Mesterséges intelligencia előadássorozat

Az előadás diái az AIMA könyvre épülve (http://aima.cs.berkeley.edu) készültek a University of California, Berkeley mesterséges intelligencia kurzusának anyagainak felhasználásával (http://ai.berkeley.edu).

These slides are based on the AIMA book (http://aima.cs.berkeley.edu) and were adapted from the AI course material of University of California, Berkeley (http://ai.berkeley.edu).

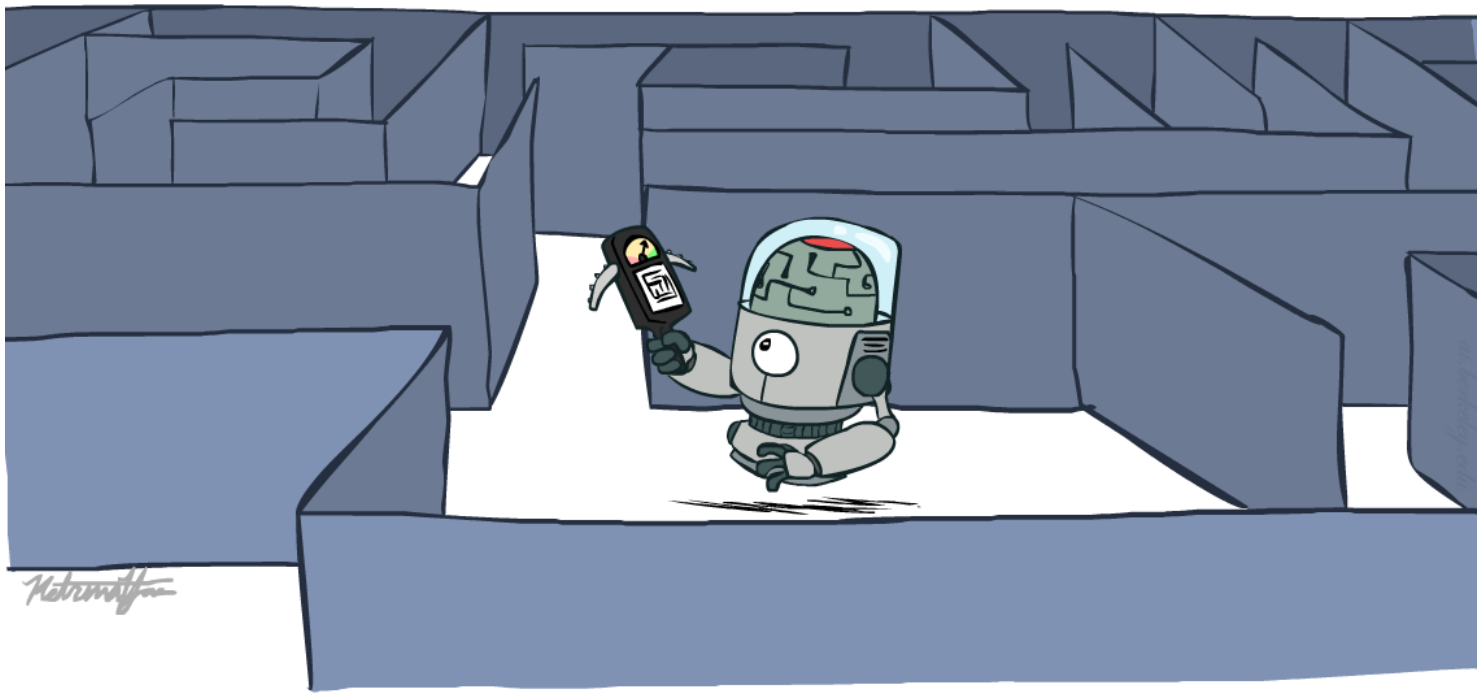
Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Mesterséges Intelligencia és Rendszertervezés Tanszék

Mesterséges intelligencia

Problémamegoldás kereséssel 2.

Előadó: Dr. Hullám Gábor

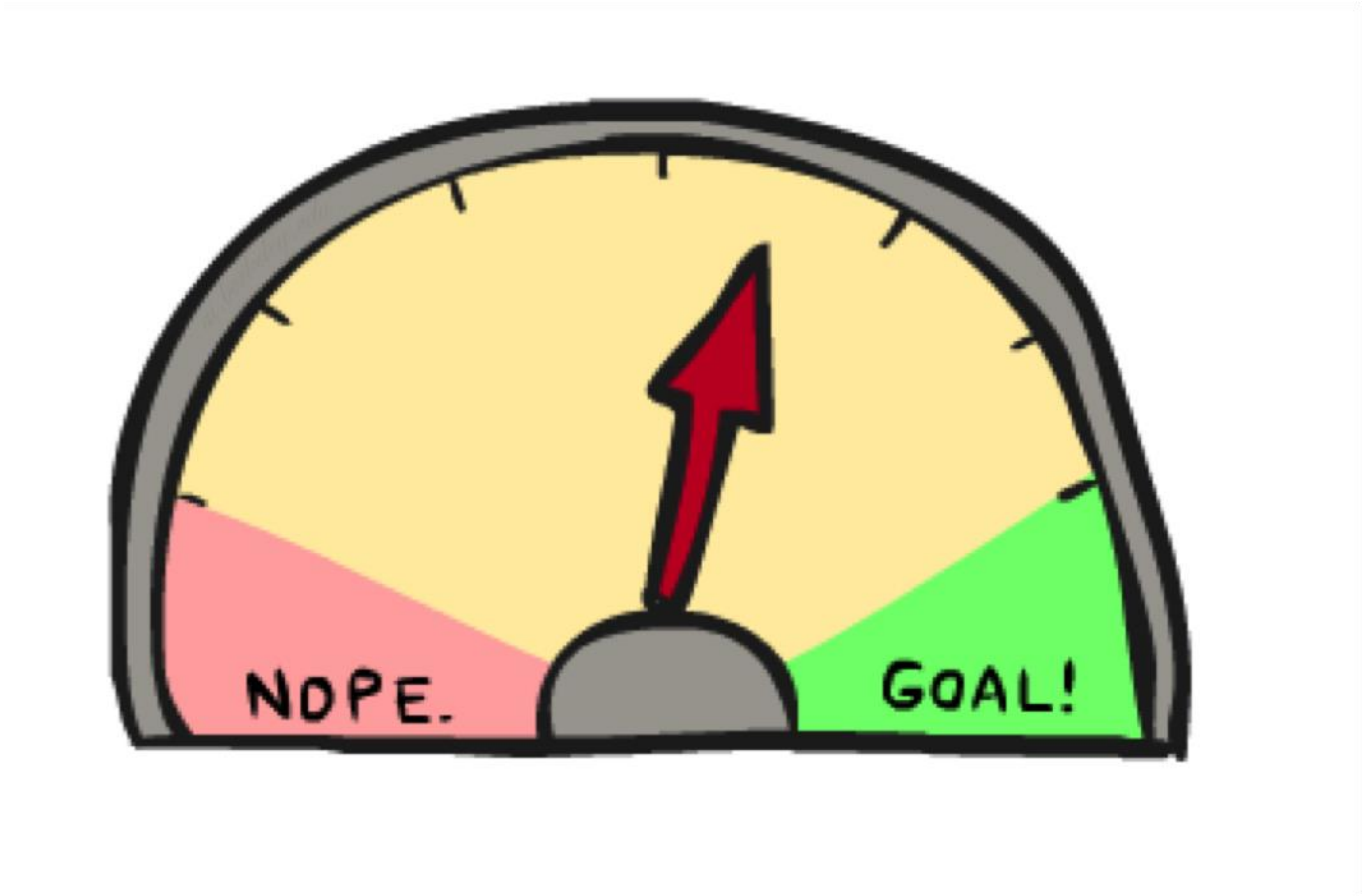
Informált keresés



Tartalom

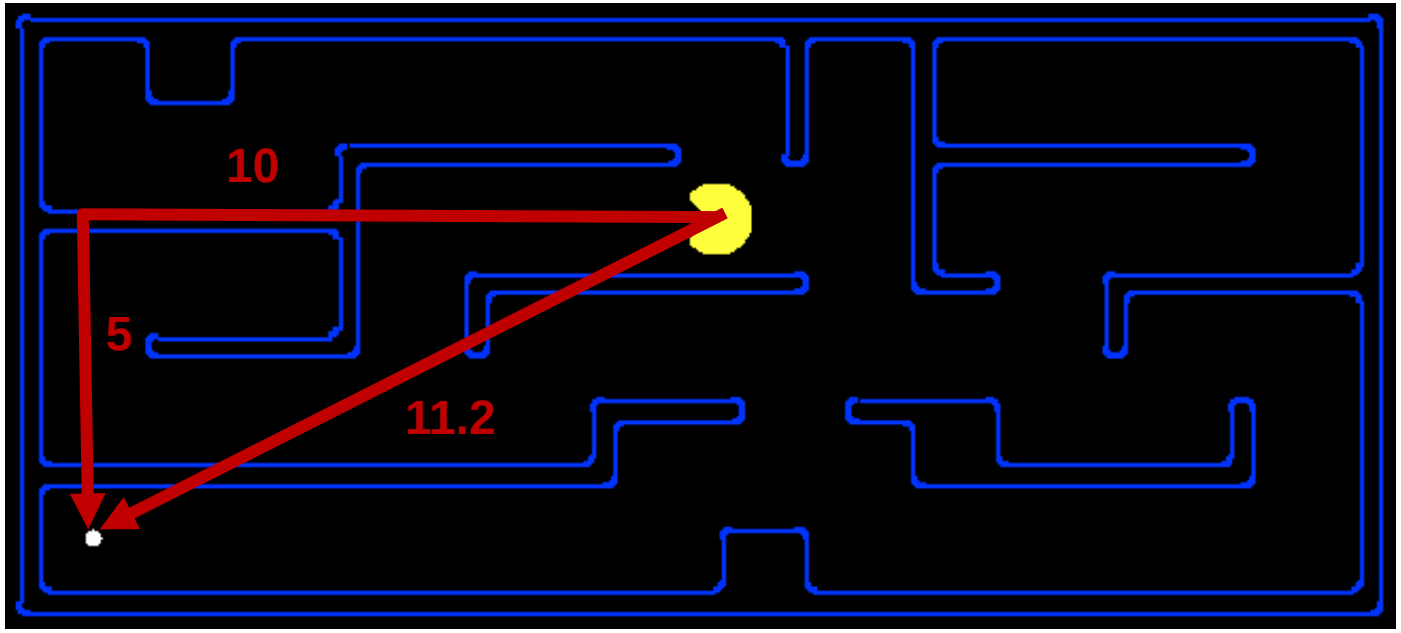
- Informált keresés
  - Heurisztikák
  - Mohó keresés
  - A* keresés
- Gráf keresés

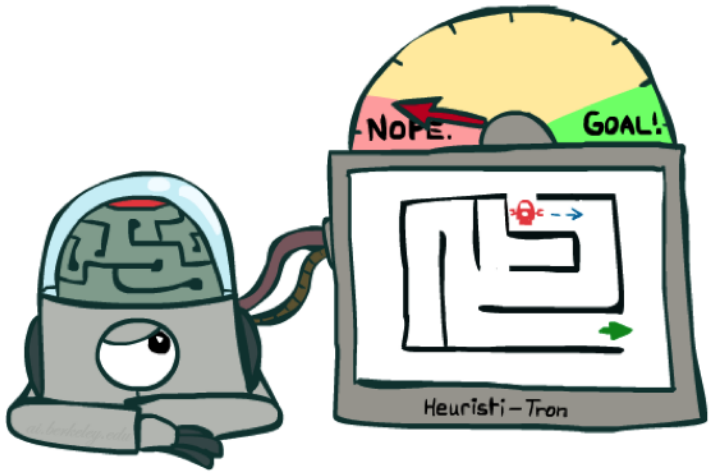
Informált keresés

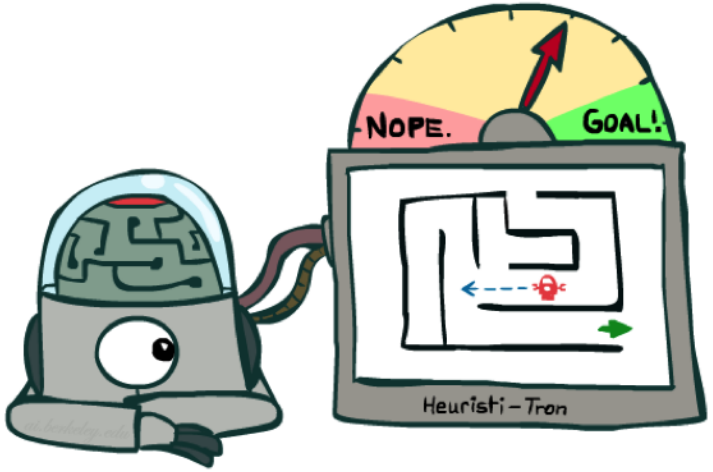


Keresési heurisztika

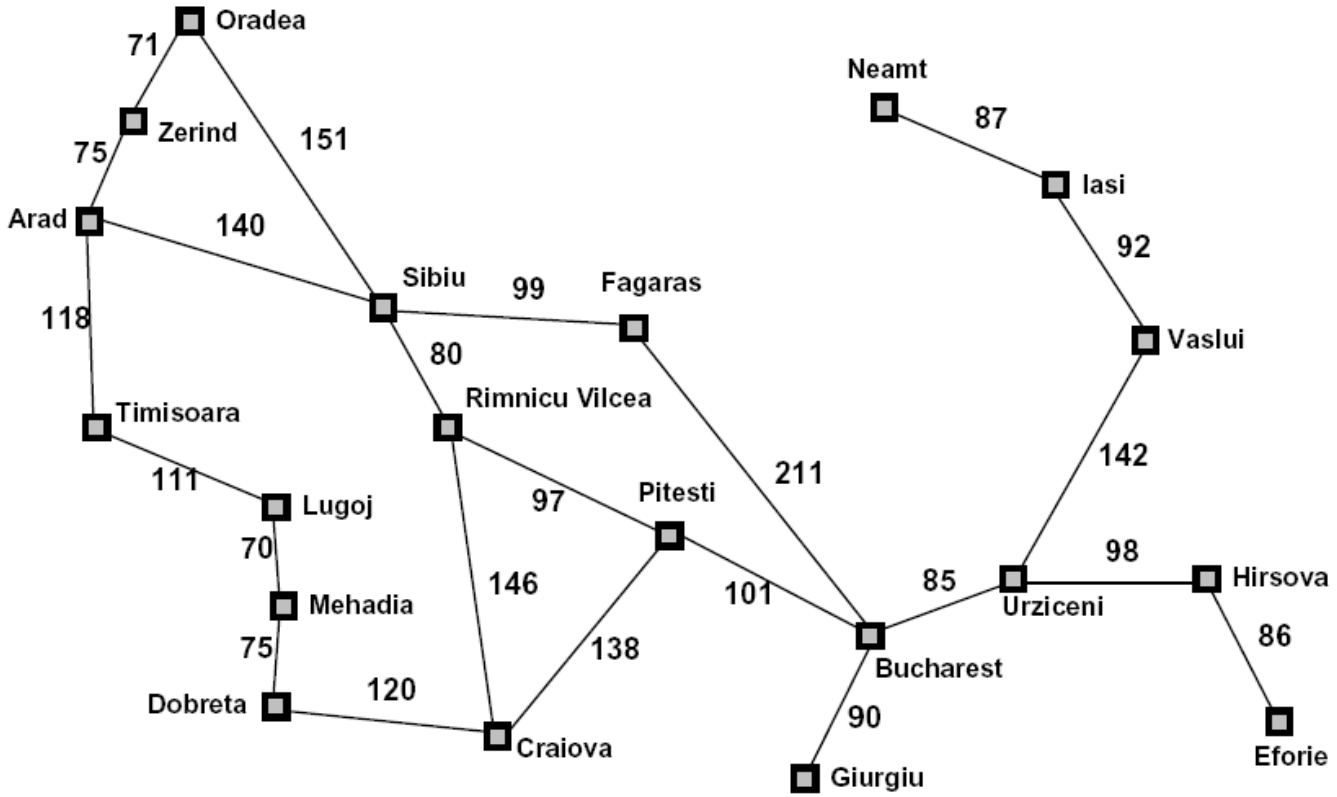
- A heurisztika:
  - Egy függvény, ami becslést ad arra, hogy az adott állapot mennyire van közel a célállapothoz
  - Egy adott keresési problémához kell tervezni
  - Példák: Manhattan távolság, euklideszi távolság (útvonal keresésnél)







