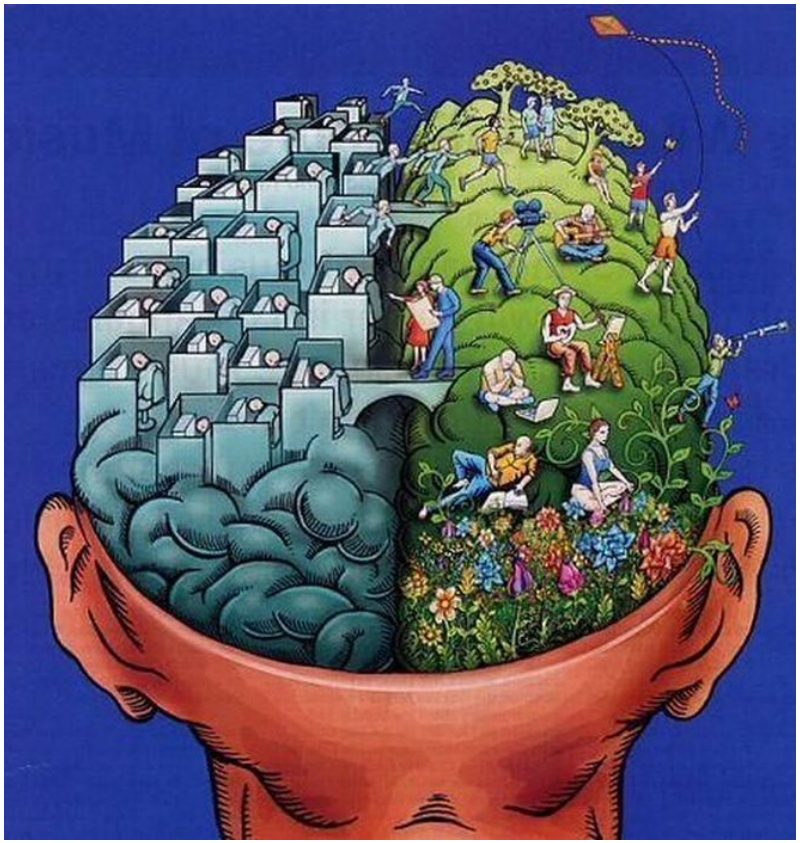
Mesterséges Intelligencia MI

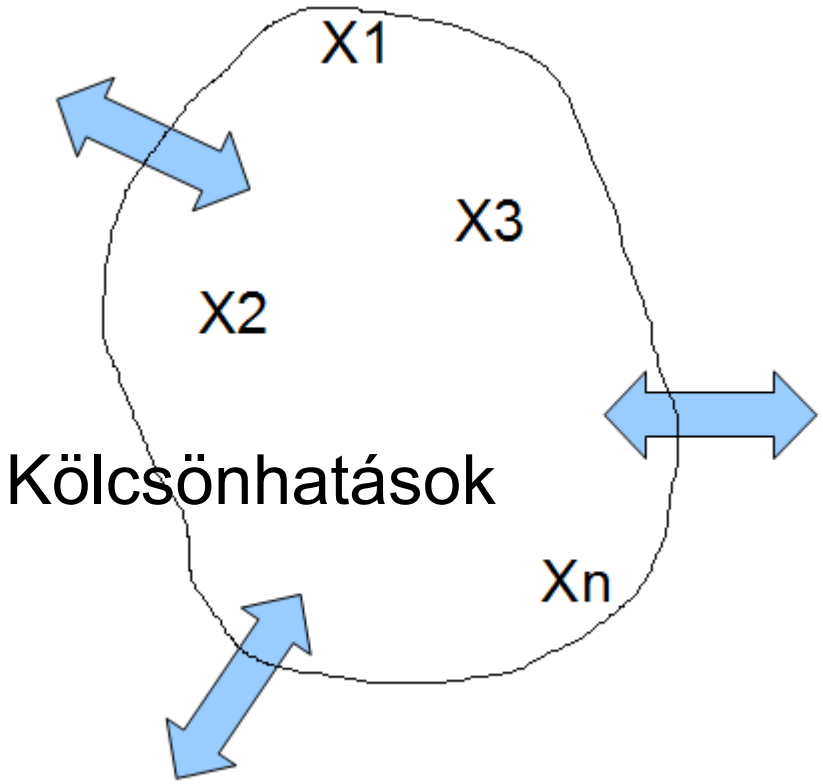
Korlát/kényszer-kielégítési problémák

Dobrowiecki Tadeusz
Eredics Péter, és mások



BME I.E. 437, 463-28-99
dobrowiecki@mit.bme.hu,
http://www.mit.bme.hu/general/staff/tade

Valós objektumok (feladatok), fizikai törvények, ... probléma-absztrakciója



Kölcsönhatások

Leíró változók (állapot)

S = (x1=..., x2=..., …, Xn=...)

Változók között relációk:
kényszerek/korlátok

pl. X2 = X1 + 1

NincsTámadás(VPoz1,VPoz2)
A OR B = 1
Vége(MunkaSz1) < Eleje(MunkaSz2)
PV = nRT
… ...

