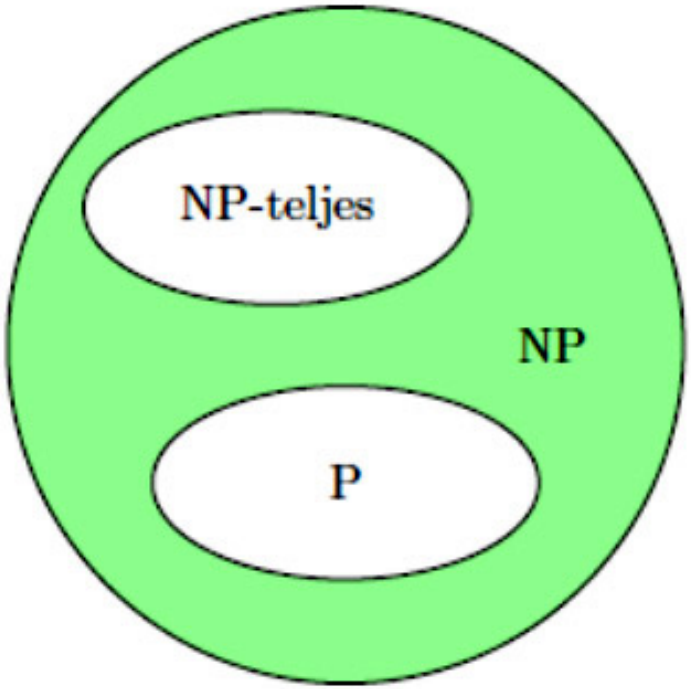
Algoritmusok bonyolultsága

5. előadás

http://www.ms.sapientia.ro/~kasa/komplex.htm

Bonyolultság mérése

- polinomiális feladat ($\mathcal{P}$)
- nemdeterminisztikusan polinomiális feladat ($\mathcal{NP}$)



$\mathcal{P} = \mathcal{NP}$ vagy $\mathcal{P} \neq \mathcal{NP}$?

LP lineáris programozási feladat
Dantzig 1945, Kacsiján (Khachiyan) 1979, Karmarkar 1984

Bonyolultság mérése

$\mathcal{NP}$-teljes feladatok:

- **logikai formulák kielégíthetősége**
  logikai formula konjunktív normál alakja (CNF):
  $(p \vee q \vee r) \wedge (\bar{q} \vee s) \wedge (\bar{r} \vee \bar{s})$, *másképpen:* $\left\{\{p, q, r\}, \{\bar{q}, s\}, \{\bar{r}, \bar{s}\}\right\}$
- **hátizsák probléma**
  Adott n tárgy (s_i tömeggel) és egy K kapacitású hátizsák.
  Töltsük meg minél jobban a zsákot!
- **ládapakolás (csomagolási probléma) (bin packing)**
  Adott n tárgy (s_i tömeggel, $0 < s_i \leq 1$) és végtelen sok 1 kapacitású láda (doboz). Helyezzük el a tárgyakat minél kevesebb ládába!
- **Hamilton-út keresése**
  Egy adott sólyozott gráfben keressünk minél rövidebb Hamilton-utat!
- **lefedőszó probléma**
  Keressük meg azt a legrövidebb szót, amely tartalmaz részszóként n adott szót! (pl. *malom, alma* esetében *almalom*)

Közelítő algoritmusok

NP-teljes feladatok esetében két lehetőségünk van:

- kicsi n-re a feladat exponenciális megoldása eredményes,
- nagy n-re megelégszünk olyan algoritmussal, amelyik megközelíti az optimális megoldást.

közelítő algoritmus

Ha C a közelítő algoritmus eredménye, C^* az optimális megoldás (minden n méretű bemenetre), és

$$\max\left(\frac{C}{C^*}, \frac{C^*}{C}\right) \leq \rho(n),$$

akkor $\rho(n)$ a közelítő algoritmus hibakorlát-függvénye.

