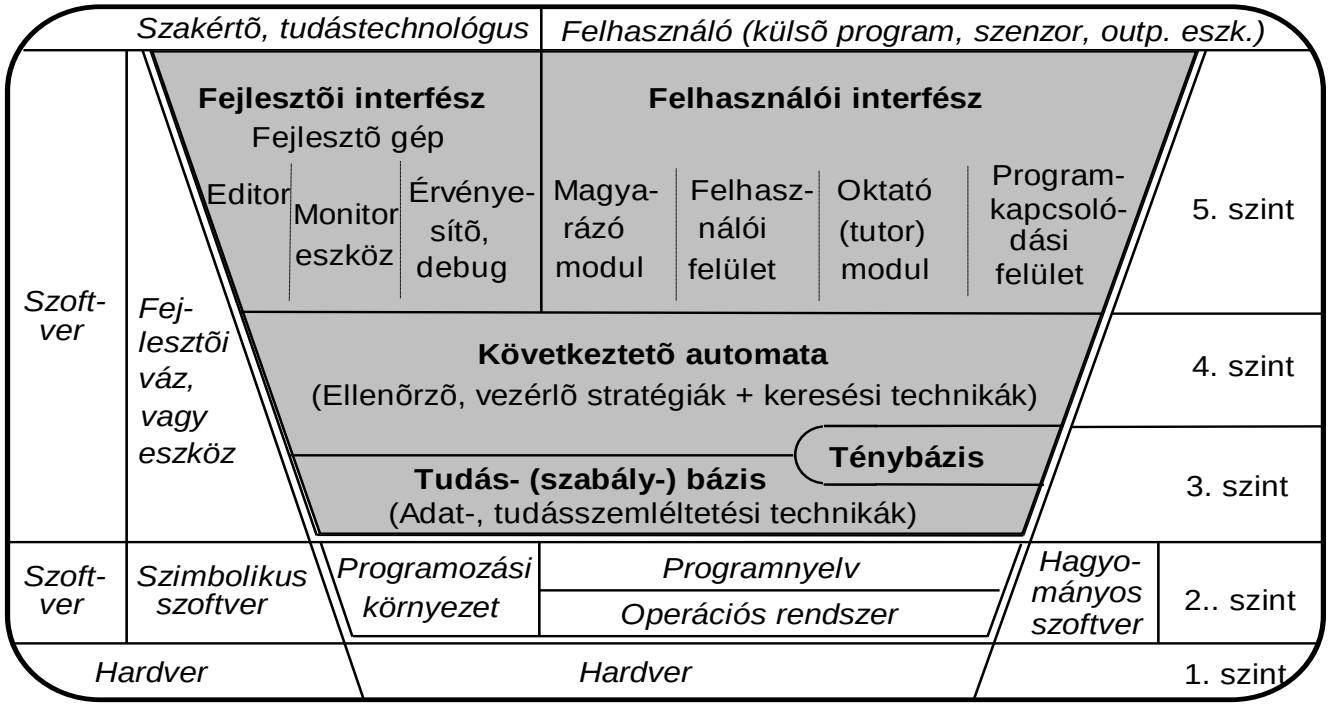
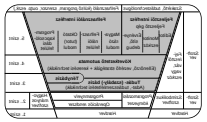
Szakértőrendszerek megvalósítása

Egy szakértőrendszer olyan eszköz, amely problémaspecifikus ismeret megértésére képes, és intelligensen használja a tématerület ismeretanyagát egy tevékenység különböző megvalósítási útjainak felvetéséhez. A szakértőrendszerek nem csak az ismeretátadás technikáit alkalmazzák, hanem analitikus, elemző eszközöket is az ismeret kiértékelésére, valamint tanulási technikákat.



Szakértőrendszer és beágyazó környezetének összetevői

Deborah D. Wolfgram - Teresa J Dear - Craig S. Galbraith: ***Expert Systems for the Technical Professional*** *John Wiley & Sons,New York, 1987.*



Szakértőrendszerek megvalósítása ..

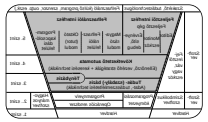
1. szint: a hardver

A szakértőrendszer alkalmazások futtatására szolgáló **dedikált** hardver jelentősen eltér a hagyományos alkalmazások futtatására előnyös hardvertől. A fő eltérések:

- Adattípus szerinti memóriaelérés
- Egyfelhasználós kivitel
- LISP célprocesszor.

A szimbolikus szoftverek futtatására szánt hardvernél kiemelt fontosságú a nagy sebesség.

Napjaink tendenciája, hogy a hagyományos hardver nagyfokú teljesítménynövekedésének, árcsökkenésének és elterjedtségének köszönhetően előtérbe kerül a mesterséges intelligencia alkalmazások céljaira, köztük a szakértőrendszerek futtatására is.



Szakértőrendszerek megvalósítása ..

1. szint: a hardver ..

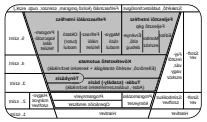
Hardverplatformok:

- Dedikált hardver, célszámítógép, elsősorban LISP processzorral
- Mainframe, fürtbe kötött, vagy hálózatban egyesített erőforrások
- Munkaállomás
- Mikroszámítógép

Dedikált hardver alkalmazásának indokai:

- Extrém nagy teljesítményigény
- Eleve erre készült szoftver

- Néhány célhardvert gyártó cég :
  - Symbolics
  - LMI (LISP Machine Inc)
  - Texas Instruments



Szakértőrendszerek megvalósítása ..

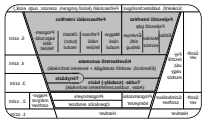
2. szint: a szoftver

A szoftver adja a rendszer intelligenciáját. Két fajtája:

- Operációs rendszer+ programozási nyelv
- Dinamikus programozási környezet dedikált hardver esetén.

A mesterséges intelligencia szoftverek a hagyományos adatfeldolgozó alkalmazásokkal szemben tudásfeldolgozó alkalmazások. Ebből származik néhány eltérés a hagyományos szoftverekhez képest:

	Szimbolikus	Hagyományos
1.	Heurisztikák	Algoritmusok
2.	Szimbolikus módon elérhető ismeretbázis	Numerikusan címzett adatbázis
3.	Szimbolikus feldolgozás orientált	Numerikus-feldolgozás orientált
4.	Emberközeli interfészek, beszédfeldolgozás	Nagyrészt önálló, interakció nélküli működés
5.	Futásközbeni magyarázatadás	Nincs futásközbeni felhasználóbarát magyarázatadás
6.	A feladat leíró módja az ember által könnyen követhető szimbolizmust alkalmaz	A feladat a programnyelv logikájára van átfogalmazva, csak a nyelvet ismerő ember képes megérteni



Szakértőrendszerek megvalósítása ..

2. szint: a szoftver ..

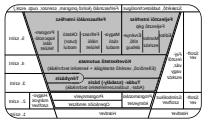
Szakértőrendszer-szoftverek felosztása:

- Általános célú MI-nyelvek: LISP, Prolog, Poplog, stb.
- Fejlesztőeszközök, szakértőrendszer-vázak, pl: KappaPC, GURU, stb.
- Kulcsrakész szakértőrendszerek, pl. Lending Advisor, stb.

Az **általános célú MI nyelveken** történő fejlesztések időigényesek és költségesek. Napjainkban egyre több rendszer készül C++ nyelven.

Fejlesztőeszközök, szakértőrendszer-vázak

Váz = üres tudás (szabály)bázisú szakértőrendszer. Létrehozását a tudásbázis és a következtető automata éles elkülönülése teszi lehetővé. Új szakértőrendszerek kifejlesztéséhez fél megoldást nyújtanak.

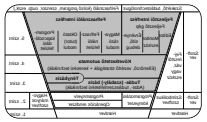


Szakértőrendszerek megvalósítása ..

2. szint: a szoftver ..

Szakértőrendszer-**vázak képességei**:

- Tudásbázis fejlesztő **editor** a tudásbázis feltöltésére és karbantartására
- A tudásbázis integritásának, **konzisztenciájának** ellenőrzése, az ellentmondó, redundáns szabályok kiszűrése, a hiányzó információk jelzése (validálás)
- **Szintaktikai**, formai ellenőrzés
- Következtetésindoklás, **magyarázatadás**
- **Hibakeresés**, nyomkövetés, töréspontok elhelyezése teszteléshez
- **Grafikus** felület az információk és kapcsolatrendszerek megjelenítésére
- **Kapcsolódási felület** más programokhoz, pl. adatbázisokhoz, táblázatkezelőkhöz, Internethez, szenzorokhoz és beavatkozó szervekhez
- **Tudáskinyerés** szöveg alapú forrásokból
- Hanggenerálás és **hangfelismerés** az emberközeli kommunikáció elősegítésére természetes nyelvi interfész formájában.



Szakértőrendszerek megvalósítása ..

2. szint: a szoftver ..

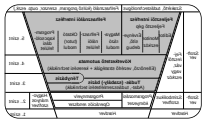
Szakértőrendszer-**vázak alkalmazásának**

- **előnye:** idő- és költségmegtakarítás
- **hátránya**: eleve adott tudásszemléltetési és következtetési technika. Ez a probléma nem merül fel a hibrid rendszereknél, pl: KES, PICON, GURU, KappaPC.

- Ismertebb szakértőrendszer-vázak: PLUS, KES, KEE, ART, RULEMASTER, EMYCIN.
- Alkalmazható szabályok száma: 100 – 10000.

Kulcsrakész szakértőrendszerek

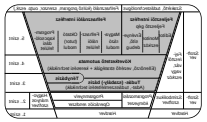
- Nem igényelnek fejlesztőmunkát, rögtön használhatók



Szakértőrendszerek megvalósítása ..

Kulcsrakész rendszer vásárlása előtt mérlegelendő

- **Létezik-e** a piacon a problémára megfelelő szakértőrendszer?
- Ki az a **szakértő**, akinek a tudását hordozza? Mások által is elfogadott szintet képvisel-e?
- **Illeszkedik-e** a tárolt tudás a megoldandó problémákhoz?
- **Megfelelő** adatszerkezeteket, keresési módszereket, **következtetési eljárásokat** alkalmaz-e?
- Kellően **teljes-e** az adott szakterület tudását illetően?
- Rugalmasan **bővíthető**, módosítható-e?
- Rendelkezik-e **interfész** felülettel más alkalmazásokhoz, hálózathoz?
- Illeszkedik-e a meglévő **hardverhez**?
- A **tudásbázis mérete** elegendően nagy-e?
- Belefér-e az **anyagi lehetőségekbe**?



Szakértőrendszerek megvalósítása ..

3. szint: a tudásbázis

A tudásbázis tartalmazza az ismeretet és szakértelmet. Minél teljesebb a tudás, annál erőteljesebb a szakértőrendszer. Fontos a tudás szemléltetésének módja:

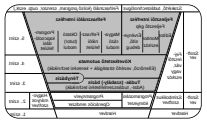
- Predikátum logika
- Szemantikus háló
- Keret
- Szabályalapú
- Hibrid modell.

4. szint: a következtető automata

Ez teszi **aktívvá** a tudásbázisban tárolt ismeretet.

Fontos jellemzője a következtetési folyamat **vezérlés stratégiája**.

Másik összetevője a **keresési módszer**, mely meghatározza, hogy milyen módon járja be a tudásbázist a következtető automata. **Lehet**: előrehaladó, visszafelé haladó, kétirányú, szembehaladó és a legkisebb kötelezettség elvén működő.



Szakértőrendszerek megvalósítása ..

5. szint: az interfész

Az interfész jelenti a kapcsolatot a külvilág felé.

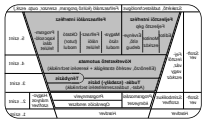
A szakértővel
A tudástechnológussal

Fejlesztői interfész

A felhasználóval
A szenzorokkal
A beavatkozószervekkel
Más alkalmazásokkal.