Korlátkielégítési problémák
Constraint Satisfaction Problems (CSPs)

Megszokott keresési feladatoknál:

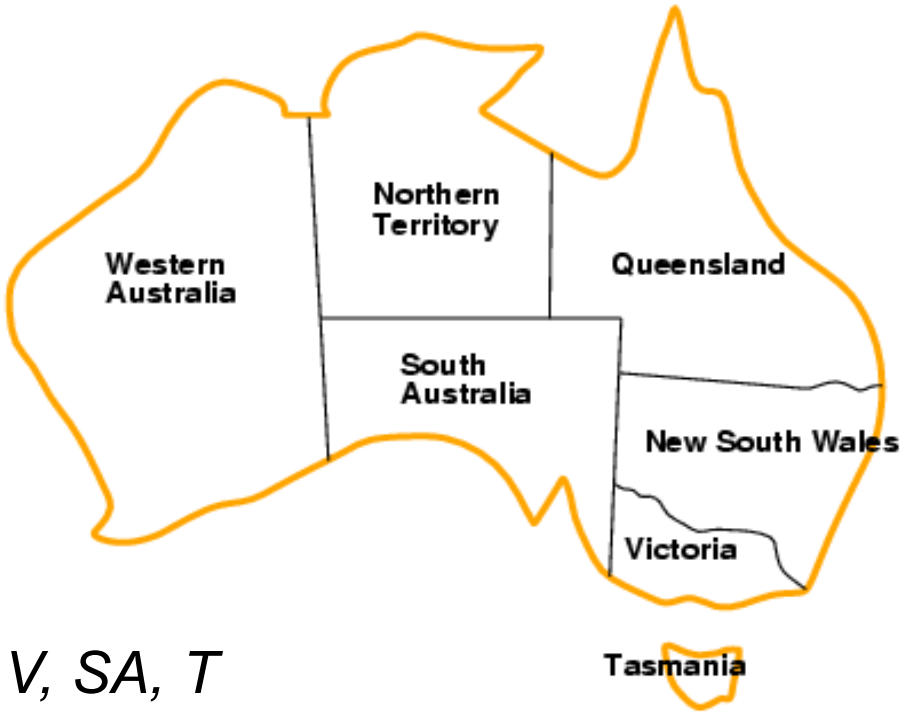
állapot: egy "**fekete doboz**" – bármely adatstruktúra, amire számítható a heurisztika, a lépésből adódó eredményállapot, és a célállapotteszt.

CSP:

állapot: egy D_i érték-tartományból értékeit felvevő X_i (állapot)változók által definiált

célállapotteszt: korlátok halmaza teljesül-e a változóértékek megengedett kombinációi révén?

pl. Térképszínezés



Változók: *WA, NT, Q, NSW, V, SA, T*

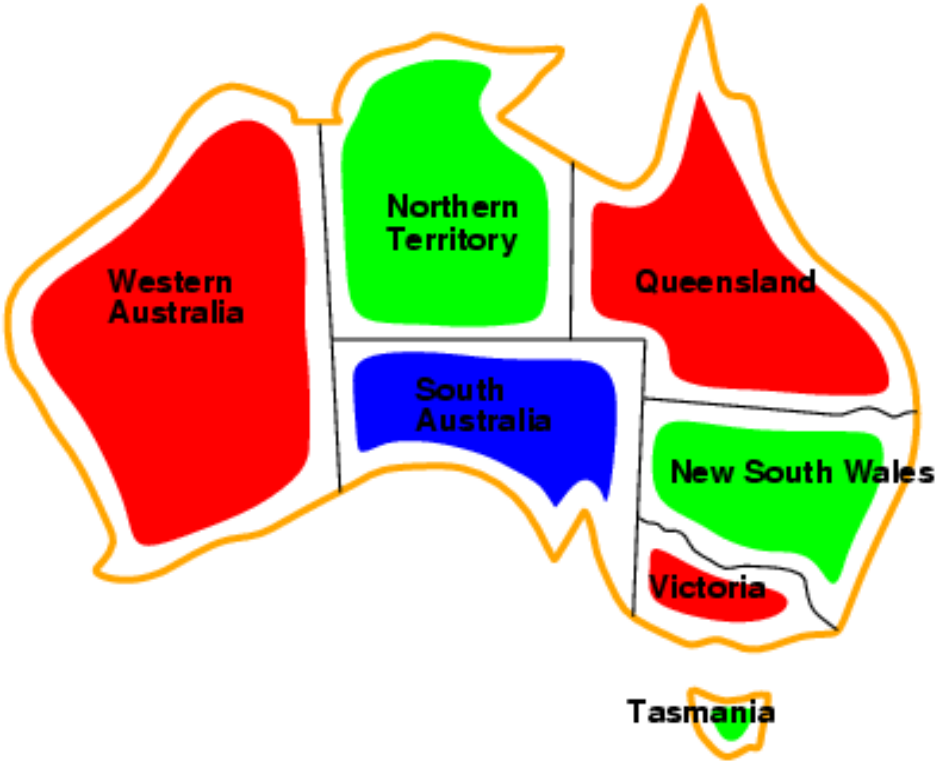
Értéktartományok D_i = {red, green, blue}

Korlátok: szomszédos területek színe legyen eltérő

pl. WA ≠ NT, ill. (WA,NT) értékét csakis a
{(red, green), (red, blue), (green, red), (green, blue), (blue, red), (blue, green)} halmazból vehet fel.

