Floyd-Warshall algoritmusa
    - általánosabb feladatot old meg
    - Későbbi
      - Robert Floyd 1962
  - Tetszőleges gráf tranzitív lezártja
    - Speciálisabb feladatot old meg
    - Bernard Roy (1959), illetve Stephen Warshall (1962)

Legrövidebb utak minden csúcspárra: a Floyd-Warshall algoritmus (FW)

1. Jelölés. $\mathbb{R}_\infty = \mathbb{R} \cup \{\infty\}$

- Az $A/1 : \mathbb{R}_\infty [n, n]$ csúcsmátrixszal adott élsúlyozott gráf alapján meghatározza a
  - $D/1 : \mathbb{R}_\infty [n, n]$ mátrixot ahol $D[i, j]$
    - az i indexű csúcsból a j indexű csúcsba vezető optimális út hossza
    - vagy ∞, ha nincs út i-ből j-be.
  - $\pi/1 : \mathbb{N}[n, n]$ mátrixot is, ahol $\pi[i, j]$
    - az algoritmus által kiszámolt, az i indexű csúcsból a j indexű csúcsba vezető optimális úton a j csúcs közvetlen megelőzője, ha $i \neq j$ és létezik út i-ból j-be
    - Különben $\pi[i, j] = 0$.

Floyd-Warshall algoritmus (FW)

- **Előfeltétel.**
  - A gráfban nincs negatív kör. (Ezt az algoritmus ellenőrzi.)
- A továbbiakban gráf csúcsait
  - 1-től *n*-ig indexeljük
  - a csúcsokat az indexükkel azonosítjuk
- Az algoritmus a $<(D^{(0)}, \pi^{(0)}),(D^{(1)}, \pi^{(1)}), \ldots ,(D^{(n)}, \pi^{(n)})>$ mátrixpársorozatot állítja elő, amit a következőképpen definiálunk

2. Jelölés. $i \overset{k}{\rightsquigarrow} j$ az *i* csúcsból a j csúcsba vezető út, azzal a megszorítással, hogy

- az úton a közbenső csúcsok indexe ≤ *k* (*i* > k és j > k megengedett, továbbá i, j ∈ 1..n és k ∈ 0..n).

3. Jelölés. $i \overset{k}{\underset{opt}{\rightsquigarrow}} j$ az i csúcsból a j csúcsba vezető legrövidebb körmentes út, azzal a megszorítással, hogy

- az úton a közbenső csúcsok indexe ≤ k. Ha több ilyen út lenne, azt vesszük, ahol a közbenső csúcsok indexeinek maximuma a lehető legkisebb.

Floyd-Warshall algoritmus (FW)

4. Megjegyzés. k szerinti indukcióval adódik, hogy

- Ha létezik $i \overset{k}{\rightsquigarrow} j$ út $\Rightarrow$ az $i \overset{k}{\underset{opt}{\rightsquigarrow}} j$ út egyértelműen definiált, ui. a negatív körök hiánya miatt
  - $i = j$ esetén $i \overset{k}{\underset{opt}{\rightsquigarrow}} j = \langle i \rangle$
  - $i \neq j$ esetén pedig
    - $\exists\, i \overset{0}{\underset{opt}{\rightsquigarrow}} j = \langle i, j \rangle \Leftrightarrow (i, j) \in G.E$
    - $k \in 1..n$ esetén pedig egy $i \overset{k}{\underset{opt}{\rightsquigarrow}} j$ úttal kapcsolatban két lehetőség van:
      - $i \overset{k}{\underset{opt}{\rightsquigarrow}} j = i \overset{k-1}{\underset{opt}{\rightsquigarrow}} j$ V $i \overset{k}{\underset{opt}{\rightsquigarrow}} j = i \overset{k-1}{\underset{opt}{\rightsquigarrow}} k \overset{k-1}{\underset{opt}{\rightsquigarrow}} j$

