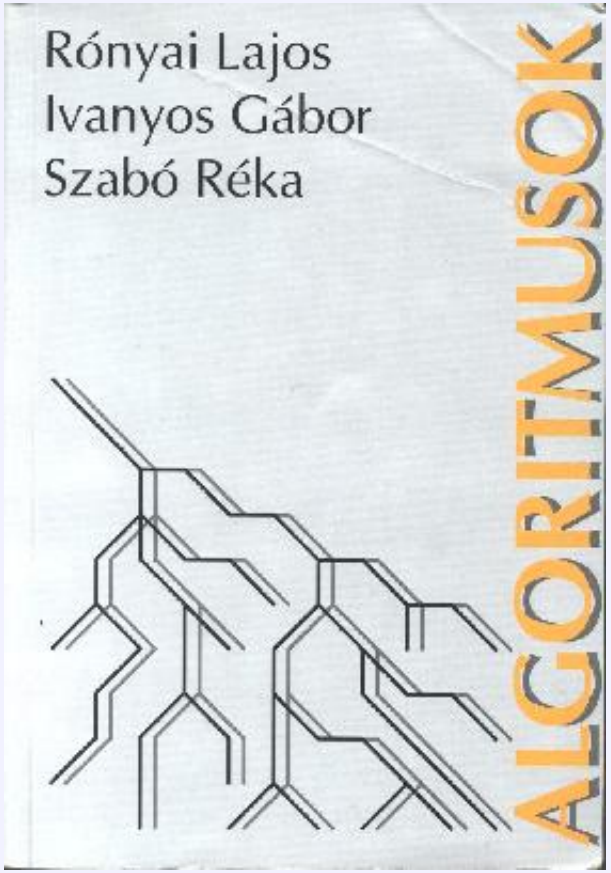
Algoritmuselmélet 1. előadás

Katona Gyula Y.
Budapesti Műszaki és Gazdaságtudományi Egyetem
Számítástudományi Tsz.
I. B. 137/b
kiskat@cs.bme.hu

2002 Február 11.

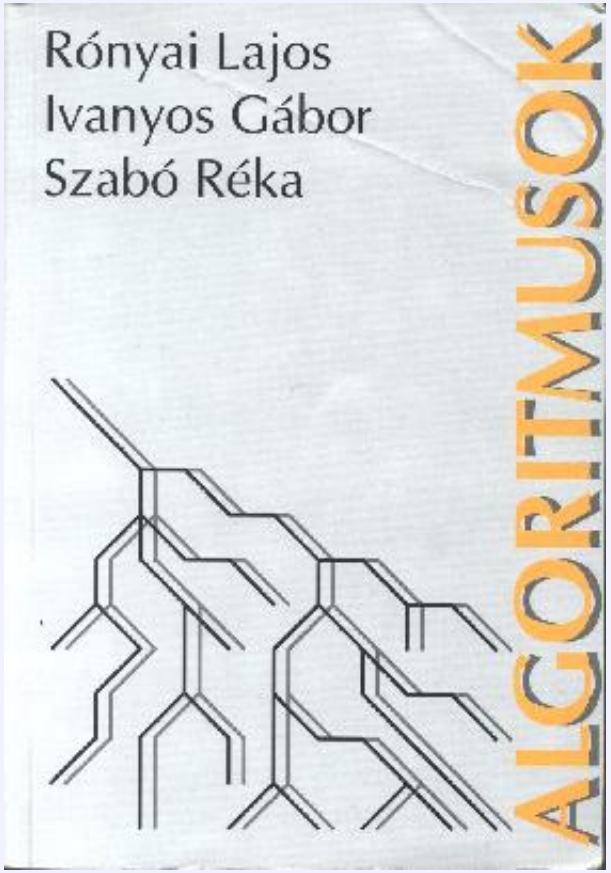
Források

Források



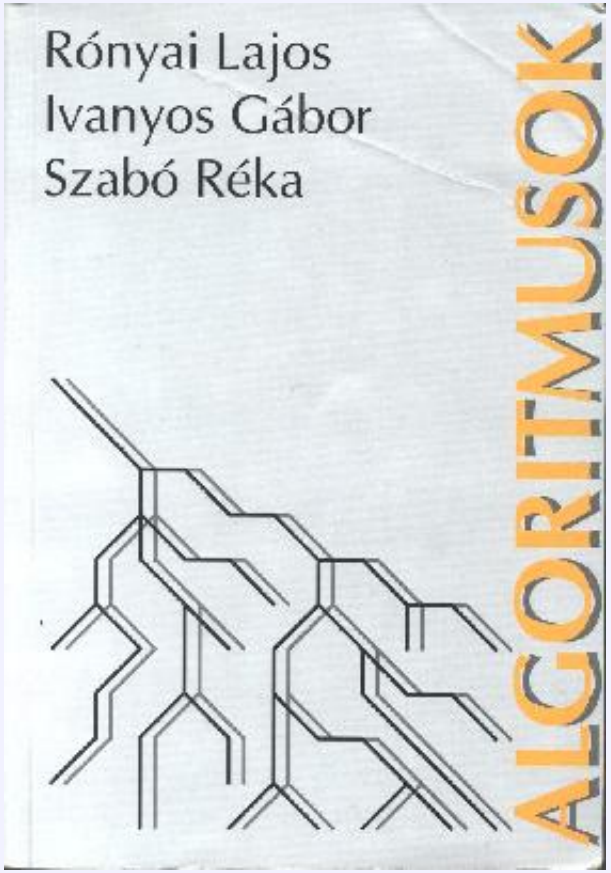
- Rónyai Lajos–Ivanyos Gábor–Szabó Réka: **Algoritmusok**, TYPOTEX, 1999

Források



- Rónyai Lajos–Ivanyos Gábor–Szabó Réka: **Algoritmusok**, TYPOTEX, 1999
- Feladatgyűjtemény letölthető: http://www.cs.bme.hu/~kiskat/algel

Források



- Rónyai Lajos–Ivanyos Gábor–Szabó Réka: **Algoritmusok**, TYPOTEX, 1999
- Feladatgyűjtemény letölthető: http://www.cs.bme.hu/~kiskat/algel
- Egyéb információk, hirdetmények ugyanitt.

Követelmények

Követelmények

Zárthelyi dolgozat: 2002. április 8. — 6 db 10 pontos feladat 100 percre, mindet lehet használni.

Pontozás: 20-29 pont 2-es, 30-39 pont 3-es, 40-49 pont 4-es, 50-60 pont 5-ös.

Aláírás: Feltétele a ZH megírása 2-esre. Szükség esetén pótZH.

Vizsga: Írásbeli — 2 elméleti kérdés, 4 példa, nem lehet semmit használni. Pontozás: mint a ZH-n.

Vizsga jegy: Ha érdemes, akkor lehet a vizsga ZH eredményét átlagolni az évközi eredménnyel. Ha aláírás csak pótZH-val született, ez a lehetőség nem áll fenn.

Javítás: Ha az így kialakult jegy nem elég jó, akkor kizárólag a vizsga eredményhirdetésének időpontjában lehet szóban felelni, amivel ± 1 jegyet lehet változtatni.

Algoritmus fogalma

- Egyelőre nem definiáljuk rendesen az algoritmus fogalmát.

Algoritmus fogalma

- Egyelőre nem definiáljuk rendesen az algoritmus fogalmát.
- Eljárás, recept, módszer.

Algoritmus fogalma

- Egyelőre nem definiáljuk rendesen az algoritmus fogalmát.
- Eljárás, recept, módszer.
- Jól meghatározott lépések egymásutánja, amelyek már elég pontosan, egyértelműen megfogalmazottak ahhoz, hogy gépiesen végrehajthatók legyenek.

A szó eredete

Al Khvarizimi (Mohamed ibn Músza) bagdadi matematikus a IX. században könyvet írt az egészekkel való alapműveletek végzéséről.

A szó eredete

Al Khvarizimi (Mohamed ibn Músza) bagdadi matematikus a IX. században könyvet írt az egészekkel való alapműveletek végzéséről.

algoritmus ⟷ számítógép program

Milyen hatékony egy algoritmus?

- Legtöbbször csak a lépésszám nagyságrendje érdekes.
  Hogyan függ a lépésszám az input méretétől?

Milyen hatékony egy algoritmus?

- Legtöbbször csak a lépésszám nagyságrendje érdekes. Hogyan függ a lépésszám az input méretétől?

- Az input méretét legtöbbször n-nel jelöljük.

Milyen hatékony egy algoritmus?

- Legtöbbször csak a lépésszám nagyságrendje érdekes.
  Hogyan függ a lépésszám az input méretétől?

- Az input méretét legtöbbször n-nel jelöljük.

- A lépésszám ennek egy f függvénye, azaz ha n méretű az input, akkor az algoritmus $f(n)$ lépést végez.

