Mohó algoritmusok

Optimalizálási probléma megoldására szolgáló algoritmus sokszor olyan lépések sorozatából áll, ahol minden lépésben adott halmazból választhatunk. Ezt gyakran dinamikus programozás alapján oldjuk meg, de előfordul, hogy a dinamikus programozás túl sok esetet vizsgál annak érdekében, hogy az optimális választást meghatározza.

A mohó algoritmus mindig az adott lépésben optimálisnak látszó választást teszi. Vagyis, a lokális optimumot választja abban a reményben, hogy ez globális optimumhoz fog majd vezetni.

Mohó algoritmus nem mindig ad optimális megoldást, azonban sok probléma megoldható mohó algoritmussal.

Egy ütemezési feladat

Feladat Vegyük azt az ütemezési modellt, ahol adottak az $\{1,\ldots,n\}$ munkák továbbá a j munkának van egy p_j végrehajtási ideje és egy w_j súlya. A cél a munkáknak egy olyan sorrendjét megadni, amely sorrendre a befejezési idők súlyozott összege minimális.

Példa $(p_1 = 2, w_1 = 2)$, $(p_2 = 4, w_2 = 3)$, $(p_3 = 2, w_3 = 1)$.
Ekkor az $1,2,3$ sorrendre a befejezési idők $2,6,8$ és a célfüggvény $2\cdot 2+6\cdot 3+8\cdot 1 = 30$.
A $2,1,3$ sorrendre a befejezési idők $4,6,8$ és a célfüggvény $4\cdot 3+6\cdot 2+8\cdot 1 = 32$.
Mohó megoldások:

- Elsőnek azt hajtsuk végre, akinek kicsi a végrehajtási ideje.
- Elsőnek azt hajtsuk végre, akinek nagy a súlya.

Egyik sem ad optimális algoritmust.

Tétel A munkákat a p_j/w_j érték szerint monoton növekvő sorrendbe rendezve egy optimális sorrendet kapunk.
Lemma Van olyan optimális megoldás, amelyben az elsőnek végrehajtott munkára minimális a p_j/w_j érték.

Bizonyítás: Vegyük azt az optimális megoldást, amelyben a legkorábban szerepel olyan munka, amelyre a p_j/w_j érték minimális. Legyen ez a megoldás OPT és tegyük fel, hogy az i-edik munka az első olyan, amire a p_j/w_j érték minimális. Ha $i = 1$, akkor a lemma teljesül.

Ellenkező esetben legyen k és r OPT i-1-edik és i-edik munkája. Vegyük azt az OPT' megoldást, amelyet úgy kapunk OPT-ból, hogy felcseréljük k és r sorrendjét. Ekkor OPT célfüggvényértékéből kivonva OPT' célfüggvényértékét a

$$w_k p_k + w_r(p_k + p_r) - w_r p_r - w_k(p_r + p_k) = w_r p_k - w_k p_r > 0$$

értéket kapjuk, ami ellentmond OPT optimalitásának.

Lemma Egy optimális megoldás tetszőleges végszelete optimális megoldása annak az ütemezési feladatnak, amelyben az input a végszeletben szereplő munkák halmaza.

Bizonyítás Ha nem lenne optimális megoldás, akkor egy jobb megoldást kiegészítve a hiányzó kezdőszelettel, a kiinduló megoldásnál jobb megoldását kapnánk az eredeti problémának.

Egy másik ütemezési feladat

Feladat Vegyük azt az ütemezési problémát, ahol egy gép van, minden munkának van egy végrehajtási ideje és egy határideje. A cél a maximális késedelem (befejezési időből kivont határidő) minimalizálása.

Mohó algoritmus: a munkákat határidő szerint rendezzük és ebben a sorrendben hajtjuk végre.

Tétel A Mohó algoritmus optimális megoldást ad.

Lemma Van olyan optimális megoldás, amelyben az elsőnek végrehajtott munkának a legkisebb a határideje.

Bizonyítás: Tekintsünk egy olyan ütemezést legyen S, amely nem egy legkisebb határidővel rendelkező munkát ütemez elsőnek. Legyen a legkisebb határidővel rendelkező munka j.