Floyd-Warshall algoritmus (FW)

5. Definíció. $D_{ij}^{(k)} = \begin{cases} w(i \overset{k}{\underset{opt}{\rightsquigarrow}} j) \text{ ha létezik } i \overset{k}{\rightsquigarrow} j \\ \infty \text{ ha nem létezik } i \overset{k}{\rightsquigarrow} j \end{cases}$

6. Definíció. $\pi_{ij}^{(k)} = \begin{cases} \text{az } i \overset{k}{\underset{opt}{\rightsquigarrow}} j \text{ úton a } j \text{ csúcs közvetlen megelőzője, ha i} \neq \text{j és létezik } i \overset{k}{\rightsquigarrow} j, \\ 0 \qquad \text{ha } i = j \text{ vagy nem létezik } i \overset{k}{\rightsquigarrow} j \end{cases}$

7. Tulajdonság. A $D^{(0)}$ mátrix megegyezik a gráf A csúcsmátrixával, a $D^{(n)}$ mátrix pedig éppen a kiszámítandó D mátrix.

8. Tulajdonság. $\pi_{ij}^{(0)} = \begin{cases} i & \text{ha } i \neq j \wedge (i,j) \text{ a gráf éle} \\ 0 & \text{ha } i = j \vee (i,j) \text{ nem éle a gráfnak} \end{cases}$

9. A $\pi^{(n)}$ mátrix pedig megegyezik a kiszámítandó π mátrixszal.

Floyd-Warshall algoritmus (FW)

9. Tulajdonság. A 4. megjegyzés alapján, $k \in 1..n$ esetén:

- Ha $D_{ij}^{(k-1)} > D_{ik}^{(k-1)} + D_{ij}^{(k-1)}$
- akkor $D_{ij}^{(k)} = D_{ik}^{(k-1)} + D_{kj}^{(k-1)} \wedge \pi_{ij}^{(k)} = \pi_{kj}^{(k-1)}$
- különben $D_{ij}^{(k)} = D_{ij}^{(k-1)} \wedge \pi_{ij}^{(k)} = \pi_{ij}^{(k-1)}$