Milyen hatékony egy algoritmus?

- Legtöbbször csak a lépésszám nagyságrendje érdekes.
  Hogyan függ a lépésszám az input méretétől?
- Az input méretét legtöbbször n-nel jelöljük.
- A lépésszám ennek egy f függvénye, azaz ha n méretű az input, akkor az algoritmus $f(n)$ lépést végez.
- Igazából az f függvény az érdekes.

Milyen hatékony egy algoritmus?

- Legtöbbször csak a lépésszám nagyságrendje érdekes.
  Hogyan függ a lépésszám az input méretétől?
- Az input méretét legtöbbször n-nel jelöljük.
- A lépésszám ennek egy f függvénye, azaz ha n méretű az input, akkor az algoritmus $f(n)$ lépést végez.
- Igazából az f függvény az érdekes.
- $100n$ vagy $101n$, általában mindegy

Milyen hatékony egy algoritmus?

- Legtöbbször csak a lépésszám nagyságrendje érdekes.
  Hogyan függ a lépésszám az input méretétől?
- Az input méretét legtöbbször n-nel jelöljük.
- A lépésszám ennek egy f függvénye, azaz ha n méretű az input, akkor az algoritmus $f(n)$ lépést végez.
- Igazából az f függvény az érdekes.
- $100n$ vagy $101n$, általában mindegy
- n^2 vagy n^3 már sokszor nagy különbség, de néha mindegy

Milyen hatékony egy algoritmus?

- Legtöbbször csak a lépésszám nagyságrendje érdekes.
  Hogyan függ a lépésszám az input méretétől?
- Az input méretét legtöbbször n-nel jelöljük.
- A lépésszám ennek egy f függvénye, azaz ha n méretű az input, akkor az algoritmus $f(n)$ lépést végez.
- Igazából az f függvény az érdekes.
- $100n$ vagy $101n$, általában mindegy
- n^2 vagy n^3 már sokszor nagy különbség, de néha mindegy
- n^2 vagy 2^n már mindig nagy különbség

Függvények nagyságrendje

Definíció. *Ha $f(x)$ és $g(x)$ az $\mathbb{R}^+$ egy részhalmazán értelmezett valós értékeket felvevő függvények, akkor $f = O(g)$ jelöli azt a tényt, hogy vannak olyan $c, n > 0$ állandók, hogy $|f(x)| \leq c|g(x)|$ teljesül, ha $x \geq n$.*

Függvények nagyságrendje

Definíció. *Ha $f(x)$ és $g(x)$ az $\mathbb{R}^+$ egy részhalmazán értelmezett valós értékeket felvevő függvények, akkor $f = O(g)$ jelöli azt a tényt, hogy vannak olyan $c, n > 0$ állandók, hogy $|f(x)| \leq c|g(x)|$ teljesül, ha $x \geq n$.*

Például:

- $100n + 300 = O(n)$, hiszen $n = 300,\ c = 101$-re teljesülnek a feltételek, $100n + 300 \leq 101n$, ha $n \geq 300$

Függvények nagyságrendje

Definíció. *Ha $f(x)$ és $g(x)$ az $\mathbb{R}^+$ egy részhalmazán értelmezett valós értékeket felvevő függvények, akkor $f = O(g)$ jelöli azt a tényt, hogy vannak olyan $c, n > 0$ állandók, hogy $|f(x)| \leq c|g(x)|$ teljesül, ha $x \geq n$.*

Például:

- $100n + 300 = O(n)$, hiszen $n = 300,\ c = 101$-re teljesülnek a feltételek, $100n + 300 \leq 101n$, ha $n \geq 300$
- $5n^2 + 3n = O(n^2)$
- $n^4 + 5n^3 = O(n^5)$
- $n^{1000} = O(2^n)$

Függvények nagyságrendje

Definíció. *Ha $f(x)$ és $g(x)$ az $\mathbb{R}^+$ egy részhalmazán értelmezett valós értékeket felvevő függvények, akkor $f = \Omega(g)$ jelöli azt a tényt, hogy vannak olyan $c, n > 0$ állandók, hogy $|f(x)| \geq c|g(x)|$ teljesül, ha $x \geq n$.*

Függvények nagyságrendje

Definíció. *Ha $f(x)$ és $g(x)$ az $\mathbb{R}^+$ egy részhalmazán értelmezett valós értékeket felvevő függvények, akkor $f = \Omega(g)$ jelöli azt a tényt, hogy vannak olyan $c, n > 0$ állandók, hogy $|f(x)| \geq c|g(x)|$ teljesül, ha $x \geq n$.*

Például:

- $100n - 300 = \Omega(n)$, hiszen $n > 300,\ c = 99$-re teljesülnek a feltételek

Függvények nagyságrendje

Definíció. *Ha $f(x)$ és $g(x)$ az $\mathbb{R}^+$ egy részhalmazán értelmezett valós értékeket felvevő függvények, akkor $f = \Omega(g)$ jelöli azt a tényt, hogy vannak olyan $c, n > 0$ állandók, hogy $|f(x)| \geq c|g(x)|$ teljesül, ha $x \geq n$.*

Például:

- $100n - 300 = \Omega(n)$, hiszen $n > 300,\ c = 99$-re teljesülnek a feltételek
- $5n^2 - 3n = \Omega(n^2)$
- $n^4 - 5n^3 = \Omega(n^4)$
- $2^n = \Omega(n^{1000})$

Függvények nagyságrendje

Definíció. *Ha* $f = O(g)$ *és* $f = \Omega(g)$ *is teljesül, akkor* $f = \Theta(g)$.

Függvények nagyságrendje

Definíció. *Ha* $f = O(g)$ *és* $f = \Omega(g)$ *is teljesül, akkor* $f = \Theta(g)$.

Például:

- $100n - 300 = \Theta(n)$

Függvények nagyságrendje

Definíció. *Ha* $f = O(g)$ *és* $f = \Omega(g)$ *is teljesül, akkor* $f = \Theta(g)$.

Például:

- $100n - 300 = \Theta(n)$
- $5n^2 - 3n = \Theta(n^2)$
- $n^4 - 5n^3 = \Theta(n^4)$
- $1000 \cdot 2^n = \Theta(2^n)$

Függvények nagyságrendje

Definíció. *Legyenek $f(n)$ és $g(n)$ a pozitív egészeken értelmezett valós értékű függvények. Ekkor az $f = o(g)$ jelöléssel rövidítjük azt, hogy*

$$\frac{f(n)}{g(n)} \to 0, \text{ ha } n \to \infty.$$

Függvények nagyságrendje

Definíció. *Legyenek $f(n)$ és $g(n)$ a pozitív egészeken értelmezett valós értékű függvények. Ekkor az $f = o(g)$ jelöléssel rövidítjük azt, hogy*

$$\frac{f(n)}{g(n)} \to 0, \text{ ha } n \to \infty.$$

Például:

- $100n + 300 = o(n^2)$, hiszen $\frac{100n+300}{n^2} \to 0$ ha $n \to \infty$

Függvények nagyságrendje

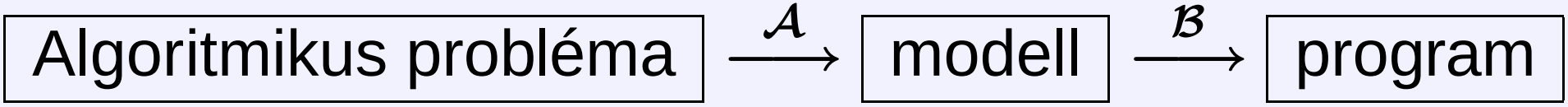
Definíció. *Legyenek $f(n)$ és $g(n)$ a pozitív egészeken értelmezett valós értékű függvények. Ekkor az $f = o(g)$ jelöléssel rövidítjük azt, hogy*

$$\frac{f(n)}{g(n)} \to 0, \text{ ha } n \to \infty.$$

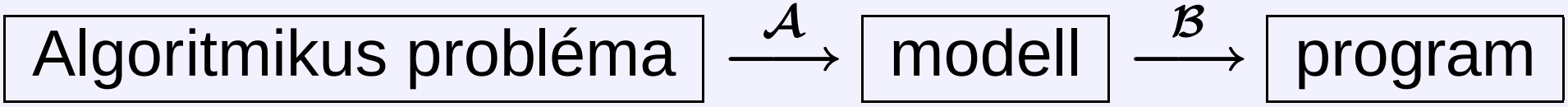
Például:

- $100n + 300 = o(n^2)$, hiszen $\frac{100n+300}{n^2} \to 0$ ha $n \to \infty$
- $5n^2 + 3n = o(n^3)$
- $n^4 + 5n^3 = o(n^4 \log_2 n)$
- $n^{1000} = o(2^n)$

