**1. Milyen formában jelennek meg az általános keresőrendszer komponensei egy gráfkereső algoritmus esetén?**

1. keresőgráf (G) : a reprezentációs gráf eddig bejárt és eltárolt része
2. nyílt csúcsok halmaza (NYÍLT) : kiterjesztésre várakozó csúcsok, amelyeknek gyerekeit még nem vagy nem eléggé jól ismerjük
3. kiterjesztett csúcsok halmaza (ZÁRT ) : azok a csúcsok, amelyeknek a gyerekeit már előállítottuk
4. kiterjesztés (Γ) : egy csúcs összes gyermekének előállítása a hozzájuk vezető élekkel együtt
5. kiértékelő függvény (f: NYÍLT → R) : kiválasztja a megfelelő nyílt csúcsot kiterjesztésre

**2. Visszalépéses keresés első változatának algoritmusa**

Recursive procedure VL1(akt) return megoldás

if cél(akt) then return(nil) endif

for ∀új ∈ Γ(akt) loop

megoldás := VL1(új)

if megoldás != hiba then

return fűz((akt,új),megoldás) endif

endloop

return(hiba)

end

**3. Általános keresorendszer komponensei**

A produkciós rendszer (keresőrendszer) három összetevője:

1. Globális adatbázis

A globális adatbázis a feladat reprezentációs gráfjának a megoldása során előállítottrészét tartalmazza.

1. produkciós szabályok

A produkciós szabályok a globális adatbázison értelmezett operátorok. Egy produkciós szabály az alkalmazása során módosítja az adatbáis tartalmát.

1. vezérlési stratégia

A vezérlési stratégia egy elsőbbségi sprrendet állít fel az alkalmazható produkciós szabályok között.

**7. Probléma-dekompoziciós eljárás reprezentációja**

Probléma-dekompozició reprezentációjának nevezzük a probléma részekre bontásának szeléletére épülő feladat reprezentációt. Ez, a probléma illetve a részproblémák leírása mellett, a közöttük kapcsolatot teremtő úgynevezett redukciós operátorok megadását is jelenti. Probléma-dekompozició reprezentációjának szemmléltetésére is gráfokat használunk, a megfogalmazott feladatok egy ÉS/VAGY gráfon történő keresési problémaként értelmezhetők.

A reprezentációhoz meg kell adnunk:

1. a feladat részproblémáinak általános leírását,
2. az eredeti problémát,
3. az egyszerű problémákat, amelyekről könnyen eldönthető, hogy megoldhatók-e vagy sem, és a dekomponáló műveleteket:

D: probléma --> probléma+ és D(p)=<p1 , ... , pn>

**9. Milyen okai vannak a rezolúciós eljárás nemdeterminisztikusságának?**

A rezolúció nem-determinisztikus, ha egy lépésben

1. több klóz párt is rezolválhatunk
2. egy klóz párban több komplemens literál pár is lehet
3. ugyanannak a literálnak több előfordulása lehet

{P(x,f(a))P(x,f(y))Q(y), ¬P(z,f(a))¬Q(z), P(u,f(a))¬Q(a)}

**10. Hogyan készíthető gráfreprezentáció az előre és visszafelé haladó szabályalapú reprezentációhoz? (Tk. 34 oldal)**

1. Előre:

A kezdőállapotból elindulva fokozatosan építi fel a célállapotba vezető műveletsorozatot, mközben esetleg sok felesleges műveletet is kipróbál. Az ilyen irnyányba működő produkciós rendszereket előrrehaladó vagy adatvezérelt produkciós rendszereknek nevezzük.

1. Visszafele:

Amennyiben a feladat egyetlen és ismert célálapottal rendelkezik, akkor alkalmazhatunk úgynevezett visszafele haladó vagy célvezérelt produkciós rendszert is. A visszafelé haladó produkciós rendszer globális adatbázisát kezdetben a célállapot tartalmazza és a produkciós szabályok inverzeit alkalmazza egészen addig amíg a kezdőállapotot el nem éri.

**1. Adja meg az általános keresorendszer algoritmusát! (TK. 29. oldal)**

Procedure KR

ADAT := kezdeti érték

while !terminálási feltétel(ADAT) loop

SELECT SZ FROM alkalmazható szabályok

ADAT := SZ(ADAT)

endloop

end

**2. Milyen feltételek mellett terminál, és talál megoldást a visszalépéses keresés első változata? (TK. 41. oldal)**

A visszalépéses keresés akkor ér véget, ha talál egy célcsúcsba vezető utat, vagy már minden lehetséges utat végignézett. Ebből látszik, hogy a keresés befejeződhet sikeresen és sikertelenül. Véges körmentes irányított gráfokon a VL1 mindig terminál, és ha létezik megoldás, akkor talál egyet.

Visszalépés feltételei VL1 nél:  
zsákutca: az aktuális csúcsból (azaz az aktuális út végpontjából) nem vezet tovább él

zsákutca torkolat: az aktuális csúcsból kivezető utak nem vezettek célba

**3. Milyen feltételek mellett talál megoldást az általános gráfkereso algoritmus?**

A GK véges delta-gráfban mindig terminál.

Ha egy véges delta-gráfban létezik megoldás, akkor a GK megoldás megtalálásával terminál.

**4. Milyen okok akadályozzák egy probléma dekompoziciós reprezentációjának megfogalmazását?**

Dekomponáló műveleteket nagyon nehéz megtalálni.

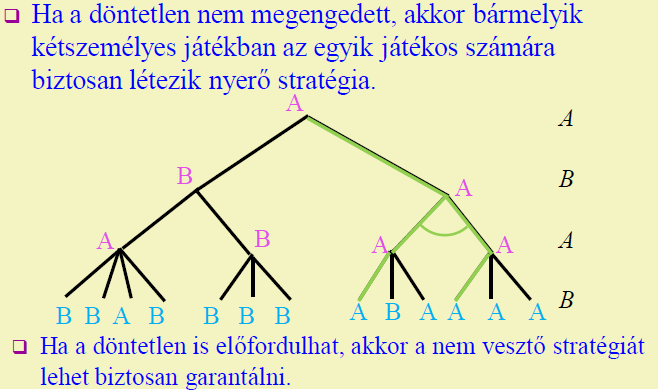
* Nem minden feladat dekomponálható.
* Könnyen kaphatunk hamis dekomponáló műveleteket.

Az egyszerű probléma felismerése sem mindig egyértelmű

A megoldás kiolvasása sem nyilvánvaló.