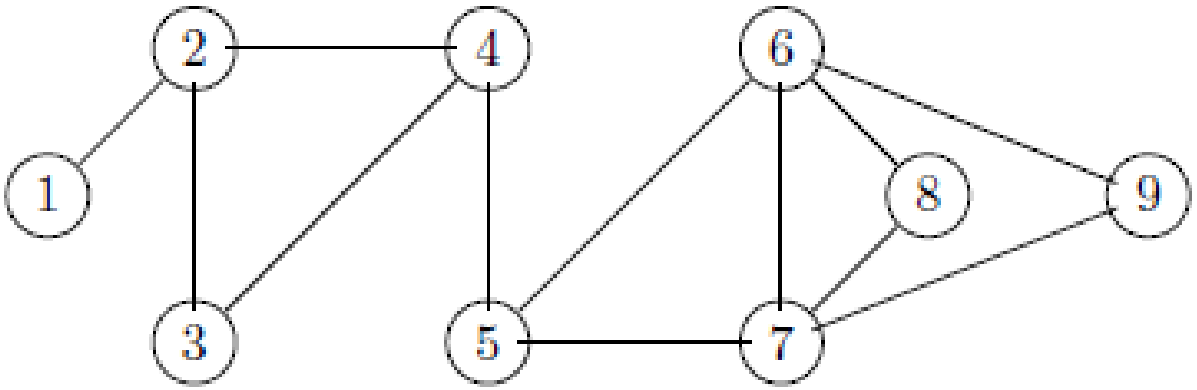
$\rho(n)$-közelítő algoritmus

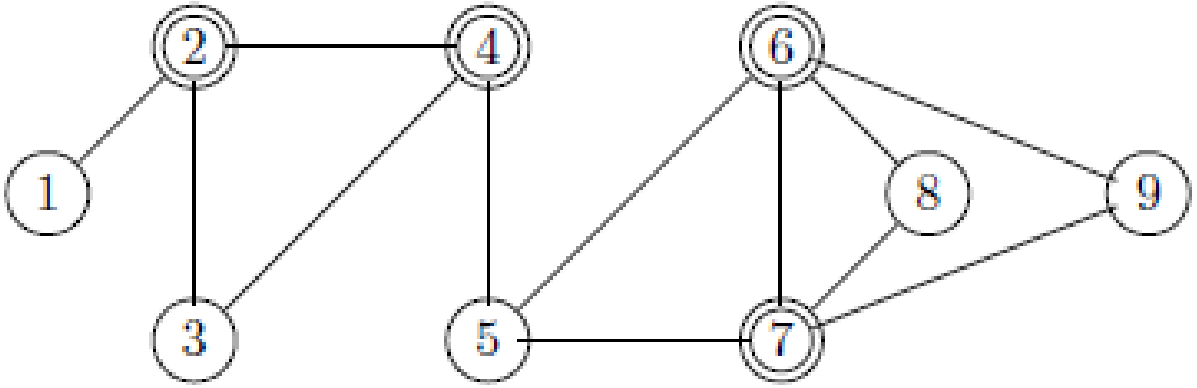
Közelítő séma

A közelítő séma olyan közelítő algoritmus, amelynek hatékonysága növelhető. Adott $\varepsilon > 0$ értékre az algoritmus $(1 + \varepsilon)$-közelítő algoritmus.
Polinomiális (idejű közelítő) séma, ha a futási ideje a bemenet méretére nézve polinomiális.

Minimális lefedő (vagy lefogó) csúcshalmaz

Határozzuk meg azt a minimális csúcshalmazt, amely elemeihez a gráf minden éle illeszkedik.





optimális lefedés (4 csúcs)

Példa

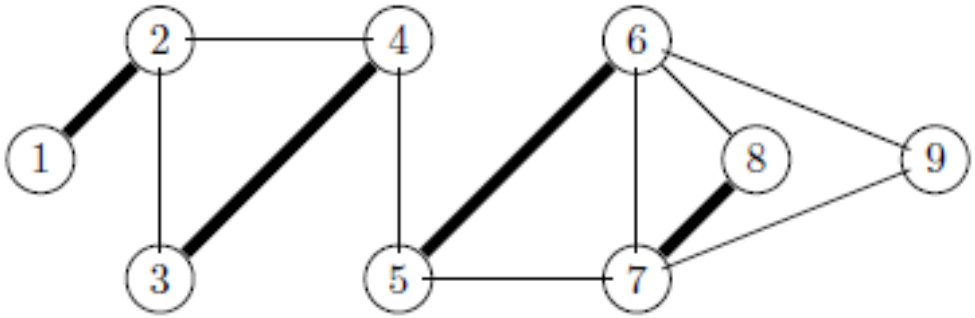
NP-teljes feladat — közelítő algoritmust keresünk

MINIMÁLIS-CSÚCSLEFEDÉS(G)

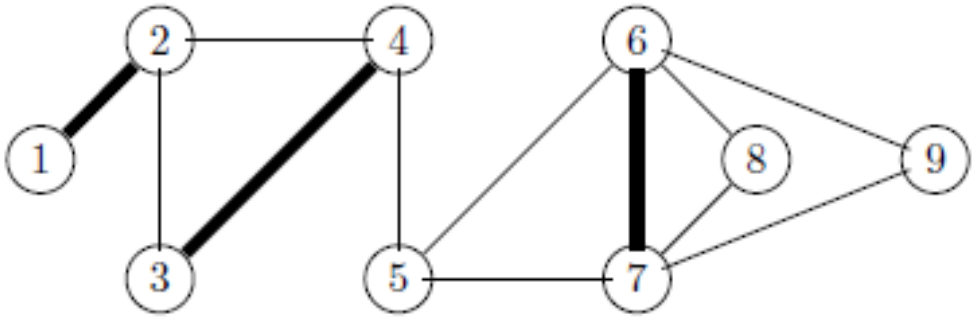
1. $C := \emptyset$
2. while $E(G) \neq \emptyset$
3. do legyen $\{u, v\}$ egy él $E(G)$-ből
4. $C := C \cup \{u, v\}$
5. távolítsuk el $E(G)$ minden u-hoz vagy v-hez illeszkedő élét
6. return C

C a kapott lefedő csúcshalmaz.
Bonyolultsága: $O(m)$, ahol m az élek száma (vagy $O(n^2)$).