Példa heurisztika: légvonalbeli távolság

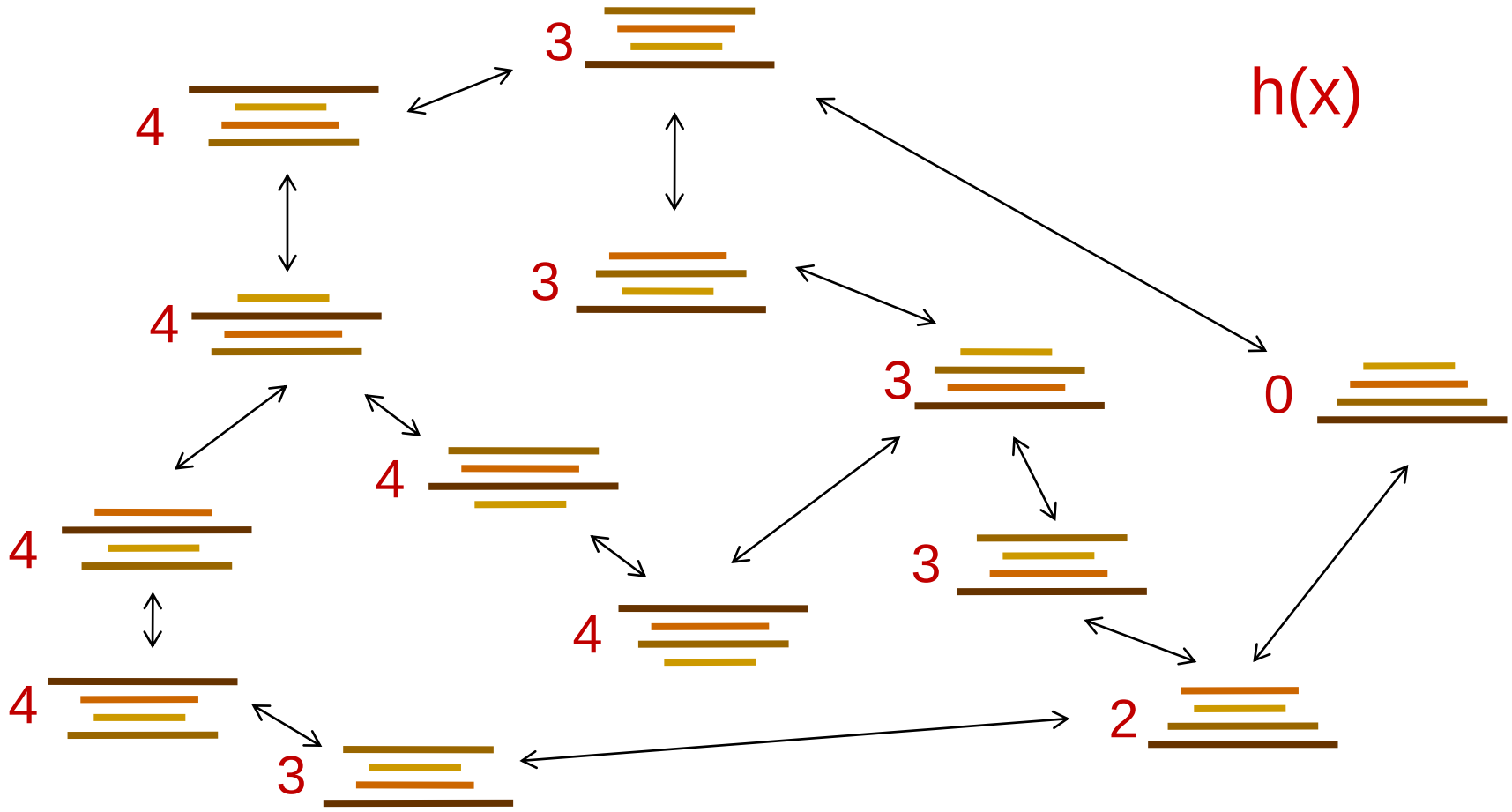


Straight−line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

h(x)

Példa heurisztika

Heurisztika: a legnagyobb rossz helyen lévő palacsinta „sorszáma”



Keresési stratégiák

Heurisztikus, vagy más néven **informált keresés**

- Büntessük a hátra (céltól elfele) keresést!
- De ugyanaz, és könnyebb díjazni az előrekeresést a célállapot irányába.

Ehhez kell valami elképzelés, hogy a cél:

- ***merrefelé és,***
- ***nagyjából milyen messzire fekszik*.**

Ez az információ az un. **heurisztika**, **heurisztikus függvény h(*n*)**,

Keresési stratégiák – 2

Heurisztika, **heurisztikus függvény h(*n*)**

- a probléma minden *n* állapotára ki kell tudnunk számítani
  kifejezi a **célig előrehaladás becsült költség**ét

- ha pontos, akkor elvben fölöslegessé teszi a keresést

(ha nagyon pontatlan, akkor viszont semmit sem segít)

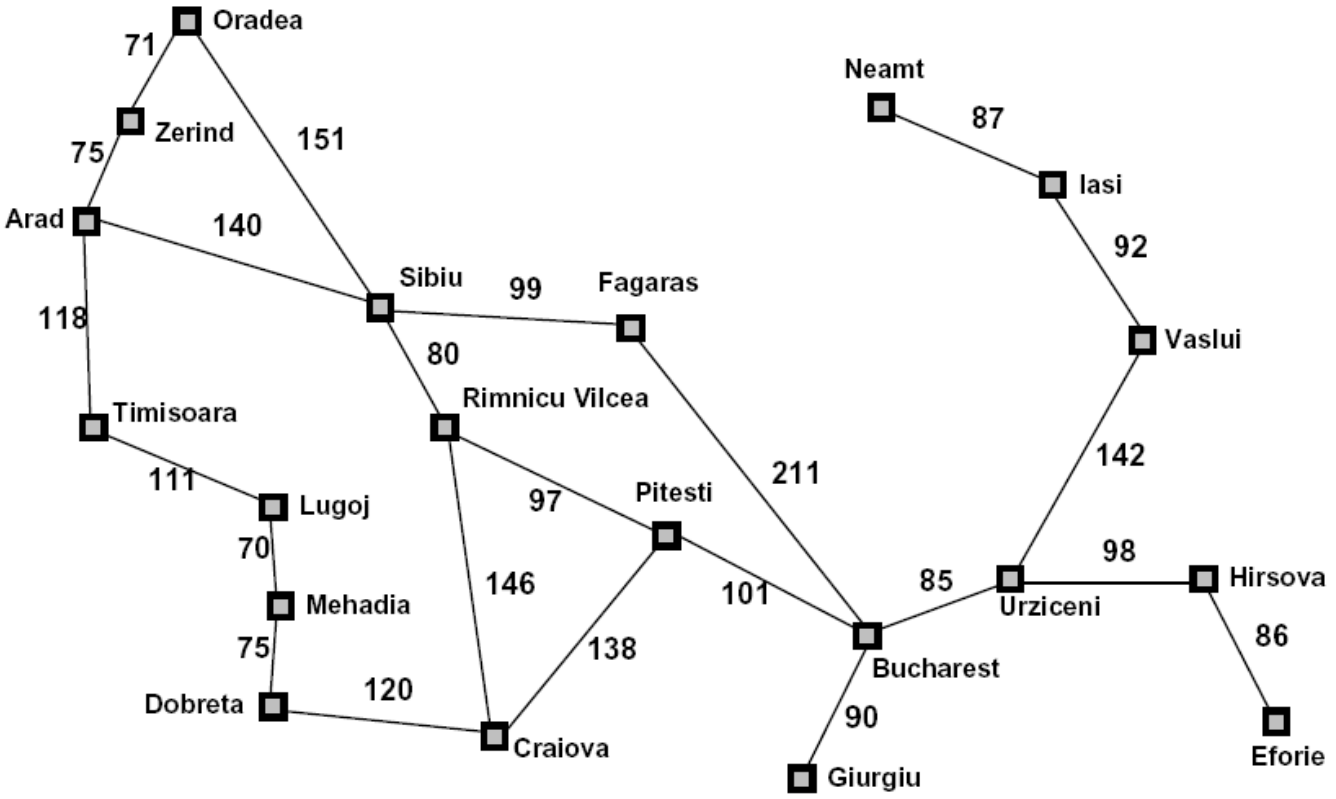
Ez minden problémára más, problémaspecifikus! $f(n) = g(n) + h(n)$

A heurisztikus függvény, h(*n*) becslést ad arra, hogy mekkora a hátralévő út legkisebb költsége.

- Ezáltal becslést ad a teljes út legjobb költségére is (f(n)).

Mohó keresés

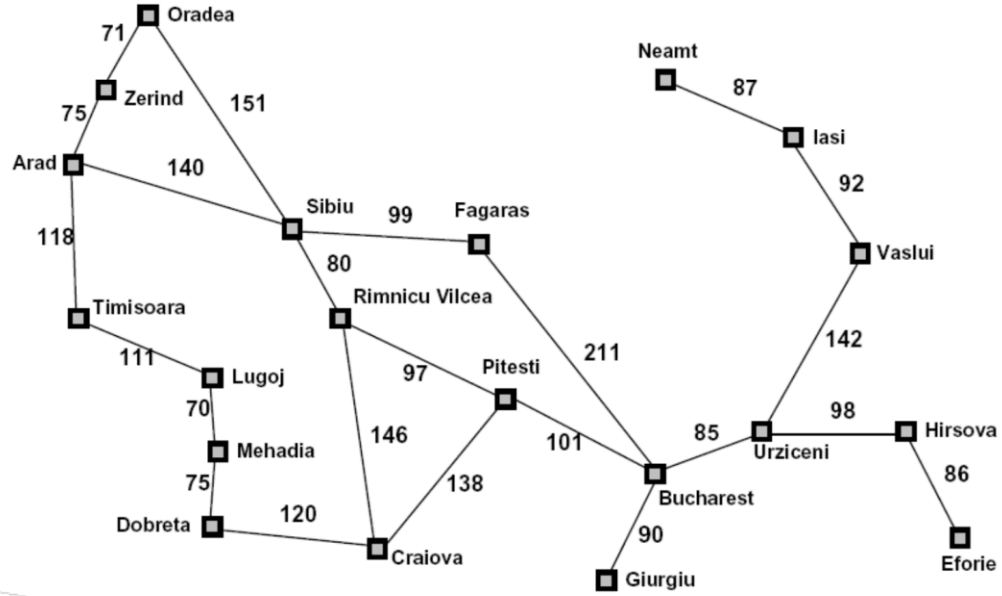
Mohó keresés - heurisztika



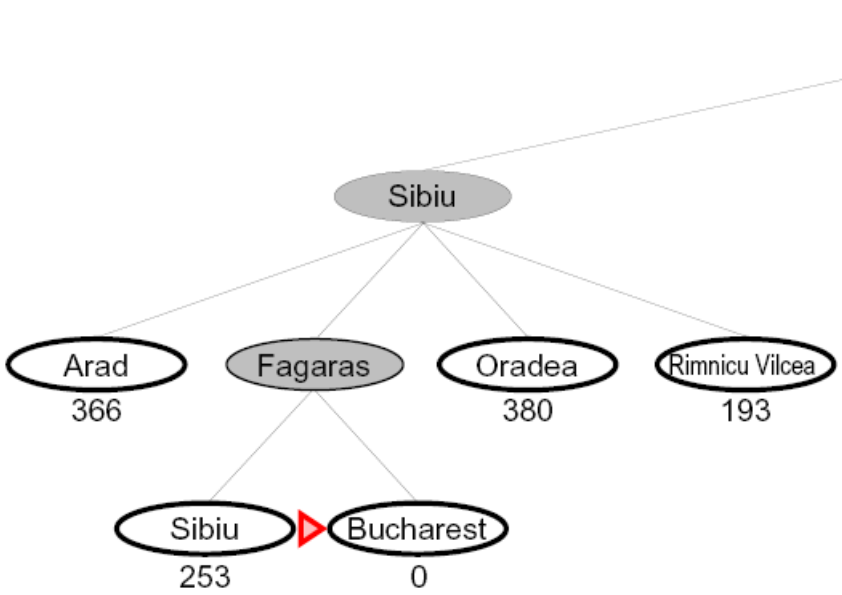
Straight−line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

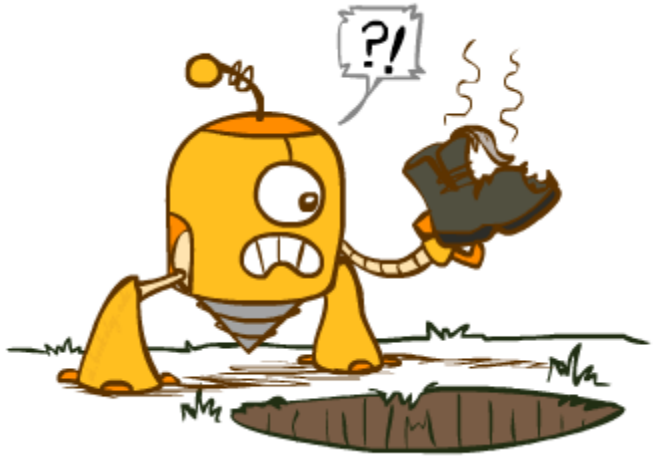
h(x)

Mohó keresés



- A legközelebbi csomópont kifejtése lenne célszerű

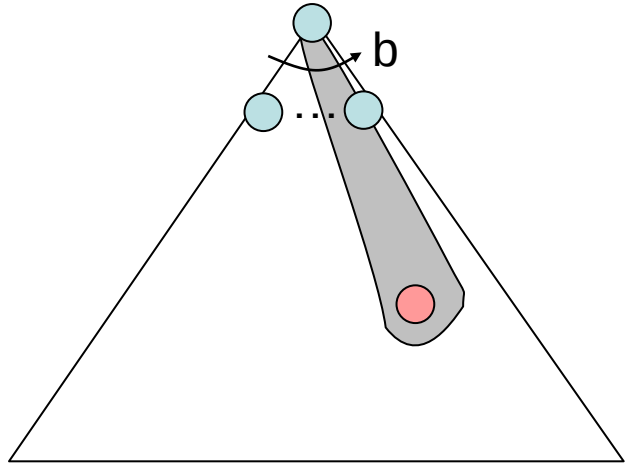




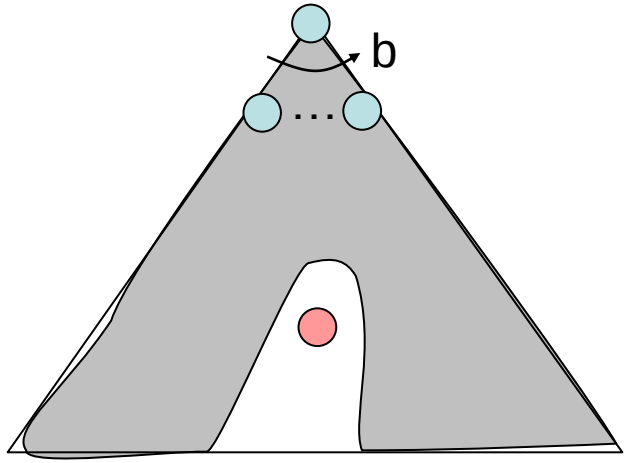
- Mi mehet félre?