Felhasználói interfész

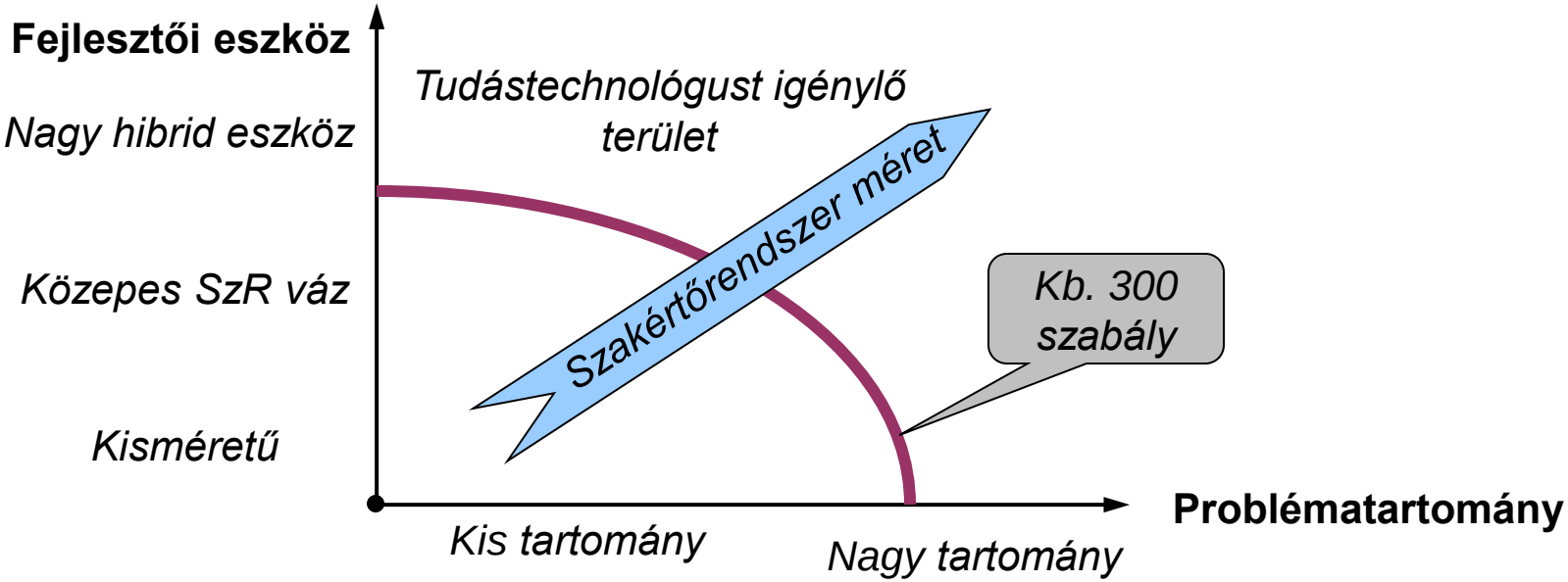
A fejlesztői és a felhasználói interfész közrefogja a szakértőrendszer fejlesztésének folyamatát..



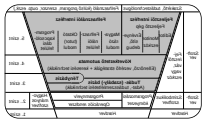
Szakértőrendszerek kifejlesztése

Méretfüggő fejlesztési megközelítés:

- Kis rendszereket: programozással nem hivatásszerűen foglalkozók, pl. a szakértő
- Közepes rendszereket: nem MI specialista programozók
- Nagy rendszereket: hivatásosak, tudástechnológus és szakértő, team.



Szakértőrendszerek kifejlesztése ..



Forrás: Paul Harmon - Rex Maus - William Morrissey:
Expert Systems: Tools & Applications
John Wiley & Sons, New York, 1988. p289.

Szakértőrendszer kifejlesztésének lépései

1. Fejlesztői (Front End) elemzés
   - A megfelelő probléma beazonosítása
   - A gazdaságosság mérlegelése
   - A vezetés támogatásának megnyerése
2. Feladatelemzés
   - A feladat beazonosítása
   - Az új rendszer viselkedésének behatárolása
   - A szükséges tudás behatárolása
3. Prototípus kifejlesztése
   - Esettanulmányok készítése (kritériumok felállítása)
   - Egy kisméretű rendszerrel bizonyítani az elv helyességét és tapasztalatot szerezni
4. A rendszer kifejlesztése
   - Újragondolni az átfogó struktúrát, ha szükséges
   - Újabb tudással bővíteni a rendszert
5. Tesztelés
   - A rendszer tesztelése valódi felhasználókkal
   - Felülvizsgálat, ha szükséges
6. Installálás
   - Telepíteni a rendszert a működési környezet hardverén
   - Betanítani a felhasználókat
7. Karbantartás
   - A rendszer aktualizálása, ha szükséges.

TERMÉSZETES NYELVEK FELDOLGOZÁSA

Szótárak – lexikonok – hatékony számítógépes tárolása

- **Tárolandó elemek:**
  - 1..43 karakteres szavak, túlnyomórészt nem hosszabbak, mint 17 karakter, átlagosan kb. 10 karakteresek.
  - A karakterkészlet egy természetes nyelv írásjeleiből áll, kb. 116 féle.
- **Követelmények** statikus és dinamikus szótáraknál egyaránt:
  - kis tárigény
  - gyors keresés.
- **További követelmények** dinamikus szótáraknál:
  - gyors beillesztés
  - gyors törlés.
- **Lehetséges szótár-adatszerkezetek:**
  - Nem hatékonyak: lista szekvenciális kereséssel,
    rendezett lista bináris kereséssel
  - Hatékony megoldás: keresőfa, ill. véges állapotú automata.

Felhasznált irodalom: Prószéky Gábor - Kis Balázs: Számítógéppel emberi nyelven SZAK Kiadó 1999

Szótárak tárolása keresőfával, vagy véges állapotú automatával

- A véges állapotú automata (finite state automaton, FSA) **hálós gráf**fal ábrázolható, mely kibontható a kevésbé hatékony memóriafoglalású keresőfának megfelelő **fagráf** alakba.

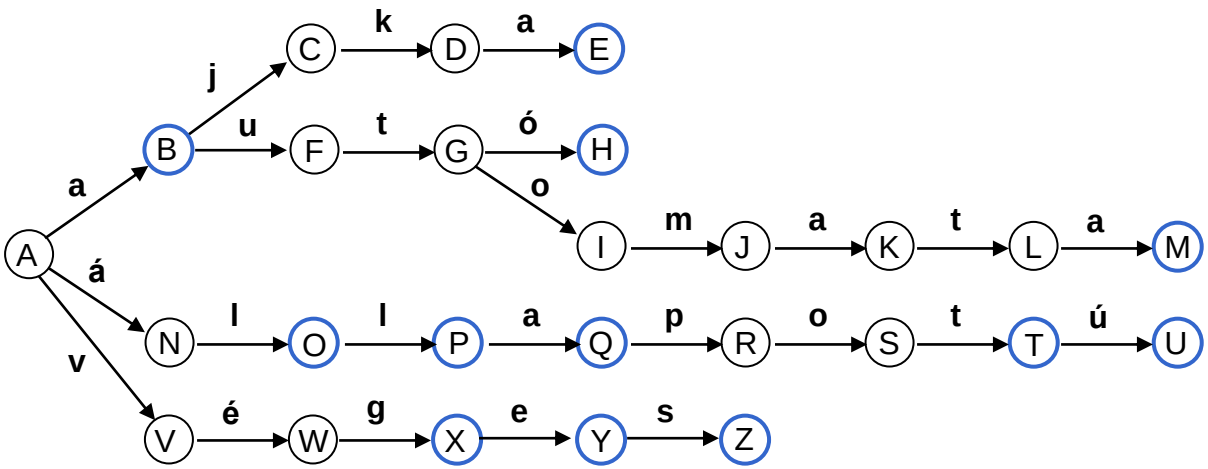
- **Előny:** a szó hosszának megfelelő lépésben képesek megállapítani, hogy a szó a szótár része, vagy sem.

- **Működés:** a bemenetként kapott szó karaktereit sorra véve, a már beolvasott karakterek által meghatározott állapotból egy lépésben elérhető állapotok közül az éppen kapott karakter által meghatározott újabb állapotba lépnek át. Amennyiben az utolsó karakter ún. elfogadó állapotba juttatja, akkor a szót elfogadva állnak le, egyébként a szót nem elfogadva állnak le, azaz a szó nem tartozik a fa, ill az automata által ismert szavak közé.

Felhasznált irodalom: Prószéky Gábor - Kis Balázs: Számítógéppel emberi nyelven SZAK Kiadó 1999
Képforrás: www.contemplativemind.org

Szótárak tárolása keresőfával, vagy végesállapotú automatával ..

- **Példaként** adjunk meg egy véges állapotú automatát állapotátmenet-gráffal:



- Az állapotokat a körökbe írt nagybetűk azonosítják.
- A szótárban található szavakat elfogadó állapotok vastag körrel jelölve.
- Az elfogadott szavak: **a, ajka, autó, automata, ál, áll, állapot, állapotú, vég, vége, véges**.
- Nem elfogadott szavak pl.: **baj, aj, ajkain**.

Felhasznált irodalom: Prószéky Gábor - Kis Balázs: Számítógéppel emberi nyelven SZAK Kiadó 1999

- **Szótárak tárolása keresőfával, vagy végesállapotú automatával ..**
  - Látható, hogy az automata **tárigénye csökkenthető**, ha lemondunk a fagráf-szerkezetről, pl. az E állapotot törölve és D-t L-lel összevonva.
  - Automatás tárolás esetén valós természetes nyelvek esetén a tárigény a listás tárigény **tizedére** csökkenhet.

Felhasznált irodalom: Prószéky Gábor - Kis Balázs: Számítógéppel emberi nyelven SZAK Kiadó 1999
Képforrás: www.elmar-lorey.de

Természetes nyelvek feldolgozása: szóelemzés

Természetes nyelvek:

- **Mondatépítés eszközei:** kötött szórend
  elöljárószavak
  toldalékolás.

- **A mondatépítési mód kihatása** a nyelv szavainak sokszínűségére:
  - Nem toldalékoló nyelvek (pl. angol): ~500 000 szóalak
  - Toldalékoló (agglutináló) nyelvek (pl. magyar): ~500 000 000 szóalak.

  meg-szent-ség-telen-ít-het-etlen-ség-es-kedés-e-i-tek-et

- **Morféma:** - tőmorféma = alapszó (szótő, az a rész, mely leginkább meghatározza a toldalékolt szó jelentését)
  - toldalékmorféma, affixum (prefixum: előtag, igekötő; szuffixum: képző, jel, rag).

Felhasznált irodalom: Prószéky Gábor - Kis Balázs: Számítógéppel emberi nyelven SZAK Kiadó 1999

Természetes nyelvek feldolgozása: szóelemzés ..

- **Szöveg morfológiai elemzése:** a szöveg morfémákra **bontása** és a morfématípusok **beazonosítása**. Szükség esetén a morféma alap (lexikális) alakjának meghatározása.

  Pl.:

  labdákat → labdá = labda [főnév] **+** k [többesszám jele] **+** at [tárgyrag]

- **Morfoszintaktikai szabályok:** megadják a morfémák kapcsolódásának szabályait. Gond: a nyelvészeti és számítógépes nyelvészeti optimális megoldások eltérése.

- **A morfológiai elemzés céljai:**

  - a kezelendő szóalakok számának **redukálása** (a toldalékmorfémák száma százas nagyságrendű csak)
  - a mondat szintaktikai, **nyelvtani elemzése** (főnév?, melléknév?, ige?, birtokosjel?, tárgyrag?, stb.)

Felhasznált irodalom: Prószéky Gábor - Kis Balázs: Számítógéppel emberi nyelven SZAK Kiadó 1999

Természetes nyelvek feldolgozása: szóelemzés ..

- **A természetes nyelvek szókészletének implicit tárolása**:
  - szóösszetételek generálásának szabályaival +
  - kivételek kezelésével.

- **A módszer használható:**
  - szintaktikailag (nyelvtanilag) helyes (összetett) **szavak generálására**
  - megadott szavak szintaktikai **helyességének ellenőrzésére**.

- **A módszer hibái:**
  - **Túlgenerálás** – a kivételkezelés hiányosságai és a jelentés tárolásának hiánya miatt olyan szavakat is létrehoznak, illetve elfogadnak, amelyek nincsenek az élő nyelvben: pl. almavaj
  - Inkább zártak, mint nyíltak: a nyelvben megjelenő **új szavak kezelése** nem automatikus.
    Fokozatok a nyitottságban:
    - nyitott új **szó**ra (gyakori igény)
    - nyitott új nyelvtani **szabály**ra (közepesen gyakori)
    - nyitott új **toldalék** megjelenésére (igen ritka).

Felhasznált irodalom: Prószéky Gábor - Kis Balázs: Számítógéppel emberi nyelven SZAK Kiadó 1999

Morfológia típusok

A szóösszetételek, a helyes morfémaláncolatok képzésének megadására alkalmas szabályokat megadó modellek típusai.