Lehetséges-e?

pl. Térképszínezés

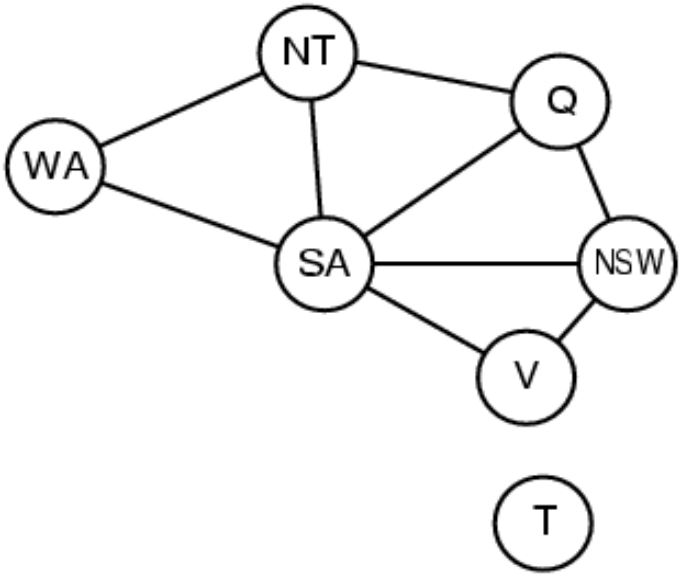


Megoldás: **teljes** és **konzisztens** változó-érték hozzárendelés

pl. WA = red, NT = green, Q = red, NSW = green, V = red,
SA = blue, T = green

(T lehet akármi, mert nem határos senkivel)
(különben is több megoldás van)

Korlátok gráfja



korlátgráf: csomópontjai a változók és élei a korlátok

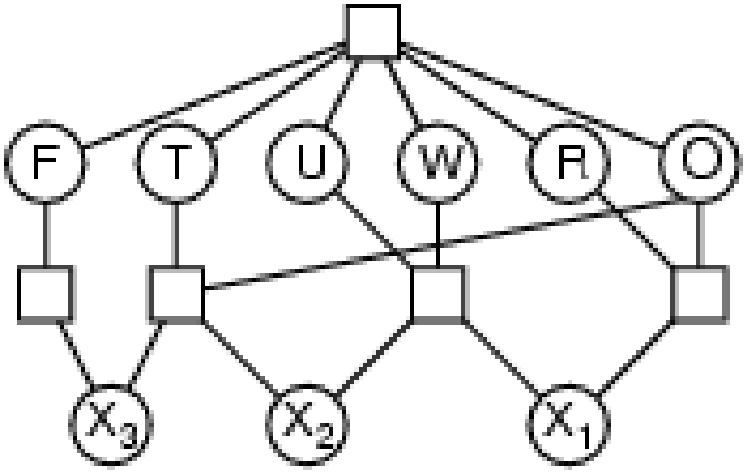
unáris CSP: egy-egy korlát csak egy változóra vonatkozik

bináris CSP: egy-egy korlát 2 változót köt össze

...