Algoritmikus problémák megoldása

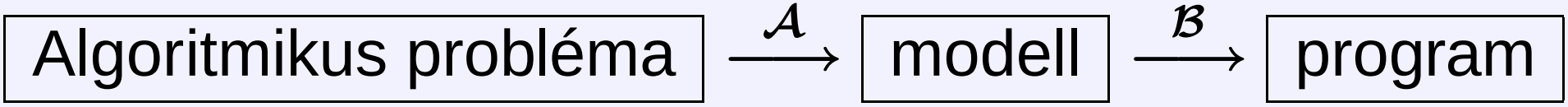


Algoritmikus problémák megoldása



$\mathcal{A}$: pontosítás, egyszerűsítés, absztrakció, lényegtelen elemek kiszűrése, a lényeg kihámozása

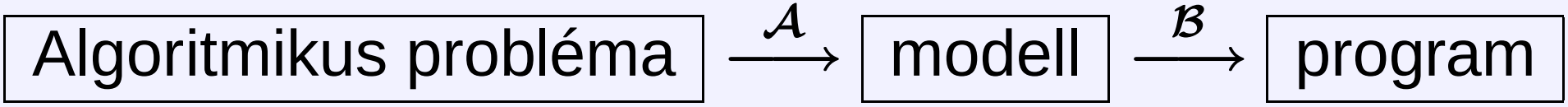
Algoritmikus problémák megoldása



$\mathcal{A}$**:** pontosítás, egyszerűsítés, absztrakció, lényegtelen elemek kiszűrése, a lényeg kihámozása

Modell: sokféle lehet, elég tág, de elég egyszerű, formalizált, pontos

Algoritmikus problémák megoldása

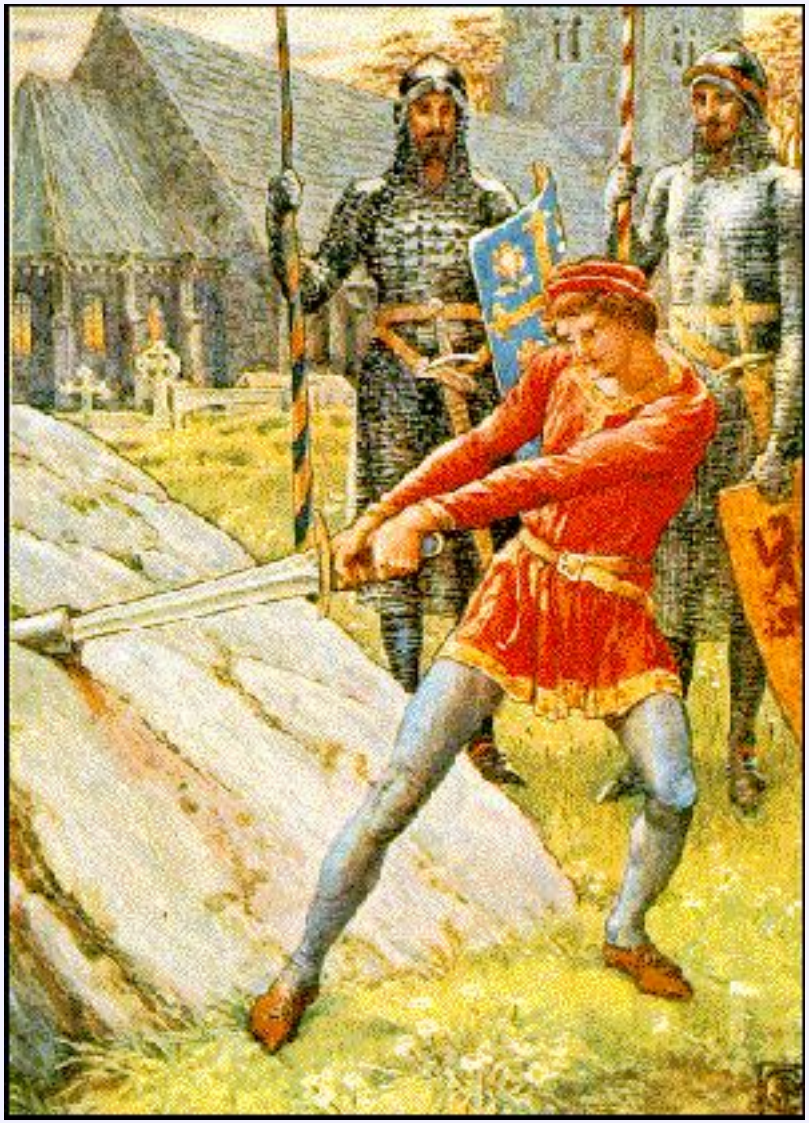


$\mathcal{A}$**:** pontosítás, egyszerűsítés, absztrakció, lényegtelen elemek kiszűrése, a lényeg kihámozása

Modell: sokféle lehet, elég tág, de elég egyszerű, formalizált, pontos

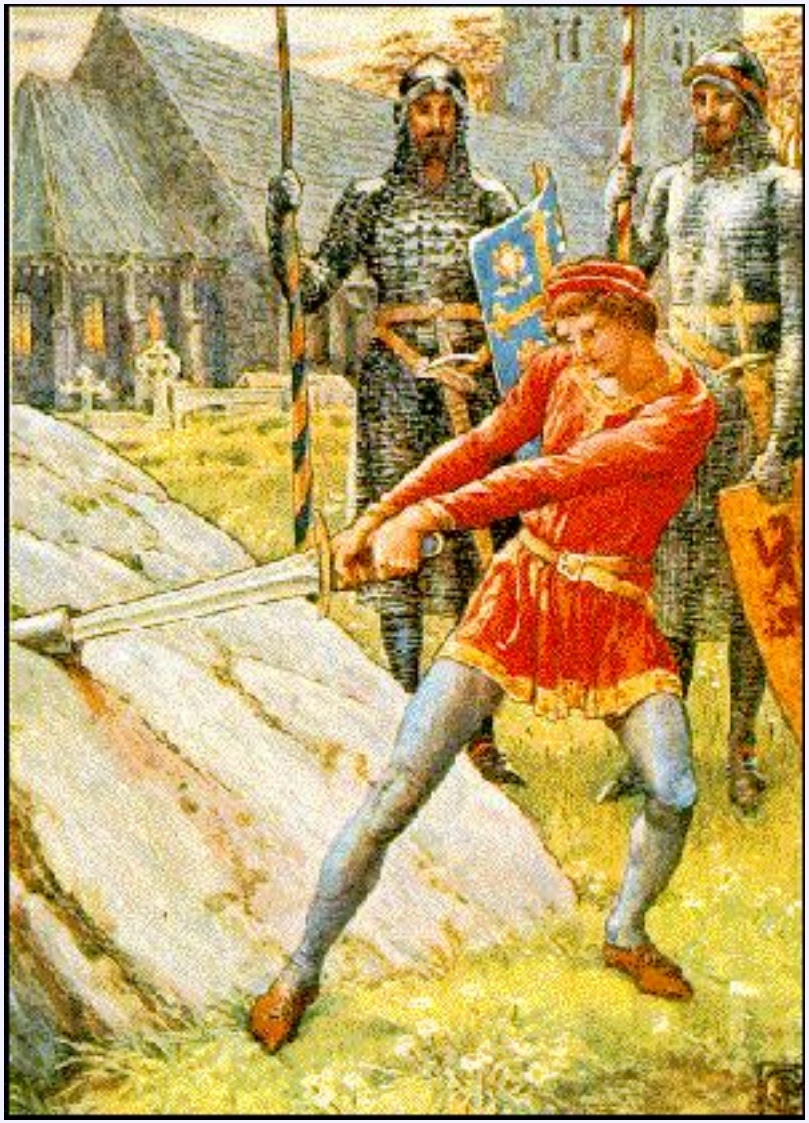
$\mathcal{B}$**:** hatékony algoritmus, bemenő adatok $\longrightarrow$ eredmény, érdemes foglalkozni a kapott algoritmus *elemzésével, értékelésével*, megvizsgálva, hogy a módszer mennyire hatékony

Arthur király civilizációs törekvései



Arthur király fényes udvarában 150 lovag és 150 udvarhölgy él. A király, aki közismert civilizációs erőfeszítéseiről, elhatározza, hogy megházasítja jó lovagjait és szép udvarhölgyeit. Mindezt persze emberségesen szeretné tenni. Csak olyan párok egybekelését akarja, amelyek tagjai kölcsönösen vonzalmat éreznek egymás iránt. Hogyan fogjon hozzá?

Arthur király civilizációs törekvései



Arthur király fényes udvarában 150 lovag és 150 udvarhölgy él. A király, aki közismert civilizációs erőfeszítéseiről, elhatározza, hogy megházasítja jó lovagjait és szép udvarhölgyeit. Mindezt persze emberségesen szeretné tenni. Csak olyan párok egybekelését akarja, amelyek tagjai kölcsönösen vonzalmat éreznek egymás iránt. Hogyan fogjon hozzá?