1. **Kétszintes morfológiák** (Koskenniemi, 1983):
   A felbontott szó morfémájához megadja az (esetleg kissé eltérő) alap (lexikális) **morfémát** a morféma **nyelvtani kategóriájával együtt**. Pl.:
   labdá+k → labda [főnév] + k [többesszám jele].
   A karakterről karakterre haladó elemzés véges automatát alkalmaz és jellemzője, hogy **megfordítható**: pl.:
   labda [főnév] + k [többesszám jele] → labdá+k .
2. **Folytatási osztályok:** egy morfémához megadja a lehetséges folytatómorfémákat. Pl.:
   labda [főnév] **(**+t [tárgyrag], +val [eszközhatározó rag], +nak [birtokosrag], … **);**
   +k [többesszám jele] (+at[tárgyrag], +nak [birtokosrag], +val [eszközhatározó rag],+ból [helyhatározó rag] …)
   A morfémák osztályozhatók az egyes folytatási osztályok tartalma alapján.
3. **Unifikációs modellek:** nem a kapcsolódó morfémákat, hanem a morfémát **megelőző és követő morfémák jellemzőit** (morfoszintaktikai és fonológiai, hangképzési tulajdonságok) tárolja. Igekötőknél a megelőző, ragoknál a következő morféma hiányzik, így jellemzői is hiányoznak.

Felhasznált irodalom: Prószéky Gábor - Kis Balázs: Számítógéppel emberi nyelven SZAK Kiadó 1999

Kétszintes morfológia

- Az egyszerű véges automata csak a szintaktikai elemzés legelső lépését képes elvégezni egy toldalékolt szóval: megállapítja, hogy része a modellezett nyelvnek, vagy sem. (Jól írt szó – hibásan írt szó.)

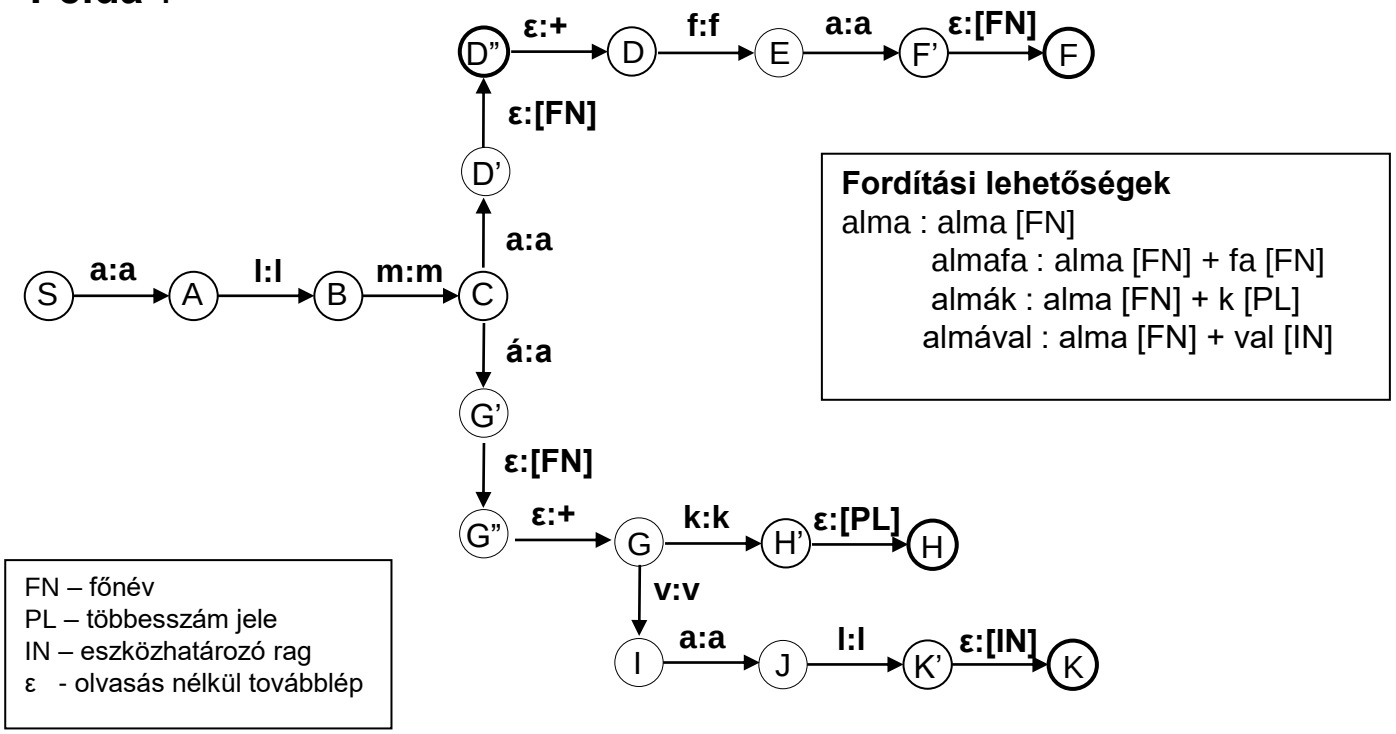
- **Az automata kibővítésével** az automata kimenete gazdagítható: képes megadni
  - a morfémák alap lexikális alakját (értelm -> értelem),
  - a morfémák morfoszintaktikai kódját (-ja -> [birtokos jelző])
  - stb.

- Ilyenkor az automata az input szöveghez **egy output leírást ad**, azaz az inputot lefordítja az outputra. A szöveget (surface level) és a hozzátartozó leírást (lexical level) az automata által kezelt két szintnek tekinthetjük, mivel még az is jellemzi, hogy a leírás is szolgálhat inputként és ekkor az automata azt a szövegre fordítja. Az ilyen, két szint közötti transzformációt végző automatát **véges fordító**nak (finite state transducer, FST) nevezzük.

Felhasznált irodalom: Prószéky Gábor - Kis Balázs: Számítógéppel emberi nyelven SZAK Kiadó 1999

Kétszintes morfológia ..

- **Példa***:



- Bár a **visszafelé fordítás** első ránézésre nem egyértelmű, a C állapot után mindkét irányba folytatva a fordítást, a D, ill a G állapotok után a választás egyértelmű.

** Forrás: Prószéky Gábor - Kis Balázs: Számítógéppel emberi nyelven SZAK Kiadó 1999*

Unifikációs morfológia (Prószéky, 1994)

- **Szóelemzés, morfémákra bontás:**
  - morfémák szótári keresésével, plusz
  - a szomszédos morfémák illeszkedésének vizsgálatával.

  Ezáltal a keresés és az illeszkedési szabályok alkalmazása különválik.

- Egy morfémához le van tárolva a megelőző morfémákhoz való kapcsolódás szükséges feltételeit megadó és a rákövetkező morfémához való kapcsolódás feltételeit megadó ismeret. Kivétel: szókezdő és szózáró morfémák, ahol csak a rákövetkezőhöz, ill. a megelőzőhöz való kapcsolódás létezhet.

- Példa*:

  szó [ ]

  [+névszó +fn +szótári –elöl –kerek –PL –PERS +ACC –ACCkötő +DAT +INS:V]

** Forrás: Prószéky Gábor - Kis Balázs: Számítógéppel emberi nyelven SZAK Kiadó 1999*

Unifikációs morfológia ..

szav [ ] amire a megelőzőnek mind fogadókésznek kell lennie

[+névszó +fn -szótári –elöl –kerek +PL +PLkötő +PERS -ACC +DAT -INS] amit el tud fogadni a rákövetkezőnél, nem kell mind

képez [ ]

[-névszó +szótári +elöl –kerek –ÁS]

képz [ ]

[-névszó +szótári +elöl –kerek –ÁS]

nak [+névszó –elöl +DAT]

[ ]

val [+névszó –elöl +INS:V]

[ ]

kal [+névszó –elöl +INS:K]

[ ]

ak [+névszó –elöl –kerek +PL +PLkötő]

[+névszó –elöl –kerek –PL –PERS +ACC +ACCkötő +DAT +INS:K]

at [+névszó –elöl –kerek +ACC +ACCkötő]

[ ]

Forrás: Prószéky Gábor - Kis Balázs: Számítógéppel emberi nyelven SZAK Kiadó 1999

Unifikációs morfológia ..

A + a morfoszintaktikai-fonológiai tulajdonság meglétét, a – a hiányát jelzi. Pl.:

+fn: főnév	
+szótári : torzítatlan, szótári alapalak	-szótári : nem ilyen
+elöl: hangképzése a szájban elöl történik	-elöl: nem elöl képzett
+kerek: ajakkerekítéses	-kerek: nem ajakkerekítéses
+PL: többesszám lehet	-PL: többesszám nem lehet
+ACC: tárgyrag (-t) lehet	
+ACCkötő: tárgyrag kötőhanggal (-at, -et) lehet	-ACCkötő: kötőhanggal nem lehet
+DAT: részeshatározó rag (-nak, -nek)	
+INS: V v betoldás lehet	-INS: nem lehet betoldás
+ÁS: -ás, -és képző lehet	-ÁS: nem lehet
(- PERS: a forrásban nem definiált)	

Forrás: Prószéky Gábor - Kis Balázs: Számítógéppel emberi nyelven SZAK Kiadó 1999

Unifikációs morfológia ..

- A morfológiai elemzés, azaz a megfelelő morfémák egymásra-következésének meghatározásához két dolgot kell nézni:

  1. a morfémák szón belüli sorrendjét

  2. a szomszédos morfémák egymáshoz illeszkedését a kapcsolódási pontban: a rákövetkező morféma igényeit a megelőző morfémának teljesítenie kell.

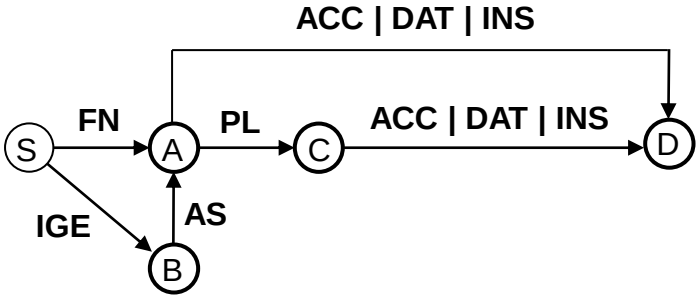
Forrás: Prószéky Gábor - Kis Balázs: Számítógéppel emberi nyelven SZAK Kiadó 1999

Unifikációs morfológia ..

1. **A morfémák szón belüli sorrendjének meghatározása**

   A morfémák lehetséges sorrendjei modellezhetők véges automatával, melynek átmeneti **operátorai most az egyes morfématípusok**. Egy lehetséges morfématípus-kapcsolódási sorozat az automata egy bejárási útját adja.

   Pl.:



Az automata például a következő morfématípus-sorozatokat fogadja el:

FN+PL (ház+ak);
FN+PL+ACC (ház+ak+at);
IGE+AS+DAT (vág+ás+nak)

de a következőket nem:

FN+PL+AS (ház+ak+ás);
FN+DAT+PL (ház+nak+ok)

Forrás: Prószéky Gábor - Kis Balázs: Számítógéppel emberi nyelven SZAK Kiadó 1999

Unifikációs morfológia ..

2. **Illeszkedésvizsgálat, unifikációval**

   Az 1. lépés még nem ad minden esetben jó megoldást. Ezért a második lépéssel **ki kell zárni a helytelen elfogadásokat** a kapcsolódó morfémák morfoszintaktikai-fonológiai elvárásainak egyeztetésével. A kapcsolódás feltétele, hogy a két jegyszerkezetben, elvárássorban **ne legyenek azonos jellemzők eltérő értékkel** (előjellel)
   Pl. a következő szavak felbontása, morfológiai felbontása **helyes**nek vehető a vastagon jelzett elvárások egyezése miatt:

szó
[+névszó +fn **+szótári** –elöl –kerek –PL –PERS +ACC –ACCkötő +DAT +INS:V]

szó+nak
[**+névszó** +fn +szótári **–elöl** –kerek –PL –PERS +ACC –ACCkötő **+DAT** +INS:V]
[**+névszó –elöl +DAT**]

Forrás: Prószéky Gábor - Kis Balázs: Számítógéppel emberi nyelven SZAK Kiadó 1999

- **Unifikációs morfológia ..**

szó+val

[**+névszó** +fn +szótári **–elöl** –kerek –PL –PERS +ACC –ACCkötő +DAT **+INS:V**]

[**+névszó –elöl +INS:V**]

szav+ak+at

[**+névszó** +fn -szótári **–elöl –kerek +PL +PLkötő** +PERS -ACC +DAT -INS]

[**+névszó –elöl –kerek +PL +PLkötő**]

[**+névszó –elöl –kerek** –PL –PERS **+ACC +ACCkötő** +DAT +INS:K]

[**+névszó –elöl –kerek +ACC +ACCkötő**]

szav+ak+kal

[**+névszó** +fn -szótári **–elöl –kerek +PL +PLkötő** +PERS -ACC +DAT -INS]

[**+névszó –elöl –kerek +PL +PLkötő**]

[**+névszó –elöl** –kerek –PL –PERS +ACC +ACCkötő +DAT **+INS:K**]

[**+névszó –elöl +INS:K**]

Forrás: Prószéky Gábor - Kis Balázs: Számítógéppel emberi nyelven SZAK Kiadó 1999

Unifikációs morfológia ..