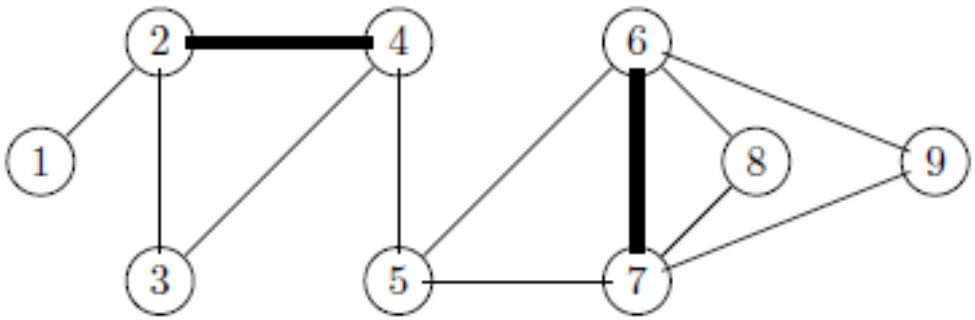
Példa



közelítő alg. (8 csúcs)



közelítő alg. (6 csúcs)



közelítő alg. (4 csúcs)

Az algoritmus elemzése

Az algoritmus legfeljebb kétszer annyi csúcsot ad meg, mint amennyit az optimális megoldás.

Tétel

A MINIMÁLIS-CSÚCSLEFEDÉS *algoritmus polinomiális 2-közelítő algoritmus.*

Bizonyítás:
Legyen A az az élhalmaz, amelyet az algoritmus 3. sora sorra vizsgál. Az A elemei függetlenek (nincs közös csúcsuk), végpontjai közül egyiknek benne kell lennie az optimális megoldásban is. Ezért $|C^*| \geq |A|$. De az algoritmus minden A-beli él mindkét végpontját beteszi a C megoldásba, tehát $|C| = 2|A|$. Innen

$$|C| = 2|A| \leq 2|C^*| \qquad \text{qu.e.d.}$$

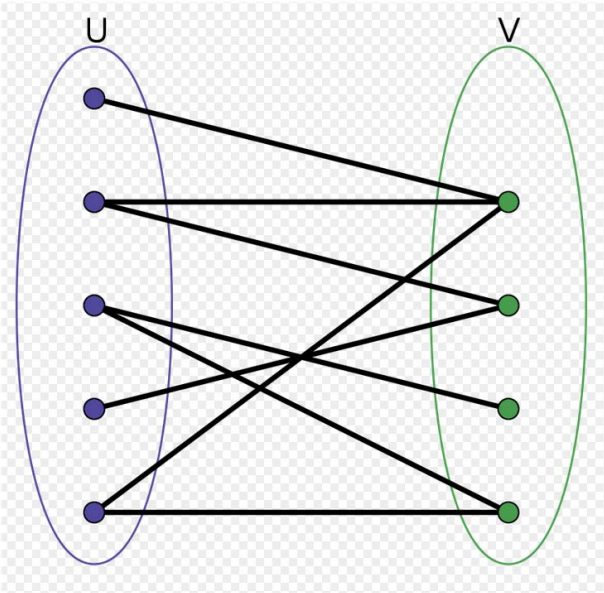
maxmatch(G) maximális párosítás éleinek száma
mincov(G) minimális lefedő csúcsok száma

Tétel

Tetszőleges G gráfban $maxmatch(G) \leq mincov(G)$

Tétel (Kőnig)

Ha G páros, akkor $maxmatch(G) = mincov(G)$



párosítás – ang. *matching*, rom. *cuplaj*
páros gráf – ang. *graph bipartite*, rom. *graf bipartit*

Az utazóügynök-feladat – Traveling Salesman Problem (TSP)

románul: Problema comis-voiajorului

$G = (E, V)$ nemirányított teljes gráf
$c(u, v) \in \mathbf{N}$ az $\{u, v\}$ él költsége

Az A élhalmaz költsége: $c(A) = \sum_{\{u,v\} \in A} c(u, v)$

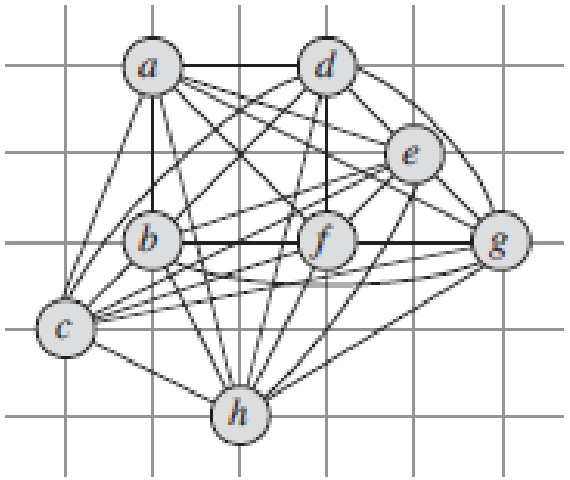
Háromszög-egyenlőtlenség:

Minden u, v, w csúcsra: $c(u, v) + c(v, w) \geq c(u, w)$.