Természetesen pártfogójához, a nagyhatalmú varázslóhoz, Merlinhez fordul. Merlin rögvest felismeri, hogy itt is bináris szimmetrikus viszonyok ábrázolásáról van szó.

Természetesen pártfogójához, a nagyhatalmú varázslóhoz, Merlinhez fordul. Merlin rögvest felismeri, hogy itt is bináris szimmetrikus viszonyok ábrázolásáról van szó.

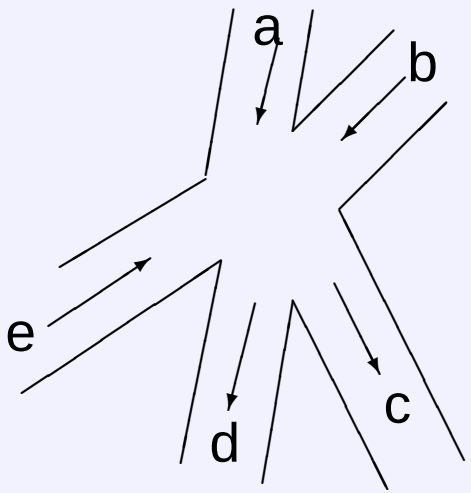
Nagy darab pergament vesz elő, és nekilát egy páros gráfot rajzolni.

Természetesen pártfogójához, a nagyhatalmú varázslóhoz, Merlinhez fordul. Merlin rögvest felismeri, hogy itt is bináris szimmetrikus viszonyok ábrázolásáról van szó.

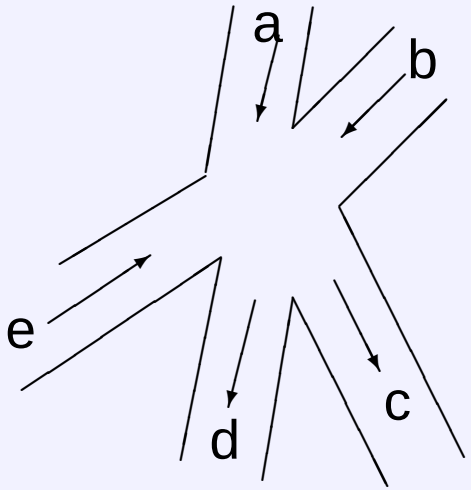
Nagy darab pergament vesz elő, és nekilát egy páros gráfot rajzolni.

A királyi parancs teljesítéséhez Merlinnek élek egy olyan rendszerét kell kiválasztania a gráf éleiből, hogy a kiválasztott élek közül a gráf minden pontjához pontosan egy csatlakozzon. A kiválasztott élek felelnek meg a tervezett házasságoknak. A gráfelmélet nyelvén teljes párosítást kell keresnie.

Közlekedési lámpák ütemezése

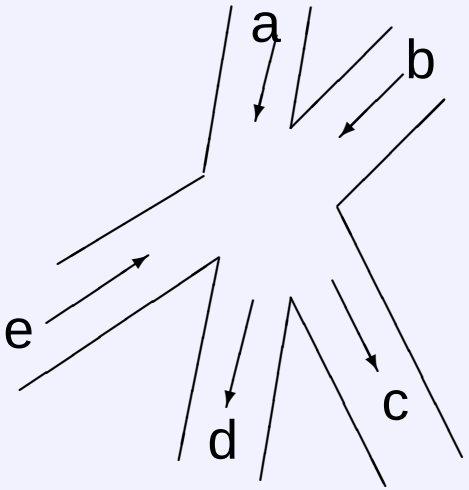


Közlekedési lámpák ütemezése



lámpák: ac, ad, bc, bd, ec és ed
állapot: lámpák $\longrightarrow \{P, Z\}$

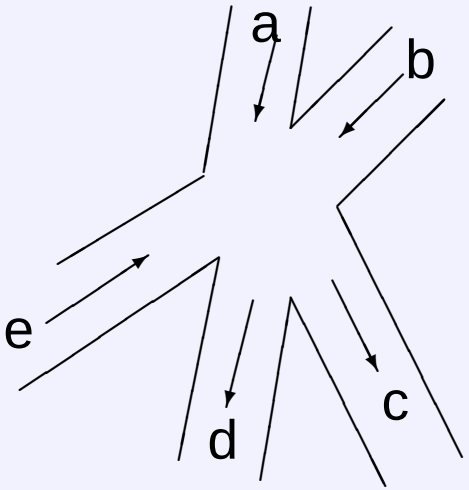
Közlekedési lámpák ütemezése



lámpák: ac, ad, bc, bd, ec és ed
állapot: lámpák $\longrightarrow \{P, Z\}$

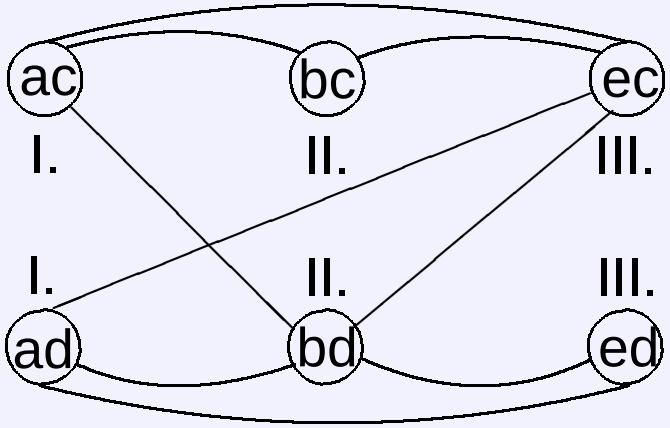
Feladat: Mennyi a minimális számú állapot, ami biztonságos és nem okoz örök dugót?

Közlekedési lámpák ütemezése

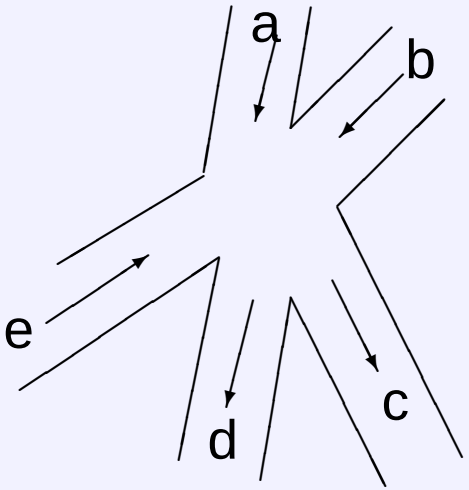


lámpák: ac, ad, bc, bd, ec és ed
állapot: lámpák $\rightarrow \{P, Z\}$

Feladat: Mennyi a minimális számú állapot, ami biztonságos és nem okoz örök dugót?

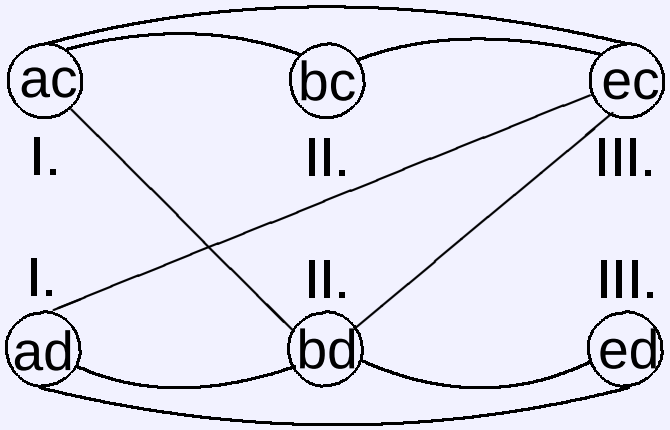


Közlekedési lámpák ütemezése



lámpák: ac, ad, bc, bd, ec és ed
állapot: lámpák $\rightarrow \{P, Z\}$

Feladat: Mennyi a minimális számú állapot, ami biztonságos és nem okoz örök dugót?