- **Példák az automata által elfogadott, de az unifikáció által a pirossal jelzett ütközések alapján kiszűrt esetekre:**

szav
[+névszó +fn **-szótári** –elöl –kerek +PL +PLkötő +PERS -ACC +DAT -INS]

szav+val
[**+névszó** +fn -szótári **–elöl** –kerek +PL +PLkötő +PERS -ACC +DAT **-INS**]
[**+névszó –elöl +INS:V**]

Forrás: Prószéky Gábor - Kis Balázs: Számítógéppel emberi nyelven SZAK Kiadó 1999

Kereső eljárások

- Cél: egy adott problémára a szóbajöhető lehetőségek közül egy adott kritériumrendszernek eleget tévő megoldás megtalálása.

- A keresést az ismeretfeldolgozási eljárások közé soroljuk, mely révén a szemléltetett ismeretet aktívvá tesszük, tudást nyerünk ki.

- Gyakran használják ismeretfeldolgozási eljárásként a keresést következtetőgráfokon következmény előállításra.

Kereső eljárások ..

- A kereséssel történő problémamegoldás lépései
  1. A probléma beazonosítása kereséssel megoldható problémaként
  2. A problémára vonatkozó ismeretek reprezentálása, szemléltetése
  3. A kereső eljárás alkalmazása.

1. Kereséssel megoldható problémák jellemzői

   Egy operátorkészlettel bejárható állapotok mindegyikén értelmezhető egy kritériumfüggvény. A probléma megoldása megfeleltethető a kritériumfüggvény adott értékének, értékhalmazának, vagy extrémumának.

2. A problémára vonatkozó ismeretek reprezentálása

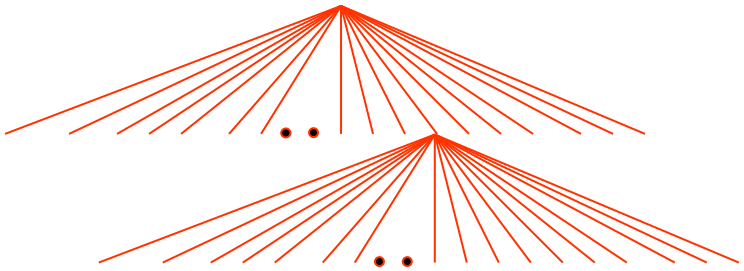
   Lásd a tudásszemléltetés elvárt jellemzőit, Patrick Winston szerint.

   Az ismeretek reprezentálása erősen kihat az ismeretfeldolgozás, jelen esetben a keresés hatékonyságára. Ennek megvilágítására tekintsük a következő példát:

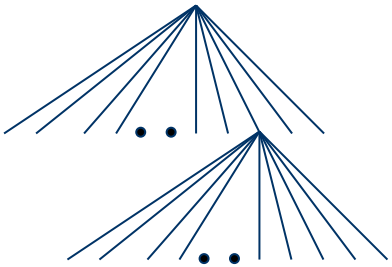
Kereső eljárások ..

- Egy 20*20 méretű négyzetrácsos játéktéren kell az amőba játékban meghatározni azt a helyet, amely a lépésen következő szempontjából jó lépésnek minősül, hat féllépésre előre tekintve.

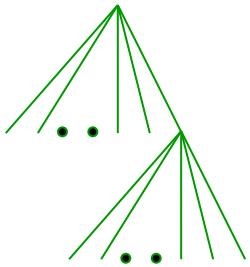
- **Lehetséges reprezentációk**



1. Az összes üres mezőt vizsgáljuk: a lehetőségek száma a játék első lépéseinél százas nagyságrendű, nagyon nagy elágazási tényezőjű fagráffal szemléltethetők a hat féllépéses részjátékokra.



2. Csak az előző lépés környezetében lévő üres helyeket vizsgáljuk egy 9*9-es mezőben, mivel az utolsónak lerakott jel csak ebben a részben alkothat nyerő ötöst. Az elágazási tényező maximum 80-as.



3. Csak az előző lépésre illesztett csillag alakzat üres helyeire végezzük a vizsgálatot, mivel ezek a helyek vannak közvetlen kapcsolatban az utolsónak lerakott jellel. Az elágazási tényező maximum 32-es.

   Megj.: a reprezentációk gyengülésétől sokkal nagyobb előnyt jelent a kivitelezhető keresések hatékonyságnövekedése. A reprezentációnak összhangban kell lennie a kereső eljárással.

dr.Dudás László

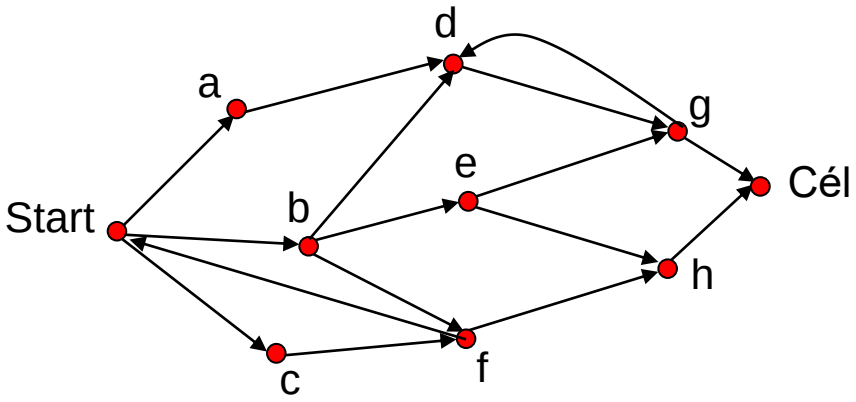
Kereső eljárások ..

3. A kereső eljárás alkalmazása
   A kereső eljárás alkalmazása a feladat megoldására az eljárások jellemzőinek ismeretében történhet. Ilyen közös jellemzők, melyek a megoldandó konkrét probléma függvényei:

   - **Teljesség:** ha létezik megoldás, azt az eljárás meg is adja.
   - **Optimalitás**: a Start-Cél útvonal egyben költségoptimális is.
   - **Időigény**: az algoritmus lépésszámára adott felső korlát.
   - **Tárigény**: a megoldás megtalálásához felhasznált memória méretére vonatkozó felső korlát.

A kereső eljárások összetevői

- **Állapottér**, reprezentálása az **állapotok**nak megfelelő csomópontokkal és az állapotok közötti mozgást jelentő **operátorok**nak megfelelő irányított élekkel.
- Kiinduló állapot: **Start**
- A kritériumnak megfelelő állapot/ állapotok: **Cél**.
- **Út, útvonal**: egy állapotból egy másik állapotba átvivő operátorsorozat, ill. érintett csomópontok rendezett listája.
- A keresés **leállási feltétele** előírhatja a kritérium teljesítését, vagy adott tűrésen belüli közelítését. Az utóbbi esetben *kvázioptimális* megoldásról beszélünk, mely gyakorlatilag jó és kompromisszumot jelent a keresés időigénye/költsége és a megoldás minősége között.



dr.Dudás László

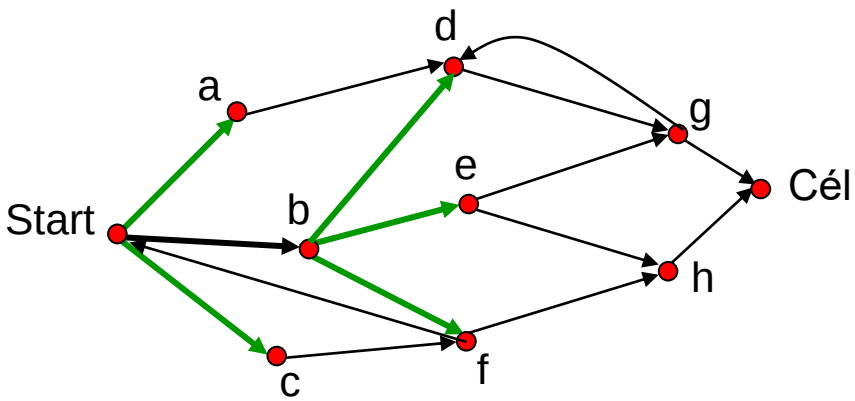
A kereső eljárások összetevői ..

- **Az operátor költsége**: az operátorok a feladat valós tartalmától függő költséggel rendelkezhetnek: pl. legrövidebb út keresése városok között – az operátor költsége az útszakasz hossza. De lehet az operátor költsége érdektelen is, például bűvös kocka kitekerésénél csak a célállapot megtalálása, illetve az odavezető út fontos.
- **Az út költsége**: az úton alkalmazott operátorok költségének összege.
- **A keresés költsége**: a keresés idő- és tárigényéhez kapcsolódó költség.
- **A keresés teljes költsége**: az út költsége + a keresés költsége. Pl. városban történő útkeresés benzinköltsége: az útszakaszokon is fogy a benzin és az elágazásoknál az útválasztási döntés meghozatala ideje alatt is jár a motor.

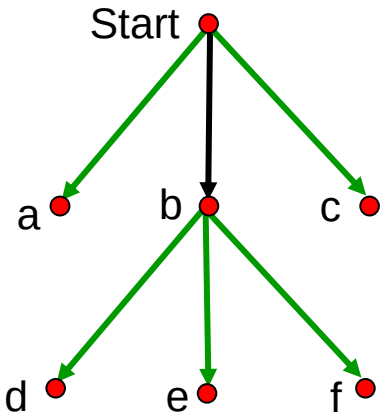
A kereső eljárások összetevői ..

- **A keresőgráf:** egy fagráf, melynek csúcsa a Start állapot, valamelyik levele pedig, a Cél állapot, amennyiben létezik megoldás.
- **Kiterjesztés**: egy állapot kiterjesztése alatt az állapotból egyetlen lépéssel elérhető állapotok feltérképezését, azokba való betekintést, de még bele nem lépést értünk. Megfelel a keresőgráf egy adott pillanatnyi levélállapotától egy szinttel lejjebb lévő állapotok keresőgráfba való megrajzolásának.

Állapotgráf

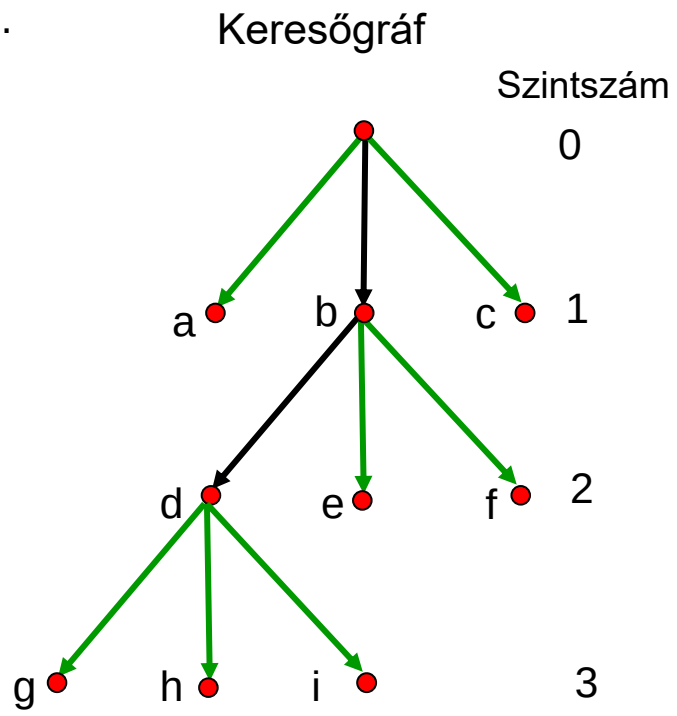


Keresőgráf



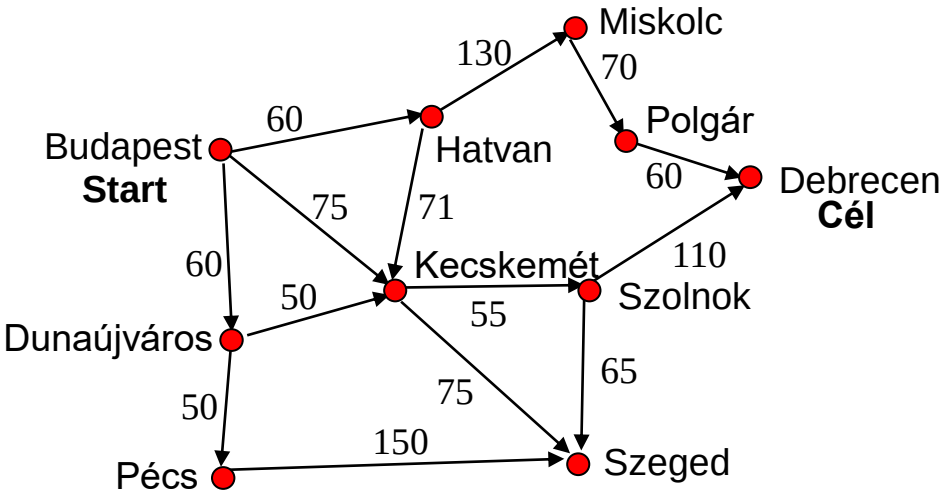
A keresőgráf jellemzői

- **Elágazási tényező (branching factor, b):** egy adott csomópontból megtett kiterjesztés ágainak száma.
- **Átlagos elágazási tényező**: több csomópontra, leggyakrabban a teljes keresőgráfra értelmezett jellemző, a figyelembe vett csomópontok elágazási tényezőinek összege osztva a figyelembe vett csomópontok számával.
- A keresőgráf **mélysége** (m): a gráf legmélyebb szintjének száma, a Start a 0. szint.
- A **megoldás mélysége** (depth, d).
- Esetleges **mélységi korlát** (l).
- A keresés **frontja**: az összes, kiterjesztéssel feltárt, de még bele nem lépett csomópont, azaz a keresési fa levelei.
- Az n állapotba vezető út költsége g(n)
- Az n állapotból a Cél elérésének becsült költsége h(n)
- Az n állapoton átvezető út (becsült) költsége f(n). (A*)