(6.2.dekomp.pdf - 1. oldal)

**5. Hogyan ábrázolható a nyerő stratégia a kétszemélyes játékok gráfreprezentációjában?**

****

(7.jatek.pdf - 3. oldal)

**7. Melyek klóz-formára hozás lépései az elsorendu predikátumkalkulusban?**

1. Kiküszöböljük az ↔ és a → műveleti jeleket.

2. Redukáljuk a negációk hatáskörét.

3. Standardizáljuk a változókat (kvantoronkénti átnevezés).

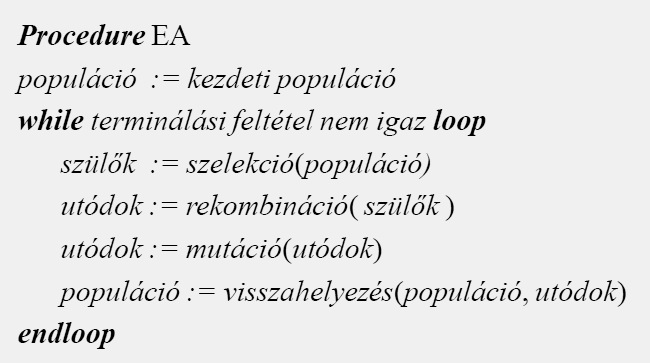
4. Egzisztenciális kvantorok Skolemizálása.

5. Univerzális kvantorok kiemelése a sorrendjük megtartásával.

6. A formula többi részét konjunktív normálformára alakítjuk.

7. Kialakítjuk a klózok halmazát (a kvantorokat és a konjunkciós műveleti jeleket elhagyjuk, a változókat klózonként egyedivé nevezzük át.)

(10.auto\_kovetkeztet.pdf - 2. oldal)

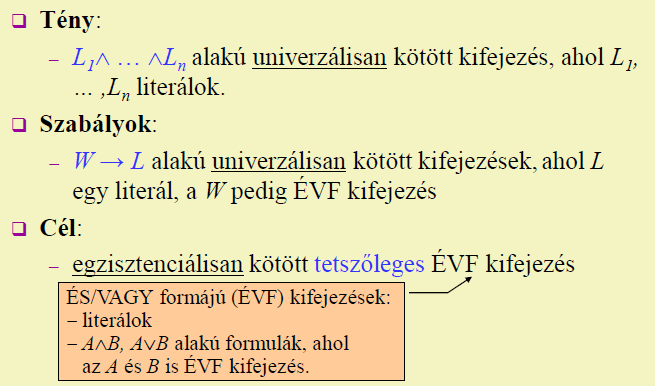
**6. írja fel az evolúciós algoritmus általános sémáját! **

**8. Adja meg a támasztó(támogató) halmaz stratégia és a lineáris input stratégia meghatározását. Teljesek-e ezek a stratégiák?**

1. Támogató-halmaz stratégia: egyik szülő az S kiinduló klózoknak egy adott T részhalmazából lett levezetve. (Teljes, ha S-T kielégíthető)
2. Lineáris input stratégia: egyik szülő az előző lépésben kapott klóz (kivéve az első lépést), a másik egy kiinduló klóz. (Teljes Horn klózok esetén)

(10.auto\_kovetkeztet.pdf - 4. oldal)

**9. Jellemezze a visszafele haladó szabályalapú reprezentáció formuláinak alakját!**

****

**4. hiperút fogalma**

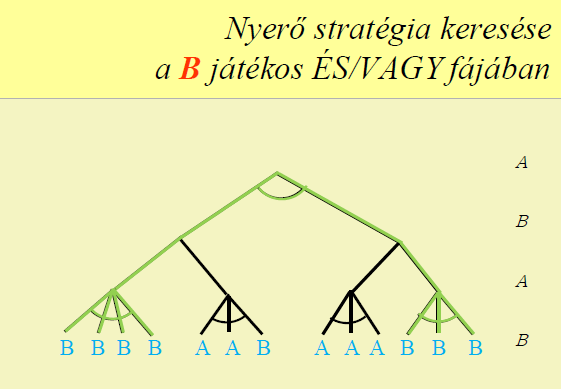
A hiper-út egy olyan véges részgráf, amelyben

* M csúcsaiból nem indul hiper-él,
* többi csúcsból egyetlen hiper-él indul,
* bármelyik csúcs elérhető az *n* csúcsból egy közönséges irányított úton.

Hiper-út hossza a benne levő élek száma.

(6.2.dekomp.pdf - 4. oldal)

**5. kétszemélyes játéknál gráfban nyerő stratégiát hogyan kell ábrázolni?**



(7.jatek.pdf - 2. oldal)

**7. lineáris input, ősre korlátozott rezolúciós stratégia**

Lineáris input stratégia: egyik szülő az előző lépésben kapott klóz (kivéve az első lépést), a másik egy kiinduló klóz. (Teljes Horn klózok esetén)

Ősre korlátozott stratégia: egy szülő az előző lépésben kapott klóz, a másik vagy egy kiinduló klóz vagy egy őse az első szülőnek. (Teljes)

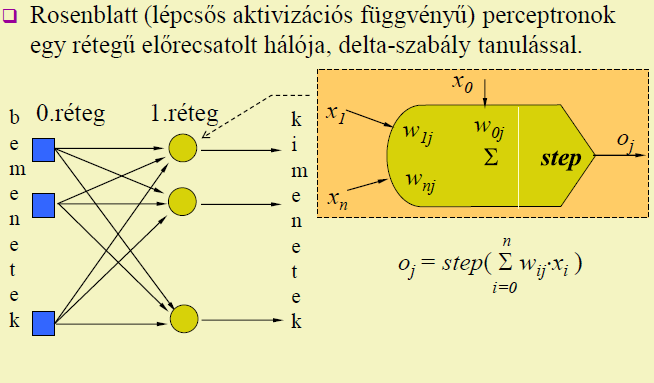
(10.auto\_kovetkeztet.pdf - 4. oldal)

**2. a visszalépéses keresés 2-es változata milyen gráfban terminál és mikor megoldással**

A visszalépéses algoritmus második változata az, amikor a visszalépés feltételei közül mindet beépítjük a kereső rendszerbe.

A VL2 delta-gráfban mindig terminál. Ha létezik a mélységi korlátnál nem hosszabb megoldás, akkor megtalál egy megoldást.

(4.visszalep.pdf - 4. oldal)

**6. perceptron modell - elemei, topológia, tanulási módszer **

Elemei:

Bemenetek, Rétegek, Kimenetek.

Tanulás:

Általánosított perceptronok esetén a neuron w súlyait tanuljuk meg, amely nemcsak a neuron számítási képletre hat ki, hanem a hálózat topológiájára is.

A minta a neuron egy súlyát a w:= w +Δw alapján módosítja.

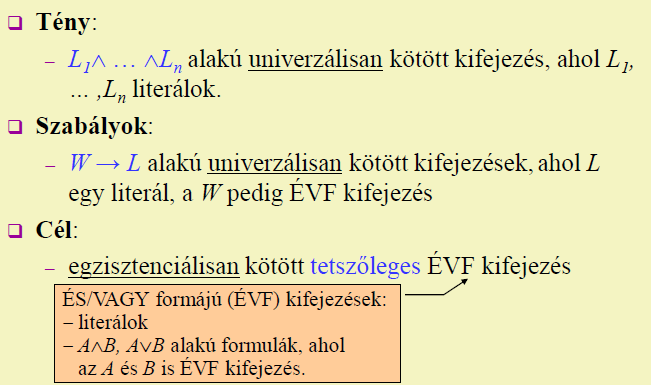
A Δw kiszámítása a neuron számított kimenetére támaszkodik, ehhez pedig a közvetlen megelőző neuronok számított kimeneteit is szükségesek.