10. Következmény. $k \in 1..n$ esetén:

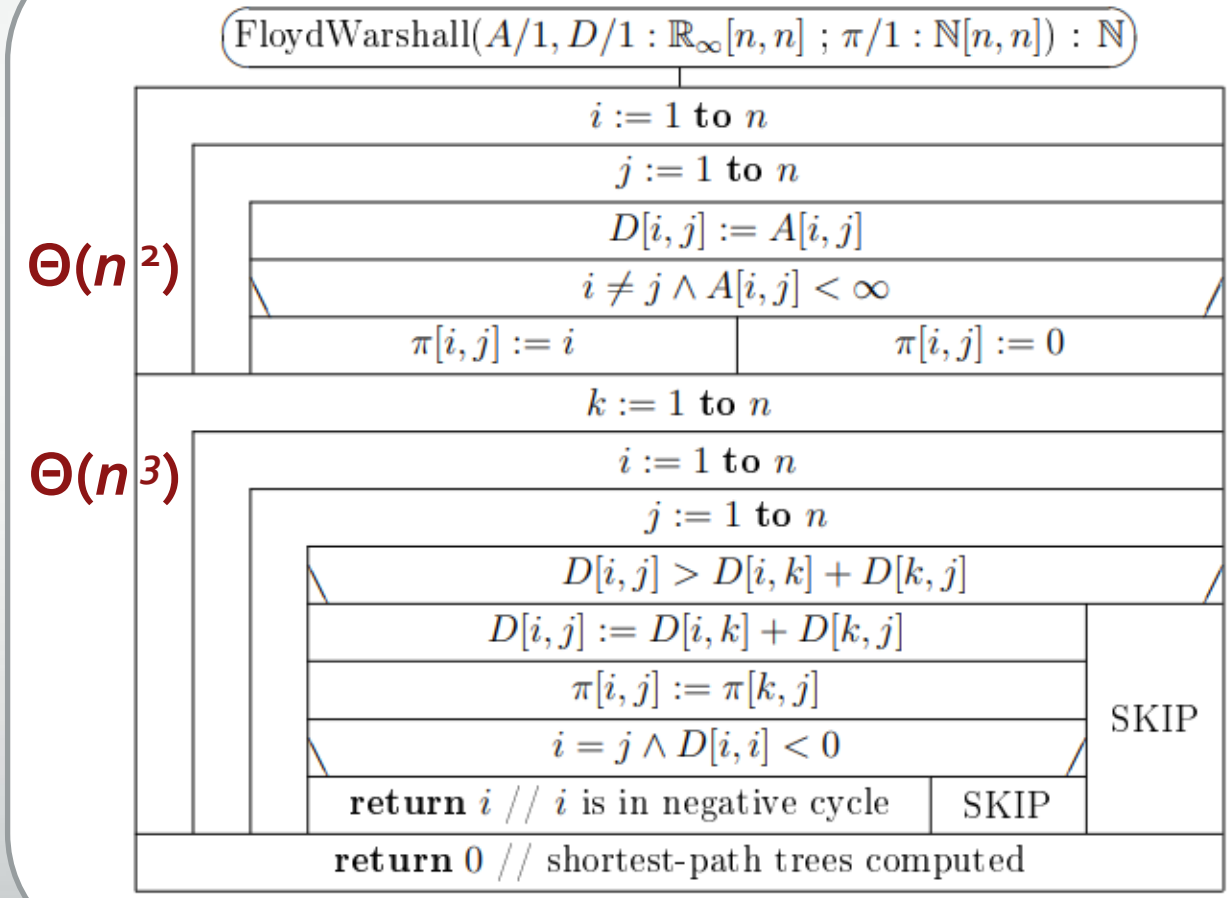
- $D_{ik}^{(k)} = D_{ik}^{(k-1)} \wedge \pi_{ik}^{(k)} = \pi_{ik}^{(k-1)} \wedge D_{kj}^{(k)} = D_{kj}^{(k-1)} \wedge \pi_{kj}^{(k)} = \pi_{kj}^{(k-1)}$
- Ez azt jelenti, hogy a $D^{(k)}$ mátrix k-adik sora és oszlopa ugyanaz, mint $D^{(k-1)}$-é, és a $\pi^{(k)}$ mátrix k-adik sora és oszlopa is ugyanaz, mint $\pi^{(k-1)}$-é.

➢ a D mátrix kiszámításához nem kell segédmátrixokat tárolni, mert

- $D_{ij}^{(k)}$ csak a $D_{ij}^{(k-1)}$, a $D_{ik}^{(k-1)}$ és a $D_{kj}^{(k-1)}$ értékektől függ
- $D_{ik}^{(k)} = D_{ik}^{(k-1)}$, ill. $D_{kj}^{(k)} = D_{kj}^{(k-1)}$,

Floyd-Warshall algoritmus (FW)

- 1. kettős ciklus
  - A fizikailag is létező *D* mátrixot az elméleti $D^{(0)}$ mátrix elemeivel inicializáljuk
- 2. háromszorosan beágyazott ciklus
  - *k* = 1-től *n*-ig kiszámoljuk $D^{(k-1)}$-ből $D^{(k)}$ –t
  - fizikailag végig ugyanazt a *D* és *π* mátrixot használva
- Ha a *D* mátrix főátlójában negatív érték jelenik meg
  - ➢ negatív kört találtunk
- Műveletidő
  - $MT(n) \in \Theta(n^3)$
  - $mT(n) \in \Theta(n^2)$
    - Ha van a gráfban negatív kör
      - 2. külső ciklus 1.iterációja alatt kiderülhet



A legrövidebb utak minden csúcspárra probléma visszavezetése a legrövidebb utak egy forrásból feladatra