Példák kereséssel megoldható feladatokra

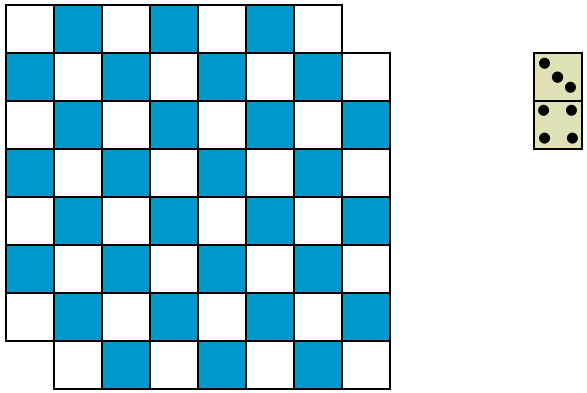
- **Útkeresés két város között**



- Probléma: melyik a legrövidebb útvonal Budapestről Debrecenbe?
- Start állapot: Budapesten vagyunk.
- Állapotok: valamelyik városban vagyunk.
- Leállási feltétel: a legrövidebb úton odajutva a Célban, Debrecenben vagyunk már? (A leállási feltétel egyben optimális utat is garantál.)
- Operátorok: utazás a szomszédos városok között.
- Költség: a megtalált Start-Cél útvonal hossza.

Példák kereséssel megoldható feladatokra ..

- **Lefedési probléma**



- Probléma: hogyan lehet 31 dominóval lefedni egy, az átellenső sarkain csonka sakktáblát?
- Start állapot: az üres sakktábla.
- Állapotok: mindig eggyel több dominó valamilyen elrendezésben a sakktáblán.
- Leállási feltétel: mind a 62 mező le van fedve, vagy a lefedés lehetetlen voltának megállapítása.
- Operátorok: egy újabb dominó elhelyezése a sakktáblán.
- Költség: nincs.

Általános kereső eljárás

- A kereső eljárások lényegi lépéseit tartalmazza az általános eljárás, amelytől a különböző eljárások csak kis részben térnek el.

1. A keresési front legyen egyenlő a Start állapottal.
2. Ha a front üres, akkor nincs megoldás, vége.
   Egyébként legyen n a front egyik állapota.
3. Ha n kielégíti a Cél-kritériumot, akkor add vissza a hozzá vezető útvonallal együtt, vége.
4. Egyébként a front n állapota helyére vedd fel a kiterjesztésével adódó új állapotokat és jegyezd fel a hozzájuk vezető útvonalakat.
   Ismételd a 2. ponttól.

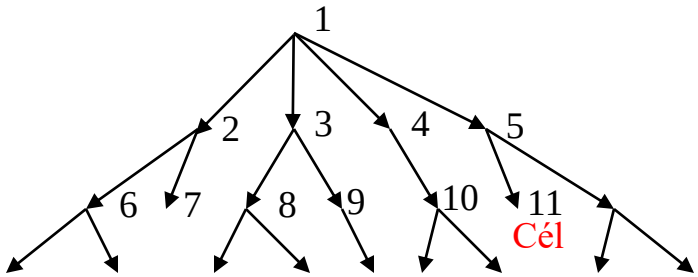
- Az algoritmus szabad kezet adó pontja a 2., amelyben arról döntünk, hogy melyik állapot szomszédai irányába terjesszük ki a keresést. Ezen döntést végezhetjük gépiesen, a neminformált eljárások esetében, vagy a keresési feladatra vonatkozó információkra támaszkodva, az informált, vagy másnéven heurisztikus kereső eljárások esetében.

Neminformált kereső eljárások (vak keresés)

- Nincs információnk az aktuális állapot és a cél távolságára (költség, vagy lépésszám). Az eljárás csak arra képes, hogy észrevegye, ha a Cél állapotban van.
- Nem hatékonyak
- Általánosan használhatók.
- Típusok:
  - Szélességben először keresés (breadth-first search)
  - Elágaztatás és ugrálás (branch and bound), vagy másnéven egyenletes költségű keresés (uniform cost search)
  - Mélységben először keresés (depth-first search)
  - Mélységben először keresés mélységi korláttal (depth limited search)
  - Iteratív mélyítés (iterative depth-first search)

Szélességben először keresés

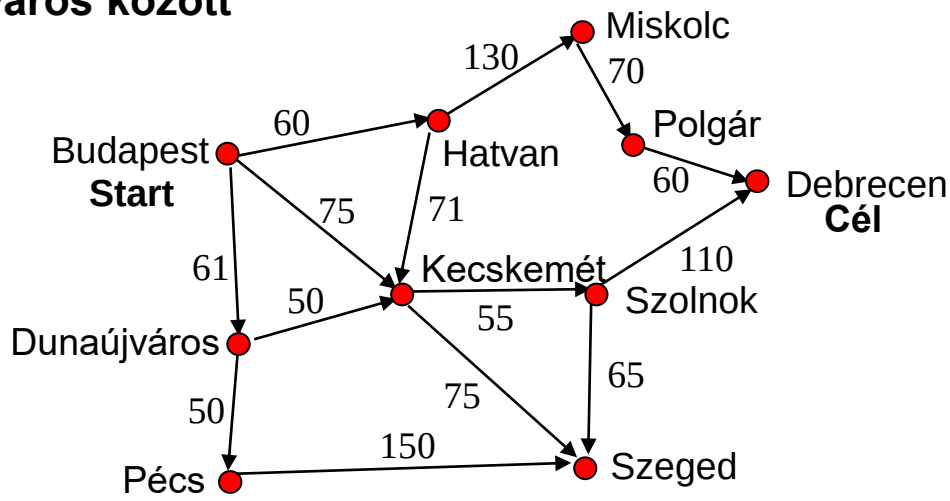
- Egy adott mélységi szint csomópontjainak mindegyikét kiterjeszti, mielőtt a következő mélységi szintre lépne.

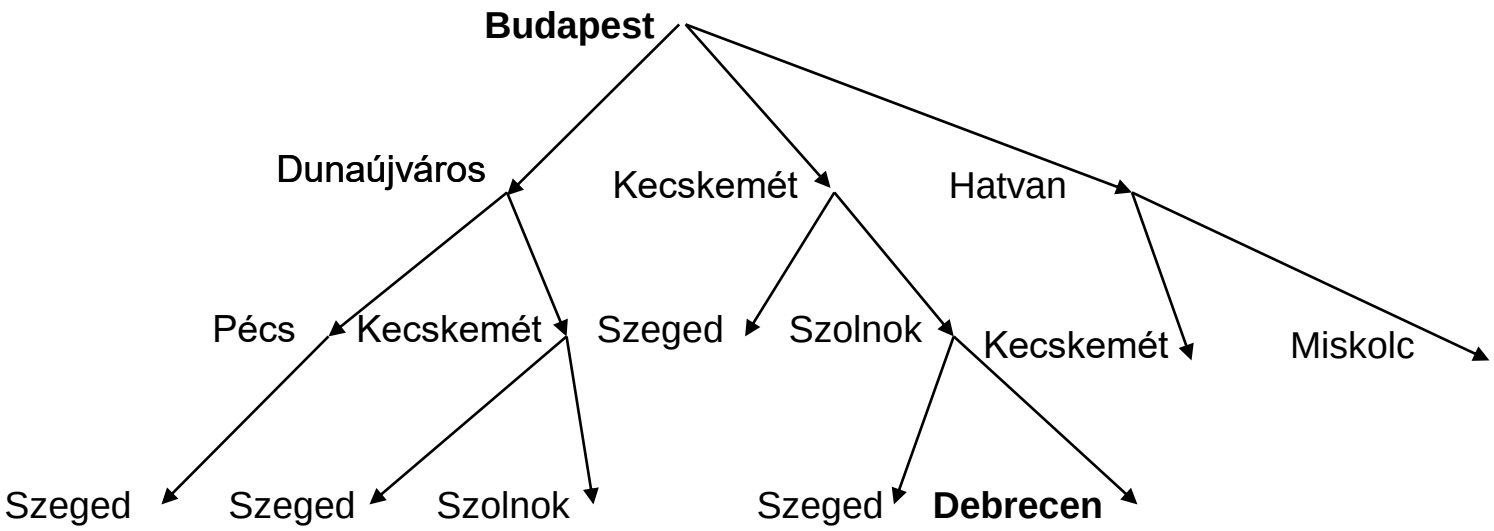


- Az általános kereső eljárás a következőképpen módosul:
  - 2. Legyen a front első állapota az n választott állapot.
  - 4. A kiterjesztéssel kapott új állapotokat csatold a frontlista végéhez.
- Az eljárás
  - Teljes
  - Optimális (amennyiben az útszakaszok egyforma költségűek)
  - Időigénye b^d, (meredeken nő a mélységgel)
  - Tárigénye b^d, (meredeken nő a mélységgel).

Példa a szélességben először keresésre

- **Útkeresés két város között**





Elágaztatás és ugrálás

- A front azon állapotába lép, amelyikhez a legkisebb költségű út vezet a Start-tól.
- Ha az útszakaszok költsége egyforma, akkor a szélességben először keresésre vezet.

- Az általános kereső eljárás a következőképpen módosul:

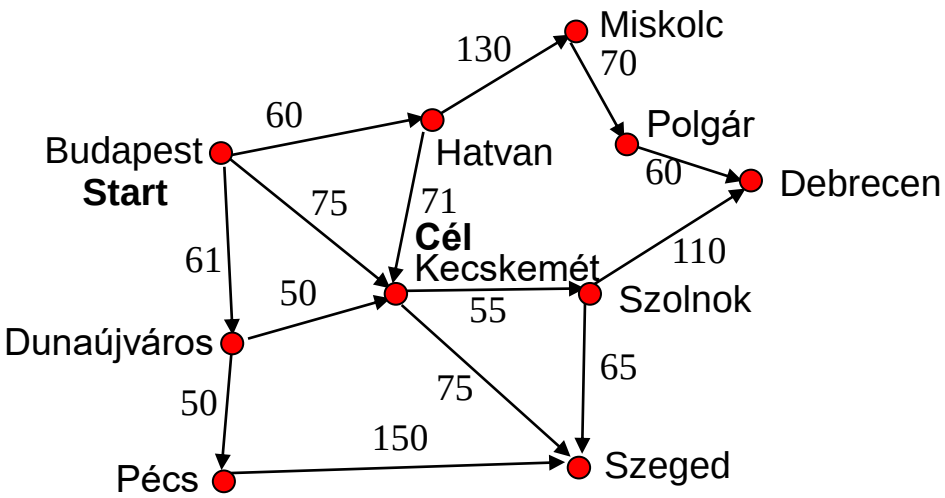
  2. Legyen a front első állapota az n választott állapot.
  4. A kiterjesztéssel kapott új állapotokat add a frontlistához, majd rendezd az állapotokat növekvő költség szerint.

- Az algoritmus leáll, ha Cél-állapotba léptünk, azaz a Front összes állapotába nagyobb költségű út vezet.