pl. Kriptoaritmetika

```
    T W O
+   T W O
-------
F O U R
```



Változók: $F\ T\ U\ W\ R\ O\ X_1\ X_2\ X_3$

Értéktartományok: {*0,1,2,3,4,5,6,7,8,9*}, {*0,1*}

Korlátok: Mind-eltérő*(F, T, U, W, R, O)*

$O + O = R + 10 \cdot X_1$

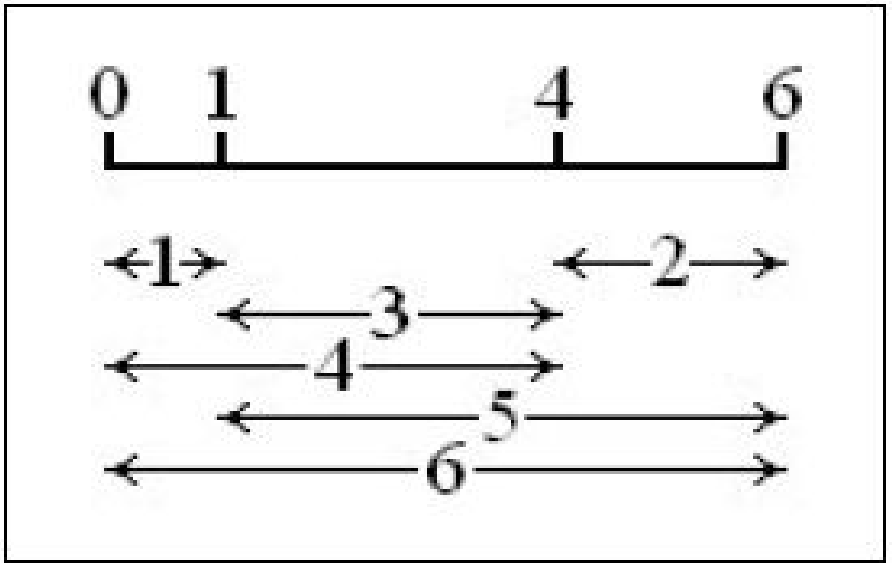
$X_1 + W + W = U + 10 \cdot X_2$

$X_2 + T + T = O + 10 \cdot X_3$

$X_3 = F,\ T \neq 0,\ F \neq 0$

Magasabb rendű korlátok gráfja egy hipergráf!

pl. Golomb vonalzó

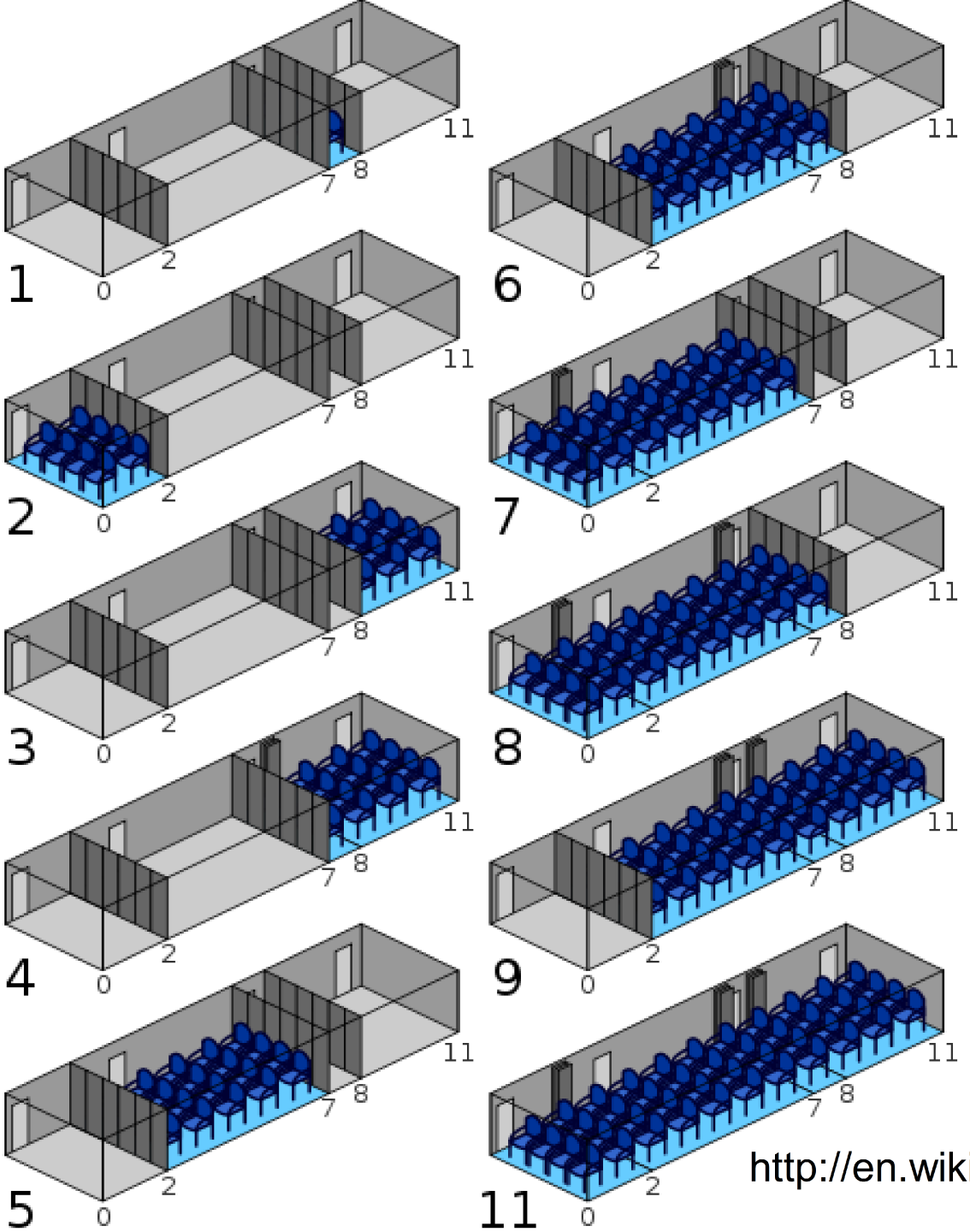


N db. marker (itt N = 4),
vonalzó hossza M (itt M = 6)
Markerek olyan elhelyezése,
hogy a páronkénti távolságuk
mind eltérő legyen
(a hosszáig bezárólag minden
táv mérhető legyen = tökéletes Gv),
nincs tökéletes Gv 5 vagy több
marker esetén, optimális Gv
megkeresésének komplexítása
ismeretlen, stb.)