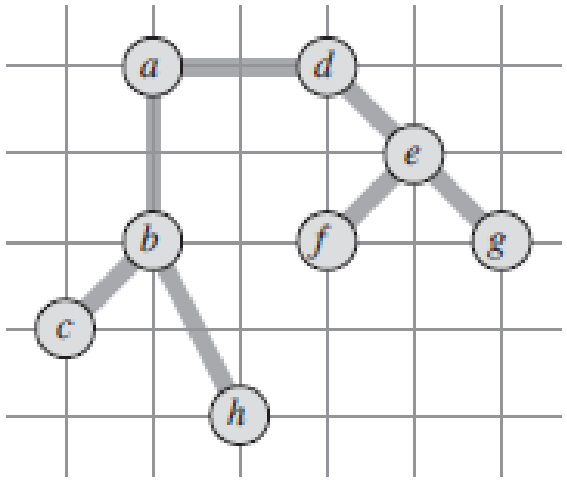
UTAZÓ-ÜGYNÖK(G, c)

1. Válasszunk ki egy $r \in V(G)$ csúcsot (gyökércsúcs).
2. Határozzuk meg G egy r gyökerű T minimális feszítőfáját (PRIM-algoritmus).
3. Legyen H a T fa csúcsainak listája preorder bejárás sorrendjében.
4. return H (a Hamilton-kör csúcsai)

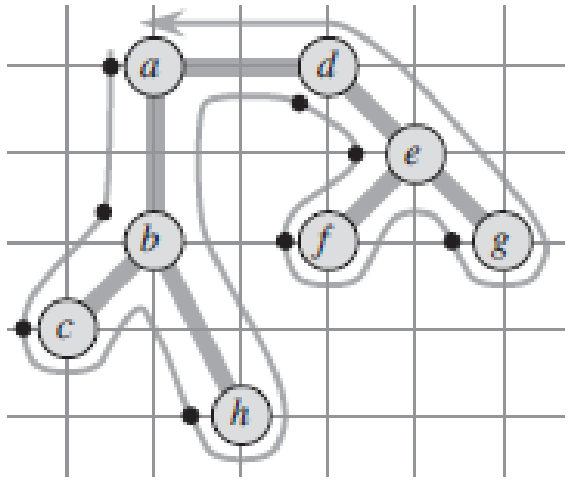
Példa



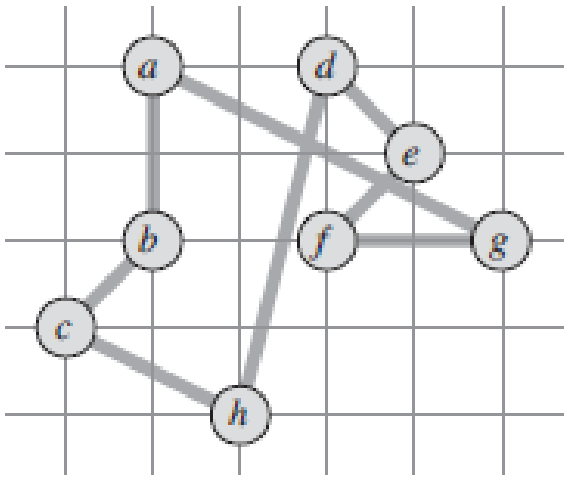
(a)



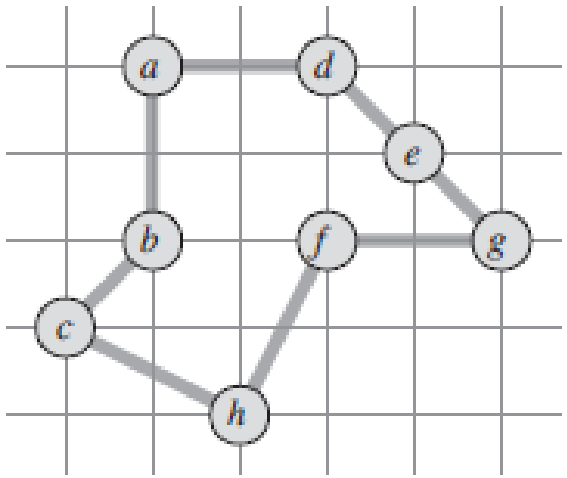
(b)



(c)



(d)



(e)

19,074 optimális: 14,715

Cormen–Leiserson–Rivest–Stein

Tétel

Ha a háromszög-egyenlőtlenség teljesül, akkor az UTAZÓ-ÜGYNÖK*-algoritmus polinomiális 2-közelítő algoritmus.*

Bizonyítás:
Az algoritmus polinomiális (Prim).
Legyen H^* egy optimális Hamilton-kör csúcsainak halmaza.
Ha T minimális feszítőfa, akkor $c(T) \leq c(H^*)$.
T teljes bejárása felsorolja a csúcsokat mind az első látogatáskor, mind a visszatéréskor. Legyen W egy ilyen teljes bejárás. Példánkban ez: $a, b, c, b, h, b, a, d, e, f, e, g, e, d, a$. Ez a bejárás T minden élén kétszer megy át, ezért:

$$c(W) = 2c(T).$$

Ekkor:

$$c(W) \leq 2c(H^*).$$

W nem mindig kör, de a háromszög-egyenlőtlenség miatt kihagyhatunk egy-egy közbeeső csúcsot, az összérték nem lesz rosszabb. Példánkban például kihagyunk minden csúcsot, amikor másodjára jelenik meg. Ezt kapjuk: a, b, c, h, d, e, f, g, és ez pont H. Tehát

$$c(H) \leq c(W) \leq 2c(H^*).$$

qu.e.d.