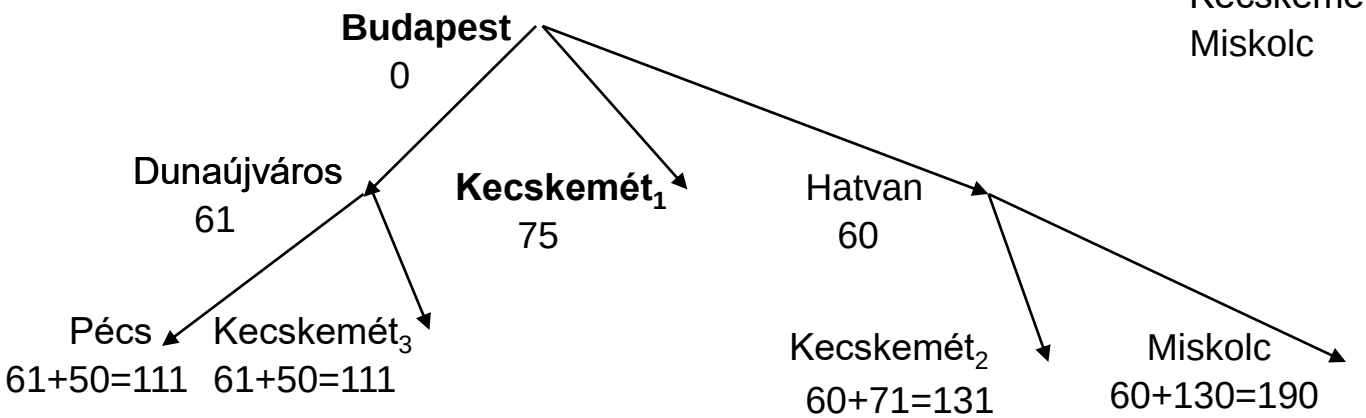
- Az eljárás
  - Teljes
  - Optimális
  - Időigénye $\approx b^d$, (meredeken nő a mélységgel)
  - Tárigénye $\approx b^d$, (meredeken nő a mélységgel).

Példa az *elágaztatás és ugrálás* keresésre

- **Útkeresés két város között**



Sorsz.	Front	Választás
0.	Budapest	Budapest
1.	Dunaújváros	
	Kecskemét$_1$	
	Hatvan	Hatvan
3.	Dunaújváros	
	Kecskemét$_1$	
	Kecskemét$_2$	
	Miskolc	Dunaújv.
4.	Pécs	
	Kecskemét$_3$	
	Kecskemét$_1$	
	Kecskemét$_2$	
	Miskolc	Kecskemét$_1$



Mélységben először keresés

- A keresési fát balra lefelé tartva járja be. Ha nem tud lefelé menni tovább, visszalép a legalsó elágazásig és a következő ágat választja. Visszalépéskor a sikertelen ág állapotait ejti. (Lásd Prolog működését.)

- Az általános kereső eljárás a következőképpen módosul:

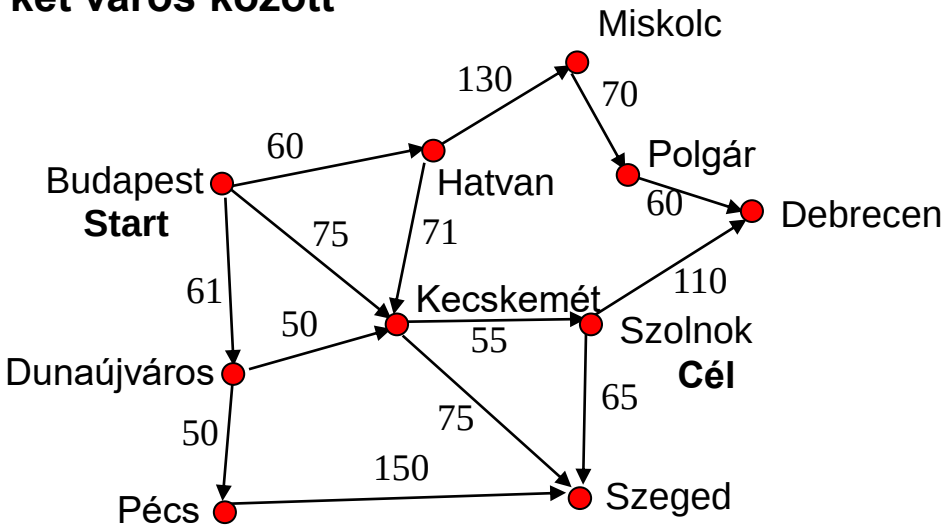
  2. Legyen a front első állapota az n választott állapot.
  4. A kiterjesztéssel kapott új állapotokat add a frontlista elejéhez.

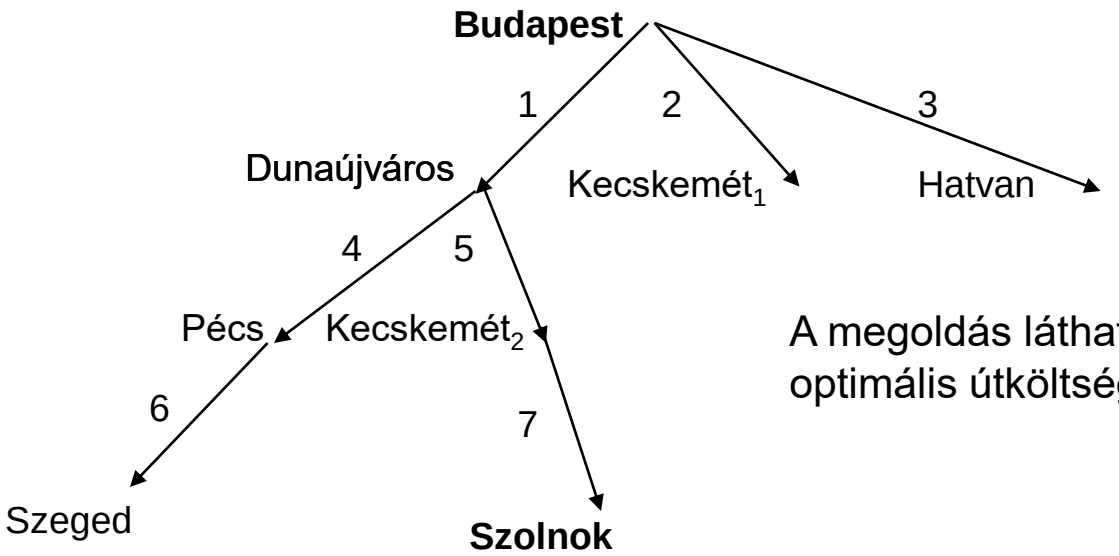
- Az eljárás
  - Teljes, ha nincs végtelen, vagy (a memóriának) túl mély ág.
  - Nem optimális
  - Időigénye $\approx b^d$, (meredeken nő a mélységgel)
  - Tárigénye b*d, (nagyon kis igény!).

- Csak egyetlen állapotba vezető út állapotait és az út leágazásain található állapotokat kell tárolnia.

Példa a mélységben először keresésre

- **Útkeresés két város között**





A megoldás láthatóan nem optimális útköltségű.

Informált kereső eljárások (heurisztikus keresés)

- A front állapotai közül belelépésre azt választjuk, amelyre vonatkozóan a konkrét megoldandó feladatra vonatkozó **információk valamilyen előnyt ígérnek**. Ilyen előny lehet: a célhoz legközelebbinek tűnik; a rajta való áthaladás a legkisebb Start-Cél költséggel kecsegtet; a legnagyobb lépést teszi a Cél irányában, stb.

- Az informált állapotválasztással a keresés hatékonysága nagyban fokozható.

- Az informált kereső eljárásokat gyakran **heurisztikus** kereső eljárásoknak nevezzük, mert a problématerületre vonatkozó tapasztalatra építenek. A heurisztika általában nagyban növeli a hatékonyságot, de tévedhet is.

- A heurisztikát a **h(n) heurisztikus kiértékelő függvény** segítségével számszerűsítjük, számítjuk az n állapot esetében.

- Heurisztikus kiértékelő függvényként gyakran alkalmazzák az n állapotból a Cél elérésének költségére vonatkozó becslést. A jó függvény nem becsüli túl a költséget és a Cél állapotra 0 értéket ad.

Informált kereső eljárások fajtái

- Globális információra támaszkodó eljárások
  - Legjobbat először keresés (Best first search)
  - A és A* keresés (A search, A* search)
  - Iteratív A* algoritmus (IDA* search)

- Lokális információra támaszkodó eljárások
  - Hegymászó keresés (Hill climbing)
  - Szimulált lehűtés (Simulated annealing)
  - Tabu keresés (Tabu search)

- A globális információ az állapottér bármely két pontja között származtatható, leggyakrabban az n állapot és a Cél-állapot között számítjuk és az n állapothoz rendeljük.
  Globális információ felhasználásával megtalálhatjuk a globális optimumot, az optimális költségű utat is.

- A lokális információ az n állapot és közvetlen szomszédai között számítható. Önmagában lokális információ alkalmazásával csak lokális extrémumot garantálnak az eljárások, gyakran azonban annak közelítésével is megelégszünk.

Legjobbat először keresés

- A legjobbat először keresés egy **heurisztikus kiértékelő függvénnyel** becsli az n állapotból **a Cél elérésének költségét** a front összes állapotára és abba az állapotba lép, amelyikre ez a költség a legkisebb. Az eljárás végződhet lokális optimumpontban is.

- Patrick **Winston** az eljárást hegymászókkal magyarázta: Egy hegymászókból álló csapat indul a hegycsúcs meghódítására. A csapat tagjai elágazásoknál szétválnak és rádiótelefonnal tartják a kapcsolatot, hogy mindig az a csapat mászhasson tovább, amely számára a csúcs elérése a legkönnyebbnek tűnik. Ha valamelyik csapat elakad, egy másik mászik tovább.

- Az általános kereső eljárás a következőképpen módosul:

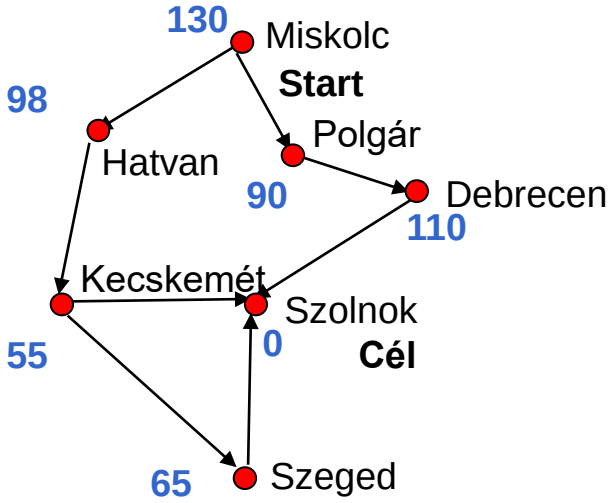
  2. … Legyen a front célhoz legközelebbinek becsült állapota az n választott állapot.

- Az eljárás
  - Teljes
  - Nem optimális
  - Időigénye $\approx b^m$, (meredeken nő a mélységgel)
  - Tárigénye $\approx b^m$, (meredeken nő a mélységgel).

dr.Dudás László

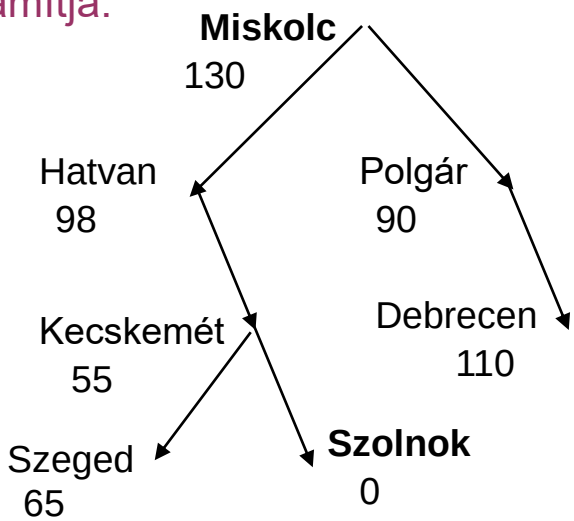
Példa a *legjobbat először* keresésre

- **Útkeresés két város között**



Sorsz.	Front	Választás
0.	Miskolc	Miskolc
1.	Hatvan Polgár	Polgár
3.	Hatvan Debrecen	Hatvan.
4.	Kecskemét Debrecen	Kecskemét
5.	Szeged Szolnok Debrecen	Szolnok

A heurisztikus kiértékelő függvény a légvonalbeli távolságot számítja.



Az A és A* keresés

- Az A keresés számítja a front összes állapotára a g(n) függvénnyel a Start-n távolságokat, valamint ugyanezen állapotokra egy h(n) **heurisztikus kiértékelő függvénnyel** becsli az n állapotból **a Cél elérésének költségét**, majd képezi ezek összegét: f(n)= g(n) + h(n).
  Abba az állapotba lép, amelyikre ez az áthaladó útvonalhossz becsült költsége a legkisebb.
  Az eljárás végződhet lokális optimumpontban is.