- A 2 probléma közötti kapcsolat
  - Floyd-Warshall (FW) algoritmus által D és π mátrixok i-edik sora a legrövidebb utak egy forrásból feladat megoldása $s = i$ esetére
  - Megfordítva, ha minden $s \in 1..n$ értékre megoldjuk a legrövidebb utak egy forrásból feladatot, megkapjuk a legrövidebb utak minden csúcspárra probléma megoldását is
- Néhány speciálisabb eset:
  - Ha a gráf DAG
    - $MT(n, m) \in \Theta(n * (n + m))$
    - Ritka gráfokon: $\Theta(n^2)$
      - ➢ nagysárenddel gyorsabb, mint az FW algoritmus
    - Sűrű gráfokon: $\Theta(n^3)$
      - feltéve, hogy a csúcsok többségéből a gráf nagy része elérhető
      - ➢ a gyakorlatban a nagyobb rejtett konstansok miatt lassúbb futást eredményez
  - Ha élsúlyozatlan gráfokra szélességi keresést alkalmazunk
    - Hasonló eredmények

A legrövidebb utak minden csúcspárra probléma visszavezetése a legrövidebb utak egy forrásból feladatra

- Ha a gráfunk élsúlyai nemnegatívak
  - a Dijkstra algoritmust minden $s \in 1..n$ értékre végrehajtva
  - $MT(n, m) \in O(n * (n + m) * \log n)$
  - Ritka gráfokra: $O(n^2 * \log n)$
    - Aszimptotikusan lényegesen jobb, mint az FW algoritmus $\Theta(n^3)$ műveletigénye
  - Sűrű gráfokra: $MT(n, m) \in O(n^3 * \log n)$
    - Aszimptotikusan rosszabb, mint az FW esetén
- Ha csak annyit tudunk, hogy nincs a gráfunkban negatív kör
  - QBF algoritmust minden $s \in 1..n$ értékre végrehajtva
  - $MT(n, m) \in O(n * n * m)$
    - a legrosszabb esetet tekintve, és feltételezve, hogy nincs lényegesen kevesebb él, mint csúcs a gráfban
    - sosem ad jobb felső becslést, mint amit az FW algoritmus esetén kaptunk

Gráf tranzitív lezártja (TC)

Feladat. egy hálózatban (gráfban) honnét hova lehet eljutni. Az eljutás módja és költsége most nem érdekel bennünket.

11. Definíció. A $G = (V, E)$ gráf *tranzitív lezártja* alatt a $T \subseteq V \times V$ relációt értjük, ahol $(u, v) \in T \Leftrightarrow$ ha G-ben az u csúcsból elérhető a v csúcs.

12. Jelölés. $\mathbb{B} = \{0; 1\}$

- A gráfot ábrázolása az $A/1 : \mathbb{B}[n, n]$ csúcsmátrixszal
  - Ha a gráfunk csúcsai az 1..n indexekkel azonosíthatók
  - (az esetleges élsúlyoktól eltekintve)
- A tranzitív lezártja ábrázolása a $T/1 : \mathbb{B}[n, n]$ mátrixszal
  - Ahol $T[i, j] \Leftrightarrow$ ha az A mátrixszal ábrázolt gráfban van út az i csúcsból a j csúcsba
  - A T mátrix kiszámításához definiáljuk a $<T^{(0)}, T^{(1)}, T^{(n)}>$ mátrixsorozatot, aminek utolsó tagja magával a T mátrixszal lesz egyenlő

13. Definíció. $T_{ij}^{(k)} \Leftrightarrow \exists\, i \overset{k}{\rightsquigarrow} j \; (k \in 0..n \wedge i, j \in 1..n)$

Gráf tranzitív lezártja (TC)