Mohó keresés

- Stratégia: fejtsük ki azt a csomópontot, amiről azt gondoljuk, hogy a legközelebb van a célállapothoz
  - Heurisztika: a legközelebbi célállapottól való becsült távolság



- Gyakori esetben:
  - A legjobbat először (mohó) keresés eljuttat egy célállapothoz, ami nem biztos, hogy a legjobb

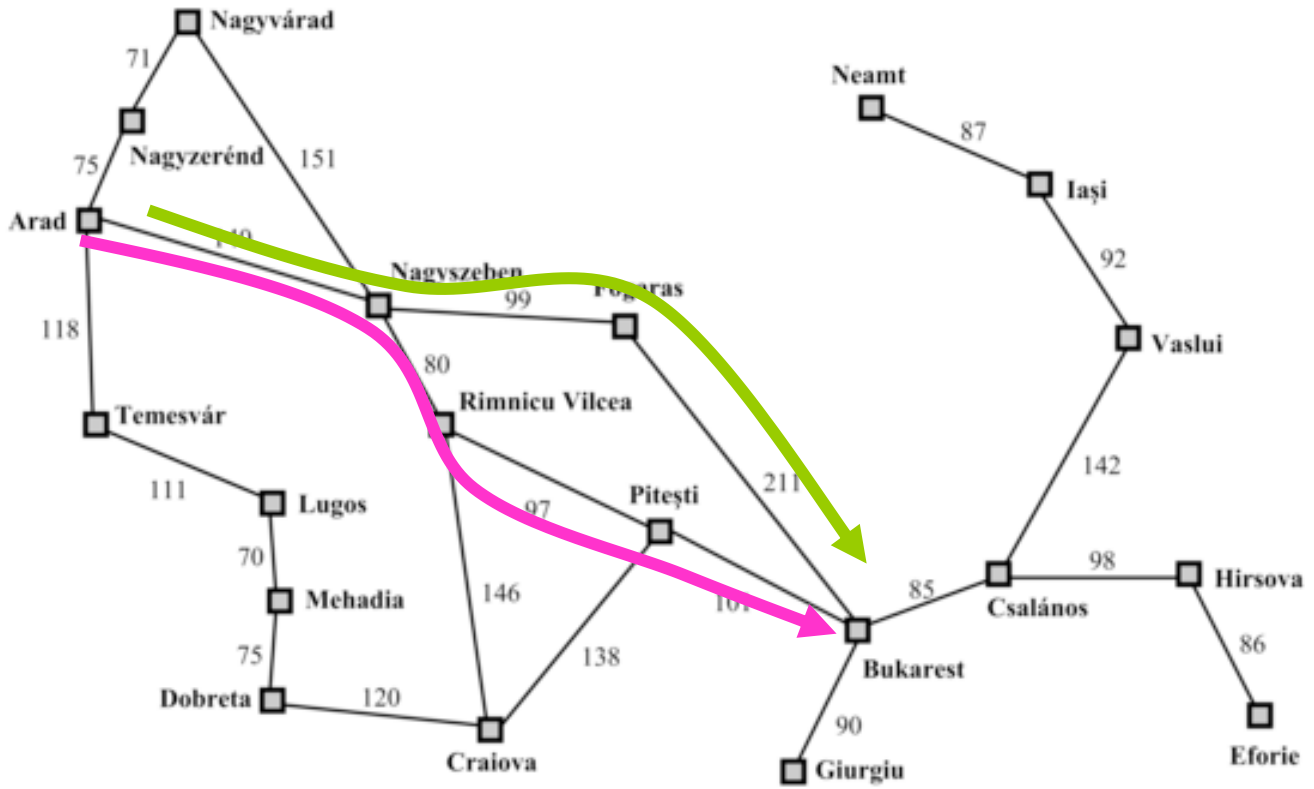


- Legrosszabb esetben:
  - egy rosszul irányított mélységi keresést kapunk

Mohó keresés

A célhoz legközelebbinek tűnő csomópont először (a legjobbnak becsültet először)

Stratégia: a következő lépésben azt a csp-t fejti ki, amelyhez rendelt probléma-állapotot a legközelebbinek ítéli a célállapothoz (legkisebb az *n*-dik csp-hoz rendelt *h*(*n*) heurisztikus érték).

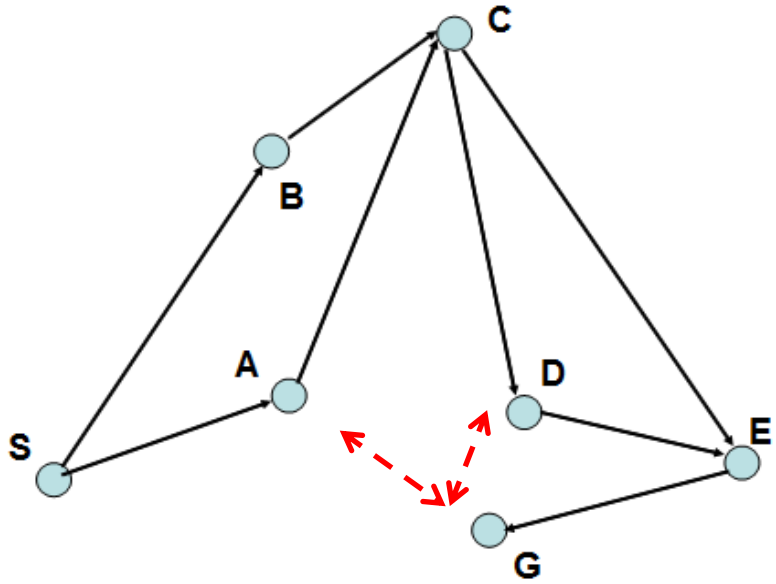


ezt kaptuk

ezt kellene kapni

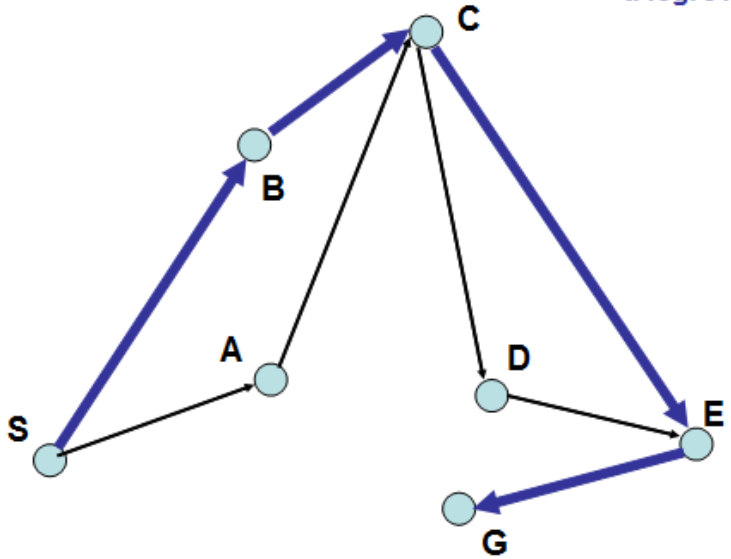
Mohó keresés

Mohó algoritmus általában **gyorsan** megtalálja a megoldást, de **nem mindig az optimális** megoldást találja meg. A mohó keresés érzékeny a hibás kezdő lépésekre is.

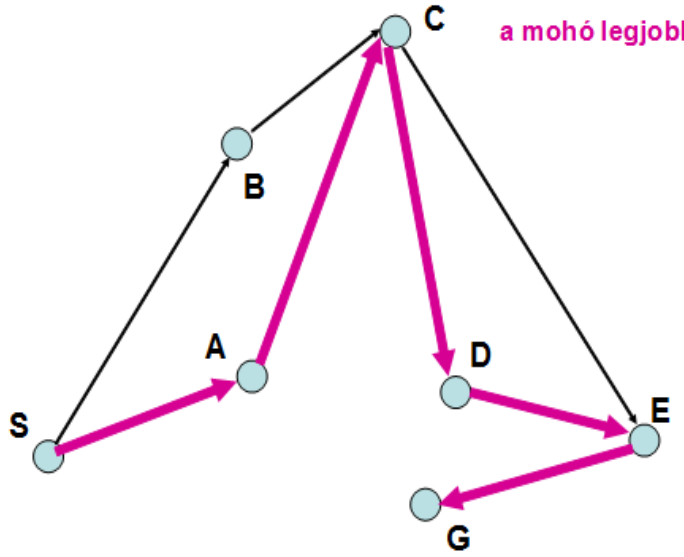


Probléma: A és D légvonalban nagyon közel van a G célhoz, de az úthálózaton nem

a legrövidebb út



a mohó legjobbat először út



Mohó keresés

- mélységi keresésre hasonlít, egyetlen út végigkövetését preferálja a célig, zsákutcából visszalép.
- Ua. a problémák, mint a mélységi keresésnél:
  - nem optimális,
  - nem teljes (elindul egy végtelen úton, és ez esetben nem tér vissza új lehetőséget kipróbálni).
- Az összes csomópontot a memóriában tartja: ezért a legrosszabb (worst-case) idő- és tárigény:
  - $O(b^m)$ – ha m a problématér mélysége

Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Méréstechnika és Információs rendszerek Tanszék

Mesterséges intelligencia

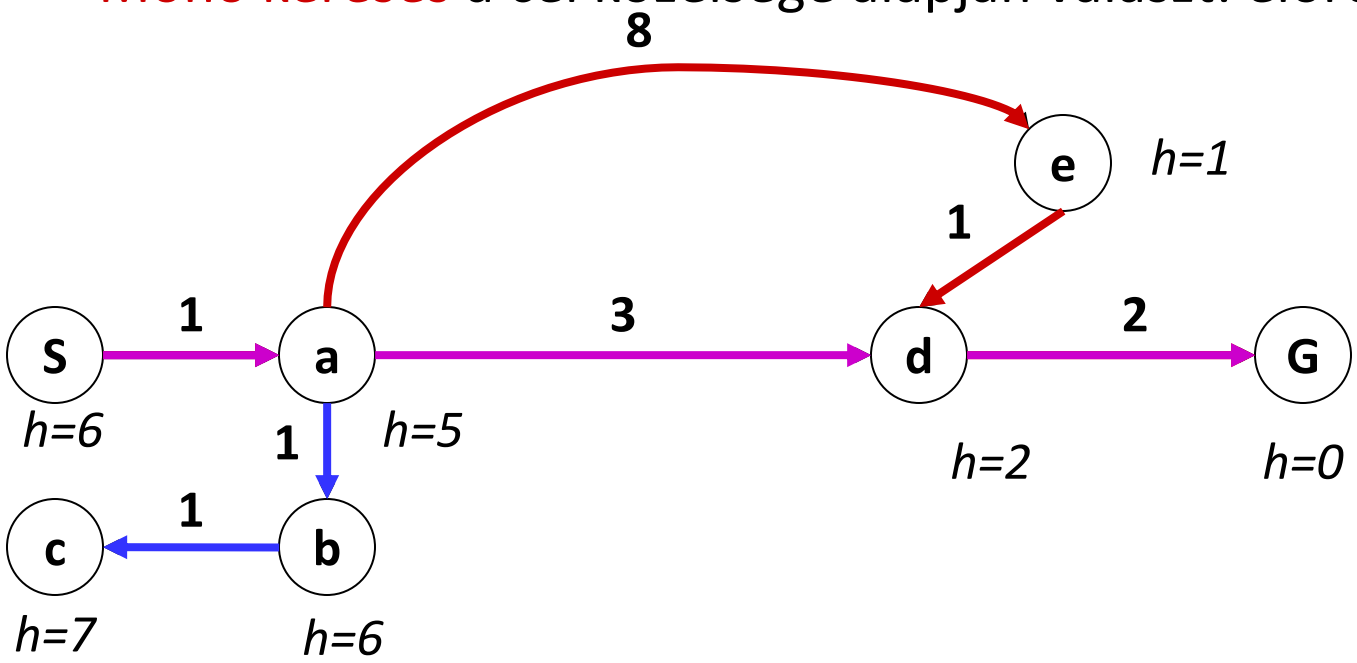
Problémamegoldás kereséssel

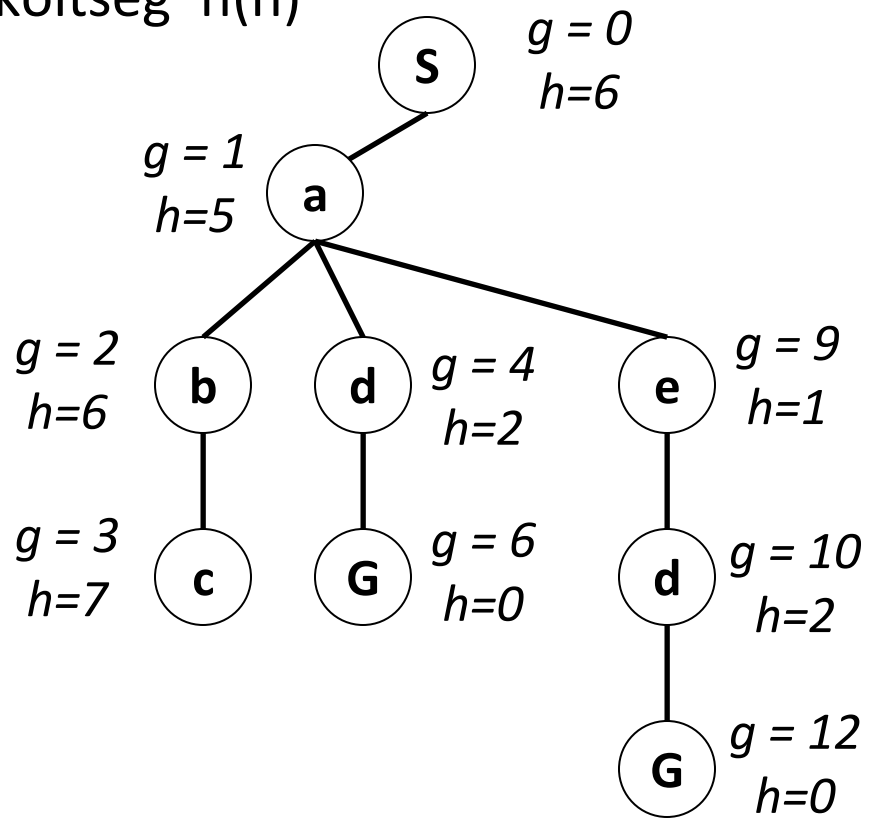
A* keresés

A* keresés

Egyenletes költségű és a mohó keresés egyesítése

- Egyenletes költségű keresés a lehetséges útvonalakat az útköltség alapján rendezi: visszamenőleges költség g(n)
- Mohó keresés a cél közelsége alapján választ: előremutató költség h(n)

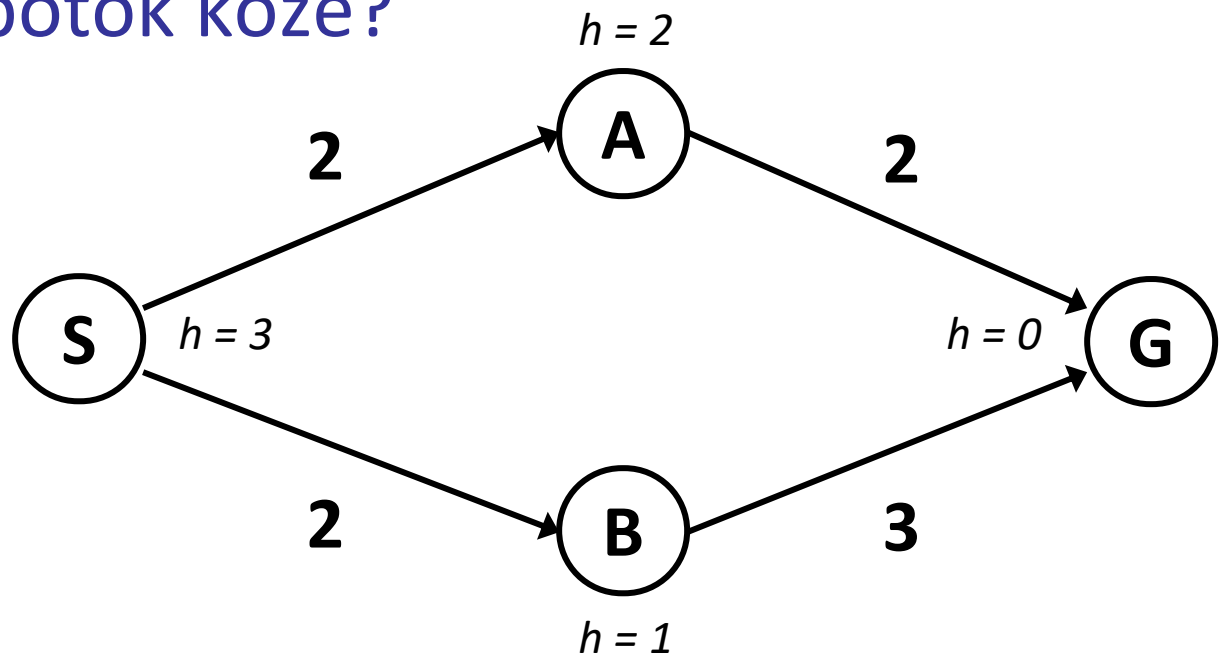




- A* keresés a kétféle költség összege alapján rendez: f(n) = g(n) + h(n)