- Bizonyítható, hogy amennyiben a h(n) költségbecslő függvény nem becsüli túl a Cél elérésének költségét egyik állapotra sem, akkor az eljárás által adott útvonal egyben optimális is. Az ilyen függvény jele: f ’(n) , az eljárás neve ekkor A*.

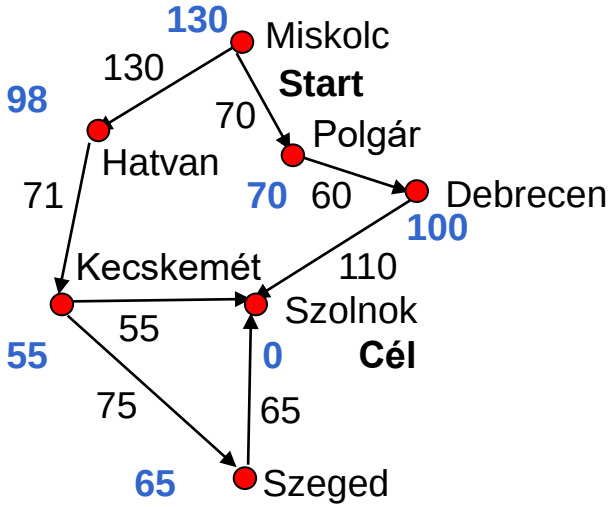
- Az általános kereső eljárás a következőképpen módosul:

  2. … Legyen a front állapotai közül kiválasztva az, amelynek legkisebb az f ‘(n) függvényértéke.

- Az eljárás
  - Teljes
  - Optimális
  - Időigénye nagy
  - Tárigénye nagy, gyakran túlcsordítja a memóriát.

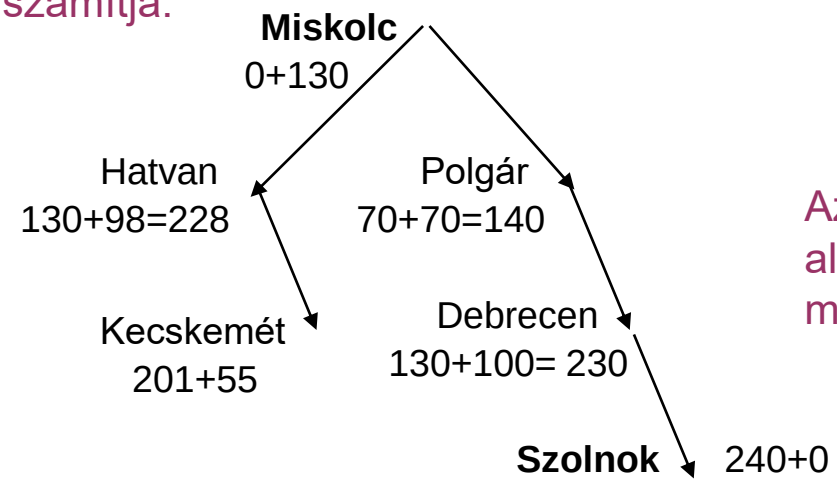
Példa az A* keresésre

- **Útkeresés két város között**



Sorsz.	Front	Választás
0.	Miskolc	Miskolc
1.	Hatvan Polgár	Polgár
3.	Hatvan Debrecen	Hatvan
4.	Kecskemét Debrecen	Debrecen
5.	Kecskemét Szolnok	Szolnok

A heurisztikus kiértékelő függvény a légvonalbeli távolságot számítja.



Az elágaztatás és ugrálás algoritmus és a legjobbat először módszer egyesítése.

A hegymászó keresés

- Lokális információt használ: csak a szomszédos állapotokat nézi.
- A legjobb szomszédot választja és gyorsan feljut egy közeli lokális csúcspontra.
- Lépései:

1. Legyen n a Start állapot.
2. Ha n Cél-állapot, add vissza a hozzávezető úttal együtt. Vége.
3. Egyébként állítsd elő n kiterjesztett állapotait. Legyen n új értéke ezek közül a legjobb.
   Ismételd 2.-től.

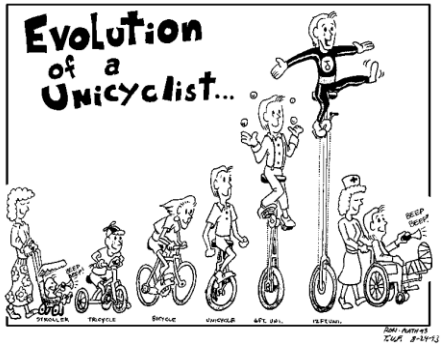
- Az eljárás:
  - Teljesége: amennyiben lokális maximum a kritérium, akkor azt megtalálja
  - Nem optimális
  - Időigénye: nagyon alacsony
  - Tárigénye: nagyon kicsi, mert nem tárol keresőgráfot.
- Probléma: erősen függ a felület alakjától.

1.

Evolúciós algoritmusok

- Az evolúciós algoritmusok a biológiai evolúciót modellezik. Az ember számtalan probléma megoldásához merítette az alapötletet a természetből. Pl. madarak szárnya → repülőgépszárny, bambuszrúd → rúdugrók üvegszálas rúdja, halak léghólyagja → tengeralattjárók merülési berendezése, emberi szemlencse → fényképezőgép optikája, stb.

- Amit az ember gondolkodással próbál megoldani, azt a természet gyakran hatékonyabban oldotta meg a változatosság és a szelekció eszközével. A többnyire kétszülős biológiai szaporodás és a többnyire külső hatásoknak köszönhető véletlenszerű megváltozás, a mutáció **az egyedek sokszínűségét** eredményezi, míg a túlélésért folyó verseny a jobb, életképesebb egyedek **kiválogatódását**, tulajdonságaik továbbörökítésének lehetőségét hordozza magában. E két működés a populáció környezeti elvárásoknak való jobb megfelelését, állandó alkalmazkodását jelenti.



1. www.ornl.gov/ORNLReview/ v34_1_01/p2b.jpg

www.tux.org/~bagleyd/unicycle_factory/ cartoons/evolution.gi

Evolúciós algoritmusok és az optimumkeresés

- Az evolúciós algoritmusokkal végzett problémamegoldást **biológiai számításnak** is nevezik.

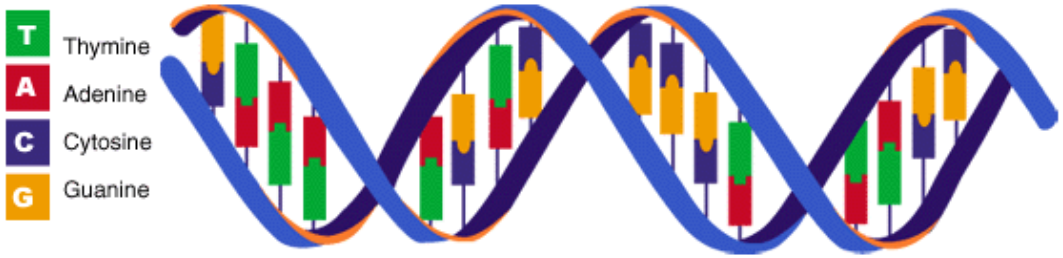
- Az evolúciós algoritmusok **populációjának** egyedei megfeleltethetők a keresés állapotterében található **állapotok** egy részhalmazának, az egyedek életképességét, jóságát mérő ún. **fittness függvény** megfeleltethető a keresés **kritériumfüggvényének**.

- Az evolúciós mechanizmus mintájára létrehozott programokkal akkor is megoldhatók a feladatok, ha a problémamegoldásra nem tudunk részletes algoritmust adni. Az irányított „próba-szerencse” módszer feladatérzéket-lenségéért, **robusztusságáért számítási idővel kell fizetnünk**. Ebben is a biológiai modellre hasonlít: az evolúció nem gyorsan zajló folyamat.

- A biológiai evolúció és az evolúciós algoritmusok **eltérése**
  A biológiai evolúció célja a környezet elvárásaihoz való minél hatékonyabb **alkalmazkodás**, míg az evolúciós algoritmusokkal globális **optimumok keresését** végezzük. Ezen optimumkeresési feladatok általában nem biológiai beágyazó környezetben zajlanak.

Evolúciós algoritmusok osztályozása

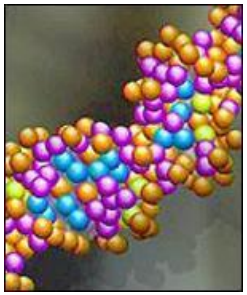
- **Evolúciós stratégiák**: a kezdet, (Rechenberg, 1973, repülőgépszárny optimalizálás.)
- **Evolúciós programozás**: programkód kifejlesztése a kódrészletek mutálódása és szelektálása által. Véges automaták automatikus kifejlesztésére (Fogel, Owens, Walsh, 1966.).
- **Genetikus algoritmus**, GA: keresztezés, mutáció és szelekció matematikai modellezése (Holland, 1975).
  - Osztályozó rendszerek
  - **Genetikus programozás**, GP: programok kitenyésztése adott feladatra. (Koza, 1992).



www.genecrc.org/site/ KO/images/helix.gif

Az Evolúciós algoritmusok általános lépései

1. Legyen P_0 a kezdeti populáció,
   k=0 a ciklusváltozó kezdőértéke.
2. Ha a megállási kritérium teljesül, add vissza a P_k populációt.
3. Egyébként bővítsd a P_k populációt új egyedekkel.
4. Szelekcióval állítsd elő a P_k populációból az új P_{k+1} populációt.
5. Inkrementáld a k ciklusváltozót és ismételd a 2. ponttól.


1.

A 3. lépésben a bővítés alkalmazhat szülőválasztást, keresztezést és mutációt.

A 4. lépés feladata a populáció létszámának eredeti értéken tartása a vesztes egyedek törlésével.

Amennyiben a populáció **egyelemű**, csak mutáció van, az algoritmus **helybeni**, és neve **sztochasztikus hegymászó**.

1. "***news.bbc.co.uk/olmedia/1520000/ images/_1522808_new150dna.jp***"

A Genetikus algoritmus

- A genetikus algoritmus egy **globális kvázioptimum** megtalálására kifejlesztett kereső algoritmus, mely alapvetően **problémafüggetlen**. A lokális informáltságú, de globális célra törő algoritmusok, a többszörös indítású hegymászó, a szimulált hűtés és a tabu keresés rokona.

1.

- A problémára vonatkozóan rendelkezésre álló **információ bevihető** az algoritmusba a valós egyedeknek megfelelő **fenotípusról** a számításban alkalmazott modelljére,
  a **genotípusra** való áttéréskor, valamint
  a keresztezés és a mutáció módjának megválasztásakor.
  Az ily módon kialakított, problémára szabott genetikus algoritmusok **hatékonysága megnő**.

- A globális optimum megtalálásában az egzakt analitikus módszerekhez (differenciálás, gradiens módszer, szimplex módszer) viszonyítva lassú, pontatlan, de diszkrét állapottereken is működik, igénytelen a feladattal szemben.
  A hasonlóan robusztus és igénytelen kimerítő kereséstől hatékonyabb.

1. www.theatlantic.com/issues/ 98jan/climate.htm

A Genetikus algoritmus ..