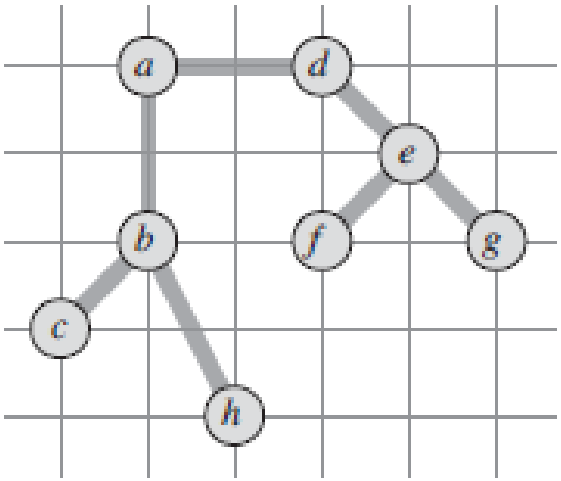
Az algoritmus szerzői: D. J. Rosenkrantz, R. E. Stearn és P. M. Lewis (konferencia:1974, cikk: 1977).
Nikosz Krisztofidesz (Nikos Christofides) (1976) javított rajta: nála a konstans 2 helyett $\frac{3}{2}$.

Christofides-algoritmus

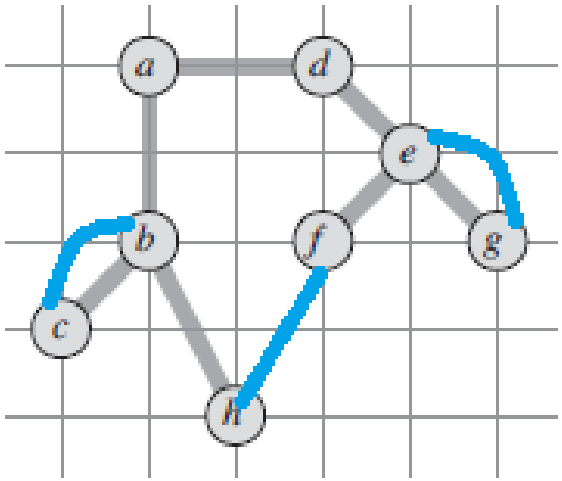
CHRISTOFIDES(G, c)

1. A G gráfban keressünk egy T minimális feszítőgráfot.
2. Jelöljük P-vel a T páratlan fokú csúcsait.
3. Határozzunk meg az eredeti teljes gráf P csúcsain egy M teljes minimális párosítást.
4. Képezzük M és T egyesítésével a H (multi)gráfot.
5. Keressünk H-ban egy zárt Euler-vonalat (ez mindig létezik!)
6. Képezzünk a zárt Euler-vonalból Hamilton-kört (háromszög-egyenlőtlenség segítségével).
7. return Hamilton-kör

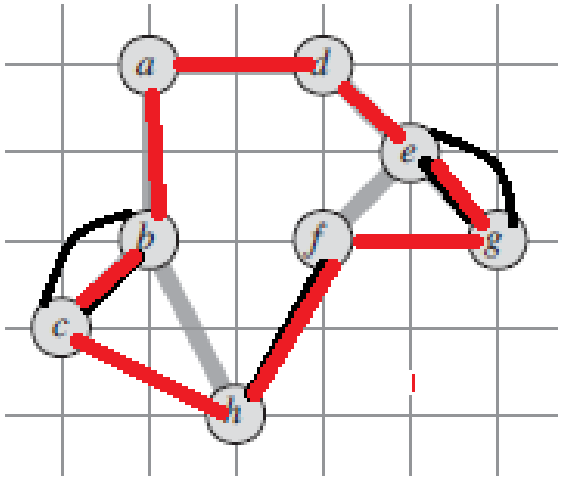
Példa



T feszítőfa



$T \cup M$



Hamilton-kör

Halmazlefedés

Legyen $(X, \mathcal{F})$, ahol X véges halmaz és $\mathcal{F} \subseteq \mathcal{P}(X)$ úgy, hogy

$$X = \bigcup_{S \in \mathcal{F}} S.$$

Lefedési feladat: Határozzunk meg egy olyan minimális méretű $\mathcal{C} \subseteq \mathcal{F}$ részhalmazcsaládot, amelyre

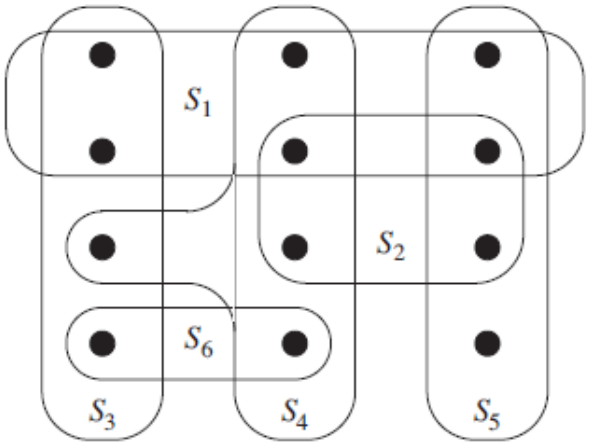
$$X = \bigcup_{S \in \mathcal{C}} S.$$

Mohó közelítő algoritmus

MINIMÁLIS-LEFEDÉS$(X, \mathcal{F})$

1. $U := X$
2. $\mathcal{C} := \emptyset$
3. while $U \neq \emptyset$
4. do Válasszunk ki egy olyan $S \in \mathcal{F}$-t, amelyre $|S \cap U|$ maximális
5. $U := U \setminus S$
6. $\mathcal{C} := \mathcal{C} \cup \{S\}$
7. return $\mathcal{C}$

Példa: $\mathcal{F} = \{S_1, S_2, S_3, S_4, S_5, S_6\}$



Minimális méretű lefedés: $\mathcal{C} = \{S_3.S_4.S_5\}$

MINIMÁLIS-LEFEDÉS eredménye: $\{S_1, S_4, S_5, S_3\}$ (esetleg $\{S_1, S_4, S_5, S_6\}$).