Tegyük fel, hogy S valamilyen $i_1, i_2, \ldots, i_k, j$ munkasorrenddel kezdődik. Vegyük S'-t, amelyet úgy kapunk S-ből, hogy a munkák ezen kezdeti sorrendjét kicseréljük a $j, i_1, i_2, \ldots, i_k$ sorrendre. Az új ütemezésre az $i_1, \ldots, i_k$ munkák később fejeződnek be, viszont a befejezési idejük nem lesz nagyobb, mint j-nek az S ütemezésben, ezért a késedelmük sem, hiszen j határideje minimális volt. Következésképpen az így kapott megoldás is optimális lesz, amivel a lemmát igazoltuk.

Gráfok, definíciók

Irányítatlan gráf: G = (V,E), ahol E rendezetlen $(a,b), a,b \in V$ párok halmaza.

Irányított gráf: G = (V,E) E rendezett (a,b) párok halmaza; $E \subseteq V \times V$.

Címkézett (súlyozott) gráf: G = (V,E,C) $C : E \to$ Címke

Minden irányítatlan G = (V,E) gráf olyan irányított gráfnak tekinthető, amelyre teljesül, hogy ha $(p,q) \in E$ akkor $(q,p) \in E$.

Jelölések:

- $Ki(G,p) = \{q \in V : (p,q) \in E\}$
- $Be(G,p) = \{q \in V : (q,p) \in E\}$
- $KiFok(G,p) = |Ki(G,p)|$
- $BeFok(G,p) = |Be(G,p)|$

Gráf absztrakt adattípus

Értékhalmaz: $Graf = \{G = (V,E) : V \subseteq PontTip, E \subseteq V \times V\}$ Műveletek: G : Graf , p, p1, p2 : PontTip, ir:boolean, I : Iterator,

- $\{Igaz\}$ Letesit(G,ir) $\{G = (\emptyset, \emptyset)\}$
- $\{G = G\}$ Megszuntet(G) $\{Igaz\}$
- $\{G = G\}$ Uresit(G) $\{G = (\emptyset, \emptyset)\}$
- $\{G = G\}$ Iranyitott(G) $\{NotIranyitott(G) \to ((p,q) \in E) \to (q,p) \in E)\}$
- $\{G = G\}$ Pontokszama(G) $\{= |V|\}$
- $\{G = G\}$ Elekszama(G) $\{= |E|\}$
- $\{G = G\}$ KiFok(G, p) $\{= |KI(G,p)|\}$
- $\{G = G\}$ BeFok(G, p) $\{= |Be(G,p)|\}$
- $\{G = (V,E)\}$ PontBovit(G, p) $\{V = Pre(V) \cup \{p\} \wedge E = Pre(E)\}$
- $\{G = (V,E) \wedge p \in V\}$ PontTorol(G, p) $\{V = Pre(V) \setminus \{p\} \wedge E = Pre(E) \setminus \{(p,q) : q \in Ki(G,p)\} \setminus \{(q,p) : q \in Be(G,p)\}\}$
- $\{G = (V,E), p1, p2 \in V\}$ ElBovit(G, p1, p2) $\{E = Pre(E) \cup \{(p1,p2)\} \wedge V = Pre(V)\}$

- $\{G=(V,E), p1, p2 \in V\}$ ElTorol(G, p1, p2) $\{E = Pre(E) \setminus \{(p1,p2)\} \wedge V = Pre(V)\}$
- $\{G=(V,E), p1, p2 \in V\}$ Vanel(G, p1, p2) $\{= (p1,p2) \in E \wedge E = Pre(E) \wedge V = Pre(V)\}$
- $\{G=G\}$ PIterator(G, I) $\{\}$
- $\{G=G\}$ KiEl(G, p) $\{= \{q : VanEl(p,q)\}\}$
- $\{G=G\}$ KiIterator(G, p) $\{\}$
- $\{G=G\}$ ElIterator(G, I) $\{\}$

Az absztrakt adattípus meg van valósítva JAVA nyelven, megvalósítása a pub konyvtárban megtalálható.

Címkézett Gráf adattípus

Értékhalmaz: $Graf = \{G = (V,E,C) : V \subseteq PontTip, E \subseteq V \times V, C : E \to CimkeTip\}$.
Műveletek: Graf-műveletek + G : Graf , P,P1,P2 : PontTip, S : CimkeTip, I : PIterator,

- $\{G=(V,E), p1, p2 \in V\}$ ElBovit(G,p1,p2,s) $\{E = Pre(E) \cup \{(p1,p2)\} \wedge C(p1,p2) = s\}$
- $\{G=(V,E), (p1,p2) \in E\}$ ElCimke(G,p1,p2,s) $\{s = Pre(C)(p1,p2) \wedge E = Pre(E)\}$
- $\{G=(V,E), (p1,p2) \in E\}$ ElCimkez(G,p1,p2,s) $\{s = C(p1,p2) \wedge E = Pre(E)\}$
- $\{G=G\}$ KiCEl(G, p) $\{= \{(q,s) : VanEl(p,q) \wedge C(p,q) = s\}\}$
- $\{G=G\}$ ElIterator(G, I) $\{\}$

Ha a CimkeTip típuson alapértelmezett lineáris rendezési reláció, akkor a címkézett gráfot súlyozott gráfnak nevezzük.