* felügyelt tanulás esetén a neuron (összetett esetben a neuront tartalmazó hálózat) várt kimenetét is használjuk,
* felügyelet nélküli tanulás esetén a várt kimenetet nem kell ismerni.

Egy egyrétegű perceptron modell csak féltereket képes felismerni (osztályozni), de egy kétrétegű már ezek kombinációit, azaz konvex poliédereket is, egy három rétegű pedig tetszőleges poliédereket.

(8.neuron.pdf - 2.,3. oldal)

**8. visszafelé szabályalapú levezetésnek a formulái milyen alakúak?**

****

**1. egy probléma állapotterét hogyan valósítjuk meg gráfreprezentációval?**

Állapot-gráf (az állapottér-reprezentáció reprezentációs gráfja):

* állapot - csúcs
* művelet hatása - irányított él
* művelet költsége - élköltség
* kezdő állapot - startcsúcs
* célállapotok - célcsúcsok
* műveletsorozat hatása - irányított út

**3. milyen nem informált gráfkereséseket ismer. definiálja ezeket feltételzve, hogy ismerjük a gráfkeresési alg-t.**

Mélységi:

* f = -g, c(n,m) = 1,
* végtelen gráfokban csak mélységi korláttal garantál megoldást

Szélességi:

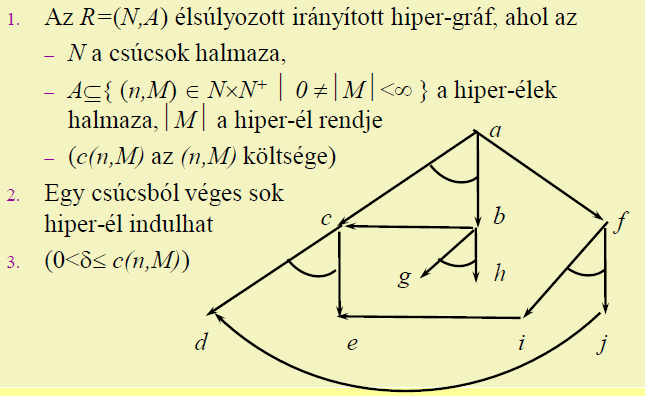
* f = g, c(n,m) = 1,
* optimális (legrövidebb) megoldást adja, ha van,
* egy csúcsot legfeljebb egyszer terjeszt ki

Egyenletes:

* f = g,
* optimális (legolcsóbb) megoldást adja, ha van,
* egy csúcsot legfeljebb egyszer terjeszt ki.

(5.grafker.pdf - 3. oldal)

**4. definiálja az ÉS\VAGY gráfot**

****

**5. ismertesse a minimax algoritmust**

1. A játékfának az adott állás csúcsából leágazó részfáját felépítjük néhány szintig.
2. A részfa leveleit kiértékeljük aszerint, hogy azok számunkra kedvező, vagy kedvezőtlen állások.
3. Az értékeket felfuttatjuk a fában:

* A saját (MAX) szintek csúcsaihoz azok gyermekeinek maximumát: szülő := max (gyerek1,..., gyerekk)
* Az ellenfél (MIN) csúcsaihoz azok gyermekeinek minimumát: szülő := min (gyerek1,..., gyerekk)

1. Soron következő lépésünk ahhoz az álláshoz vezet, ahonnan a gyökérhez felkerült a legnagyobb érték.

**6. milyen szempontokat kell figyelembe venni az evoluciós algoritmus kiválasztó operátorának (szelekció?) konstruálásakor?**

Szelekció célja: a rátermett egyedek kiválasztása úgy, hogy a rosszabbak kiválasztása is kapjon esélyt.

* Rátermettség arányos (rulett kerék algoritmus): minél jobb a rátermettségi függvényértéke egy elemnek, annál nagyobb valószínűséggel választja ki
* Rangsorolásos: rátermettség alapján sorba rendezett egyedek közül a kisebb sorszámúakat nagyobb valószínűséggel választja ki
* Versengő: véletlenül kiválasztott egyedcsoportok (pl. párok) legjobb egyedét választja ki.
* Csonkolásos v. selejtezős: a rátermettség szerint legjobb (adott küszöbérték feletti) valahány egyedből véletlenszerűen választ néhányat.

**8. ismertesse az egységklóz / támasztóhalmaz stratégiát. teljesek-e ezek?**

Egységklóz stratégia: az egyik szülő egységklóz (egy literálból álló klóz). (Teljes Horn klózok esetén)

Támogató-halmaz stratégia: egyik szülő az S kiinduló klózoknak egy adott T részhalmazából lett levezetve. (Teljes, ha S-T kielégíthető)

**1. Nevezze meg és jellemezze a kereső rendszerek fő részeit!**

A **globális munkaterület**en a reprezentációs gráf egy

részgráfja jelenik meg.

* vagy csak egy csúcs, esetleg annak közvetlen szomszédjai
* vagy a startcsúcsból kivezető egyik út kezdőszakasza
* vagy a startcsúcsból kivezető összes út kezdőszakasza

A **keresési szabályok** ezt a globális munkaterületen tárolt

részgráfot módosítják

* lecserélnek egy csúcsot valamelyik szomszédjára
* hozzávesznek egy tárolt út végéhez egy onnan kiinduló élt
* törölnek egy tárolt út végéről egy élt

A **vezérlési stratégia** a fenti szabályokból válogat.

(1.bevezetes.pdf - 3. oldal)

**2. Melyek a visszalépés feltételei a visszalépéses algoritmus legáltalánosabb változatában?**

* zsákutca: az aktuális csúcsból (azaz az aktuális út végpontjából) nem vezet tovább él
* zsákutca torkolat: az aktuális csúcsból kivezető utak nem vezettek célba

(4.visszalep.pdf - 1. oldal)

**4. Mit jelent kétszemélyes játékoknál a nyerő stratégia, és milyen állítást mondtunk ki ezzel kapcsolatban?**

Egy játékos nyerő stratégiája egy olyan elv, amelyet betartva az ellenfél minden lépésére tud olyan választ adni, hogy megnyerje a játékot.

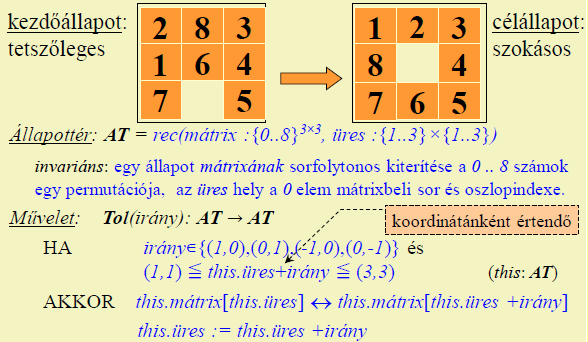
A nyerő stratégia NEM egyetlen győztes játszma, hanem olyan győztes játszmák összessége, amelyek közül az egyiket biztos végig tudja játszani az a játékos, aki rendelkezik a nyerő stratégiával.