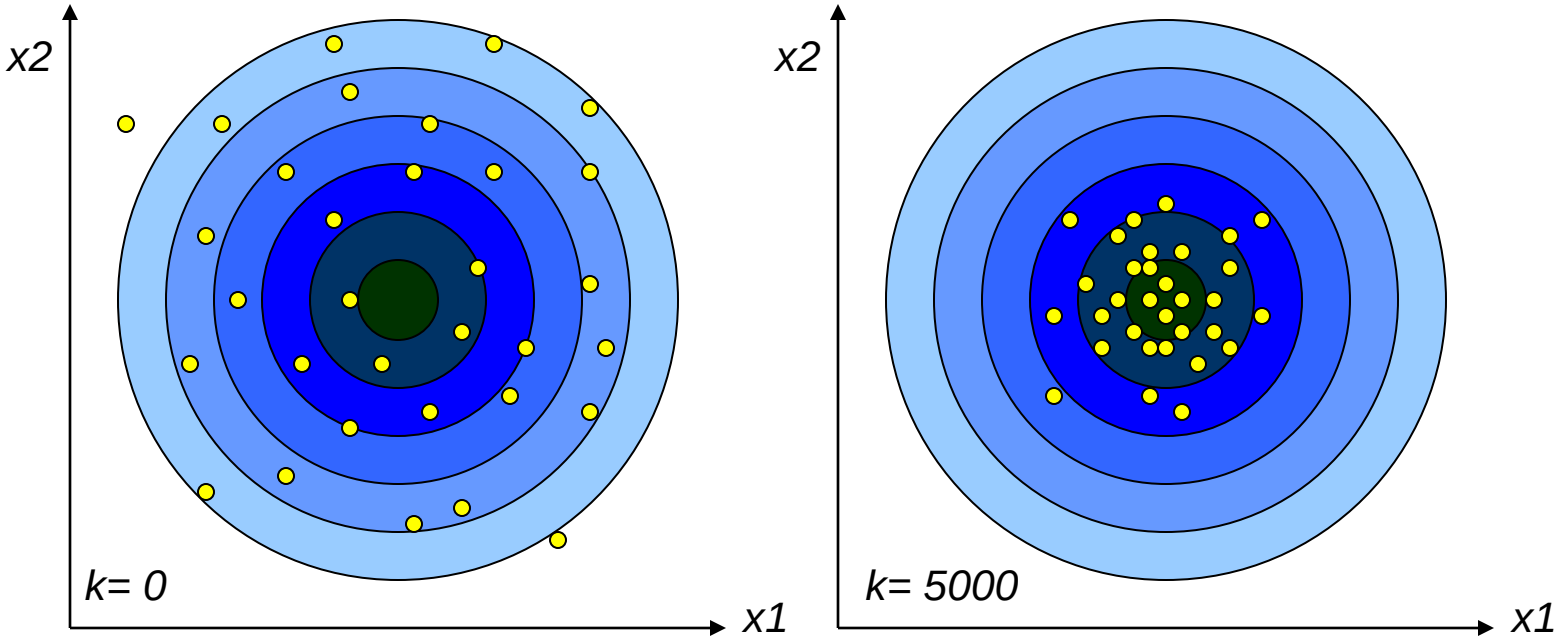
- A genetikus algoritmus **nevét** az élőlények utódaiban megjelenő kombinálódott szülői génekről, mint az utód tulajdonságainak hordozójáról kapta. A genetikus algoritmus alkalmazásának kulcskérdése a fenotípusról genotípusra való leképezés, azaz az egyed egyes tulajdonságait reprezentáló „gének”, azaz kromoszómarészletek kialakítása.

- Mivel egyszerre egy populációt, azaz egyedek egy csoportját kezeli, a globális optimum fellelésének megnő az esélye. Az állapottér feltárásában a következők játszanak szerepet:
  - A kezdeti populáció egyedeinek eloszlása az állapottéren
  - A keresztezések interpoláló hatása
  - A mutációk extrapoláló, a keresztezéssel kialakult belterjességből, lokalizáltságból kivezető hatása.

- A kezdeti mutációk az állapottér hatékony feltárását eredményezik. A globális optimum köré gyűlt késői populációkból viszont már csak gyorsan elhaló mutációk származnak, mivel azok életképessége a populáció többi tagjához képest alacsony.

A Genetikus algoritmus ..

- A **populáció fejlődése** az egyedek fejlődésén keresztül realizálódik. A szelekciónak köszönhető **látszólag spontán** fejlődés nem csak a véletlen eredménye: a jobb célfüggvény (fitness) értéket reprezentáló egyedek aránya a populációban egyre nő, míg a kevésbé életképeseké csökken. Az egyedek a **módszer memóriájaként** is felfoghatók: a jó tulajdonságú egyedek megtartásával az algoritmus **megtanulja** a jó célfüggvényértéket adó tulajdonságokat, a gyengébbeket eredményező tulajdonságok pedig **elfelejtődnek** az azokat hordozó egyedek kihalásával.



A Genetikus algoritmus ..

- Leképezés fenotípusról genotípusra

  Feladata minden egyes állapothoz egy karaktersorozat, azaz bitminta hozzárendelése.

  A leképezés kihat az alkalmazható operátorokra, végeredményben a GA hatékonyságára.

- Példák egyszerű esetekre:
  - Fenotípus: 12 zöld, vagy piros téglalapból álló sorozat.
    Genotípus: 12 bites bináris szám, piros:1, zöld: 0.
    Optimum: minden téglalap piros.

  - Fenotípus: m*n méretű fekete-fehér bittérkép alakjában adott kép.
    Genotípus: m*n elemű bináris szám, fekete: 0, fehér: 1.
    Optimum: két fekete átló a bittérképen.

- Megjegyzés: Az egyetlen **tulajdonság különböző értékeinek** tárolására alkalmazott több bites, egységként, génként kezelt kromoszómarészlet által felvehető értékeket **allél**-eknek nevezzük.

A Genetikus algoritmus ..

- **A szelekciós operátor**

  Feladata a szülőegyedek kiválasztása a keresztezés számára. A különféle szelekciós módszerekben közös, hogy a rátermettebb egyedeket **jelentősebb arányban** választják ki tulajdonságaik továbbörökítésére, de általában a gyengébb egyedek is kapnak egy **kis esélyt**. Ha a régi populáció legrátermettebb egyede mindig átkerül az új populációba, a szelekció **elitista**.

- A szelekcióban alkalmazott technikák

  1. Rátermettség-arányos választás
  2. Párok versenyeztetése
  3. Rangsorolás

1.

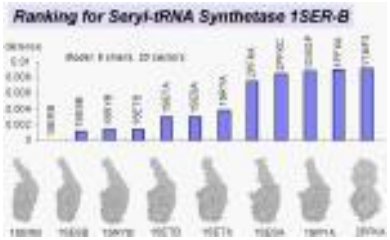
1. cbs.sportsline.com/u/includes/ olympics/2000/kids/boxing.htm

A Genetikus algoritmus ..

1. **Rátermettség-arányos választás**: az egyed kiválasztásának valószínűsége arányos a populáció rátermettségi átlagához viszonyított rátermettségével. A rátermettség gyakran azonos az egyedhez tartozó függvényértékkel, ritkábban azonban csak a szelekcióhoz alkalmazott egyedjellemző neve.

2. **Párok versenyeztetése**: véletlenszerűen kiválasztott két egyed közül a versenyt nyerő, azaz a nagyobb rátermettségi értékű lesz a kiválasztott. A módszer alkalmazható több, mint két versenyzős esetben is.

3. **Rangsorolás**: a keresztezés interpolációs hatásából eredő állapottérfeltárás hatásának lecsökkenését eredményezi, ha mindig ugyanazok az ősök szelektálódnak, örökítik tovább tulajdonságaikat. Ez a veszély fenyeget azoknál a módszereknél, amelyek a kiválasztást közvetlenül a rátermettség értékére alapozzák. Rangsorolás esetén a közvetlen **rátermettségi értékek helyett a rátermettségi sorrendre alapozott** szelekciót végzünk.



www.dbs.informatik.uni-muenchen.de/ Forschung/Bioinforma...

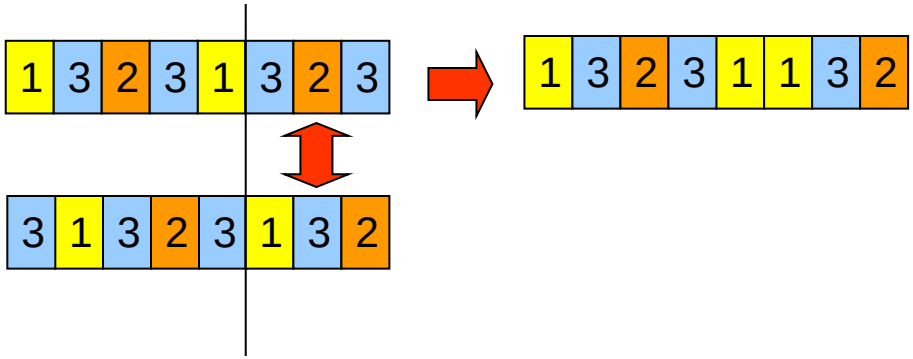
A Genetikus algoritmus ..

- **A keresztezés operátor**

  Szerepe utódok, azaz új egyedek előállítása.

1.

- Leggyakoribb módszerek
  - **Egypontos keresztezés**

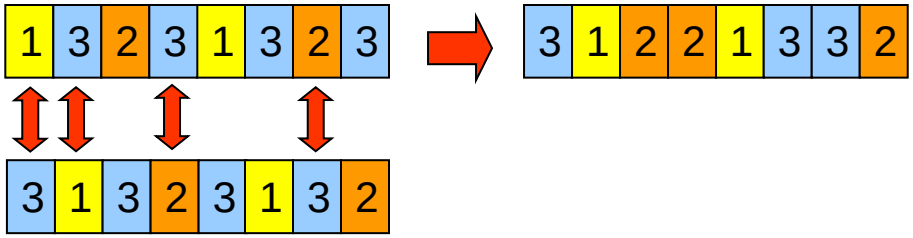


Véletlenszerű keresztezési pont választás.
(Többpontos keresztezés esetén több, mint két szülő is lehet.)

1. www.cnn.com/SPECIALS/ 2001/newamericans/

A Genetikus algoritmus ..

- **Egyenletes keresztezés**



1.

Ötven százalék eséllyel cserélődik minden egyes gén.

- Megjegyzések:
  - Nincsen általános módszer a keresztezés megválasztására, annak a feladathoz kell igazodnia.
  - Speciális keresztező operátort igényelhet az olyan eset, amikor összefüggés van a gének között (intelligens keresztezés).
  - A GA algoritmus érzékeny az operátorválasztásra.

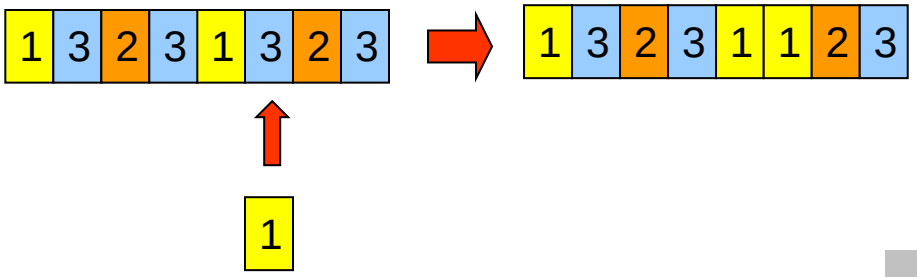
1. www.virginiasystems.com/

A Genetikus algoritmus ..

- **A mutáció operátor**

  Olyan új egyedet hoz létre, mely mentes a keresztezés belterjességétől, azaz az ősökétől merőben elütő tulajdonságokat hordozhat. A keresztezéstől kisebb gyakorisággal alkalmazott művelet.

- A módszer: az adott egyed kromoszómájának egy véletlenszerűen választott génjét egy másik génre cseréljük ki.



***Demo** található a*
http://www.taygete.demon.co.uk/java/ga/index.html
címen.

"http://www.grospixels.com/site/peve.html"

A Genetikus algoritmus lépései

1. Add meg az algoritmus fő paramétereit
2. Add meg a kezdeti populációt
3. Rendezd sorba az egyedeket csökkenő rátermettség szerint
4. Ha a megállási kritérium teljesül, add vissza az aktuális populációt
5. Egyébként válaszd ki az egyedeket a keresztezéshez
6. Hajtsd végre a keresztezéseket
7. Válaszd ki az egyedeket a mutációhoz
8. Végezd el a mutációkat
9. Rendezd csökkenő rátermettség szerint az eredeti és a keresztezéssel és mutációval kapott egyedekből álló halmazt
10. Állítsd elő az új populációt a leggyengébb rátermettségű egyedek ejtésével.
11. Ismételd a 4. ponttól.

Megjegyzések:

A kezdeti populáció feltöltése történhet véletlen értékekkel, de történhet a feladatra vonatkozó információk felhasználásával célirányosan is. Kevés ismeret felhasználása is nagy hatékonyságnövekedést eredményezhet. A kezdeti populáció számossága 50 -100.

A Genetikus algoritmus paraméterei

- A populáció számossága (pl. 16)
- Kromoszóma hossz (pl.12 bit)
- A szelekciós operátor
- A keresztezés operátor: a keresztezési ráta (új egyedek aránya, pl. 0.7), a keresztezési pontok száma, stb.
- A mutáció operátor: a mutációs ráta (pl. 0.001), a mutációs pontok száma, stb.

1.

A genetikus algoritmus alkalmazhatósága

Bár a genetikus algoritmusok nagyon igénytelenek az állapottérrel, az állapottéren értelmezett kiértékelő függvénnyel szemben, van néhány követelmény, amelynek eleget kell tenniük a megoldandó feladatoknak:

- A feladat megoldási lépéseinek ismerete nem szükséges, de bizonyosnak kell lennünk a megoldhatóságában.
- A megoldások összetevői között lehet részleges függőség, de nem lehet az összes összetevő függőségi viszonyban.

1. www.usaid.gov/bd/ Population_Health.html