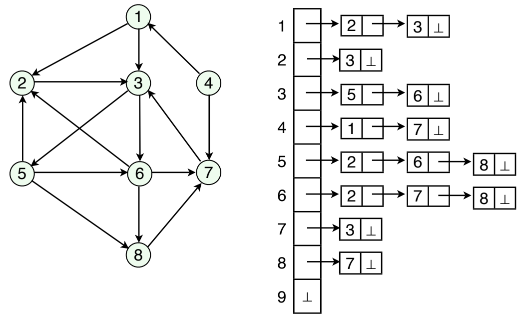
Gráfok ábrázolása

Szempontok az adatszerkezet megválasztásához.

- Az adott probléma megoldásához ténylegesen mely műveletek szükségesek.
- Melyek a releváns műveletek, amelyek alapvetően befolyásolják az algoritmus futási idejét.
- A tárigény az adott probléma esetén.

Ábrázolások

- Élhalmazlánc vagy élhalmaztömb: az élek felsorolása. Tárigény: $\Theta(|E|)$, VANEL időigénye: $\Theta(|E|)$, El-iteráció időigénye: $\Theta(|E|)$, Ki-iteráció időigénye: $\Theta(|E|)$.
- Kapcsolatmátrix (szomszédsági) mátrix: $G[i,j] = 1$ ha $(i,j) \in E$, címkézett gráf esetén G a címkéket tárolhatja. Tárigény: $\Theta(n^2)$, ahol $|V| = n$, VANEL időigénye: $\Theta(1)$, El-iteráció időigénye: $\Theta(n^2)$, Ki-iteráció időigénye: $\Theta(n)$.
- Statikus éllista: egy n elemű tömb, amelynek az i-dik eleme a $Ki(G,i)$ halmazt tartalmazza láncként. Tárigény: $\Theta(|E|+n)$, VANEL időigénye: $O(n)$, El-iteráció időigénye: $\Theta(|E|)$, Ki-iteráció időigénye: $\Theta(|Ki(G,i)|)$.



1. ábra.

Példa:

Számított gráf

Az élek halmazát explicite nem tároljuk, mert van olyan számítási eljárás, amely bármely két $p,q \in V$-re kiszámítja $VanEl(p,q)$-t. Vagy, van olyan számítási eljárás, amely minden $p \in V$-re kigenerálja a $Ki(G,p)$ halmaz elemeit.

Alapvető definíciók

Legyen G = (V,E) irányított vagy irányítatlan gráf.

Definíció $p,q \in V$-re egy p-ből q-ba vezető út G-ben, jele: $\pi : p \rightsquigarrow q$, olyan $\pi = \{p_0, p_1, \ldots, p_k\}$ pontsorozat, ahol $p_i \neq p_j$, ha $i \neq j$, $p = p_0$ és $q = p_k$, továbbá $p = q = p_0$, vagy $(\forall i \in \{1,\ldots,k\})((p_{i-1}, p_i) \in E)$

A $\pi = p \rightsquigarrow q$ út hossza, $|\pi| = |p \rightsquigarrow q| = k$. Ha G = (V,E,C) élei a $C : E \to R$ függvénnyel súlyozottak, akkor a $p \rightsquigarrow q$ út hossza $|p \rightsquigarrow q| = \sum_{i=1}^{k} = C(p_{i-1}, p_i)$.

Definíció p-ből q-ba vezető legrövidebb út hossza, p és q távolsága: $\delta(p,q) = \infty$, ha nincs $p \rightsquigarrow q$ út és $Min\{|\pi| : \pi : p \rightsquigarrow q\}$ egyébként.

Definíció A $p = \{p_0, p_1, \ldots, p_k\}$ pontsorozatot a p_0-ból p_k-ba vezető sétának nevezzük, ha $(\forall i \in \{1,\ldots,k\})(p_{i-1}, p_i) \in E$.