A nyerő stratégia keresése egy ÉS/VAGY gráfbeli hiper-út keresési probléma.

**5. Mi történik (milyen lépések hajtódnak végre) egy evolúciós algoritmus egy iterációja során?**

* **Szelekció**: Kiválasztunk néhány (lehetőleg rátermett) egyedet szülőnek.
* **Rekombináció** (keresztezés): Szülőkből utódok készülnek úgy, hogy a szülők tulajdonságait örököljék az utódok.
* **Mutáció**: Az utódok tulajdonságait kismértékben módosítjuk.
* **Visszahelyezés**: Új populációt alakítunk ki az utódokból és a régi populációból.

**I. Adjon állapottér-reprezentációt a 8-as kirakó játékra! (Egy 3x3-as keretben 8 darab 1x1-es méretű 1-től 8-ig megszámozott lapocska található és egy üres hely...)**

****

**1, vezérlési stratégiák osztályzása**

Elsődleges:

* független a feladattól és annak modelljétől: nem merít sem a feladat ismereteiből, sem a modell sajátosságaiból.

1. nemmódosítható (lokális keresések, evolúciós algoritmus, rezolúció)
2. módosítható (visszalépéses keresések, gráfkeresések)

Másodlagos:

* nem függ a feladat ismereteitől, de épít a feladat modelljének általános elemeire.

Heurisztika:

* a feladattól származó, annak modelljében nem rögzített, de a megoldást segítő speciális ismeret.

**2, visszalépéses keresés előnyei/hátrányai**

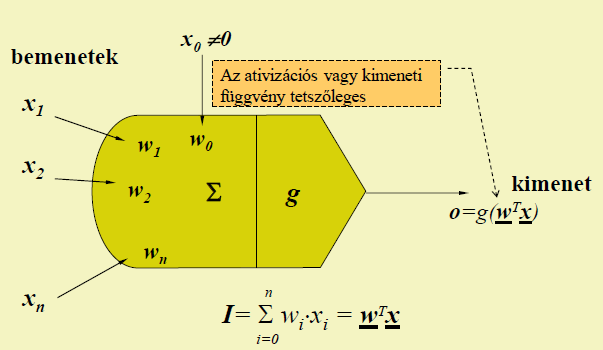
ELŐNYÖK:

* mindig terminál, talál megoldást
* könnyen implementálható
* kicsi memória igény

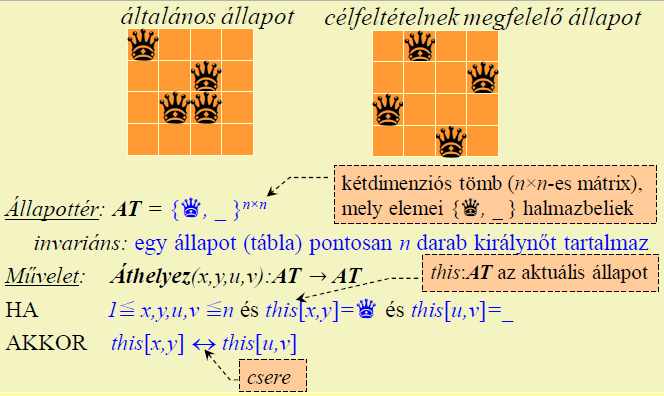
HÁTRÁNYOK:

* nem ad optimális megoldást. (iterációba szervezhető)
* kezdetben hozott rossz döntést csak sok visszalépés korrigál (visszaugrásos keresés)
* egy zsákutca részt többször is bejárhat a keresés

**5, hogy néz ki egy perceptron**



**F1 n-királynő probléma reprezentációja**



**2) visszalépéses keresés globális munkaterülete, keresési szabálya, vezérlési stratégiája.**

Globális munkaterülete:

* egy út a startcsúcsból az aktuális csúcsba (ezen kívül az útról leágazó még ki nem próbált élek)
  + Kezdetben a startcsúcsot tartalmazó nulla hosszúságú út
  + Terminálás célcsúccsal vagy startcsúcsból való visszalépéssel

Keresés szabályai:

* a nyilvántartott út végéhez egy új (ki nem próbált) él hozzáfűzése, vagy a legutolsó él törlése (visszalépés szabálya)

Vezérlés stratégiája a visszalépés szabályát csak a legvégső esetben alkalmazza.

**1) Jellemeze a problémás és a tabu keresés által adott megoldást.**

Hegymászó algoritmus a lokális optimum hely körül vagy ekvidisztans felületeten (azonos értékű szomszédos csúcsok között) található körön végtelen működésbe eshet.

Ennek a hátránynak a kiküszöbölésére megoldást nyújt a tabu keresés, mely aktuális csúcson kívül (akt) nyilvántartja az utolsó néhány érintett csúcsot (tabu halmaz), és a legjobb csúcsot (opt).

Az algoritmus minden lépésben az aktuális csúcs egy gyerekére lép, kivéve a Tabu halmazban lévőket, ha akt jobb, mint az opt, akkor az opt az akt lesz, és frissíti a sorszerkezetű tabu halmazt akt-al. Terminálási feltétel, ha az opt a célcsúcs, illetve ha opt sokáig nem változik.

Algoritmus:

*if f(akt) jobb, mint f(opt) then opt:=akt*

*endloop*

A törzs első sora helyettesíthető a következővel bővebben:

Az algoritmus előnye, hogy a tabu méreténél rövidebb köröket észleli.

Hátránya, hogy a Tabu halmaz méretét kísérletezéssel kell belőni, és zsákutcába futva beragad.

**3) Milyen eredményeket mondtunk ki az általános gráfkereső algoritmusról?**

Bebizonyítható:

* A GK ?-gráfban a működése során egy csúcsot legfeljebb véges sokszor terjeszt ki. (tehát a körökre nem érzékeny)
* A GK véges ?-gráfban mindig terminál.
* Ha egy véges ?-gráfban létezik megoldás, akkor a GK megoldás megtalálásával terminál.

(5.grafker.pdf - 2. oldal)

**1. Milyen hatással van a heurisztika általában a kereső rendszerek működésére?**

A heurisztika olyan, a feladathoz kapcsolódó ötlet, amelyet közvetlenül építünk be egy algoritmusba azért, hogy annak eredményessége és hatékonysága javuljon, habár erre általában semmiféle garanciát sem ad.