Változók: P1, P2, P3, …, PN

Értéktartomány: {*0,1,2,3, …, M*}

Korlátok:
Mind-eltérő*(|P1-P2|, |P1-P3|, |P1-P4|, |P2-P3|, |P2-P4|, |P3-P4|)*

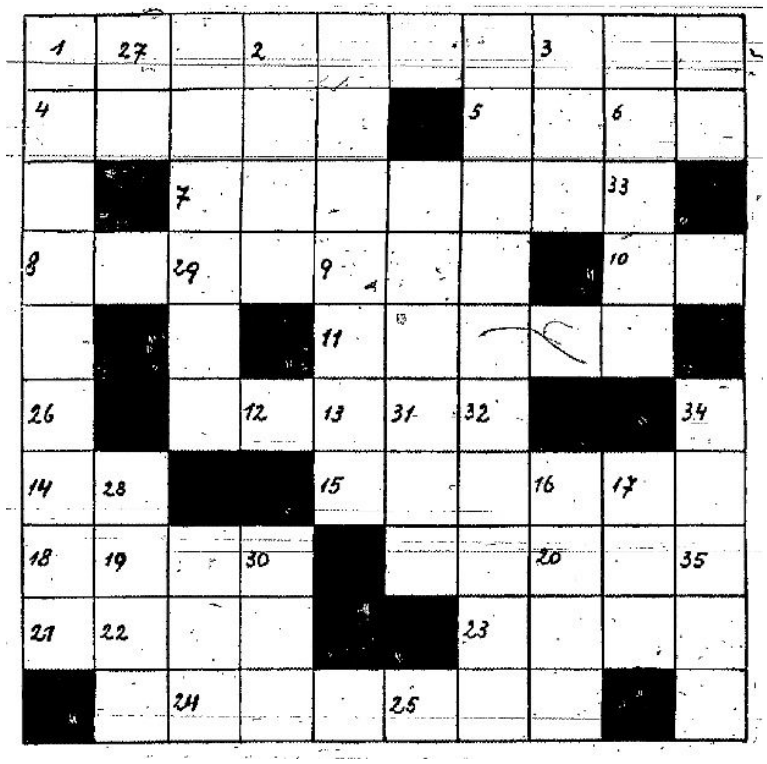


10 különböző méretre átrendezhető konferencia terem Golomb vonalzó szerinti elválasztó falakkal

http://en.wikipedia.org/wiki/Golomb_ruler

pl. Sudoku, keresztrejtvény, stb.

		8		5			1	2
			8		9			3
	6		2		4		5	
6		9		2	3	8		
8		5				3		6
		4	9	8		5		1
	3		1		2		8	
2			4		8			
4	8			9		2		



CSP problémák típusai

Diszkrét változók

véges értéktartományok:

pl. Boole-típ. CSPs, Boole-féle kielégítési vizsgálatok (NP-t)

végtelen értéktartományok:

egész számok, füzérek, stb.

pl. job scheduling, változók a munkaszakaszok kezdete/vége

Folytonos változók

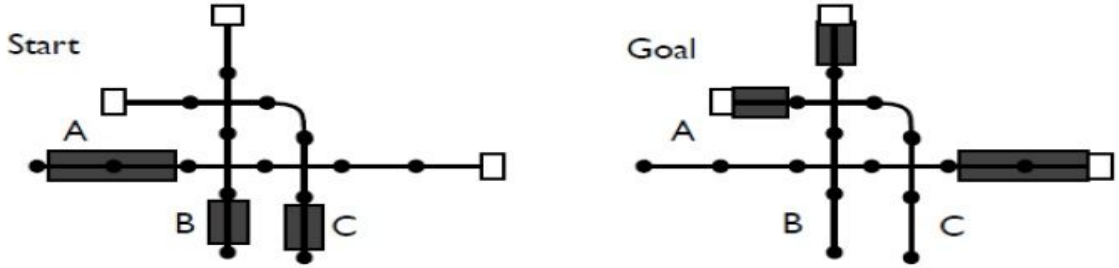
pl. a Hubble Space Telescope megfigyeléseit határoló időpontok, fizikai állapotváltozók

Unáris korlát: egyetlen egy változóra vonatkozik, pl. SA ≠ green

Bináris korlát: két változó viszonyára vonatkozik, pl. SA ≠ WA

Magasabb-rendű korlát: 3 vagy több változó viszonyára vonatkozik, pl. oszloponkénti változó korlátok kriptoaritmetikai feladványokban

Valós problémák



Hozzárendelési problémák, pl. ki, mit, hol tanítson
Menetrendi problémák, pl. az egyetemi órarend/ teremfoglalás
Szállításütemezés
Gyári ütemezések

Problémák többsége valós változókkal dolgozik

Pl. **Raktár-tervezési probléma**
Egy szupermarket-lánc raktárakat szeretne telepíteni a bolti hálózat kiszolgálására. Mit tudunk?
L1, ..., Ln lokáció, ahol raktár építhető. Csak k db. raktárra van szükség (k < n). Minden lokációra jellemző CPi kapacitás = hány boltot képes kiszolgálni. Minden bolthoz rendelni kell egy raktárt.
Si bolt ellátása Lj lokációból Sci,j-be kerül.
Az összköltséget TC konstans alatt kell tartani.