Definíció A G=(V,E) irányított vagy irányítatlan gráfnak az $F = (\bar{V}, \bar{E})$ gráf a $r \in V$ gyökerű feszítőfája, ha

- F részgráfja G-nek ($\bar{V} \subseteq V, \bar{E} \subseteq E$) és fa.
- $(\forall p \in V)$ akkor és csak akkor van $r \rightsquigarrow p$ G-ben, ha van $r \rightsquigarrow p$ F-ben.

Definíció A G = (V,E) irányított vagy irányítatlan, esetleg súlyozott gráfnak az $F = (\bar{V}, \bar{E})$ gráf a $r \in V$ gyökerű legrövidebb utak feszítőfája (LUF), ha

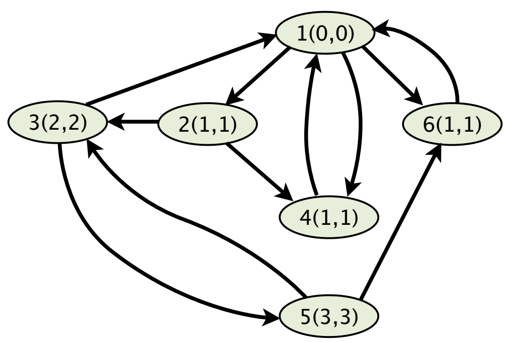
- F r-gyökerű feszítőfája G-nek, és
- $\forall p \in V$-re $\delta_G(r,p) = \delta_F(r,p)$.

Szélességi keresés

Egy adott súlyozatlan irányított vagy irányítatlan gráf egy pontjából keressük az elérhető pontokat, és az azokhoz vezető legrövidebb utakat.

Bemenet: a $G = (V,E)$ gráf és egy kiindulási pont $r \in V$. Kimenet: a legrövidebb utak fája, egy Apa függvény által megadva, és egy d függvény, amelyre $d(p) = \delta(p)$ minden $p \in V$-re.

```
SZELTKERES
 for(p in V)
     {Apa(p):=-1
      d(p):=INF}
 d(r):=0
 Apa(r):=0
 Letesit(S:Sor)
 Sorba(S,r)
 While(Elemszam(S)>0)
     {Sorbol(S,u)
      for(v in Kiel(G,u))
         {if Apa(v)<0
             Then {Apa(v):=u
                   d(v):=d(u)+1
                   Sorba(S,v)}}}
```



2. ábra.

$S = [1], u = 1, S = [], v = 2, Apa(2) = 1, d(2) = 1, S = [2], v = 4, Apa(4) = 1, d(4) = 1,$
$S = [2,4], v = 6, Apa(6) = 1, d(6) = 1, S = [2,4,6], u = 2, S = [4,6], v = 3, Apa(3) = 2,$
$d(3) = 2, S = [4,6,3], v = 4, u = 4, S = [6,3], v = 1, u = 6, S = [3], v = 1, u = 3, S = [],$
$v = 1, v = 5, Apa(5) = 3, d(5) = 3, u = 5, S = [], v = 3, v = 6$

Helyesség

Lemma Legyen G = (V,E) irányított vagy irányítatlan gráf és $s,u,v \in V$. Ekkor minden $(u,v) \in E$ élre $\delta(s,v) \leq \delta(s,u) + 1$.

Lemma Ha SZELTKERES algoritmust az r pontra alkalmazzuk, akkor a kiszámított d-re teljesül: $(\forall v \in V)(d[v] \geq \delta(r,v))$.

Bizonyítás. Az S sorba kerülés szerinti indukcióval.

Lemma Legyen a SZELTKERES algoritmust végrehajtásának egy pillanatában az S sor tartalma $S = [v_1, v_2, \ldots, v_k]$. Ekkor $d(v_k) \leq d(v_1) + 1$ és $d(v_i) \leq d(v_{i+1}) (i = 1, \ldots, k-1)$

Tétel $(\forall v \in V)(\delta(r,v) = d(v))$

Bizonyítás

- Ha $\delta(r,v) = \infty$, akkor nem kerül be v a sorba.
- Egyébként legyen v a legkisebb olyan $\delta(r,v)$ értékű pont, hogy $\delta(r,v) \neq d(v)$, és u egy olyan pont, amely közvetlenül megelőzi v-t az $r \rightsquigarrow v$ legrövidebb úton. Ekkor a lemmák alapján $d(v) > \delta(r,v) = \delta(r,u) + 1 = d(u) + 1$, amivel ellentmondáshoz jutunk megvizsgálva az algoritmus viselkedését az (u,v) él vizsgálata során.