**2. Írja le a hegymászó algoritmust**!

akt := start

while akt T loop

akt := (

endloop

while törzse a következővel helyettesíthető bővebben:

**3. Mit tesz az általános gráfkereső algoritmus akkor, amikor egy már korábban felfedezett csúcshoz talál minden addiginál olcsóbb utat?**

Ekkor annak a korábbi csúcsnak a szülője az aktuális kiterjesztés gyökere lesz (tudom, nem valami érthető), az útköltség pedig a szülő és a startcsúcs közti útköltség + a szülő és az aktuális(korábbi) csúcs útköltsége.

Formálisan:

Ha m régi csúcs, amelyhez olcsóbb utat találtunk,

azaz mᕮG és g(n)+c(n,m)<g(m) akkor

?(m) := n, g(m) := g(n)+c(n,m).

**4. Mit tartalmaz egy probléma dekompozíciós reprezentációja?**

A reprezentációhoz meg kell adnunk:

* a feladat részproblémáinak általános leírását,
* az eredeti problémát,
* az egyszerű problémákat, amelyekről könnyen eldönthető, hogy megoldhatók-e vagy sem, és
* a dekomponáló műveleteket:
  + D: probléma → probléma+ és D(p)=<p1, … , pn>

**Mikor nevezünk egy gráfkereső algoritmust A algoritmusnak? Milyen állítást mondhatunk ki vele kapcsolatban?**

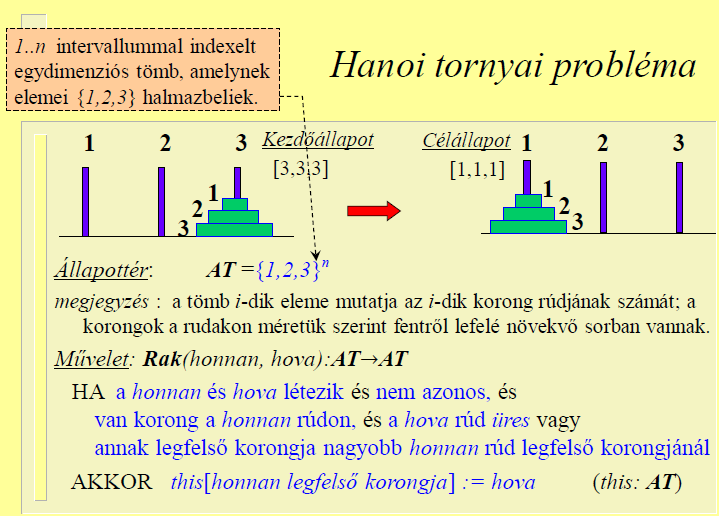
f = g+h és 0 ≦ h

megoldást ad, ha van megoldás (még végtelen gráfokban is)

**Mit értünk egy hiperút bejárásán?**

Egy irányított hiper-út bejárása is egy sorozat, de ez csúcsok sorozatainak sorozata, és ez nem egyértelmű.

**I. Adjon állapottér reprezentációt a Hanoi-tornyai problémára! (stb. stb.)**

****

**Milyen célból alkalmazhatunk heurisztikákat a visszalépéses keresésben?Adjon rájuk egy-egy példát!**

* Sorrendi szabály: sorrendet ad az aktuális út végpontjából kivezető élek (utak) vizsgálatára
* Vágó szabály: megjelöli azokat az aktuális út végpontjából kivezető éleket (utakat), amelyeket nem érdemes megvizsgálni

Sorrendi heurisztikák az n-királynő problémára:

A mezőkhöz rendelt értékek alapján rendezzük sorba a mezőket.

Diagonális: a mezőn áthaladó hosszabb átló hossza.

Vágás: Forward Checking:

FC algoritmus:

Nemcsak akkor lép vissza, ha az aktuális sorban nincs szabad (ütésben nem álló üres) mező, hanem ha a hátralevő üres sorok valamelyikében nincs már szabad mező, azaz Ǝr∈[k+1.. n]: Szabadr = ∅, akkor is visszalép.

(4.visszalep.pdf - 1,2. oldal)

**2. Mit nevezünk egy probléma állapottér-reprezentációjának?**

* Állapottér: a feladat fókuszában álló adat-együttes (objektum) lehetséges értékeinek (állapotainak) halmaza
  + egy állapot többnyire egy összetett szerkezetű érték, amelynek gyakran meg kell felelnie egy invariáns állításnak is.
* Műveletek: állapotból állapotba vezetnek
  + megadásukhoz: előfeltétel és hatás-leírás
  + invariáns tulajdonságot tartó leképezés
* Kezdő állapot(ok) vagy azokat leíró kezdőfeltétel
* Célállapot(ok) vagy célfeltétel

**4. Minimax algoritmus módosításai, melyek ezek, milyen szempontból javítanak a minimax működésén?**

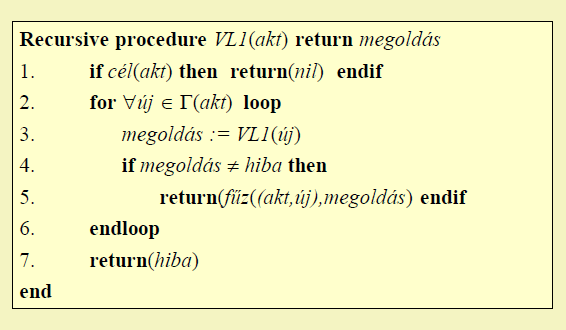
Negamax, alfa-béta

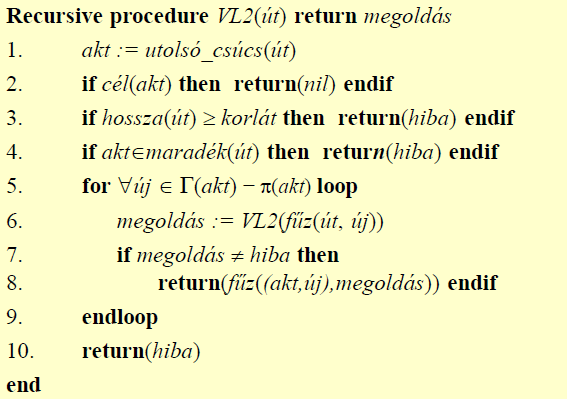
**5. Mit jelent a lineárisan szeparálhatóság fogalma, hogyan kapcsolódik a mesterséges neuronhálózatok témaköréhez?**

Egyetlen perceptron csak olyan bonyolultságú feladatot képes megoldani, ahol az azonos eredményt adó bemenetek két csoportját egyetlen hipersíkkal el lehet választani, azaz lineárisan szeparálhatók.

Azért nincs XOR műveletet megvalósító perceptron, mert ez a probléma nem lineárisan szeparálható.

**1. visszalépéses keresés: első/második változat**

****



**3. és/vagy gráfot mikor, miért használjuk**

A közönséges irányított gráfok útkereső algoritmusait szeretnénk adaptálni az ÉS/VAGY gráfbeli keresési problémákhoz: megoldás gráf kereséséhez.

A megoldás gráf egy startcsúcsból kiinduló hiper-gráf. A startcsúcsból kiinduló hiper-gráfokat megadhatjuk a bejárásaikkal, ennél fogva ábrázolhatóak közönséges irányított utakkal.