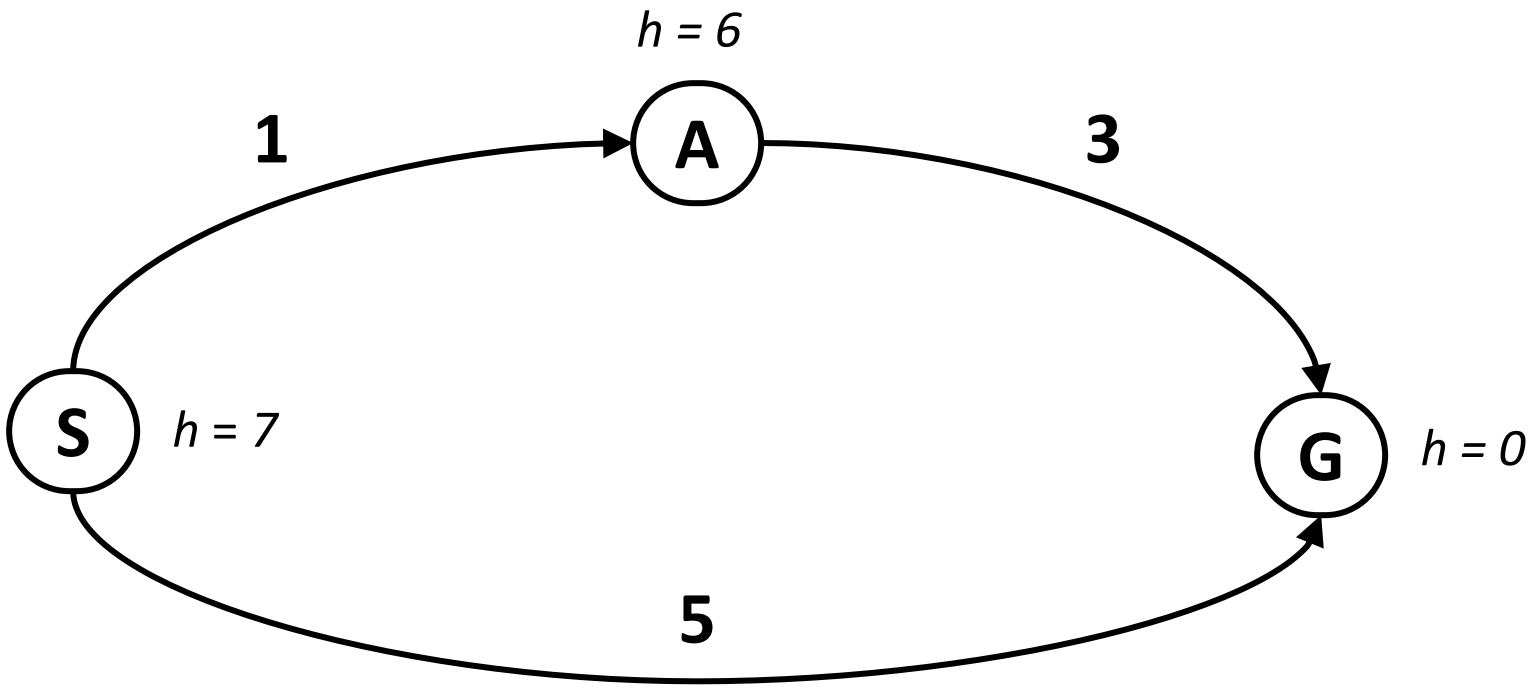
Mikor kell az A* keresésnek megállnia?

- Megállhat-e a keresés, amikor egy célállapot kerül a lehetséges követő állapotok közé?



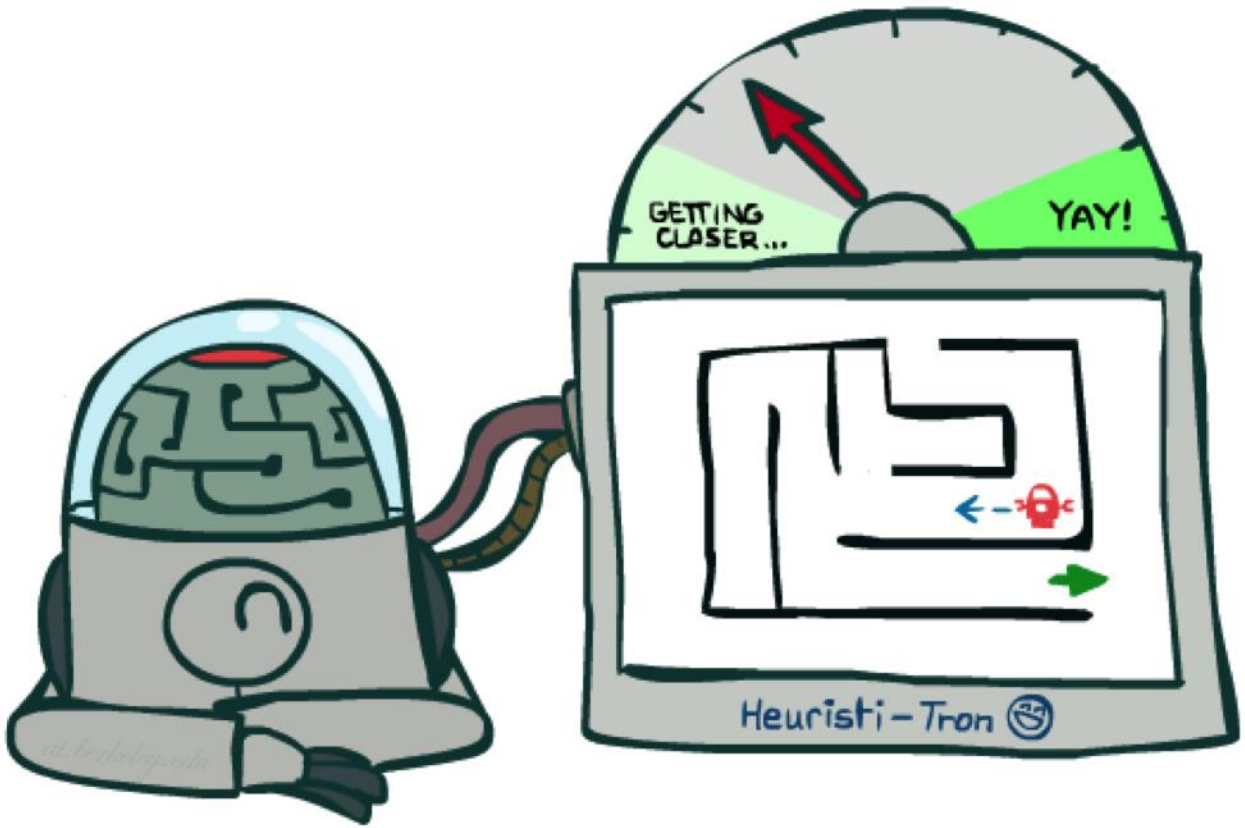
- Nem: csak akkor állhat meg, ha már kikerült az állapotok közül, és kifejtésre került

Optimális-e az A*?

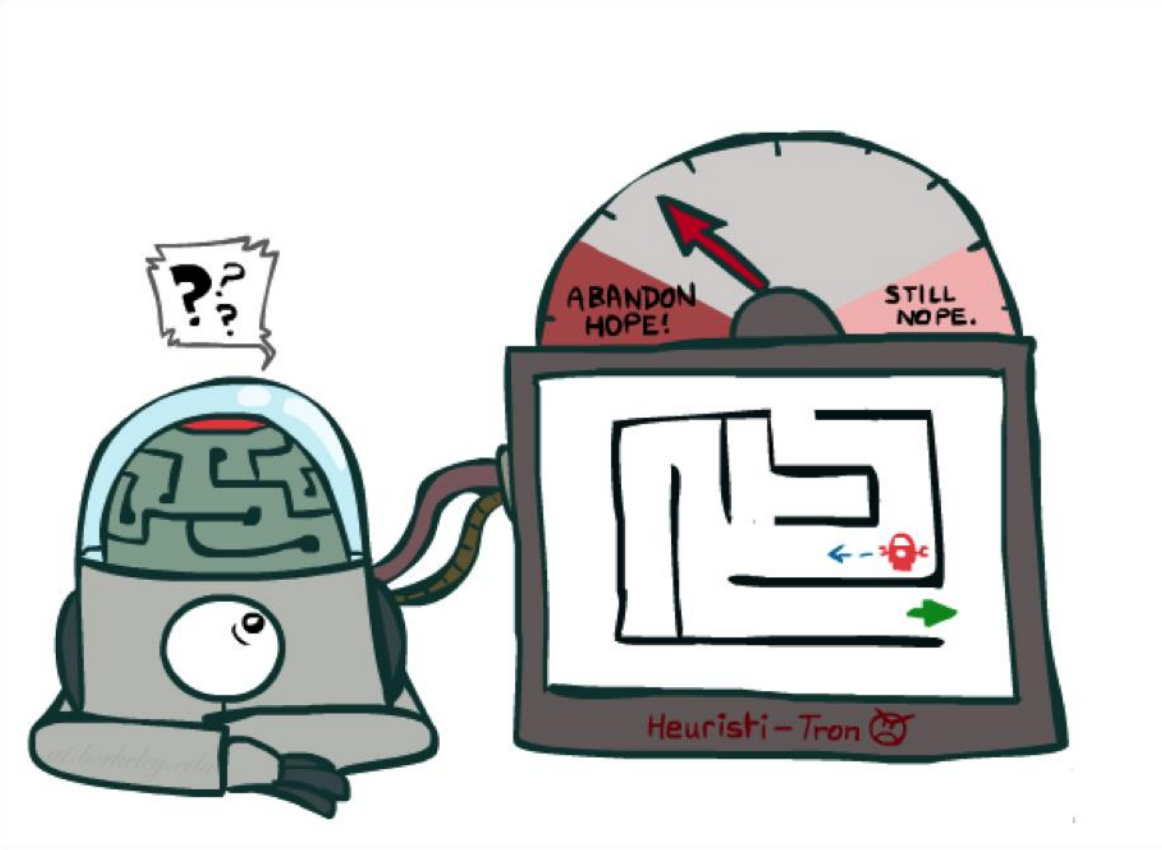


- Mi ment félre?
- A valós „rossz” célköltség < becsült „jó” célköltség
- A becsült költségnek kisebbnek kell lennie a valódi költségnél!

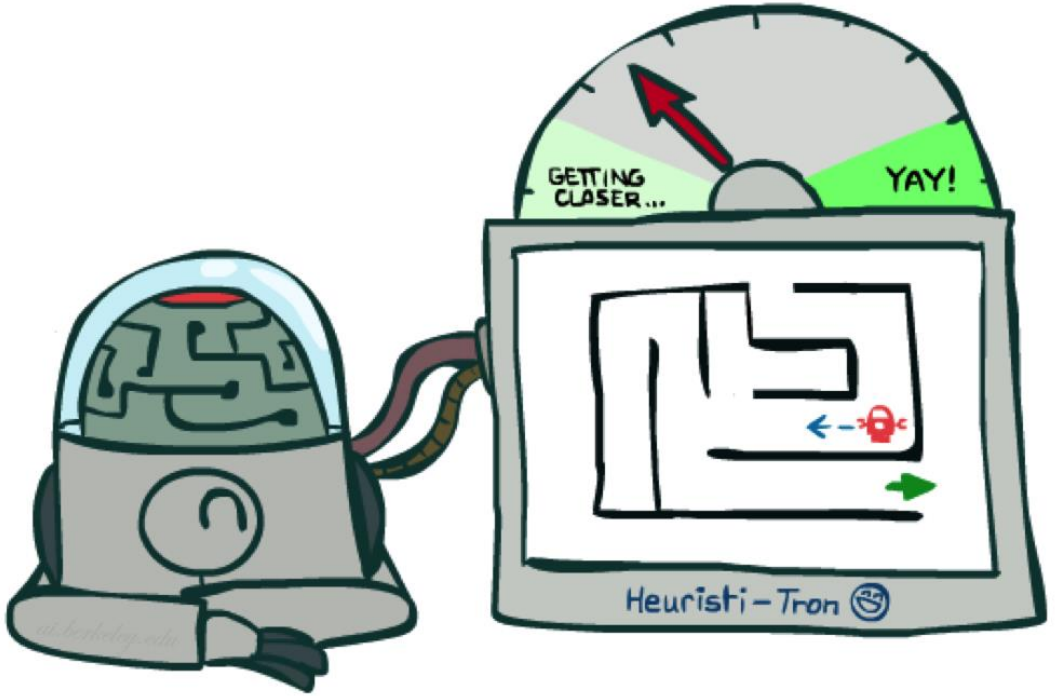
Elfogadható heurisztika



Elfogadhatóság, mint tulajdonság



Nem elfogadható (pesszimista) heurisztika megtöri az optimalitást azzal, hogy jó tervek a peremben maradnak.



Elfogadható (optimista) heurisztika „lelassítja” a rossz terveket, de sosem lép túl a valós költségeken

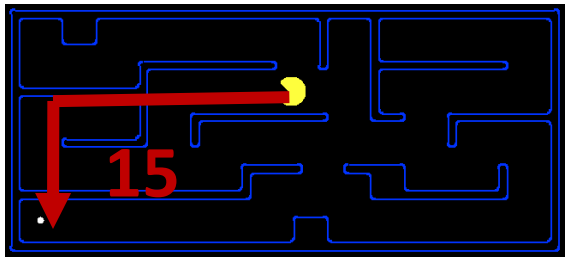
Elfogadható heurisztika

- A h heurisztika *elfogadható* (optimista) ha:

$$0 \leq h(n) \leq h^*(n)$$

ahol $h^*(n)$ a valós költség a legközelebbi célállapothoz

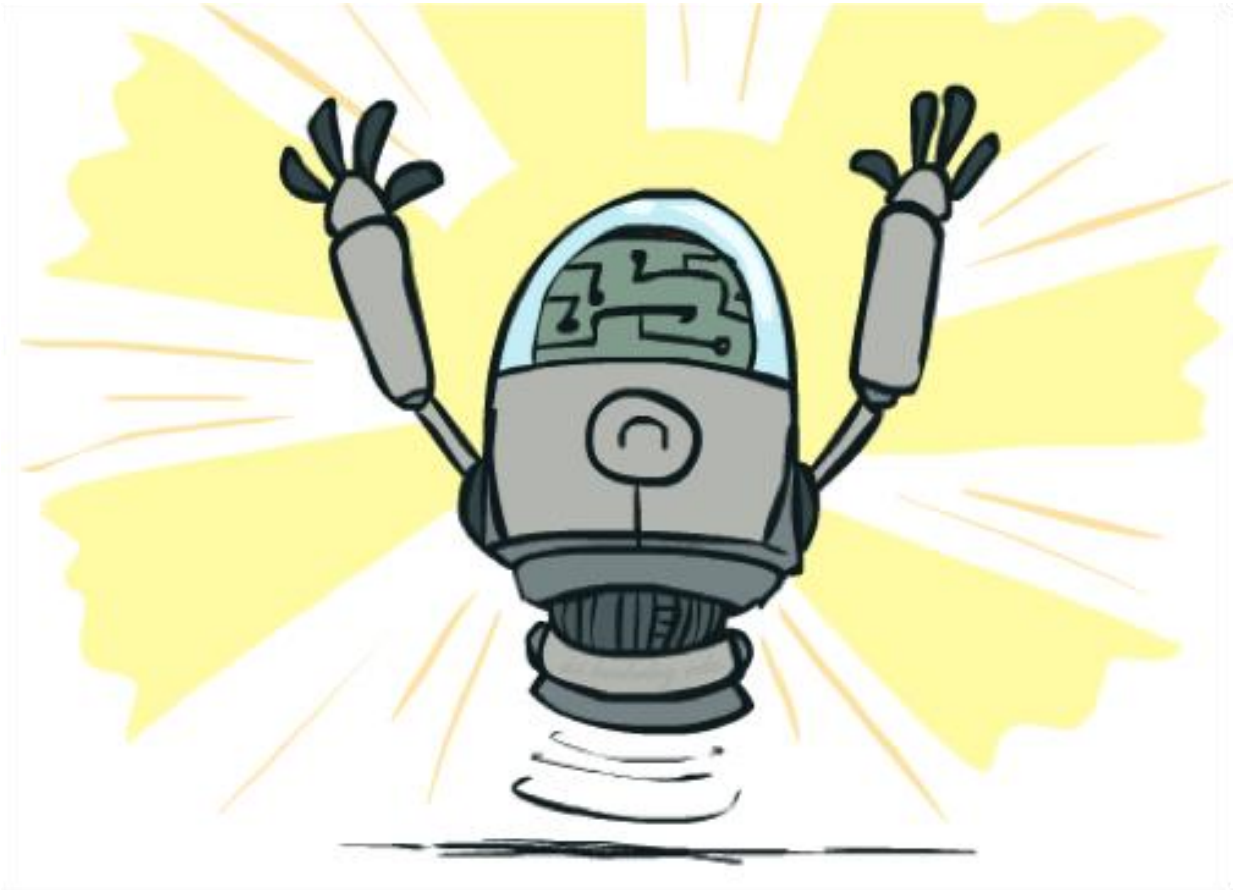
- Példák:



4

- Elfogadható heurisztikák kialakítása az A* keresés egyik lényegi pontja

Az A* keresés optimalitása



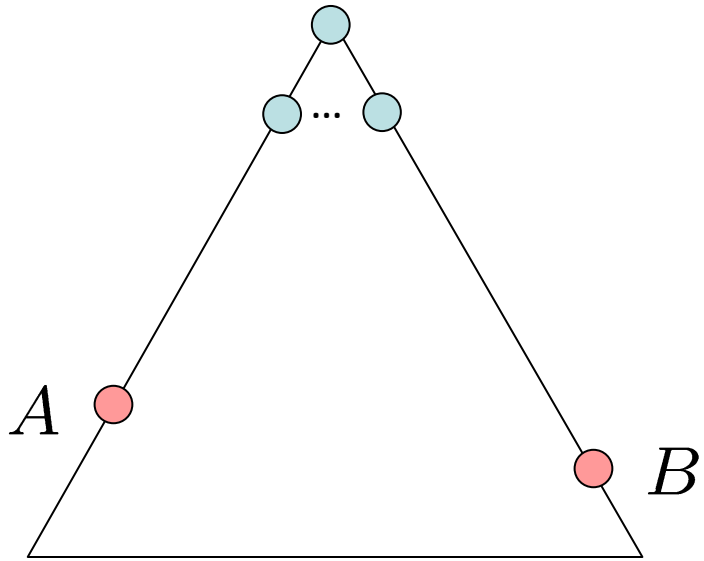
Az A* keresés optimalitása

Feltevések:

- „A” az optimális célcsomópont
- „B” a szuboptimális célcsomópont
- h elfogadható

Állítás:

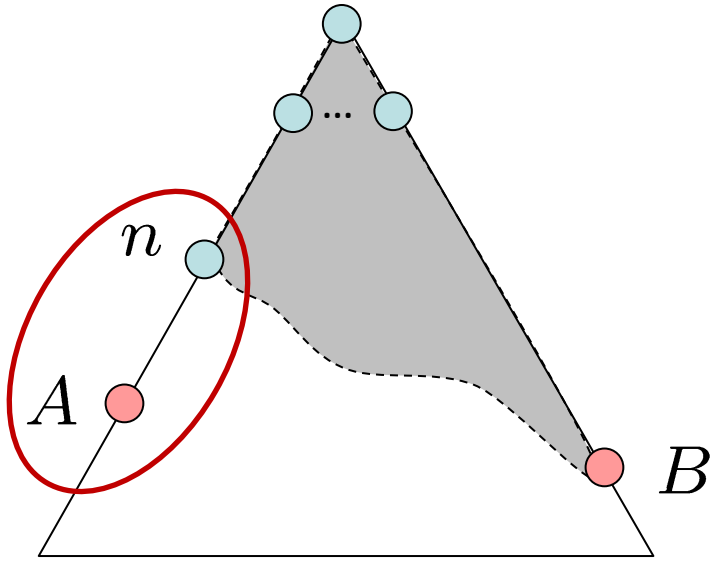
- „A” előbb kerül ki a peremből, mint „B”



Az A* keresés optimalitása: blokkolás

Bizonyítás:

- Tegyük fel, hogy „B” a peremen található
- Továbbá, hogy „A” valamely „n” őse szintén a peremen található (vagy maga „A” is)
- Állítás: *„n”* kifejtésére előbb kerül sor, mint „B” kifejtésére
  1. f(n) kisebb egyenlő, mint f(A)



$f(n) = g(n) + h(n)$ f-költség definíció

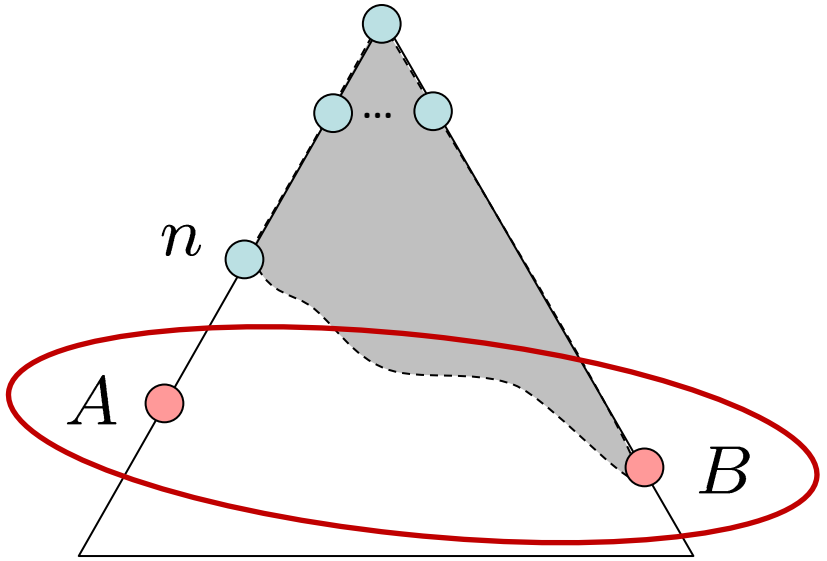
$f(n) \leq g(A)$ h elfogadhatósága

$g(A) = f(A)$ a célnál h = 0

Az A* keresés optimalitása: blokkolás

Bizonyítás:

- Tegyük fel, hogy „B” a peremen található
- Továbbá, hogy „A” valamely „n” őse szintén a peremen található (vagy maga „A” is)
- Állítás: *„n”* kifejtésére előbb kerül sor, mint „B” kifejtésére
  1. f(n) kisebb egyenlő, mint f(A)
  2. f(A) kisebb, mint f(B)



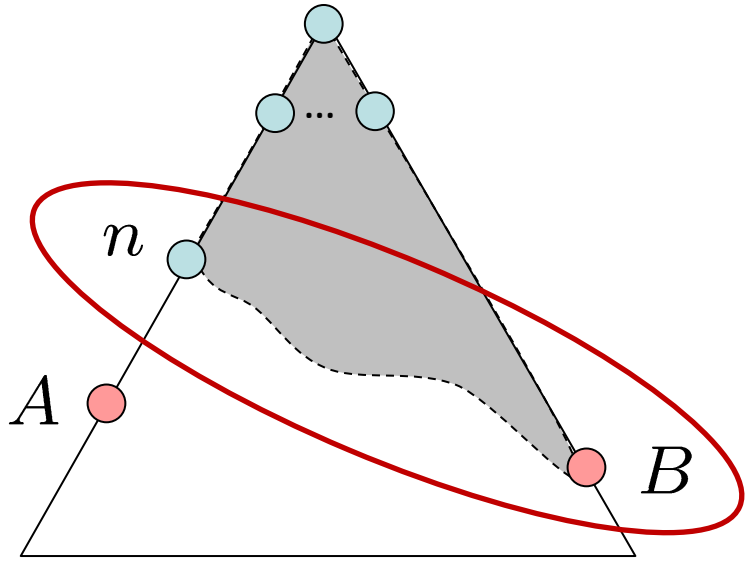
$g(A) < g(B)$ B szuboptimális

$f(A) < f(B)$ h = 0 a célnál

Az A* keresés optimalitása: blokkolás

Bizonyítás:

- Tegyük fel, hogy „B” a peremen található.
- Továbbá, hogy „A” valamely „n” őse szintén a peremen található (vagy maga „A” is).
- Állítás: *„n”* kifejtésére előbb kerül sor, mint „B” kifejtésére.
  1. f(n) kisebb egyenlő, mint f(A)
  2. f(A) kisebb, mint f(B)
  3. *„n”* előbb kerül kifejtésre, mint „B”
- „A” minden őse előbb kerül kifejtésre, mint „B”.
- „A” kifejtésre kerül „B” előtt.
- A* keresés optimális.



$$f(n) \leq f(A) < f(B)$$

Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Méréstechnika és Információs rendszerek Tanszék

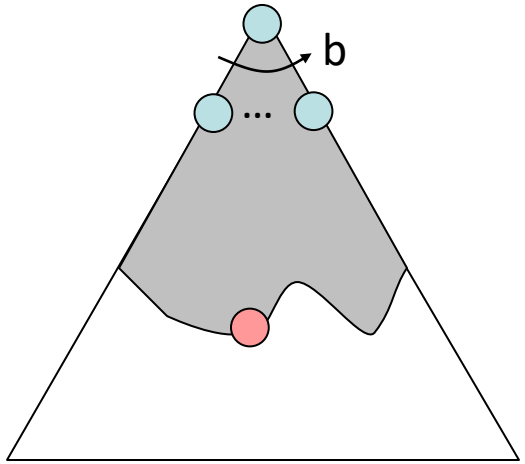
Mesterséges intelligencia

Problémamegoldás kereséssel
Informált keresés

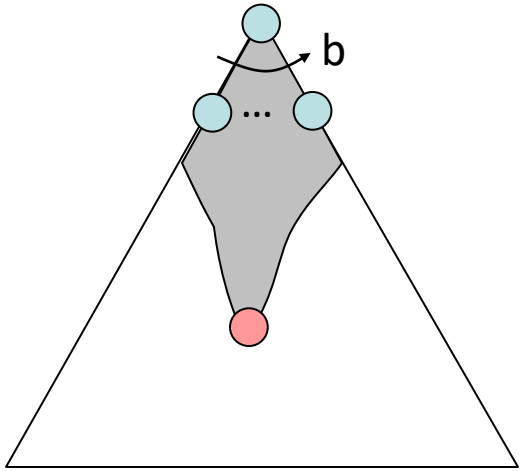
A* keresés további tulajdonságai

A* keresés tulajdonságai

Egyenletes költségű keresés

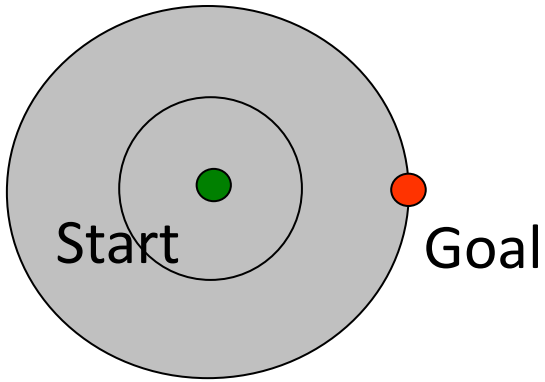


A*

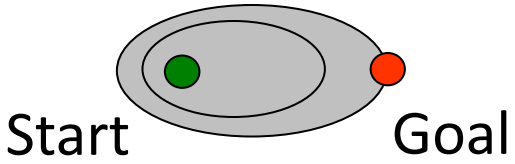


Egyenletes keresés vs A* határvonalak

- Egyenletes költségű keresés minden „irányban” egyenlően végez kifejtést



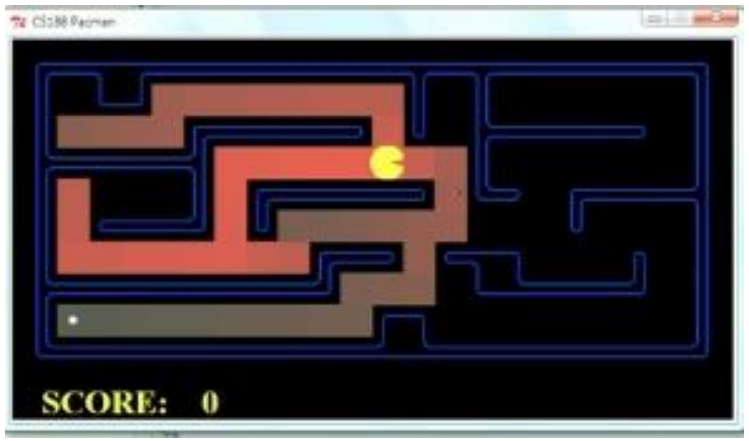
- A* főleg a cél irányában végez kifejtést



Összehasonlítás

Mohó

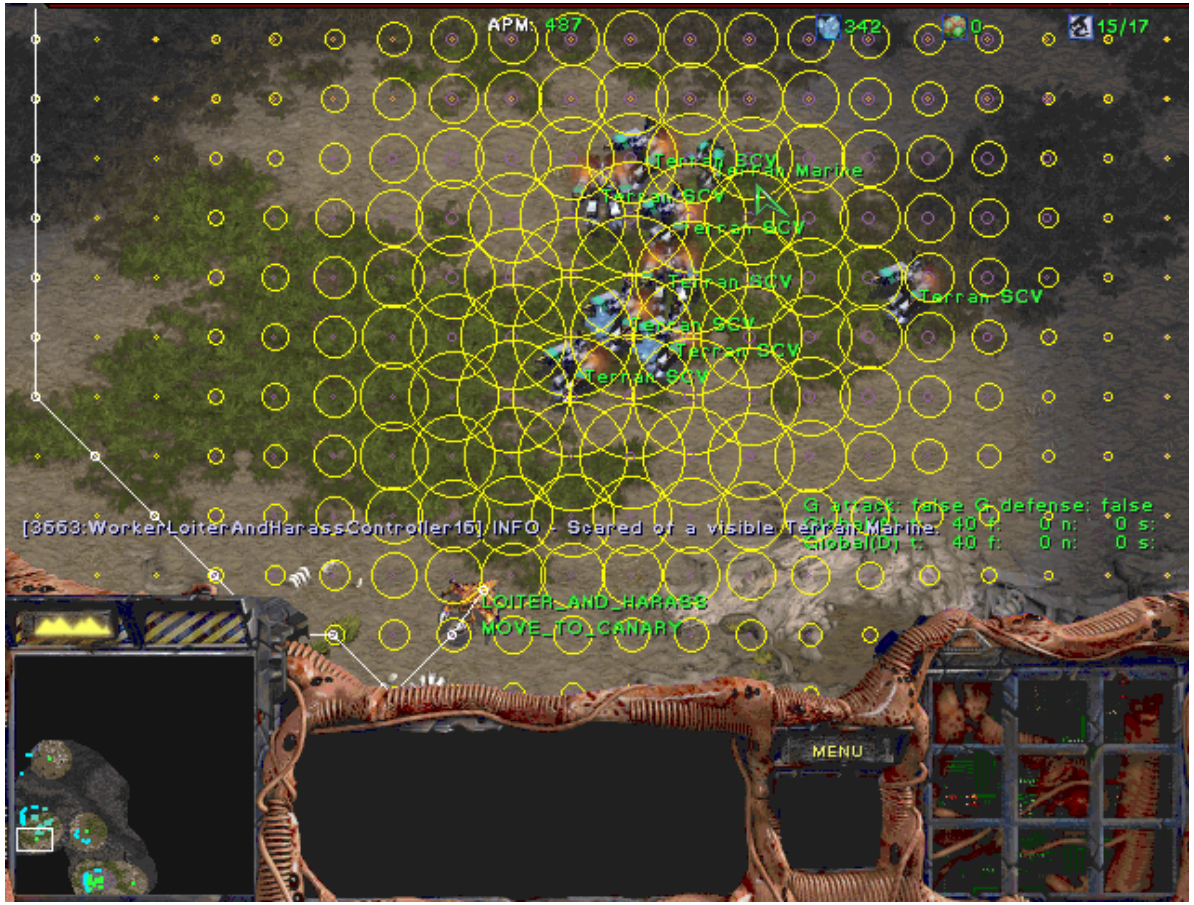
Egyenletes költségű keresés



A*

A* alkalmazásai

- Videójátékok
- Útvonalkeresési problémák
- Erőforrástervezési problémák
- Robot mozgás tervezése
- Nyelvi elemzés
- Gépi fordítás
- Beszédfelismerés
- …