Legyen $H(d) = \sum_{i=1}^{d} \frac{1}{i}$ (harmonikus sor részösszege).

Tétel

A MINIMÁLIS-LEFEDÉS *algoritmus polinomiális $\rho(n)$-közelítő algoritmus, ahol*

$$\rho(n) = H\Big(\max\{|S| : S \in \mathcal{F}\}\Big)$$

Következmény

A MINIMÁLIS-LEFEDÉS *algoritmus polinomiális $(\ln|X| + 1)$-közelítő algoritmus.*

Közelítő algoritmusok — ládapakolás (bin packing)

románul: problema împachetării

Adott n tárgy (s_i tömeggel) és végtelen sok 1 kapacitású láda.
Helyezzük el a tárgyakat minél kevesebb ládába! Legyen $S = (s_1, s_2, \ldots, s_n)$ úgy, hogy $1 \geq s_1 \geq s_2 \geq \cdots \geq s_n > 0$. Eredmény: $B = (B_1, B_2, \ldots, B_n)$: a B_i halmaz tartalmazza az i-edik ládába került tárgyak indexét.

LÁDAPAKOLÁS(S)

```
1. for i := 1 to n
2.     do B_i = ∅          // az i-edik láda tárgyainak indexhalmaza
3.        b_i := 0         // mennyire van megtelve az i-edik láda
4. for t := 1 to n
5.     do j := 1
6.        while b_j + s_t > 1
7.              do j := j + 1
8.        B_j := B_j ∪ {t}
9.        b_j := b_j + s_t
10. return B
```

Ládapakolás — elemzés

Az algoritmus négyzetes (a két egymásba ágyazott ciklus miatt): $\Theta(n^2)$.

Tétel

A LÁDAPAKOLÁS *algoritmus 2-közelítő algoritmus.*

Legyen az optimális megoldás C^* láda, és az algoritmus eredménye C láda (nyilván $C^* \leq C$.)
Két egymás melletti láda esetében nem fordulhat elő, hogy mindkettő csak legfeljebb félig legyen megtöltve, hisz ekkor a második láda tartalma belefért volna az elsőbe.
Tehát legalább $C-1$ láda felénél jobban meg van töltve. Azaz $\sum_{i=1}^{n} s_i > \frac{C-1}{2}$, de $\sum_{i=1}^{n} s_i < C^*$, innen pedig $C - 1 < 2C^*$,
azaz $C \leq 2C^*$.

1. példa:
0.5, 0.4, 0.3, 0.2, 0.2, 0.2, 0.2

algoritmus szerint:
0.5, 0.4
0.3, 0.2, 0.2, 0.2
0.2 3 láda

optimális:
0.5, 0.3, 0.2
0.4, 0.2, 0.2, 0.2 2 láda

2. példa
0.8, 0.7, 0.6, 0.6, 0.4, 0.4, 0.3, 0.2

algoritmus szerint:

0.8, 0.2
0.7, 0.3
0.6, 0.4
0.6, 0.4 amely optimális

Közelítő sémák — hátizsákfeladat (knapsack, prob. rucsacului)

Adott n tárgy (s_i tömeggel) és egy K kapacitású hátizsák. Töltsük meg minél jobban a zsákot!

$S = (s_1, s_2, \ldots, s_n)$ úgy, hogy $s_1 \geq s_2 \geq \cdots \geq s_n$.

HÁTIZSÁK$_k(S, K)$

1. $H := \emptyset$; $max := 0$
2. for $\{1, 2, \ldots, n\}$ minden legfeljebb k elemű T részhalmazára
3. do $sum := \sum_{i \in T} s_i$
4. for $j := 1$ to n
5. do if $(j \notin T)$ and $(sum + s_j \leq K)$
6. then $sum := sum + s_j$
7. $T := T \cup \{j\}$
8. if $max < sum$
9. then $max := sum$
10. $H := T$
11. return H

Hátizsákfeladat — példa

S=(50,42,32,26,20,17,6,3), K=100

Optimális megoldás: 42, 32, 26 összeg: 100

HÁTIZSÁK$_0$:	50, 42, 6	összeg: 98
HÁTIZSÁK$_1$:	32, 50, 17	összeg: 99
HÁTIZSÁK$_2$:	42, 32, 26	összeg: 100

Hátizsákfeladat — elemzés

Az $\{1, 2, \ldots, n\}$ halmaznak $\binom{n}{i}$ darab i elemű részhalmaza van.

Az algoritmus 3–10. sorait $\sum_{i=0}^{k} \binom{n}{i}$-szer végezzük el.