Ez lehetővé teszi, hogy minden ÉS/VAGY gráfnak megfeleltethessünk egy olyan közönséges ?-gráfot, amelynek megoldási útjai az ÉS/VAGY gráfbeli megoldás gráfok bejárásai, és ezeket közönséges útkereső algoritmusokkal megkereshetjük.

**(Nem biztos, hogy ez jó válasz)**

(6.2.dekomp.pdf - 4. oldal)

**4. evolúciós alg: visszahelyezés**

A visszahelyezés a populációnak az utódokkal történő frissítése: Kiválasztja (újabb szelekció) a populációnak a lecserélendő egyedeit, és azok helyére a kiválasztott utódokat teszi.

utódképzési ráta (u) = utódok száma / populáció száma

visszahelyezési ráta (v) = lecserélendő egyedek száma / populáció száma

* ha u=v, akkor feltétlen cseréről van szó
* ha u<v , akkor egy utód több példányban is bekerülhet
* ha u>v, akkor az utódok közül szelektál

**1. ket szemelyes jatekoknal tetel a nyero strategiarol**

Egy játékos nyerő stratégiája egy olyan elv, amelyet betartva az ellenfél minden lépésére tud olyan választ adni, hogy megnyerje a játékot.

**4. evolucios algoritmus lepesei**

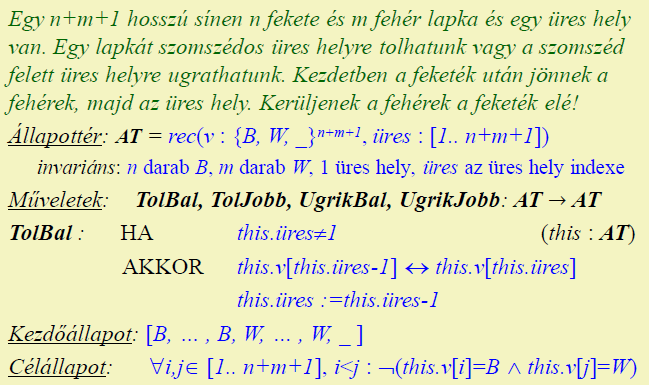
1. Szelekció: Kiválasztunk néhány (lehetőleg rátermett) egyedet szülőnek.
2. Rekombináció (keresztezés): Szülőkből utódok készülnek úgy, hogy a szülők tulajdonságait örököljék az utódok.
3. Mutáció: Az utódok tulajdonságait kismértékben módosítjuk.
4. Visszahelyezés: Új populációt alakítunk ki az utódokból és a régi populációból.

**2. Mikor nevezünk egy gráfkereső algoritmust szélességi keresésnek? Milyen állítást mondhatunk ki vele kapcsolatban?**

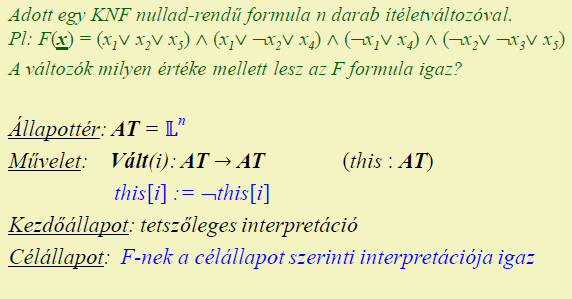
Optimális (legrövidebb) megoldást adja, ha van.

Egy csúcsot legfeljebb egyszer terjeszt ki

**I. Fekete-fehér kirakó állapottér-reprezentációja. Problématér és állapotér méretének becslése.**

****

**I. állapottér rep. SAT problémára**

****

**1. Milyen eredményre képes a visszalépéses keresés első, illetve második változata?**

Első:

* Véges körmentes irányított gráfokon (itt nem kell ?-gráf) a VL1 mindig terminál, és ha létezik megoldás, akkor talál egyet.

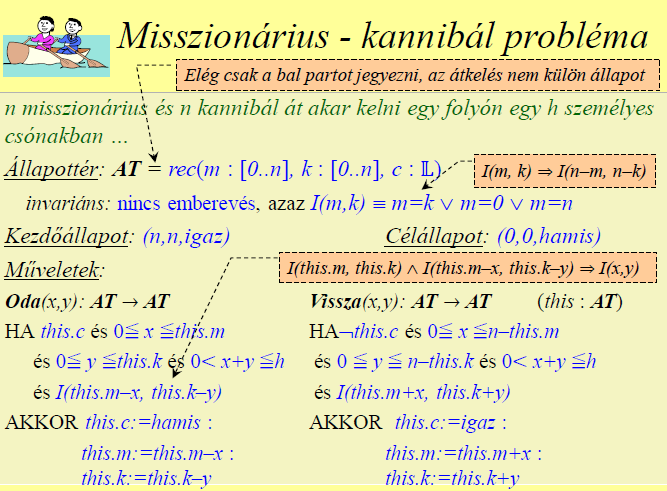
Második:

* A VL2 delta-gráfban mindig terminál. Ha létezik a mélységi korlátnál nem hosszabb megoldás, akkor megtalál egy megoldást. UI: véges sok adott korlátnál rövidebb startból induló út van.

**5. A rezolúció fő részei**

* globális munkaterület: aktuális klózhalmaz
* kiindulási érték: az „axiómák ⇒ célállítás” klózai
* terminálási feltétel:
  + sikeres: üres klóz
  + sikertelen: nincs újabb rezolvens klóz
* kereső szabály: rezolvens képzés
* vezérlési stratégia: elég a nem-módosítható
* heurisztika: jó lenne a hatékonyság miatt, de sajnos nincs

**I. állapottér repr. a misszionárius - kannibál problémára**



**3. Mikor nevezünk egy gráf kereső algoritmust A^c algoritmusnak, és melyek a legfőbb tulajdonságai**

f = g+h és 0 ≦ h és

h ≦ h\* és

h(n)-h(m) ≦ c(n,m)

* optimális megoldást ad, ha van
* ugyanazt a csúcsot nem terjeszti ki kétszer

1. **mi az elsőrendű stratégia, csoportjai példák**

Ha ez az elsődleges vezérlési stratégiákra vonatkozik, akkor:

független a feladattól és annak modelljétől: nem merít sem a feladat ismereteiből, sem a modell sajátosságaiból.

* módosítható: visszalépéses keresések, gráfkeresések
* nem módosítható: lokális keresések, evolúciós algoritmus, rezolúció

**1, Heurisztika, mire jó, KR-ben hol és mire használjuk?**

A heurisztika olyan, a feladathoz kapcsolódó ötlet, amelyet közvetlenül építünk be egy algoritmusba azért, hogy annak eredményessége és hatékonysága javuljon, habár erre általában semmiféle garanciát sem ad.