Gyári ütemezés – erőforrások (gépek, szerszámok) időben való allokációja

- végrehajtható tervek, a legjobb alternatívák

„Tiszta” ütemezés – tevékenységek kezdő, vég pontjai

„Tiszta” erőforrás allokáció – tevékenységek és erőforrások hozzárendelése

Együttes probléma – tevékenységek versengenek szűkös erőforrásokért.

Erőforrások

- egyetlen tevékenység használhatja
- diszkrét egységekben igénybe vehetők

Rendelkezésre állás (kapacitás)

- rögzített
- változó

Alternatív erőforrásokkal, de más-más idő alatt

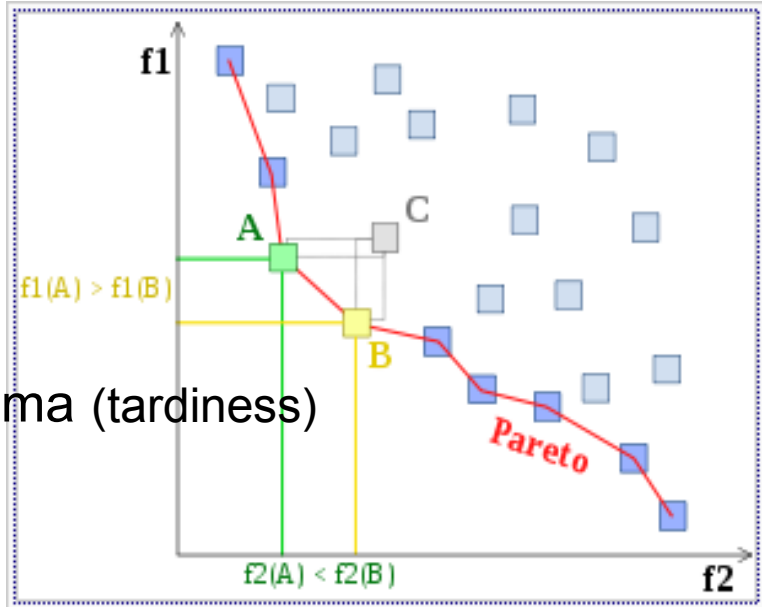
Korlátok

- időablakok, határidők
- precedenciák tevékenységek között (termékszerkezet, technológia)
- mellékproblémák (hőkezelés a legvégén, azonos vevő összes rendelése egyszerre szállítandó, egyes gépek korlátjai, pl. max. súly a darunál, ...)

Gyári ütemezés – erőforrások (gépek, szerszámok) időben való allokációja

Optimalizálási kritérium

Egyetlen, jól definiált kritérium
- szállítóképesség
  - átfutási idő minimuma (makespan)
  - késések számának minimuma
  - késések (súlyozott) összegének minimuma (tardiness)
- költségek minimuma
  - tevékenységek összköltsége
  - work-in-process (WIP)
- erőforrás kihasználtság



(http://en.wikipedia.org/wiki/Pareto_efficiency)

Jól definiált kritériumok kombinációja, súlyozott összege, ...

Pareto optimum

Preferenciák, puha korlátok, ...

Brute-force? pl. 1000 bin. változó: 2^{1000} kb. 10^{301} művelet

10^{15} op/sec (peta FLOPS), 10^{286} sec (világűr kora 10^{18} sec)

(Váncza József, SZTAKI, http://www.sztaki.mta.hu/~vancza/)

Hardver/ szoftver formális verifikáció, protokoll-tervezés, stb.

Logikai kifejezések kielégíthetősége (SAT)
- mi a (bináris) változók az az értékkészlete,
  hogy az összes kifejezés (un. klóz) egyszerre lehessen igaz?

$$(a \vee \neg b \vee \neg c) \text{ AND } (b \vee \neg c) \text{ AND } (a \vee c)$$

a = igaz/ hamis?
b = igaz/ hamis?
c = igaz/ hamis?

$2 \times 2 \times 2 = 2^3 = 8$

itt: a = igaz, b = igaz, c közömbös, vagy
a = igaz, c = hamis, b közömbös, stb.

Hardver/ szoftver formális verifikáció, protokoll-tervezés, stb.