De $\binom{n}{0} = 1$ és $\binom{n}{i} \leq n^i$, ezért

$$\sum_{i=0}^{k} \binom{n}{i} \leq 1 + kn^k$$

A ciklus belsejét legfeljebb n-szer végezzük el, ezért az algoritmus bonyolultsága:

$$O\left(n + kn^{k+1}\right), \quad \text{azaz} \quad O\left(kn^{k+1}\right).$$

Tétel

A HÁTIZSÁK$_k$ *közelítő algoritmus esetében az optimális megoldás* C^* *tömegösszege és az algoritmus által adott* C *tömegösszeg aránya* $1+\frac{1}{k}$. $\left(\textit{Az algoritmus } \left(1+\frac{1}{k}\right)\textit{-közelítő algoritmus}\right)$.

Bizonyítás:

Legyen $s_1 \geq s_2 \geq \cdots \geq s_n$. Az optimális megoldás $s_{i_1} \geq s_{i_2} \geq \ldots \geq s_{i_p}$.
Ekkor $C^* = \sum_{j=1}^{p} s_{i_j}$. Innen: Ha $1 \leq j \leq k$, akkor $s_{i_j} \leq \frac{C^*}{j}$. és ha $k+1 \leq j \leq p$, akkor $s_{i_j} \leq \frac{C^*}{k+1}$.
Ha $p \leq k$, akkor $C = C^*$.
Ha $p > k$, akkor legyen s_q az első tömeg, amelyik nincs benne az optimális megoldásban.
Ekkor $C + s_q > K$, innen $\frac{C}{C^*} + \frac{s_q}{C^*} > \frac{K}{C^*} \geq 1$. Innen $\frac{C}{C^*} + \frac{1}{k+1} > 1$ és $\frac{C}{C^*} > \frac{k}{k+1}$ vagy $\frac{C^*}{C} < 1 + \frac{1}{k}$.

Részletösszeg-feladat — (változat a hátizsákfeladatra) – CLRS

Adott S=$\{s_1, s_2, \ldots, s_n\}$, ahol minden $s_i > 0$, és adott $K > 0$.
A feladat: létezik-e $X \subseteq S$ úgy, hogy $\sum_{x \in X} x = K$? NP-teljes feladat

Gyakorlatban: keressük azt az $X \subseteq S$ részhalmazt, amelyre $\sum_{x \in X} x \leq K$, és az összeg a lehető legnagyobb.

Lista: $L =< s_1, s_2, \ldots, s_k >$, ahol $s_1 \leq s_2 \leq \ldots \leq s_k$.
Ekkor $L + x =< s_1 + x, s_2 + x, \ldots, s_k + x >$.

Exponenciális idejű, pontos algoritmus

Csak a részletösszeget határozza meg. Az ÖSSZEFÉSÜL két listát egybefésül, kihagyva az ismétlődő elemeket.

PONTOS-RÉSZLETÖSSZEG(S, K)

1. $n := |S|$
2. $L_0 :=< 0 >$
3. for $i := 1$ to n
4. do $L_i :=$ ÖSSZEFÉSÜL($L_{i-1}, L_{i-1} + s_i$)
5. töröljük L_i-ből a K-nál nagyobb elemeket
6. return L_n legnagyobb eleme

Példa:

$S = \{1, 2, 4, 6\}$, $K = 9$

$L_0 =< 0 >$

$L_1 =< 0, 1 >$

$L_2 =< 0, 1, 2, 3 >$

$L_3 =< 0, 1, 2, 3, 4, 5, 6 >$

$L_4 =< 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 >$

Polinomiális közelítő séma

Legyen $0 < \delta < 1$, és ha

$$\frac{y}{1+\delta} \leq z \leq y,$$

akkor y-t kitöröljük a listából, z az y „képviselője” az új listában. Ez a művelet a ritkítás.

Ha $\delta = 0,1$, akkor $L =< 10, 11, 12, 15, 20, 21, 22, 23, 24, 29 >$
ritkítása: $L =< 10, 12, 15, 20, 23, 29 >$, mert pl. $\frac{24}{1,1} < 23 < 24$.

$L =< y_1, y_2, \ldots, y_m >$ elemei növekvő sorrendbe vannak

RITKÍT(L, δ)

1. $m := |L|$
2. $L' :=< y_1 >$
3. $last := y_1$
4. for $i := 2$ to m
5. do if $y_i > last \cdot (1 + \delta)$
6. then tegyük y_i-t L' végére
7. $last := y_i$
8. return L'

KÖZELÍTŐ-RÉSZLETÖSSZEG(S, K, ε)

1. $n := |S|$
2. $L_0 :=< 0 >$
3. for $i := 1$ to n
4. do ÖSSZEFÉSÜL($L_{i-1}, L_{i-1} + s_i$)
5. $L_i :=$ RITKÍT($L_i, \varepsilon/2n$)
6. töröljük L_i-ből a K-nál nagyobb elemeket
7. return L_n legnagyobb eleme

Példa

$$
\begin{aligned}
L_0 &= \langle 0 \rangle \\
L_1 &= \langle 0, 104 \rangle \\
L_1 &= \langle 0, 104 \rangle \\
L_1 &= \langle 0, 104 \rangle \\
L_2 &= \langle 0, 102, 104, 206 \rangle \\
L_2 &= \langle 0, 102, 206 \rangle \\
L_2 &= \langle 0, 102, 206 \rangle \\
L_3 &= \langle 0, 102, 201, 206, 303, 407 \rangle \\
L_3 &= \langle 0, 102, 201, 303, 407 \rangle \\
L_3 &= \langle 0, 102, 201, 303 \rangle \\
L_4 &= \langle 0, 101, 102, 201, 203, 302, 303, 404 \rangle \\
L_4 &= \langle 0, 101, 201, 302, 404 \rangle \\
L_4 &= \langle 0, 101, 201, 302 \rangle
\end{aligned}
$$

$S =< 104, 102, 201, 101 >$,
$K = 308$, $\varepsilon = 0,40$, ekkor
$\varepsilon/8 = 0,05$

Eredmény: 302
Az optimális megoldás:
307= 104+102+101.