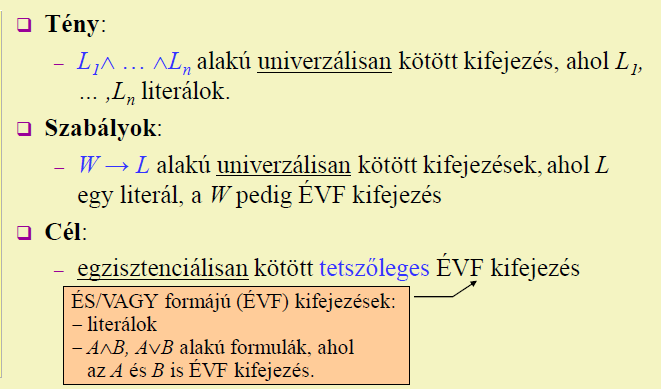
Erős heurisztika nélkül nincs sok esély a cél megtalálására.

* Jó heurisztikára épített kiértékelő függvénnyel elkerülhetőek a zsákutcák.

**Hiányos**

(3. lokalker.pdf - 4. oldal)

**6, Visszafelé haladó szabályalapú logikai reprezentáció, formai megszorításai.**

****

**III, Adott ÉS/VAGY gráfból közönséges irányított gráfot készíteni.**

1. Előre:

ÉS/VAGY formájú (ÉVF) kifejezések:

* literálok
* AB, AB alakú formulák, ahol az A és B is ÉVF kifejezés.

Tény: univerzálisan kötött tetszőleges ÉVF kifejezés

Szabályok: L → W alakú univerzálisan kötött kifejezések, ahol L egy literál, a W pedig ÉVF kifejezés

Cél: L1 V … V Ln alakú egzisztenciálisan kötött kifejezés, ahol L1, ... ,Ln literálok.

1. Visszafele:

Tény: L1 ᐱ … ᐱ Ln alakú univerzálisan kötött kifejezés, ahol L1, ... ,Ln literálok.

Szabályok: W → L alakú univerzálisan kötött kifejezések, ahol L egy literál, a W pedig ÉVF kifejezés

Cél: egzisztenciálisan kötött tetszőleges ÉVF kifejezés

ÉS/VAGY formájú (ÉVF) kifejezések:

* literálok
* AB, AB alakú formulák, ahol az A és B is ÉVF kifejezés.

1. **milyen modellezési megoldásokat vettünk a félév során**

megoldások:

állapotreprezentáció

keresési rendszerek

redukció

dekompozíció

modellezés mesterséges neurális hálókkal (perceptron, MLP és Hopfield)

1. **szelekció és arra 3 példa hogy mire használtuk - vmi ilyesmi**

Szelekció: Kiválasztunk néhány (lehetőleg rátermett) egyedet szülőnek.

**+példák**

(9.evolucio.pdf - 1. oldal)

1. **visszalépéses keresés leírása és heurisztikái**

A visszalépéses keresés egy olyan KR

* globális munkaterülete:
  + egy út a startcsúcsból az aktuális csúcsba (ezen kívül az útról leágazó még ki nem próbált élek)
  + Kezdetben a startcsúcsot tartalmazó nulla hosszúságú út
  + terminálás célcsúccsal vagy startcsúcsból való visszalépéssel
* keresés szabályai:
  + a nyilvántartott út végéhez egy új (ki nem próbált) él hozzáfűzése, vagy a legutolsó él törlése (visszalépés szabálya)
* vezérlés stratégiája a visszalépés szabályát csak a legvégső esetben alkalmazza

Heurisztikák:

* sorrendi szabály: sorrendet ad az aktuális út végpontjából kivezető élek (utak) vizsgálatára
* vágó szabály: megjelöli azokat az aktuális út végpontjából kivezető éleket (utakat), amelyeket nem érdemes megvizsgálni

1. **rezolúció másodlagos stratégiái**

A rezolúciós stratégia a rezolúció alapalgoritmusát kiegészítő olyan előírás, amely

* korlátozza egy adott pillanatban előállítható rezolvensek körét, vagy
* szabályozza a rezolvens képzés sorrendjét.

A rezolúciós stratégiák a rezolúciónak, mint kereső rendszernek, a másodlagos vezérlési stratégiái: csak klóz alapú reprezentáció esetén értelmezhetőek. A korlátozásnál megsérülhet a teljesség: azaz nem vezethető le minden esetben az üres klóz.

1. **és/vagy fát mire használtuk a félév során**

A közönséges irányított gráfok útkereső algoritmusait szeretnénk adaptálni az ÉS/VAGY gráfbeli keresési problémákhoz: megoldás gráf kereséséhez.

A megoldás gráf egy startcsúcsból kiinduló hiper-gráf. A startcsúcsból kiinduló hiper-gráfokat megadhatjuk a bejárásaikkal, ennél fogva ábrázolhatóak közönséges irányított utakkal.

Ez lehetővé teszi, hogy minden ÉS/VAGY gráfnak megfeleltethessünk egy olyan közönséges -gráfot, amelynek megoldási útjai az ÉS/VAGY gráfbeli megoldás gráfok bejárásai, és ezeket közönséges útkereső algoritmusokkal megkereshetjük.

**Nem tudom, hogy jó válasz-e**

(6.2.dekomp.pdf - 4. oldal)

**1) Sorolja fel a hegymászó algoritmus hátrányait, és írja le, milyen módon lehet ezeken javítani! (GT 3. dia)**

hátrányok:

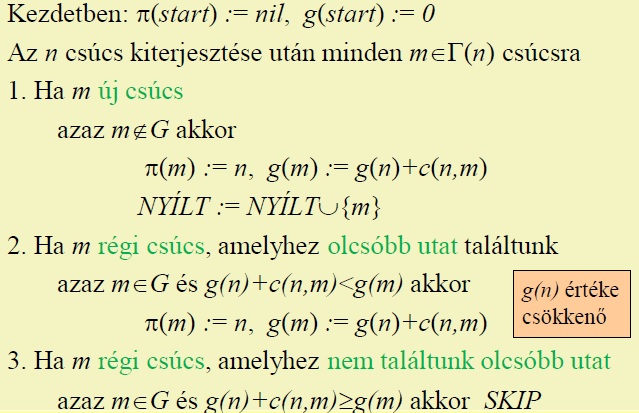
1. Csak erős heurisztika esetén lesz sikeres (zsákutcába kerulhet)
2. (azonos értékű szomszédos csúcsok között) található körön, végtelen működésbe eshet.

javitasok:

1. zsákutca elkerüléséhez szimultán hűtés
2. deadlock kiszűrése tabu halmazzal

**2. Mikor veszítheti el a gráfkeresés során a feszítőfa a korrektséget? Hogyan korrigálható ez, és mi melyik megoldást választottuk?**

Abban az esetben vesziti el a fa a korrektseget, ha egy adott csucsot mar az algoritmus futasa folyaman egyszer kiterjesztettunk, majd ezt kovetoen ujra kiterjesztesre kerul.

(Csokkeno kiterjesztes használata.)