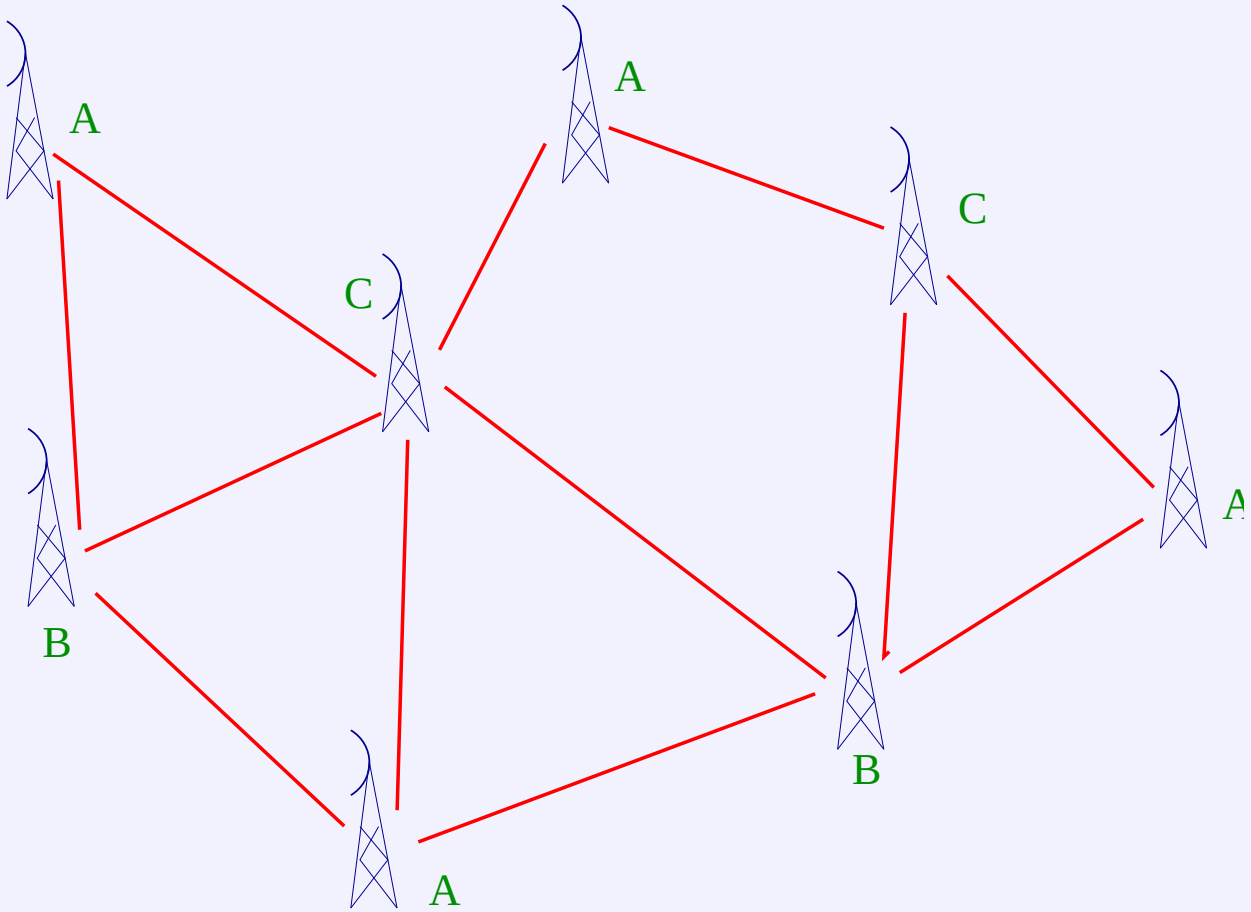
Gráfelméleti nyelven: Mennyi G kromatikus száma?

Mobil telefon átjátszók frekvencia kiosztása

Egy adott átjátszóhoz egy adott frenkvenciát rendelnek.
Egy telefon a közelben levő átjátszók közül választ.
„Közel levő” átjátszók frekvenciája különbözzön.

Mobil telefon átjátszók frekvencia kiosztása

Egy adott átjátszóhoz egy adott frenkvenciát rendelnek.
Egy telefon a közelben levő átjátszók közül választ.
„Közel levő” átjátszók frekvenciája különbözzön.



Szuperforrás keresése

Definíció. *A G irányított gráf $s \in V$ csúcsa szuperforrás, ha minden s-től különböző $y \in V$ csúcs esetén teljesül, hogy $(s, y) \in E$ és $(y, s) \notin E$. A szuperforrás olyan s csúcs, amiből a gráf minden más csúcsába él vezet, az s-be pedig egyetlen más csúcsból sem megy él.*

Szuperforrás keresése

Definíció. *A G irányított gráf $s \in V$ csúcsa szuperforrás, ha minden s-től különböző $y \in V$ csúcs esetén teljesül, hogy $(s, y) \in E$ és $(y, s) \notin E$. A szuperforrás olyan s csúcs, amiből a gráf minden más csúcsába él vezet, az s-be pedig egyetlen más csúcsból sem megy él.*

A feladat a következő: tegyük fel, hogy az A adjacencia mátrixával adott a $G = (V, E)$ irányított gráf, aminek a csúcshalmaza $V = \{1, \ldots, n\}$. Döntsük el, hogy van-e G-ben szuperforrás. Ha igen, találjuk meg.

Szuperforrás keresése

Definíció. *A G irányított gráf $s \in V$ csúcsa szuperforrás, ha minden s-től különböző $y \in V$ csúcs esetén teljesül, hogy $(s, y) \in E$ és $(y, s) \notin E$. A szuperforrás olyan s csúcs, amiből a gráf minden más csúcsába él vezet, az s-be pedig egyetlen más csúcsból sem megy él.*

A feladat a következő: tegyük fel, hogy az A adjacencia mátrixával adott a $G = (V, E)$ irányított gráf, aminek a csúcshalmaza $V = \{1, \ldots, n\}$. Döntsük el, hogy van-e G-ben szuperforrás. Ha igen, találjuk meg.

Első ötlet: Sorra vesszük az $i \in V$ csúcsokat, mindegyikről megnézve, hogy szuperforrás-e.

Szuperforrás keresése

Definíció. *A G irányított gráf $s \in V$ csúcsa szuperforrás, ha minden s-től különböző $y \in V$ csúcs esetén teljesül, hogy $(s, y) \in E$ és $(y, s) \notin E$. A szuperforrás olyan s csúcs, amiből a gráf minden más csúcsába él vezet, az s-be pedig egyetlen más csúcsból sem megy él.*

A feladat a következő: tegyük fel, hogy az A adjacencia mátrixával adott a $G = (V, E)$ irányított gráf, aminek a csúcshalmaza $V = \{1, \ldots, n\}$. Döntsük el, hogy van-e G-ben szuperforrás. Ha igen, találjuk meg.

Első ötlet: Sorra vesszük az $i \in V$ csúcsokat, mindegyikről megnézve, hogy szuperforrás-e.

Ennek költsége: az A mátrix $2(n^2 - n)$ elemét vizsgáljuk meg.

Szuperforrás keresése

Definíció. *A G irányított gráf $s \in V$ csúcsa szuperforrás, ha minden s-től különböző $y \in V$ csúcs esetén teljesül, hogy $(s, y) \in E$ és $(y, s) \notin E$. A szuperforrás olyan s csúcs, amiből a gráf minden más csúcsába él vezet, az s-be pedig egyetlen más csúcsból sem megy él.*

A feladat a következő: tegyük fel, hogy az A adjacencia mátrixával adott a $G = (V, E)$ irányított gráf, aminek a csúcshalmaza $V = \{1, \ldots, n\}$. Döntsük el, hogy van-e G-ben szuperforrás. Ha igen, találjuk meg.

Első ötlet: Sorra vesszük az $i \in V$ csúcsokat, mindegyikről megnézve, hogy szuperforrás-e.

Ennek költsége: az A mátrix $2(n^2 - n)$ elemét vizsgáljuk meg.

Jobb módszer: Ha $(i, j) \in E$, akkor j nem lehet szuperforrás, ha $(i, j) \notin E$ akkor i nem lehet szuperforrás

Gyorsabb algoritmus

$i := 1, j := n$;
while $i \neq j$ **do**
    if $A[i, j] = 1$ **then** $j := j - 1$

    else $i := i + 1$;
(Amikor ideérünk, már csak i lehet szuperforrás,*
*ezt ellenőrizzük a továbbiakban. *)*
for $k = 1$ **to** n **do**
    if $k \neq i$ és ($A[i, k] \neq 1$ vagy $A[k, i] \neq 0$) **then return**(nincs szuperforrás)
return(i szuperforrás).

Gyorsabb algoritmus

$i := 1, j := n$;
while $i \neq j$ **do**
    if $A[i, j] = 1$ **then** $j := j - 1$

    else $i := i + 1$;
(Amikor ideérünk, már csak i lehet szuperforrás,*
*ezt ellenőrizzük a továbbiakban. *)*
for $k = 1$ **to** n **do**
    if $k \neq i$ és ($A[i, k] \neq 1$ vagy $A[k, i] \neq 0$) **then return**(nincs szuperforrás)
return(i szuperforrás).