Tétel

A KÖZELÍTŐ-RÉSZLETÖSSZEG *algoritmus teljesen polinomiális közelítő séma.*

teljesen polinomiális közelítő séma: n-ben és $1/\varepsilon$-ban is polinomiális.

Mihai Oltean
An optical solution for the set splitting problem
Acta Universitatis Sapientiae, Informatica **9,** 2 (2017) 134–143

https://acta.sapientia.ro/content/docs/an-optical-solution-for-the-set-splittin.pdf

Gráfszínezés

Feladat: minimális számú színnel kifesteni a gráf elemeit (csúcsok, élek, tartományok) úgy, hogy két szomszédos elem nem legyen ugyanolyan színű.

– síkgráfok tartományainak színezése – négyszínprobléma, 1976-ban megoldották (K. Appel és W. Haken)

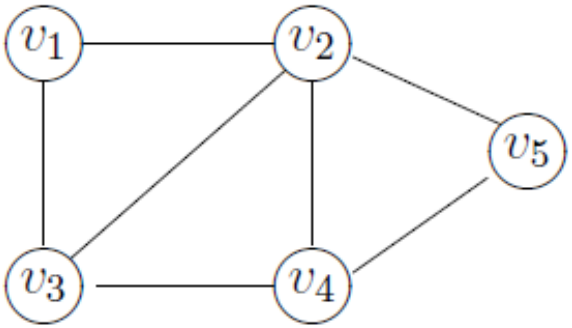
Közelítő algoritmus

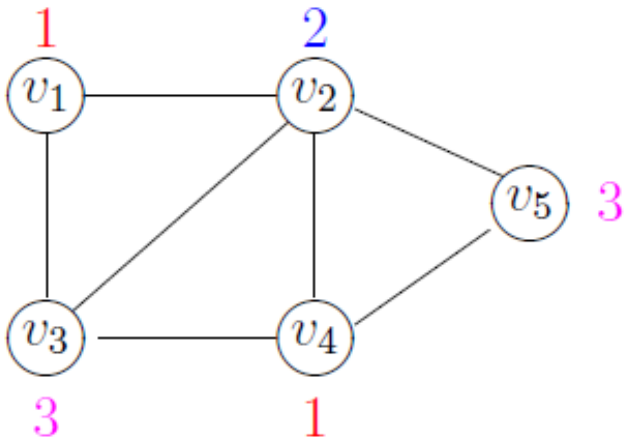
Adott a $G = (V, E)$ gráf, ahol $V = \{v_1, v_2, \ldots, v_n\}$. Az algoritmus hozzárendel minden v_i csúcshoz egy c_i színt. Eredmény a $C = (c_1, c_2, \ldots, c_n)$ vektor.

SZEKVENCIÁLIS_SZÍNEZÉS(G)

```
1. for i = 1 to n
2.     do s := 1
3.        while N(v_i)-ben van s színű csúcs
4.              do s := s + 1
5.        c_i := s                          ▷ színezzük ki v_i-t az s színnel
6. return C
```

Példa



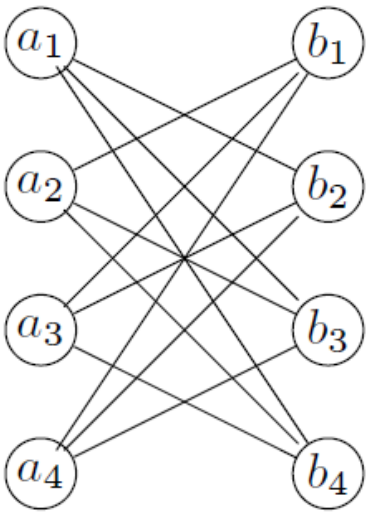


Eredmény: $C = (1, 2, 3, 1, 2)$.

Kromatikus szám: Az a legkisebb c, amennyi színnel ki lehet színezni a gráf csúcsait. Jelölése: $\chi(G)$. Fenti példában $\chi(G) = 3$, mivel kevesebb színnel nem lehet kiszínezni, ugyanis v_2 szomszédai egy úton vannak, tehát két szín szükséges a kiszínezésükre, így v_2 már csak egy harmadik színnel színezhető.

Az algoritmus polinomiális, $O(n^2)$ bonyolultságú.
Eredményessége azonban függ a csúcsok sorrendjétől.

Például, legyen $G = (V, E)$, ahol $V = \{a_1, b_1, a_2, b_2, \ldots, a_n, b_n\}$,
$E = \Big\{\{a_i, b_j\} \mid i \neq j\Big\}$.



A közelítő algoritmus eredménye:

2, ha a csúcsok sorrendje $a_1, a_2, \ldots, a_n, b_1, b_2, \ldots, b_n$,
n, ha a csúcsok sorrendje $a_1, b_1, a_2, b_2, \ldots, a_n, b_n$.