The instance bmc-ibm-6.cnf, IBM LSU 1997:

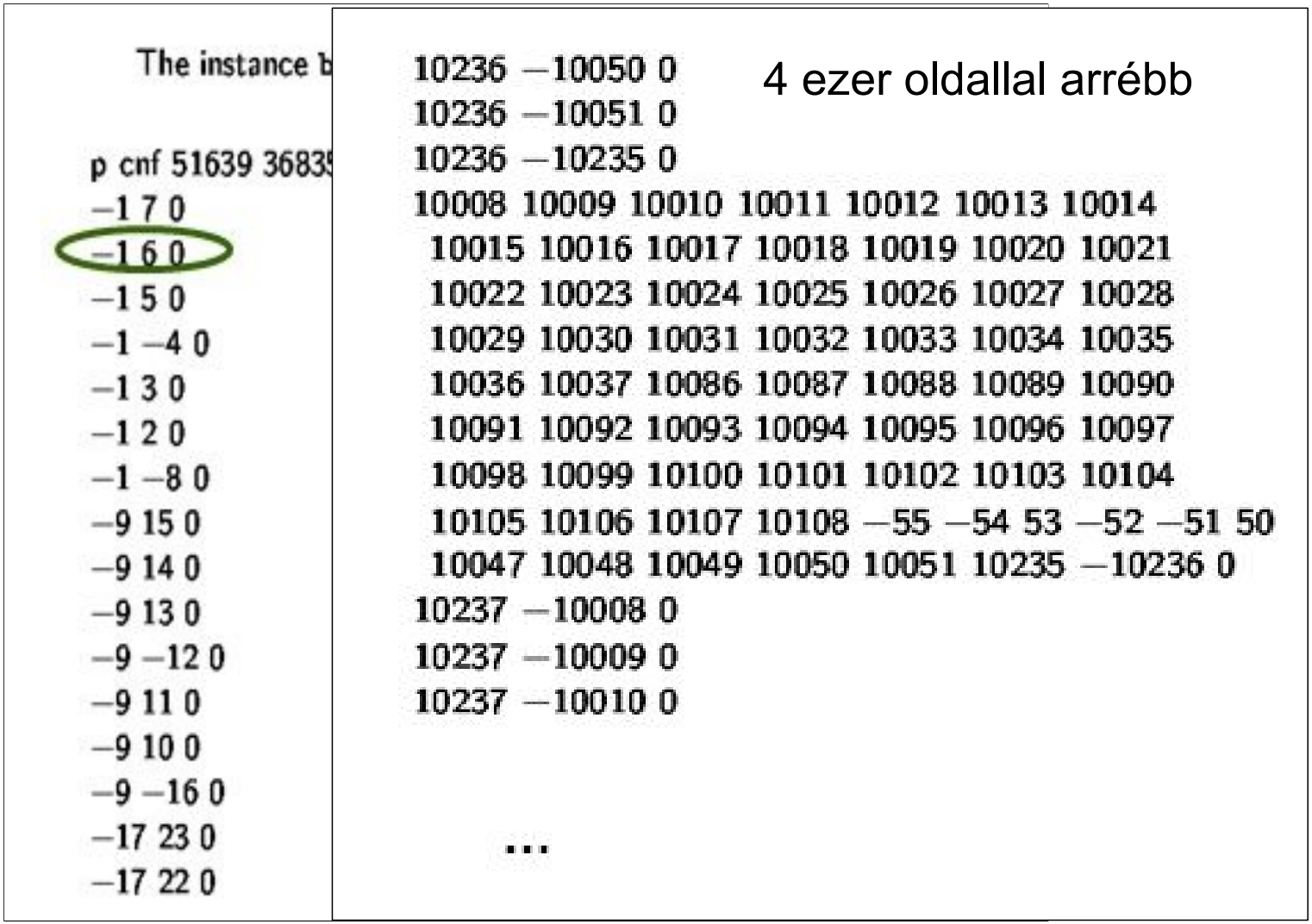
```
p cnf 51639 368352
-1 7 0
-1 6 0
-1 5 0
-1 -4 0
-1 3 0
-1 2 0
-1 -8 0
-9 15 0
-9 14 0
-9 13 0
-9 -12 0
-9 11 0
-9 10 0
-9 -16 0
-17 23 0
-17 22 0
```

i.e., ((not x_1) or x_7)
((not x_1) or x_6)
etc.