Ennek költsége: összesen $3n - 3$ elemet vizsgálunk meg, $n - 1$-et a while ciklusban, $2n - 2$-t az ellenőzésnél

Gyorsabb algoritmus

$i := 1, j := n;$
while $i \neq j$ **do**
    if $A[i, j] = 1$ **then** $j := j - 1$

    else $i := i + 1;$
(Amikor ideérünk, már csak i lehet szuperforrás,*
*ezt ellenőrizzük a továbbiakban. *)*
for $k = 1$ **to** n **do**
    if $k \neq i$ és ($A[i, k] \neq 1$ vagy $A[k, i] \neq 0$) **then return**(nincs szuperforrás)
return(i szuperforrás).

Ennek költsége: összesen $3n - 3$ elemet vizsgálunk meg, $n - 1$-et a while ciklusban, $2n - 2$-t az ellenőzésnél

Mennyire jó ez?

A költség elemzése

Jelölje $T(n)$ a legjobb (leggyorsabb) algoritmus által megvizsgált mátrix-elemek számának maximumát az összes n pontú gráfra.

A költség elemzése

Jelölje $T(n)$ a legjobb (leggyorsabb) algoritmus által megvizsgált mátrix-elemek számának maximumát az összes n pontú gráfra.

Tudjuk, hogy $T(n) \leq 3n - 3$.

A költség elemzése

Jelölje $T(n)$ a legjobb (leggyorsabb) algoritmus által megvizsgált mátrix-elemek számának maximumát az összes n pontú gráfra.

Tudjuk, hogy $T(n) \leq 3n - 3$.

Nyilvánvaló, hogy $2n - 2 \leq T(n)$, mert le kell ellenőrizni, hogy szuperforrás-e.

A költség elemzése

Jelölje $T(n)$ a legjobb (leggyorsabb) algoritmus által megvizsgált mátrix-elemek számának maximumát az összes n pontú gráfra.

Tudjuk, hogy $T(n) \leq 3n - 3$.

Nyilvánvaló, hogy $2n - 2 \leq T(n)$, mert le kell ellenőrizni, hogy szuperforrás-e.

1 él megkérdezése legfeljebb 1 csúcsot zár ki mint lehetséges szuperforrást.

A költség elemzése

Jelölje $T(n)$ a legjobb (leggyorsabb) algoritmus által megvizsgált mátrix-elemek számának maximumát az összes n pontú gráfra.

Tudjuk, hogy $T(n) \leq 3n - 3$.

Nyilvánvaló, hogy $2n - 2 \leq T(n)$, mert le kell ellenőrizni, hogy szuperforrás-e.

1 él megkérdezése legfeljebb 1 csúcsot zár ki mint lehetséges szuperforrást.

Egy algoritmus első $n - 2$ kérdése után még legalább két csúcs – mondjuk i és j – lehet szuperforrás. Ahhoz, hogy belássuk, hogy i szuperforrás, meg kell vizsgálni az i-edik sor és i-edik oszlop minden elemét.

A költség elemzése

Jelölje $T(n)$ a legjobb (leggyorsabb) algoritmus által megvizsgált mátrix-elemek számának maximumát az összes n pontú gráfra.

Tudjuk, hogy $T(n) \leq 3n - 3$.

Nyilvánvaló, hogy $2n - 2 \leq T(n)$, mert le kell ellenőrizni, hogy szuperforrás-e.

1 él megkérdezése legfeljebb 1 csúcsot zár ki mint lehetséges szuperforrást.

Egy algoritmus első $n - 2$ kérdése után még legalább két csúcs – mondjuk i és j – lehet szuperforrás. Ahhoz, hogy belássuk, hogy i szuperforrás, meg kell vizsgálni az i-edik sor és i-edik oszlop minden elemét.

Vagy i-re vagy j-re igaz lesz, hogy az első $n - 2$ kérdés közül legfeljebb $(n - 2)/2$ kérdés vonatkozott rá. Így még $2n - 2 - (n - 2)/2$ legalább kérdés kell.

A költség elemzése

Jelölje $T(n)$ a legjobb (leggyorsabb) algoritmus által megvizsgált mátrix-elemek számának maximumát az összes n pontú gráfra.

Tudjuk, hogy $T(n) \leq 3n - 3$.

Nyilvánvaló, hogy $2n - 2 \leq T(n)$, mert le kell ellenőrizni, hogy szuperforrás-e.

1 él megkérdezése legfeljebb 1 csúcsot zár ki mint lehetséges szuperforrást.

Egy algoritmus első $n - 2$ kérdése után még legalább két csúcs – mondjuk i és j – lehet szuperforrás. Ahhoz, hogy belássuk, hogy i szuperforrás, meg kell vizsgálni az i-edik sor és i-edik oszlop minden elemét.

Vagy i-re vagy j-re igaz lesz, hogy az első $n - 2$ kérdés közül legfeljebb $(n - 2)/2$ kérdés vonatkozott rá. Így még $2n - 2 - (n - 2)/2$ legalább kérdés kell. Tehát

$$T(n) \geq 2n - 2 - \frac{n-2}{2} + n - 2 = 3n - 4 - \frac{n-2}{2}.$$

Azaz a fenti algoritmus közel optimális.

Rendezési reláció

Legyen U egy halmaz, és $<$ egy kétváltozós reláció U-n. Ha $a, b \in U$ és $a < b$, akkor azt mondjuk, hogy a kisebb, mint b.

Rendezési reláció

Legyen U egy halmaz, és $<$ egy kétváltozós reláció U-n. Ha $a, b \in U$ és $a < b$, akkor azt mondjuk, hogy a kisebb, mint b. A $<$ reláció egy rendezés, ha teljesülnek a következők:

1. $a \not< a$ minden $a \in U$ elemre ($<$ irreflexív);
2. Ha $a, b, c \in U$, $a < b$, és $b < c$, akkor $a < c$ ($<$ tranzitív);
3. Tetszőleges $a \neq b \in U$ elemekre vagy $a < b$, vagy $b < a$ fennáll ($<$ teljes).

Rendezési reláció

Legyen U egy halmaz, és $<$ egy kétváltozós reláció U-n. Ha $a, b \in U$ és $a < b$, akkor azt mondjuk, hogy a kisebb, mint b. A $<$ reláció egy rendezés, ha teljesülnek a következők:

1. $a \not< a$ minden $a \in U$ elemre ($<$ irreflexív);
2. Ha $a, b, c \in U$, $a < b$, és $b < c$, akkor $a < c$ ($<$ tranzitív);
3. Tetszőleges $a \neq b \in U$ elemekre vagy $a < b$, vagy $b < a$ fennáll ($<$ teljes).

Ha $<$ egy rendezés U-n, akkor az $(U, <)$ párt rendezett halmaznak nevezzük.

Példák:

- $\mathbb{Z}$ az egész számok halmaza. A $<$ rendezés a nagyság szerinti rendezés.

Rendezési reláció

Legyen U egy halmaz, és $<$ egy kétváltozós reláció U-n. Ha $a, b \in U$ és $a < b$, akkor azt mondjuk, hogy a kisebb, mint b. A $<$ reláció egy rendezés, ha teljesülnek a következők:

1. $a \not< a$ minden $a \in U$ elemre ($<$ irreflexív);
2. Ha $a, b, c \in U$, $a < b$, és $b < c$, akkor $a < c$ ($<$ tranzitív);
3. Tetszőleges $a \neq b \in U$ elemekre vagy $a < b$, vagy $b < a$ fennáll ($<$ teljes).

Ha $<$ egy rendezés U-n, akkor az $(U, <)$ párt rendezett halmaznak nevezzük.

Példák:

- $\mathbb{Z}$ az egész számok halmaza. A $<$ rendezés a nagyság szerinti rendezés.
- Az abc betűinek Σ halmaza; a $<$ rendezést az abc-*sorrend* adja. Az x betű kisebb, mint az y betű, ha x előbb szerepel az abc-sorrendben, mint y.

Rendezési reláció

Legyen U egy halmaz, és $<$ egy kétváltozós reláció U-n. Ha $a, b \in U$ és $a < b$, akkor azt mondjuk, hogy a kisebb, mint b. A $<$ reláció egy rendezés, ha teljesülnek a következők:

1. $a \not< a$ minden $a \in U$ elemre ($<$ irreflexív);
2. Ha $a, b, c \in U$, $a < b$, és $b < c$, akkor $a < c$ ($<$ tranzitív);
3. Tetszőleges $a \neq b \in U$ elemekre vagy $a < b$, vagy $b < a$ fennáll ($<$ teljes).

Ha $<$ egy rendezés U-n, akkor az $(U, <)$ párt rendezett halmaznak nevezzük.

Példák:

- $\mathbb{Z}$ az egész számok halmaza. A $<$ rendezés a nagyság szerinti rendezés.
- Az abc betűinek Σ halmaza; a $<$ rendezést az abc-*sorrend* adja. Az x betű kisebb, mint az y betű, ha x előbb szerepel az abc-sorrendben, mint y.