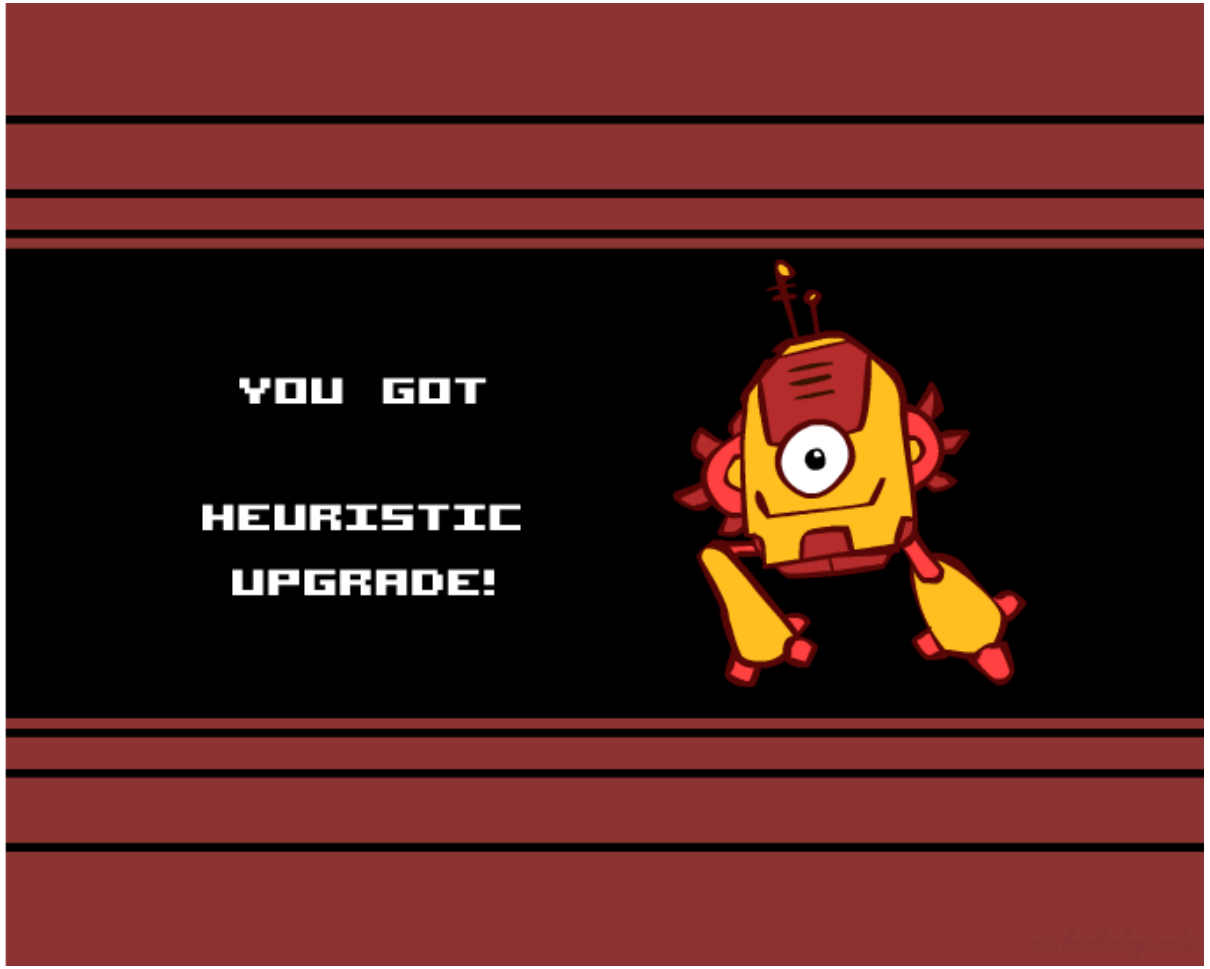
Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Méréstechnika és Információs rendszerek Tanszék

Mesterséges intelligencia

Problémamegoldás kereséssel
Informált keresés

Heurisztikák

Heurisztikák kialakítása

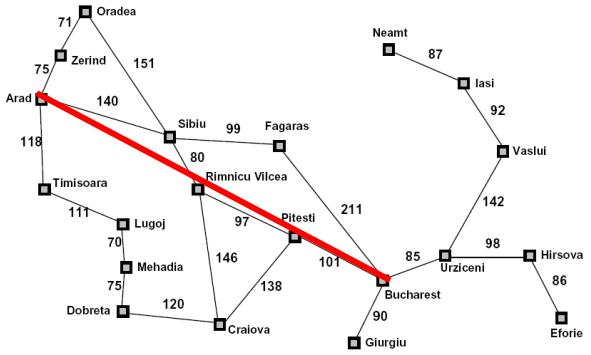


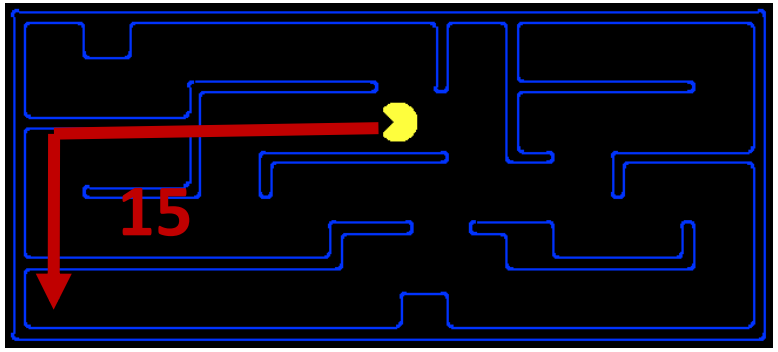
Elfogadható heurisztikák kialakítása

- Nehéz problémák optimális megoldásának a legnehezebb lépése a megfelelő elfogadható heurisztika kialakítása

- Az elfogadható heurisztikák gyakran egy relaxált probléma megoldásai, ahol további cselekvések lehetségesek.

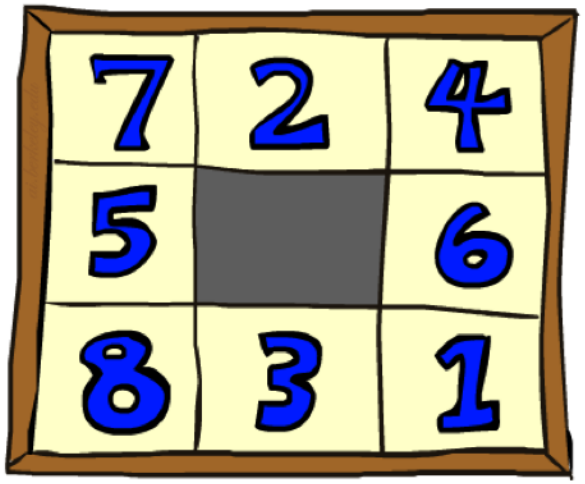
366



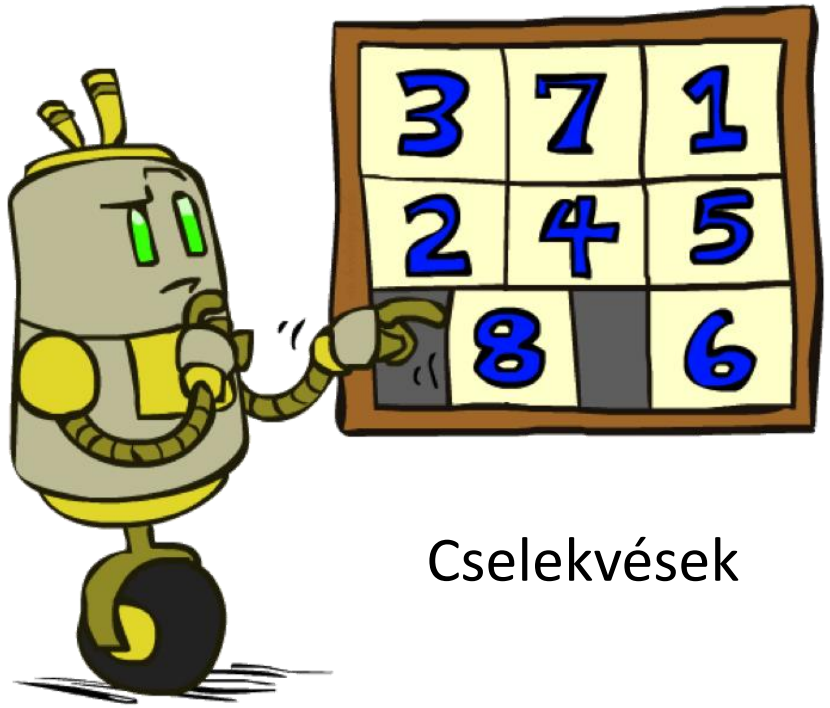


- Nem elfogadható heurisztikák is hasznosak lehetnek esetenként

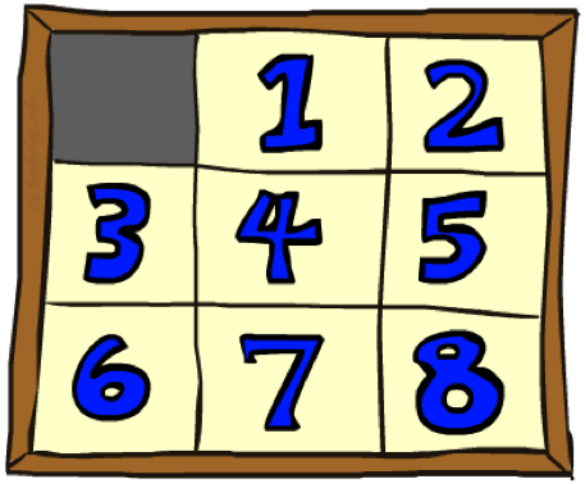
Példa: 8-as kirakójáték



Kiindulási állapot



Cselekvések

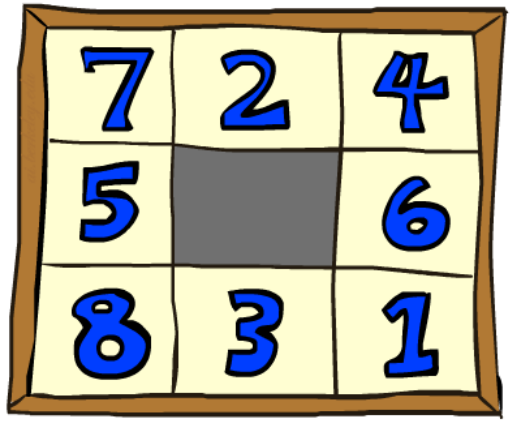


Célállapot

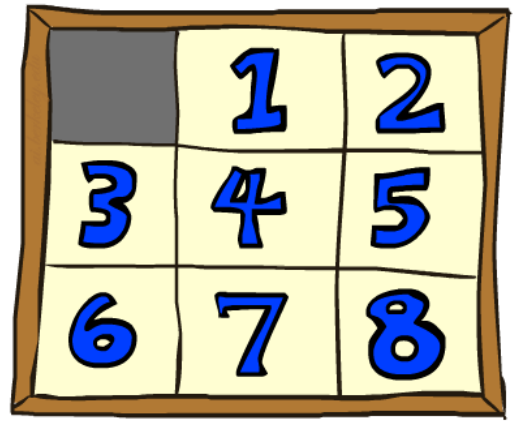
- Melyek a lehetséges állapotok?
- Hány állapot van?
- Melyek a lehetséges cselekvések?
- Hány követőállapot létezik a kiindulási állapotból?
- Mi legyen a költség?

8-as kirakójáték - 1

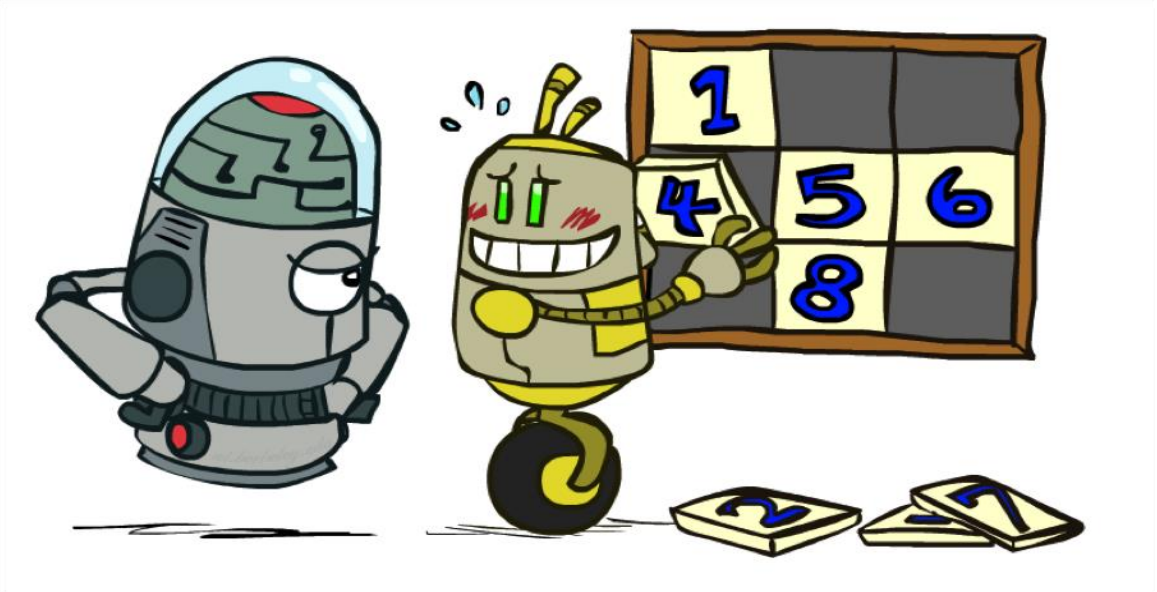
- Heurisztika: a rossz helyen lévő elemek száma
- Ez miért elfogadható?
- h(start) = 8
- Ez egy relaxált probléma alapú heurisztika



Kiindulási állapot



Célállapot

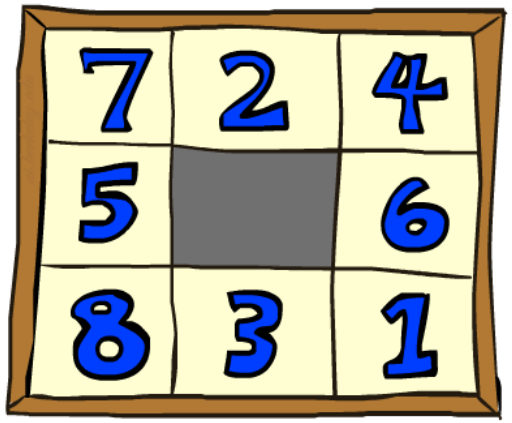


	Átlagos kifejtett csomópontszám, amikor az optimális út tartalmaz…		
	4 lépést	8 lépést	12 lépést
EKK	112	6,300	3.6×10^6
Elemszám	13	39	227

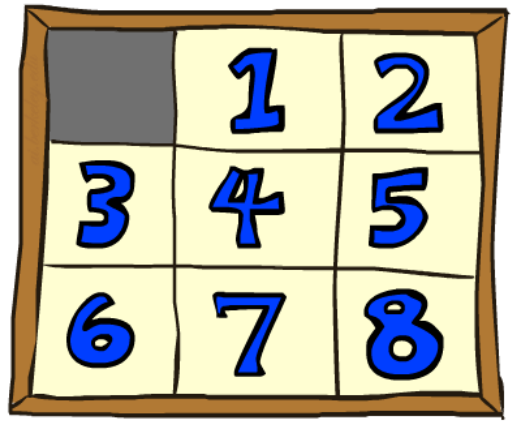
A. Moore

8-as kirakójáték - 2

- Mi lenne, ha lenne egy egyszerűbb 8-as kirakó, ahol minden elem minden irányban elmozdulhatna függetlenül a többi elemtől?
- *Manhattan* távolság



Kiindulási állapot



Célállapot

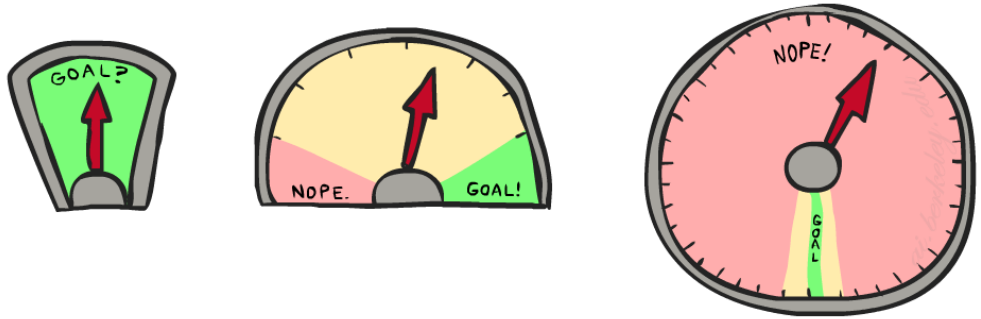
- Miért elfogadható?

- h(start) = 3 + 1 + 2 + … = 18

	Átlagos kifejtett csomópontszám, amikor az optimális út tartalmaz…		
	4 lépést	8 lépést	12 lépést
Elemszám	13	39	227
MANHATTAN	12	25	73

8-as kirakójáték - 3

- Mi lenne akkor, ha a tényleges költséget használnák heurisztikának?
  - Elfogadható lenne?
  - Csökkentené a kifejtett csomópontok számát?
  - Mi a probléma vele?



- A*: egyensúlyt teremt a becslés minősége és a csomópontban elvégzendő számítások mennyisége között
  - Ahogy egy heurisztika megközelíti a valós költséget, kevesebb csomópontot kell kifejteni, ugyanakkor a csomópontonként végzendő számítások nagysága nagyobb lesz.