Következmény Az algoritmus egy r gyökerű legrövidebb utak feszítőfát ad vissza.

Futási idő Az algoritmus futási ideje $O(V + E)$ ha a ki iteráció lineáris idejű (pl éllistás esetben).

Példa Adott egy sziget ahol 42 kaméleon lakik. A kaméleonok három színt vehetnek fel piros, kék és zöld. Ha két különböző színű kaméleon találkozik, akkor megijednek és mindkettő átváltoztatja a színét a harmadik színre. A szigeten a kiindulási időpontban 13,14,15 darab piros zöld és kék kaméleon van. Előfordulhat -e, hogy a szigeten minden kaméleonnak ugyanaz legyen a színe?

Megoldás: Definiáljunk egy gráfot, a gráf pontjai az (x,y,z) számhármasok, ahol minden komponens nemnegatív egész és $x+y+z = 42$. Az élek a lehetséges átváltozásoknak felelnek meg. Tehát (x,y,z)-nek három szomszédja lesz $(x-1,y-1,z+2)$, $(x-1,y+2,z-1)$, $(x+2,y-1,z-1)$. A feladat, hogy meghatározzuk van -e a gráfban a $(13,14,15)$ pontból a $(42,0,0)$, $(0,42,0)$, $(0,0,42)$ pontok valamelyikébe út. Ezt megoldhatjuk szélességi kereséssel.

Példa Adott egy $n \times n$-es sakktábla. Az $(1,1)$ mezőn áll egy huszár. Határozzuk meg eljuthat -e az (u,v) mezőre, ha igen adjunk meg egy legkevesebb lépésből álló utat! Adjunk algoritmust, ami megoldja a feladatot.

Megoldás Konstruáljunk a feladathoz egy gráfot. A gráf csúcsai (a lehetséges konfigurációk) a sakktábla mezői, két n-nél nem nagyobb pozitív koordinátából álló vektorok. Két csúcs össze van kötve, ha a huszár átléphet egyikből a másikba. Tehát az (x,y) szomszédjai az $(x-1,y+2)$, $(x-1,y-2)$, $(x+1,y+2)$, $(x+1,y-2)$, $(x+2,y-1)$, $(x+2,y+1)$, $(x-2,y-1)$, $(x-2,y+1)$ párosok közül azok, amelyekben mindkét koordináta pozitív és nem nagyobb, mint n. A feladat megtalálni a legrövidebb utat az $(1,1)$ pontból az (u,v) pontba. Ezt egy szélességi kereséssel, amit az $(1,1)$ pontból indítunk a fenti gráfban megtaláljuk.

Megjegyzés Ha adott mezők egy T halmaza, ahova nem léphet, akkor is hasonlóan oldható meg a feladat. Ez esetben csak a megengedett mezők a gráf pontjai.

Példa

Erdős szám kiszámítása. Erdős Pál matematikus tiszteletére definálták az Erdős szám fogalmát. Erdős Pál Erdős száma 0, azoknak a matematikusoknak, akiknek van közös cikkük Erdőssel az Erdős számúk 1. Azoknak, akiknek nincs közös cikkük Erdős Pállal, de van közös cikkük olyan matematikussal, akinek az Erdős száma 1, az Erdős száma 2. Általában egy matematikus Erdős száma a társszerzői Erdős számának minimuma +1. Adjunk algoritmust, amely a cikkek adatbázisa alapján (feltételezhetjük, hogy adott egy TÁRS algoritmus, ami minden matematikusra visszaadja a társszerzői halmazát) meghatározza egy adott személy Erdős számát. A futási idő lineáris kell legyen $R+n$-ben, ahol n az adatbázisban a matematikusok száma R pedig a $t(i)$ értékek összege, ahol $t(i)$ az i személy társsszerzőinek száma.

Definiálunk egy gráfot, amelyben midnenkinek a társszerzői a szomszédjai. Ebben keressük a legrövidebb Erdős szerző utat.

Kiskérdések a jövő hétre
Kiskérdések a ZH utáni hétre

- Szintszerinti bejárás
- Eseményválasztás algoritmusa
- Huffman kód algoritmusa
- Huffman kód algoritmusának végrehajtása, az optimális fa felépítése, a kódok megadása egy megadott példán