- A Σ betűiből alkotott szavak Σ^* halmaza a szótárszerű vagy lexikografikus rendezéssel. $\Rightarrow$ legyen $X = x_1x_2\cdots x_k$ és $Y = y_1y_2\cdots y_l$ két szó. Az X kisebb mint Y, ha vagy $l > k$ és $x_i = y_i$ ha $i = 1, 2, \ldots, k$; vagy pedig $x_j < y_j$ teljesül a legkisebb olyan j indexre, melyre $x_j \neq y_j$. Tehát például $kar < karika$ és $bor < bot$.

- A Σ betűiből alkotott szavak Σ^* halmaza a szótárszerű vagy lexikografikus rendezéssel. $\Rightarrow$ legyen $X = x_1x_2\cdots x_k$ és $Y = y_1y_2\cdots y_l$ két szó. Az X kisebb mint Y, ha vagy $l > k$ és $x_i = y_i$ ha $i = 1, 2, \ldots, k$; vagy pedig $x_j < y_j$ teljesül a legkisebb olyan j indexre, melyre $x_j \neq y_j$. Tehát például $kar < karika$ és $bor < bot$.

A rendezés feladata: *adott az* $(U, <)$ *rendezett halmaz elemeinek egy* $u_1, u_2, \ldots, u_n$ *sorozata; rendezzük ezt át egy nem csökkenő* $v_1, v_2, \ldots, v_n$ *sorrendbe.*

- A Σ betűiből alkotott szavak Σ^* halmaza a szótárszerű vagy lexikografikus rendezéssel. $\Rightarrow$ legyen $X = x_1x_2\cdots x_k$ és $Y = y_1y_2\cdots y_l$ két szó. Az X kisebb mint Y, ha vagy $l > k$ és $x_i = y_i$ ha $i = 1, 2, \ldots, k$; vagy pedig $x_j < y_j$ teljesül a legkisebb olyan j indexre, melyre $x_j \neq y_j$. Tehát például $kar < karika$ és $bor < bot$.

A rendezés feladata: *adott az $(U, <)$ rendezett halmaz elemeinek egy $u_1, u_2, \ldots, u_n$ sorozata; rendezzük ezt át egy nem csökkenő $v_1, v_2, \ldots, v_n$ sorrendbe.*

Input: tömb, láncolt lista, (vagy bármi)

- A Σ betűiből alkotott szavak Σ^* halmaza a szótárszerű vagy lexikografikus rendezéssel. $\Rightarrow$ legyen $X = x_1x_2\cdots x_k$ és $Y = y_1y_2\cdots y_l$ két szó. Az X kisebb mint Y, ha vagy $l > k$ és $x_i = y_i$ ha $i = 1, 2, \ldots, k$; vagy pedig $x_j < y_j$ teljesül a legkisebb olyan j indexre, melyre $x_j \neq y_j$. Tehát például $kar < karika$ és $bor < bot$.

A rendezés feladata: *adott az $(U, <)$ rendezett halmaz elemeinek egy $u_1, u_2, \ldots, u_n$ sorozata; rendezzük ezt át egy nem csökkenő $v_1, v_2, \ldots, v_n$ sorrendbe.*

Input: tömb, láncolt lista, (vagy bármi)

Output: általában, mint az Input

- A Σ betűiből alkotott szavak Σ^* halmaza a szótárszerű vagy lexikografikus rendezéssel. $\Rightarrow$ legyen $X = x_1x_2\cdots x_k$ és $Y = y_1y_2\cdots y_l$ két szó. Az X kisebb mint Y, ha vagy $l > k$ és $x_i = y_i$ ha $i = 1, 2, \ldots, k$; vagy pedig $x_j < y_j$ teljesül a legkisebb olyan j indexre, melyre $x_j \neq y_j$. Tehát például $kar < karika$ és $bor < bot$.

A rendezés feladata: *adott az* $(U, <)$ *rendezett halmaz elemeinek egy* $u_1, u_2, \ldots, u_n$ *sorozata; rendezzük ezt át egy nem csökkenő* $v_1, v_2, \ldots, v_n$ *sorrendbe.*

Input: tömb, láncolt lista, (vagy bármi)

Output: általában, mint az Input

Lépések: elemek mozgatása, cseréje, összehasonlítása

- A Σ betűiből alkotott szavak Σ^* halmaza a szótárszerű vagy lexikografikus rendezéssel. $\Rightarrow$ legyen $X = x_1x_2\cdots x_k$ és $Y = y_1y_2\cdots y_l$ két szó. Az X kisebb mint Y, ha vagy $l > k$ és $x_i = y_i$ ha $i = 1, 2, \ldots, k$; vagy pedig $x_j < y_j$ teljesül a legkisebb olyan j indexre, melyre $x_j \neq y_j$. Tehát például $kar < karika$ és $bor < bot$.

A rendezés feladata: *adott az $(U, <)$ rendezett halmaz elemeinek egy $u_1, u_2, \ldots, u_n$ sorozata; rendezzük ezt át egy nem csökkenő $v_1, v_2, \ldots, v_n$ sorrendbe.*

Input: tömb, láncolt lista, (vagy bármi)

Output: általában, mint az Input

Lépések: elemek mozgatása, cseréje, összehasonlítása

A rendezés önmagában is előforduló feladat, de előjön, mint hasznos adatstruktúra is.

- A Σ betűiből alkotott szavak Σ^* halmaza a szótárszerű vagy lexikografikus rendezéssel. $\Rightarrow$ legyen $X = x_1x_2\cdots x_k$ és $Y = y_1y_2\cdots y_l$ két szó. Az X kisebb mint Y, ha vagy $l > k$ és $x_i = y_i$ ha $i = 1, 2, \ldots, k$; vagy pedig $x_j < y_j$ teljesül a legkisebb olyan j indexre, melyre $x_j \neq y_j$. Tehát például $kar < karika$ és $bor < bot$.

A rendezés feladata: *adott az* $(U, <)$ *rendezett halmaz elemeinek egy* $u_1, u_2, \ldots, u_n$ *sorozata; rendezzük ezt át egy nem csökkenő* $v_1, v_2, \ldots, v_n$ *sorrendbe.*

Input: tömb, láncolt lista, (vagy bármi)

Output: általában, mint az Input

Lépések: elemek mozgatása, cseréje, összehasonlítása

A rendezés önmagában is előforduló feladat, de előjön, mint hasznos adatstruktúra is.
Rendezett halmazban könnyebb keresni (pl. telefonkönyv).

Keresés rendezett halmazban

Bar Kochba játék: gondolok egy számot $1 - 100$-ig, hány eldöntendő kérdésből lehet kitalálni?

Keresés rendezett halmazban

Bar Kochba játék: gondolok egy számot $1 - 100$-ig, hány eldöntendő kérdésből lehet kitalálni?

Adott az $(U, <)$ rendezett halmaz véges $S = \{s_1 < s_2 < \ldots < s_{n-1} < s_n\}$ és $s \in U$. részhalmaza.

Összehasonlításokkal akarjuk eldönteni, hogy igaz-e $s \in S$. Hány összehasonlítás kell?

Keresés rendezett halmazban

Bar Kochba játék: gondolok egy számot $1 - 100$-ig, hány eldöntendő kérdésből lehet kitalálni?

Adott az $(U, <)$ rendezett halmaz véges $S = \{s_1 < s_2 < \ldots < s_{n-1} < s_n\}$ és $s \in U$. részhalmaza.

Összehasonlításokkal akarjuk eldönteni, hogy igaz-e $s \in S$. Hány összehasonlítás kell?

Lineáris keresés

Sorban mindegyik elemmel összehasonlítjuk.

Keresés rendezett halmazban

Bar Kochba játék: gondolok egy számot $1 - 100$-ig, hány eldöntendő kérdésből lehet kitalálni?

Adott az $(U, <)$ rendezett halmaz véges $S = \{s_1 < s_2 < \ldots < s_{n-1} < s_n\}$ és $s \in U$. részhalmaza.

Összehasonlításokkal akarjuk eldönteni, hogy igaz-e $s \in S$. Hány összehasonlítás kell?

Lineáris keresés

Sorban mindegyik elemmel összehasonlítjuk.
Költség a legrosszabb esetben: n, mert lehet, hogy pont az utolsó volt.

Keresés rendezett halmazban

Bar Kochba játék: gondolok egy számot $1 - 100$-ig, hány eldöntendő kérdésből lehet kitalálni?

Adott az $(U, <)$ rendezett halmaz véges $S = \{s_1 < s_2 < \ldots < s_{n-1} < s_n\}$ és $s \in U$. részhalmaza.

Összehasonlításokkal akarjuk eldönteni, hogy igaz-e $s \in S$. Hány összehasonlítás kell?