* **Mit tartalmaz egy probléma dekompozíciós reprezentációja?**
  + A feladat részproblémáinak általános leírása
  + A kiinduló probléma
  + Az egyszerű problémák, amelyikről könnyen eldönthető, hogy megoldhatóak-e, vagy sem
  + A dekomponáló műveleteket:  
    D : probléma -> probléma+  
    D(p) = <p1, ..., pn>
* **Adjon példát legalább három rekombinációs operátorra!**
  + Egypontos keresztezés
  + Többpontos keresztezés
  + Egyenletes keresztezés
  + Parciálisan illesztett keresztezés
  + Ciklikus keresztezés
* **Visszalépéses keresés munkaterülete, keresési szabályai, vezérlési stratégia**
  + Munkaterülete: egy út az aktuális csúcsba a startcsúcsból + a leágazó, még ki nem próbált élek
  + Keresés szabályai: a nyilvántartott úthoz egy új él hozzáfűzése, vagy az utoló él törlése (visszalépés)
  + Vezérlési stratégia: A visszalépés szabályát csak legvégső esetben alkalmazza
  + A visszalépés feltételei:
    - Zsákutca
    - Zsákutca torkolat
    - Kör
    - Mélységi korlát
* **Általános gráf ker. eredményei**
  + δ-gráfokban egy csúcsot véges sokszor terjeszt ki
  + *véges* δ-gráfban terminál
  + *véges* δ-gráfban, ha van megoldás, megtalálja és terminál
* **Mikor nevezünk egy gráfkereső algoritmust szélességi keresésnek? Milyen állítást mondhatunk ki vele kapcsolatban?**

| **Elnevezés** | **Definíció** | **Eredmények** |
| --- | --- | --- |
| Mélységi | f=-g, c(n,m)=1 | Végtelen gráfokban mélységi korláttal megoldást garantál |
| Szélességi | f=g, c(n,m)=1 | Végtelen gráfokban a legrövidebb megoldást adja, egy csúcsot csak egyszer terjeszt ki. |
| Egyenletes | f=g | Végtelen gráfokban a legolcsóbb megoldást adja, egy csúcsot legfeljebb egyszer terjeszt ki. |

* **Sorolja fel, milyen módosításait ismerte meg a minimax algoritmusnak, és írja melléjük, hogy ezek milyen szempontból javítanak annak működésén?**
  + Átlagoló kiértékelés
    - A kiértékelő függvény esetleges tévedéseit simítja ki
  + Váltakozó mélységű kiértékelés
    - A kiértékelő függvény minden ágon reális értéket mutat
  + Szelektív kiértékelés
    - A memóriaigényt csökkenti (csak a lényeges lépéseket értékeli)

Változatok:

* + *Negamax algoritmus:* könnyebb implementálni
  + *Alfa-béta algoritmus:* kisebb a memóriaigénye, mert csak 1 utat tárol és a vágások miatt jobb a futási ideje is
* **Hogyan történik az evolúciós algoritmusokban a visszahelyezés?**  
  A populáció az utódokkal történő frissítése: kiválasztja a populáció lecserélendő egyedeit, és azok helyére az utódokat teszi.  
  *utódképzési ráta(u) = utódok száma/populáció száma*  
  *visszahelyezési ráta(v) = lecserélendő egyedek száma/populáció száma*
  + Ha *u=v:* feltétlenül csere
  + Ha *u<v:* egy utód több példányban is bekerülhet
  + Ha *u>v:* az utódok közül szelektál
* **A rezolúció módszere egy speciális kereső rendszer. Írja le, hogy ebben az esetben mik lesznek a keresőrendszer fő részei!**
  + Globális munkaterület -> aktuális klózhalmaz
  + Kiindulási érték -> *"axioómák -> célállítás"* klózai
  + Terminálási feltétel -> *"sikeres"* – üres klóz; *"sikertelen"* – nincs újabb rezolvens klóz
  + Kereső szabály -> rezolvens képzés
  + Vezérlési stratégia -> nem módosítható
  + Heurisztika -> nincs
* **Mikor nevezünk egy gráf kereső algoritmust Ac algoritmusnak, és melyek a legfőbb tulajdonságai**

| **Elnevezés** | **Definíció** | **Eredmények** |
| --- | --- | --- |
| Előre tekintő gráfkeresés | f=h | Nincs említésre méltó tulajdonsága |
| A algoritmus | f=g+h, h≤h\* | Megoldást ad, ha van, még végtelen gráfban is |
| A\* algoritmus | f=g+h, h≤h\*, h≥0 | Optimális megoldást ad, ha van, még végtelen gráfokban is |
| Ac algoritmus | f=g+h, h≤h\*, h≥0, h(n)-h(m)≤c(n,m) | Optimális megoldást ad, ha van és ugyanazt a csúcsot nem terjeszti ki kétszer |

* **Felügyelt és felügyelet nélküli tanulás def + példák**  
  Egy neuron esetében a *Δw* kiszámítása az éppen vizsgált minta bemeneti értékeire és a neuron által kiszámított kimenetre támaszkodik.
  + *Felügyelt tanulás* esetén felhasználjuk a neuronnal a vizsgált minta alapján elvárt kimeneti értékét is
  + *Felügyelet nélküli tanulás* esetén a várt kimenetre nincs szükség

**Példák:**

| **Felügyelt tanulás (Delta szabály)** | **Felügyelet nélküli tanulás (Hebb szabály)** |
| --- | --- |
| Delta szabály | Hebb szabály |

***Jelmagyarázat:*** *η: tanulási együttható, a tanulás sebességét befolyásolja.  
oi: az i-edik neuron számított kimenete, és egyben a j-edik neuron i-edik bemenete is. ||| wij: a j-edik neuron bemenetének súlya  
oj: a j-edik neuron számított kimenete ||| tj: a j-edik neuron várt kimenete*

* **Mi a nem-determinisztikus rezolúció?**  
  A rezolúció nem determinisztikus. Egy lépésben:
  + Egyszerre több rezolváló klóz pár is lehet
  + Egy klóz párban több komplemens literál is lehet
  + Ugyanannak a literálnak több előfordulása is lehet
* **Hegymászó algoritmus és tabu keresés közti különbség**  
  A hegymászó algoritmus lokális optimum hely körül, illetve ekvidisztans felületen lévő körön végtelen működésbe eshet, ezt hivatott kiküszöbölni a tabu keresés a memória növelése által.  
  A tabu keresés nyilvántart egy tabu halmazt és az optimális csúcsot.  
  Mindig az aktuális csúcs legjobb, de nem a tabu halmazban lévő gyerekére lép. Ha az aktuális csúcs jobb, mint a jelenlegi optimális csúcs, akkor kicseréljük az optimális csúcsot az aktuális csúcsra. Az algoritmus minden lépésben frissíti a sorszerkezetű tabu halmazt az aktuális csúccsal.  
  Az algoritmus terminál, ha az optimális csúcs célcsúcs, vagy ha az optimális csúcs már régóta nem változott.