11. Tulajdonság. (A T mátrixok rekurzív megadása.)

- $T_{ij}^{(0)} = A[i,j] \vee (i = j) \quad (i,j \in 1..n)$
- $T_{ij}^{(k)} = T_{ij}^{(k-1)} \vee T_{ik}^{(k-1)} \wedge T_{kj}^{(k-1)} \; (k \in 1..n \wedge i,j \in 1..n)$

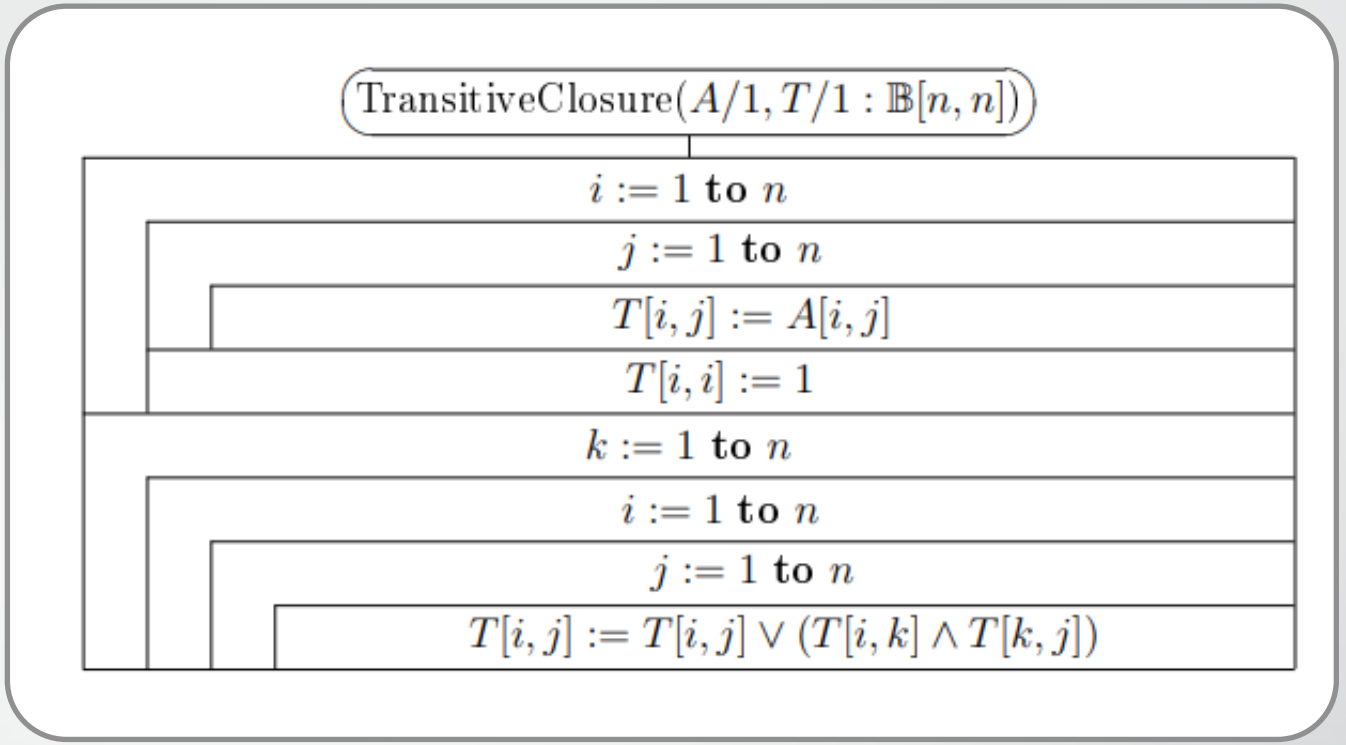
12. Következmény.