Triviális heurisztikák, dominancia

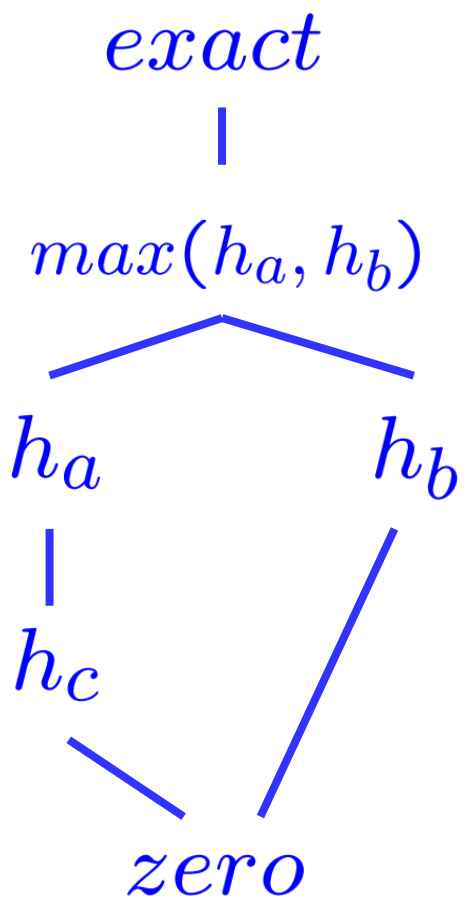
- Dominancia: $h_a \geq h_c$ ha

$$\forall n : h_a(n) \geq h_c(n)$$

- A heurisztikák egy félhálót (semi-lattice) alkotnak (speciális részben rendezett halmazt):
  - A maximuma az elfogadható heurisztikáknak szintén elfogadható

$$h(n) = max(h_a(n), h_b(n))$$

- Triviális heurisztikák
  - A félháló „alja” a zeró heurisztika
  - A teteje pedig az egzakt heurisztika



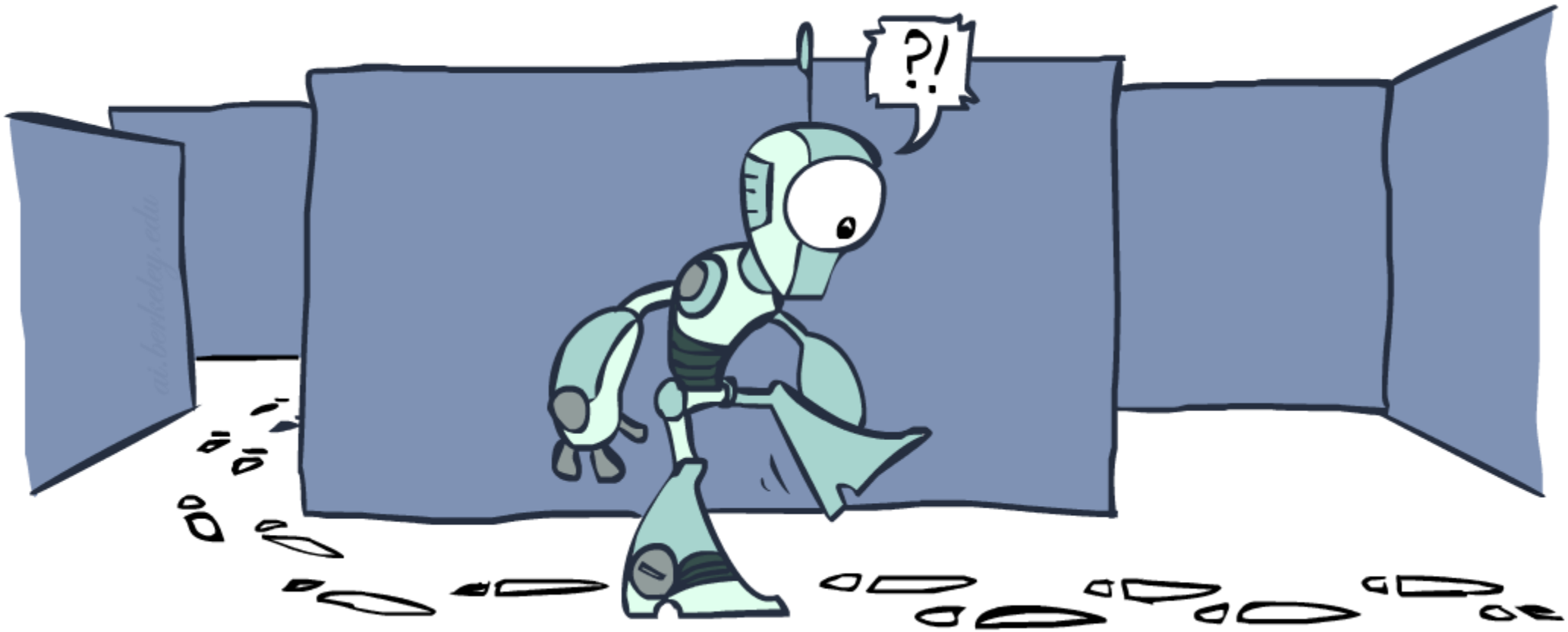
Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Méréstechnika és Információs rendszerek Tanszék

Mesterséges intelligencia

Problémamegoldás kereséssel
Informált keresés

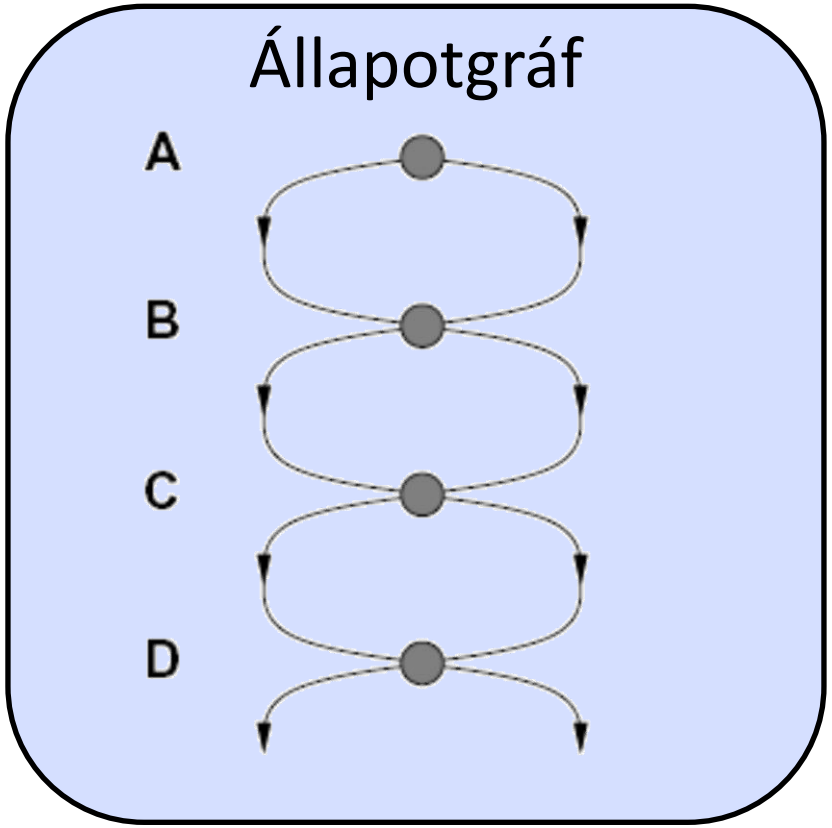
Gráfalapú keresés

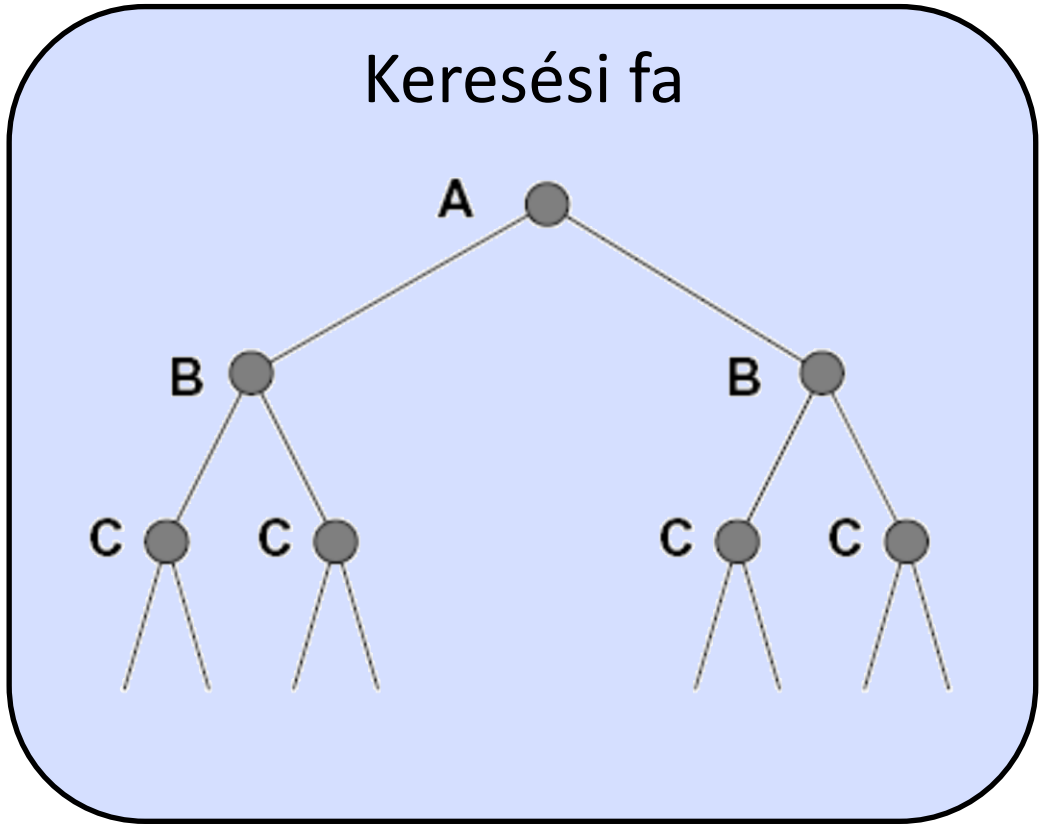
Gráfalapú keresés



A fa alapú keresés többlet munkával jár

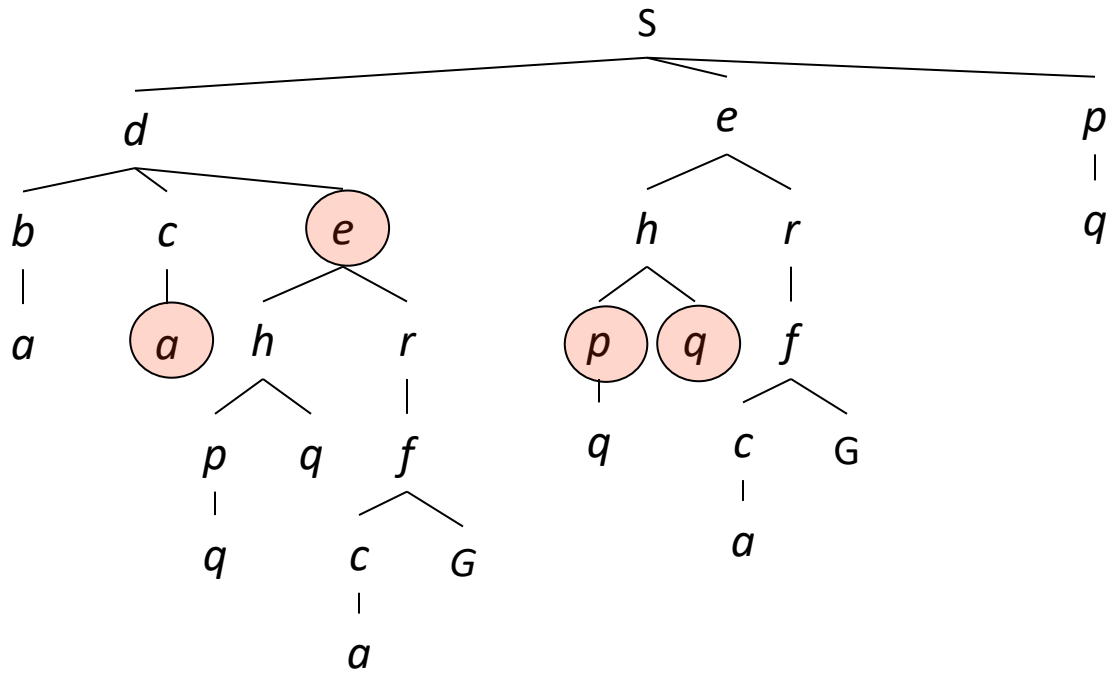
- Ismétlődő állapotok felismerésének a hiánya exponenciálisan több munkát eredményezhet.





Gráf alapú keresés

- Például az alábbi esetben szélességi keresésnél nem kellene foglalkoznunk a bekarikázott csomópontok kifejtésével

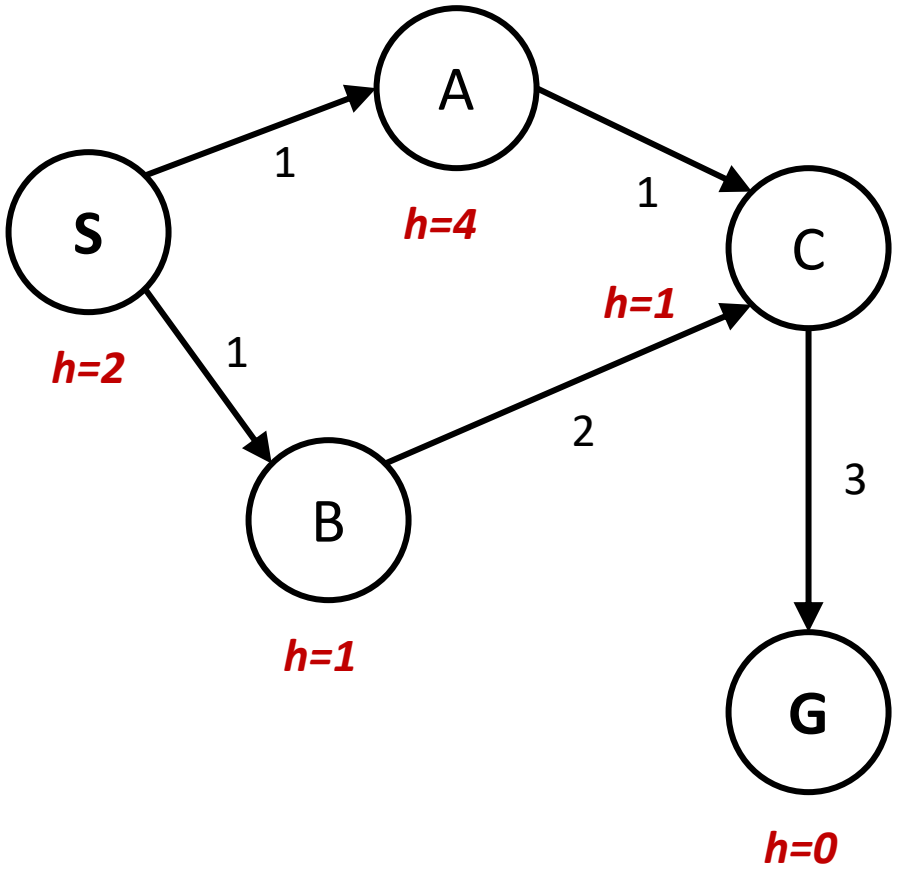


Gráf alapú keresés

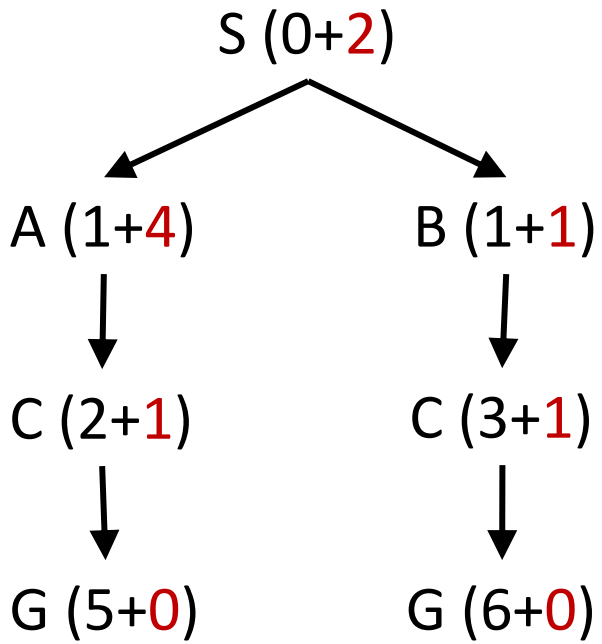
- Alapgondolat: soha ne fejtsük ki ugyanazt az állapotot kétszer
- Megvalósítás:
  - Fa keresés + kifejtett csomópontok halmaza (“zárt halmaz”)
  - Keresési fa kifejtése csomópontonként, de…
  - Kifejtés előtt ellenőrizzük, hogy nem volt-e korábban kifejtve
    - Ha nem új, kihagyjuk,
    - Ha új, hozzáadjuk a zárt halmazhoz.
- Lényeges: a zárt halmazt valóban halmazként tároljuk, ne listaként.
- A gráf alapú keresés befolyásolja-e a teljességet?
- Mi a helyzet az optimalitással?