Lineáris keresés

Sorban mindegyik elemmel összehasonlítjuk.
Költség a legrosszabb esetben: n, mert lehet, hogy pont az utolsó volt.
Költség átlagos esetben esetben: $(n/2) + 1$

Bináris keresés

Oszd meg és uralkodj: először a középső s_i-vel hasonlítunk.

Bináris keresés

Oszd meg és uralkodj: először a középső s_i-vel hasonlítunk.
Hasonló feladatot kapunk egy S_1 halmazra, amire viszont $|S_1| \leq |S|/2$.

Bináris keresés

Oszd meg és uralkodj: először a középső s_i-vel hasonlítunk.
Hasonló feladatot kapunk egy S_1 halmazra, amire viszont $|S_1| \leq |S|/2$.

És így tovább:

$$|S_2| \leq \frac{|S|}{4}, |S_3| \leq \frac{|S|}{2^3}, \ldots |S_k| \leq \frac{|S|}{2^k}$$

Bináris keresés

Oszd meg és uralkodj: először a középső s_i-vel hasonlítunk.
Hasonló feladatot kapunk egy S_1 halmazra, amire viszont $|S_1| \leq |S|/2$.

És így tovább:

$$|S_2| \leq \frac{|S|}{4}, |S_3| \leq \frac{|S|}{2^3}, \ldots |S_k| \leq \frac{|S|}{2^k}$$

Pl. keressük meg, benne van-e 21 az alábbi sorozatban!

$$15, 22, 25, 37, \boxed{48}, 56, 70, 82 \quad (1)$$

Bináris keresés

Oszd meg és uralkodj: először a középső s_i-vel hasonlítunk.
Hasonló feladatot kapunk egy S_1 halmazra, amire viszont $|S_1| \leq |S|/2$.

És így tovább:

$$|S_2| \leq \frac{|S|}{4}, |S_3| \leq \frac{|S|}{2^3}, \ldots |S_k| \leq \frac{|S|}{2^k}$$

Pl. keressük meg, benne van-e 21 az alábbi sorozatban!

$$15, 22, 25, 37, \boxed{48}, 56, 70, 82 \quad (1)$$

$$15, 22, \boxed{25}, 37, 48, 56, 70, 82 \quad (2)$$

Bináris keresés

Oszd meg és uralkodj: először a középső s_i-vel hasonlítunk.
Hasonló feladatot kapunk egy S_1 halmazra, amire viszont $|S_1| \leq |S|/2$.

És így tovább:

$$|S_2| \leq \frac{|S|}{4}, |S_3| \leq \frac{|S|}{2^3}, \ldots |S_k| \leq \frac{|S|}{2^k}$$

Pl. keressük meg, benne van-e 21 az alábbi sorozatban!

$$15, 22, 25, 37, \boxed{48}, 56, 70, 82 \quad (1)$$

$$15, 22, \boxed{25}, 37, 48, 56, 70, 82 \quad (2)$$

$$15, \boxed{22}, 25, 37, 48, 56, 70, 82 \quad (3)$$

Bináris keresés

Oszd meg és uralkodj: először a középső s_i-vel hasonlítunk.
Hasonló feladatot kapunk egy S_1 halmazra, amire viszont $|S_1| \leq |S|/2$.

És így tovább:

$$|S_2| \leq \frac{|S|}{4}, |S_3| \leq \frac{|S|}{2^3}, \ldots |S_k| \leq \frac{|S|}{2^k}$$

Pl. keressük meg, benne van-e 21 az alábbi sorozatban!

$$15, 22, 25, 37, \boxed{48}, 56, 70, 82 \quad (1)$$

$$15, 22, \boxed{25}, 37, 48, 56, 70, 82 \quad (2)$$

$$15, \boxed{22}, 25, 37, 48, 56, 70, 82 \quad (3)$$

$$\boxed{15}, 22, 25, 37, 48, 56, 70, 82 \quad (4)$$

Bináris keresés

Addig kell csinálni, amig $|S_k| \geq 1$, vagyis $1 \leq \frac{n}{2^k}$.

Bináris keresés

Addig kell csinálni, amig $|S_k| \geq 1$, vagyis $1 \leq \frac{n}{2^k}$.

$\implies 2^k \leq n \implies k \leq \lfloor \log_2 n \rfloor$

Bináris keresés

Addig kell csinálni, amig $|S_k| \geq 1$, vagyis $1 \leq \frac{n}{2^k}$.

$\implies 2^k \leq n \implies k \leq \lfloor \log_2 n \rfloor$

Ez $k+1$ összehasonlítás volt. $\implies k+1 \leq \lfloor \log_2 n \rfloor + 1 \leq \lceil \log_2(n+1) \rceil$

Bináris keresés

Addig kell csinálni, amig $|S_k| \geq 1$, vagyis $1 \leq \frac{n}{2^k}$.
$\implies 2^k \leq n \implies k \leq \lfloor \log_2 n \rfloor$
Ez $k+1$ összehasonlítás volt. $\implies k+1 \leq \lfloor \log_2 n \rfloor + 1 \leq \lceil \log_2(n+1) \rceil$
Tétel. *Ez optimális, nincs olyan kereső algoritmus, ami minden esetben kevesebb mint* $\lceil \log_2(n+1) \rceil$ *kérdést használ.*

Bizonyítás: Az ellenség nem is gondol egy számra, csak mindig úgy válaszol, hogy minél többet kelljen kérdezni.

Bináris keresés

Addig kell csinálni, amig $|S_k| \geq 1$, vagyis $1 \leq \frac{n}{2^k}$.
$\implies 2^k \leq n \implies k \leq \lfloor \log_2 n \rfloor$
Ez $k+1$ összehasonlítás volt. $\implies k+1 \leq \lfloor \log_2 n \rfloor + 1 \leq \lceil \log_2(n+1) \rceil$
Tétel. *Ez optimális, nincs olyan kereső algoritmus, ami minden esetben kevesebb mint* $\lceil \log_2(n+1) \rceil$ *kérdést használ.*

Bizonyítás: Az ellenség nem is gondol egy számra, csak mindig úgy válaszol, hogy minél többet kelljen kérdezni.
Ha egy kérdést felteszek, és az igen válasz után mondjuk szóba jön x lehetőség, akkor a nem esetén szóba jön még $n - x$ lehetőség.

Bináris keresés

Addig kell csinálni, amig $|S_k| \geq 1$, vagyis $1 \leq \frac{n}{2^k}$.
$\implies 2^k \leq n \implies k \leq \lfloor \log_2 n \rfloor$
Ez $k+1$ összehasonlítás volt. $\implies k+1 \leq \lfloor \log_2 n \rfloor + 1 \leq \lceil \log_2(n+1) \rceil$
Tétel. *Ez optimális, nincs olyan kereső algoritmus, ami minden esetben kevesebb mint* $\lceil \log_2(n+1) \rceil$ *kérdést használ.*

Bizonyítás: Az ellenség nem is gondol egy számra, csak mindig úgy válaszol, hogy minél többet kelljen kérdezni.
Ha egy kérdést felteszek, és az igen válasz után mondjuk szóba jön x lehetőség, akkor a nem esetén szóba jön még $n - x$ lehetőség.
Az ellenség úgy válaszol, hogy minél több lehetőség maradjon, így el tudja érni, hogy legalább $n/2$ marad. $\implies k$ kérdés után is marad még $\frac{n}{2^k}$ lehetőség.

Bináris keresés

Addig kell csinálni, amig $|S_k| \geq 1$, vagyis $1 \leq \frac{n}{2^k}$.
$\implies 2^k \leq n \implies k \leq \lfloor \log_2 n \rfloor$
Ez $k+1$ összehasonlítás volt. $\implies k+1 \leq \lfloor \log_2 n \rfloor + 1 \leq \lceil \log_2(n+1) \rceil$
Tétel. *Ez optimális, nincs olyan kereső algoritmus, ami minden esetben kevesebb mint* $\lceil \log_2(n+1) \rceil$ *kérdést használ.*

Bizonyítás: Az ellenség nem is gondol egy számra, csak mindig úgy válaszol, hogy minél többet kelljen kérdezni.
Ha egy kérdést felteszek, és az igen válasz után mondjuk szóba jön x lehetőség, akkor a nem esetén szóba jön még $n - x$ lehetőség.
Az ellenség úgy válaszol, hogy minél több lehetőség maradjon, így el tudja érni, hogy legalább $n/2$ marad. $\implies k$ kérdés után is marad még $\frac{n}{2^k}$ lehetőség.
Ha tehát $\frac{n}{2^k} > 1$, akkor nem tudom, hogy az-e a gondolt szám, vagy nincs benne a sorozatban. Tehát még egy kérdésre szükség van.
$\implies \frac{n}{2^k} > 1 \implies n > 2^k \implies \log_2 n > k$.