- $T_{ik}^{(k)} = T_{ik}^{(k-1)} \wedge T_{kj}^{(k)} = T_{kj}^{(k-1)} \; (k \in 1..n \wedge i,j \in 1..n)$
- Azaz a $T^{(k)}$ mátrix k-adik oszlopa és sora ugyanaz, mint a $T^{(k-1)}$ mátrixé. ($k \in 1..n$)

➢ a T mátrix kiszámításához nem kell segédmátrixokat tárolni, mert

- $T_{ij}^{(k)}$ csak a $T_{ij}^{(k-1)}$, a $T_{ik}^{(k-1)}$ és a $T_{kj}^{(k-1)}$ értékektől függ
- $T_{ik}^{(k)} = T_{ik}^{(k-1)}$, $T_{kj}^{(k)} = T_{kj}^{(k-1)}$

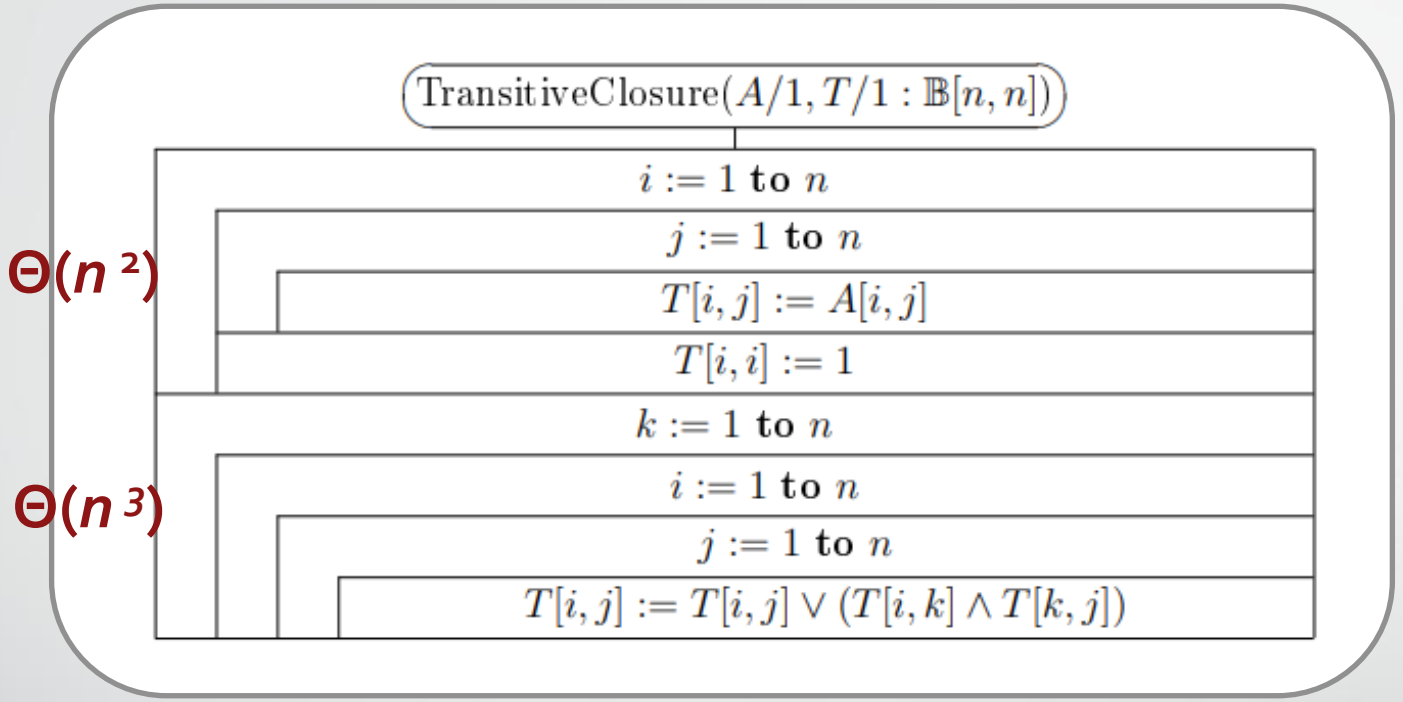
A tranzitív lezárt algoritmusa



- 1. kettős ciklusa
  - A fizikailag is létező *T* mátrixot az elméleti $T^{(0)}$ mátrix elemeivel inicializáljuk
- 2. háromszorosan beágyazott ciklusa
  - *k* = 1-től *n*-ig kiszámoljuk $T^{(k-1)}$-ből $T^{(k)}$ –t
  - Fizikailag végig ugyanazt a *T* mátrixot használva
    - *T*[*i*, *j*] új értékének kiszámolásakor az elméleti mátrixsorozat $T_{ij}^{(k)}$ elemét határozzuk meg
    - Az értékadás végrehajtása előtt még
      - $T[i, j] = T_{ij}^{(k-1)}$,
      - $T[i, k] = T_{ik}^{(k)} = T_{ik}^{(k-1)}$ és $T[k, j] = T_{kj}^{(k)} = T_{kj}^{(k-1)}$

➢ mindegy, hogy a második külső ciklus adott iterációján belül a *T*[*i*, *k*], illetve a *T*[*k*, *j*] már értéket kapott-e

A tranzitív lezárt algoritmus elemzése



- Műveletidő: $T(n) \in \Theta(n^3)$
- A TC és az FW kapcsolata: $T[i, j] \Leftrightarrow D[i, j] < \infty$ $(i, j \in 1..n)$
  - TC: egyszerűbb feladat, hatékonyabb megoldás (az egyszerűbb ciklusmagok miatt)

A tranzitív lezárt kiszámításának visszavezetése szélességi keresésre

- A szélességi keresés után
  - $s = i$ esetben éppen azok a csúcsok lesznek feketék, amelyek elérhetők i-ből