A* gráf alapú keresés - probléma

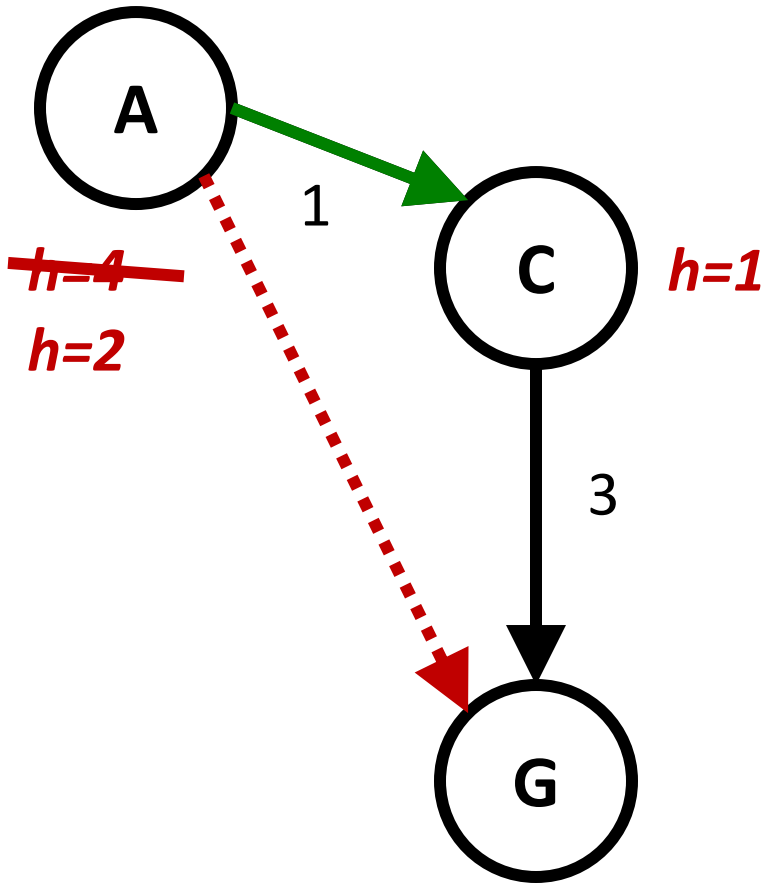
Állapotgráf



Keresési fa



Heurisztikák konzisztenciája



- Központi gondolat:

  becsült heurisztika költsége ≤ valós költség

  - Elfogadhatóság: heurisztika költsége ≤ valós költség a célig

    h(A) ≤ valós költség A-tól G-ig

  - Konzisztencia: heurisztika “élköltsége” ≤ valós élköltség

    h(A) – h(C) ≤ valós költség(A-> C)

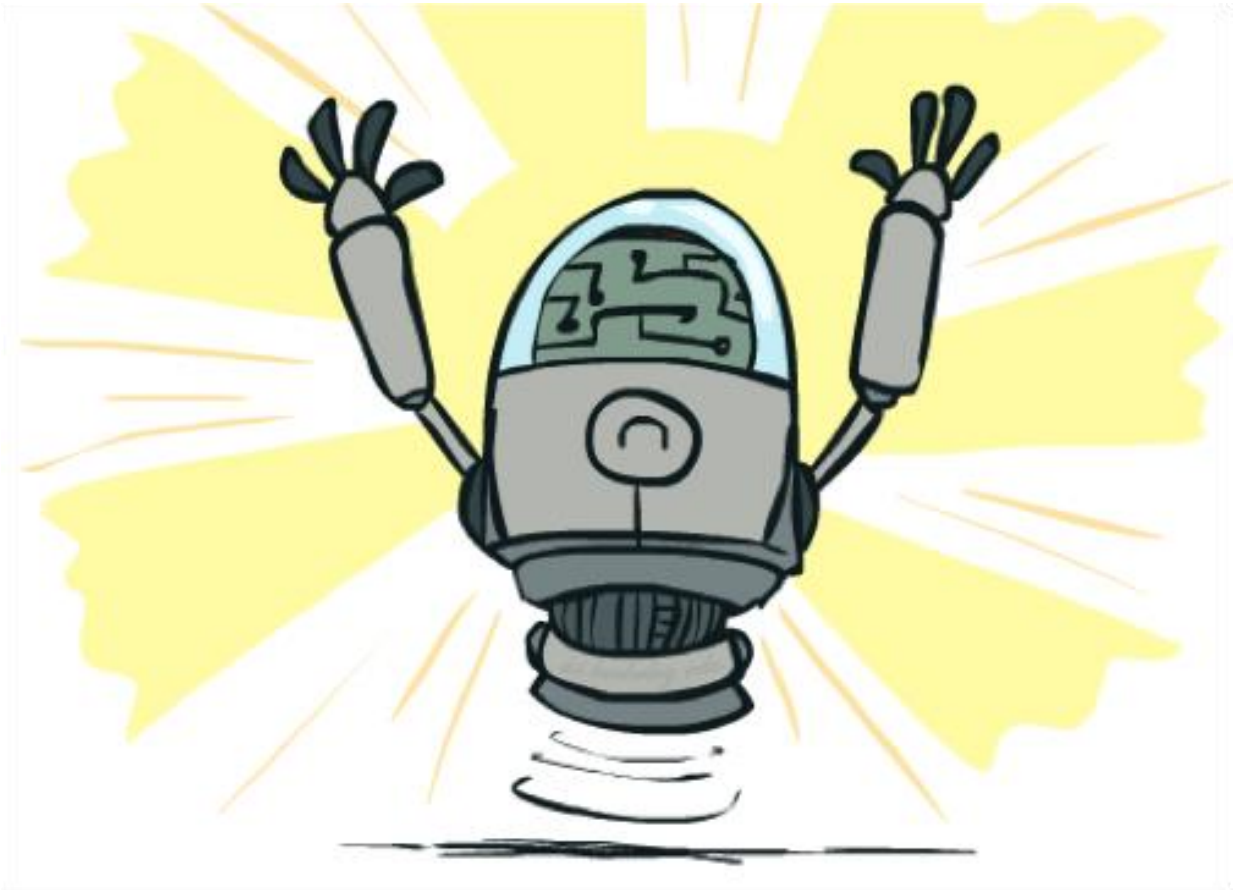
- Konzisztencia következménye:

  - Az f értéke nem csökken az útvonal mentén

    h(A) ≤ valós költség(A-> C) + h(C)

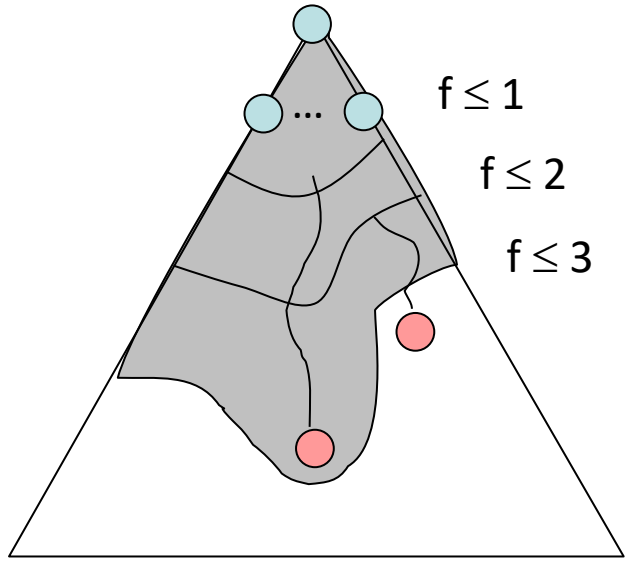
  - A* gráf alapú keresés optimális

A* gráf keresés optimalitása



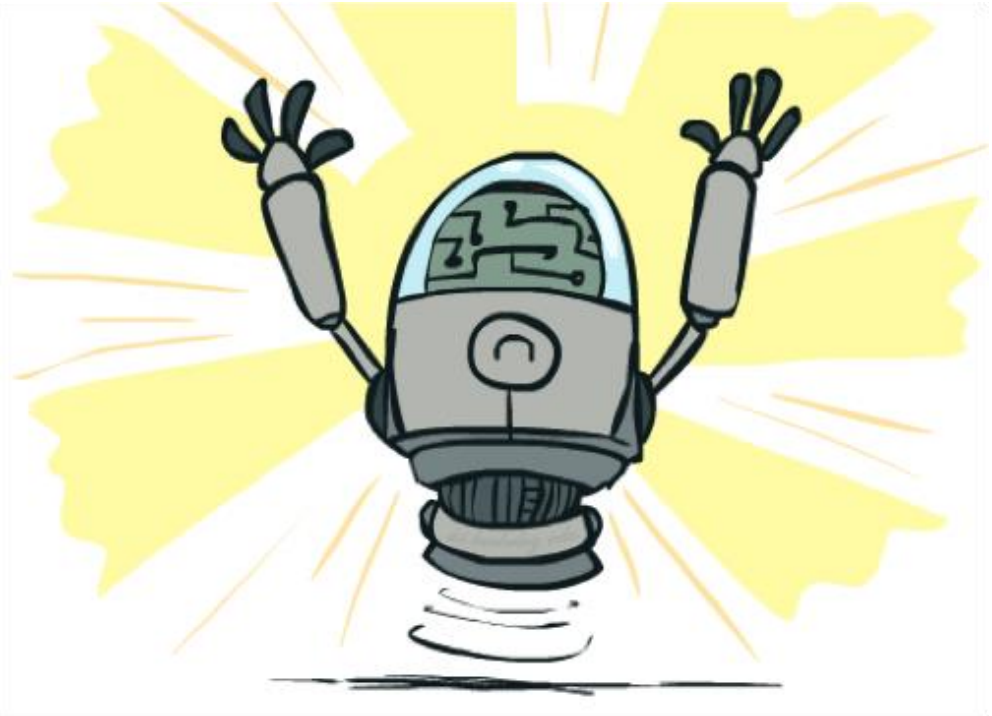
A* gráf keresés optimalitása

- Vázlat: tekintsük át mit tesz az A* keresés konzisztens heurisztikával:
  - 1. Tény: Fa keresés esetén az A* a csomópontokat növekvő f érték szerint fejti ki. (f-határvonal)
  - 2. tény: Minden *s* állapotra, azokat a csomópontokat, amiken keresztül *s* optimálisan elérhető, hamarabb fejti ki, mint azokat, melyeken keresztül *s* szuboptimálisan elérhető.
  - Eredmény: A* gráf keresés optimális



Optimalitás

- Fa alapú keresés:
  - A* optimális, ha a heurisztika elfogadható
  - Az egyenletes költségű keresés ennek egy speciális esete (h = 0)
- Gráf alapú keresés:
  - A* optimális, ha a heurisztika konzisztens
  - Az egyenletes költségű keresés optimális (h = 0 konzisztens)
- Konzisztenciából következik az elfogadhatóság
- Általában a legtöbb természetesen adódó, elfogadható heurisztika konzisztens, főleg ha relaxált problémából származnak

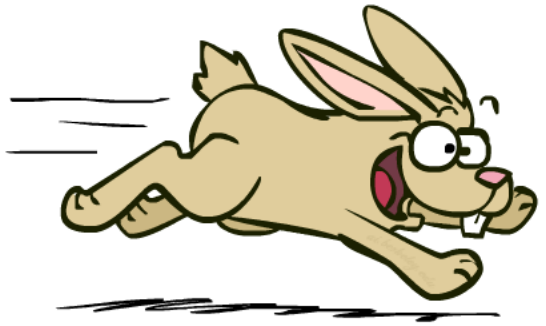


A*: Összefoglalás

A*: Összefoglalás

- A* figyelembe veszi az adott pontig megtett út költségét (backward cost), és az adott pontból a célállapotig tartó út becsült költségét (forward cost)
- A* optimális elfogadható / konzisztens heurisztika alkalmazása esetén
- A heurisztika megtervezése a lényeges pont, gyakran egy relaxált probléma alapján alakítjuk ki

Egyenletes költségű keresés



Mohó keresés

A* keresés

Fa keresés pszeudókódja

```
function FA-KERESÉS (probléma, perem) return egy megoldás vagy kudarc
        perem ← BEILLESZT(ÚJ-CSOMÓPONT(KEZDŐ-ÁLLAPOT[probléma]), perem)
        loop do
                if perem üres then return kudarc
                csomópont ← ELSŐ-ELEM-KIVÁLASZT(perem)
                if CÉLÁLLAPOT-TESZT(probléma, ÁLLAPOT[csomópont]) then return csomópont
                for gyermek-csomópont in KIFEJTÉS(ÁLLAPOT[csomópont], probléma) do
                perem ← BEILLESZT(gyermek-csomópont, perem)
                end
        end
```

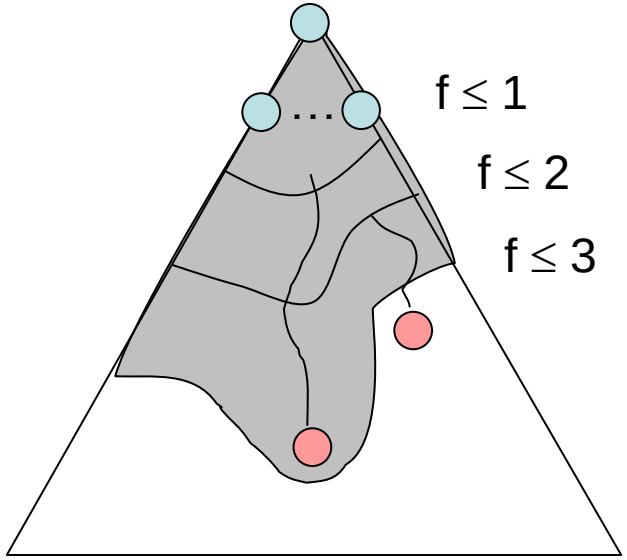
Gráf keresés pszeudókódja

```
function GRÁF-KERESÉS (probléma, perem) return egy megoldás vagy kudarc
        zárt-halmaz ←üres
        perem ← BEILLESZT(ÚJ-CSOMÓPONT(KEZDŐ-ÁLLAPOT[probléma]), perem)
        loop do
                if perem üres then return kudarc
                csomópont ← ELSŐ-ELEM-KIVÁLASZT(perem)
                if CÉLÁLLAPOT-TESZT(probléma, ÁLLAPOT[csomópont]) then return csomópont
                if ÁLLAPOT[csomópont] not in zárt-halmaz then
                        zárt-halmaz ← HOZZÁAD(ÁLLAPOT[csomópont], zárt-halmaz)
                        for gyermek-csomópont in KIFEJTÉS(ÁLLAPOT[csomópont], probléma) do
                                perem ← BEILLESZT(gyermek-csomópont, perem)
                        end
        end
```

A* Gráf keresés optimalitása

- Mit tesz az A*:
  - Csomópontokat fejt ki növekvő *f* érték szerint (f-határvonal)
    f(n) = g(n) + h(n) = „n”-be jutás költsége + heurisztika
  - Bizonyítás alapgondolata: Az optimális cél(ok) rendelkeznek a legkisebb *f* értékkel tehát ezeket kell előbb kifejtenie.

Ez jó lenne, de elég? Élnünk kell-e még további feltételezésekkel?



A* Gráf keresés optimalitása

Bizonyítás:

- Új probléma: a G*-hoz tartó útvonalon egy ***n*** csomópont nincs benne kifejtendő csomópontok sorában, mert egy rosszabb ***n'*** került ki előbb és azt fejtette ki az algoritmus
- Vegyük a legmagasabban lévő ***n*** *-et* a keresési fából
- Legyen ***p*** az ***n*** őse,ami benne volt a sorban, amikor ***n'*** kikerült onnan**.**
- ***f(p) < f(n)*** a konzisztencia miatt
- ***f(n) < f(n')*** mivel ***n'*** szuboptimális
- ***p*** csomópontot előbb kellett volna kifejtenie, mint ***n'***-et
- Ellentmondásra jutottunk!