- A szélességi keresés egyszerűsített változatát alkalmazva
  - ahol $T[i, j] \Leftrightarrow \text{color}(j) \neq \text{white} \Rightarrow$ a T mátrix megfelelő sorát kapjuk meg
  - $MT(n, m) \in \Theta(n + m)$ műveletigénnyel
- A mátrix összes sorát így $\Theta(n * (n + m))$ maximális futási idővel kapjuk meg
  - Ritka gráfok esetén $\Theta(n^2)$
    - azaz aszimptotikusan jobb, mint a TC algoritmus
  - Sűrű gráfok esetén $\Theta(n^3)$
    - ami a gyakorlatban hosszabb futási időt eredményez, a nagyon egyszerű TC algoritmussal összehasonlítva

Ellenőrző kérdések

1. Mit számol ki a Floyd-Warshall algoritmus? Mekkora a műveletigénye n csúcsú gráf esetén? Miért?
   - Szemléltessük a működését az alábbi irányított gráfon a $(D^{(0)}; \pi^{(0)}); \ldots ; (D^{(3)}; \pi^{(3)})$ mátrix párok megadásával!
   - Rajzoljuk le a legrövidebb utak fáit, amiket az eredményből kiolvashatunk!
     - 1-> 3, 1. 2-> 1, 0 ; 3, 2. 3-> 1, 1 ; 2, 2.
2. Mit jelent egy gráf tranzitív lezártja, amit Warshall algoritmusa számol ki?
   - Tegyük fel, hogy a gráfnak *n* csúcsa van! Mi a $\langle T^{(k)}: k \in 2\ 0..n \rangle$ mátrix sorozat definíciója? Mi a rekurzív képlete?
   - Adja meg az algoritmus struktogramját! Mekkora a műveletigénye n csúcsú gráf esetén? Miért elegendő egyetlen T mátrix a programban?
   - Mutassa be az algoritmus működését a 4 -> 3. 3->2. 2 -> 1 ; 4. irányított gráfon!

Köszönöm a figyelmet!

Pusztai Kinga

A bemutató Ásványi Tibor: Algoritmusok és adatszerkezetek II. eladásjegyzet:Élsúlyozott gráfok és algoritmusaik alapján készült.