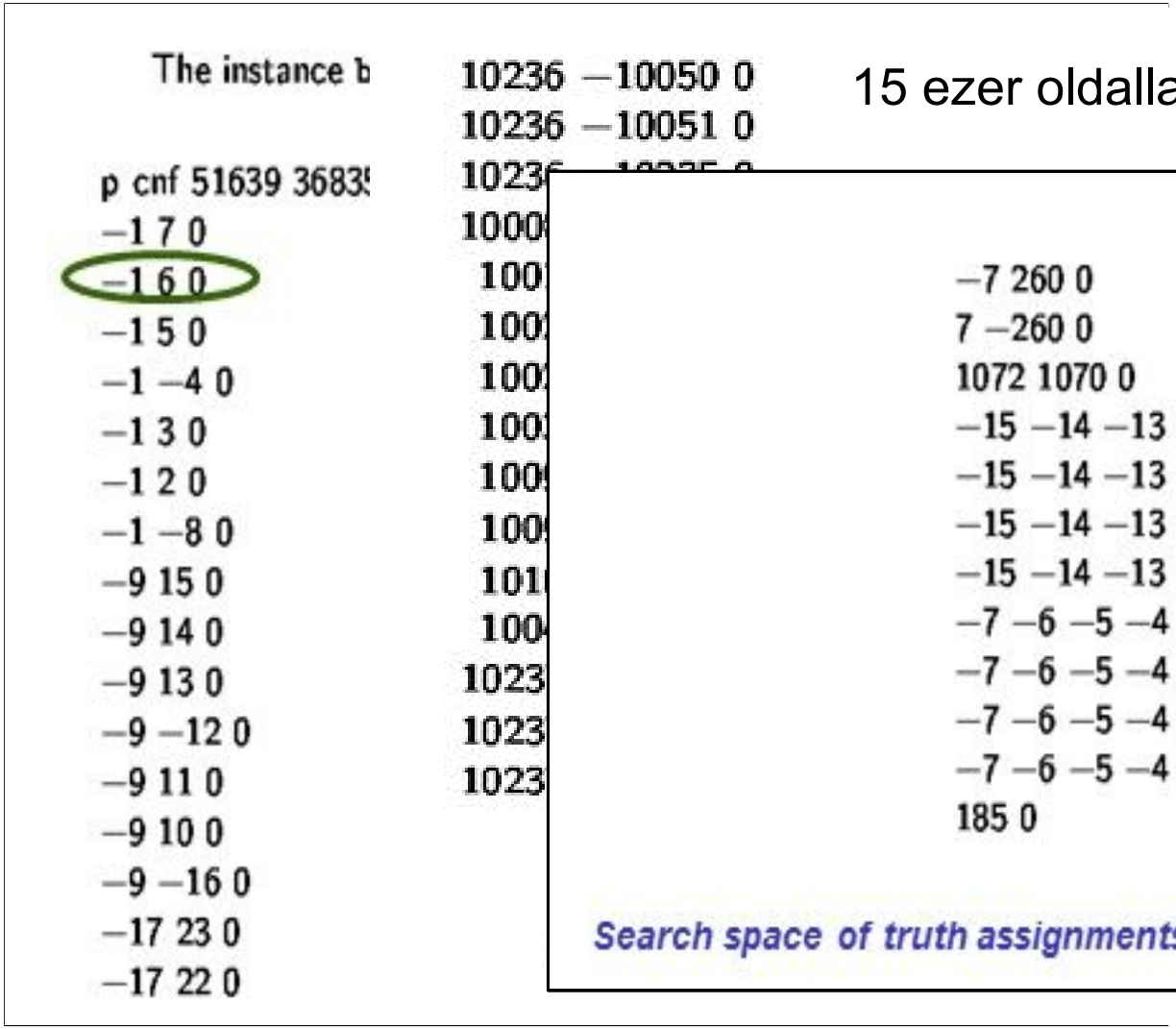
x_1, x_2, x_3, etc. are our Boolean variables (to be set to True or False)

Should x_1 be set to False??

Hardver/ szoftver formális verifikáció, protokoll-tervezés, stb.



Hardver/ szoftver formális verifikáció, protokoll-tervezés, stb.



15 ezer oldallal arrébb

kb.. 10 mp alatt!

Megfogalmazás problémamegoldáshoz

Kezdeti állapot: változó-hozzárendelés üres { }

Operátor: értéket hozzárendelni egy még nem hozzárendelt változóhoz úgy, hogy az eddigi hozzárendelésekkel ne ütközzön

→ kudarc, ha nincs megengedett hozzárendelés

Célállapotteszt: ha az aktuális hozzárendelés teljes

n db változó esetén minden megoldás n mélységben fekszik

→ használjuk a **mélységi keresést**

a pálya (út) itt nem számít

elágazási tényező *L* mélységben, *d* számosságú domain mellett

$b = (n - L)\, d$, azaz $n!\, d^n$ levélcsomópont

Keresés

Változó hozzárendelés **kommutatív**, azaz

(WA = red) majd (NT = green)
u.u, mint (NT = green) majd (WA = red)

Egy-egy csomópontban csakis egyetlen egy változó hozzárendelése megtörténthet:

→ b = d és fának d^n levele van

Visszalépéses keresés, azaz

egy mélységi keresés egyetlen egy változó hozzárendeléssel

alapvető nem informált algoritmus (keresés) CSP problémák megoldására

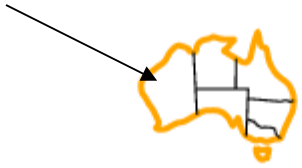
Megfogalmazás problémamegoldáshoz

Az n-királynő probléma tanulsága: legyen három modell

1. változók: x_{ij}
   domén: {0, 1}
   kényszerek (sorok, oszlopok, átlók)
2. változók: $x_1, \ldots, x_n$ (egy királynő mező poziciója)
   domén: {0, 1, 2, …, n^2-1} (mezőindex)
   kényszerek (sorok, oszlopok, átlók)
3. változók: $x_1, \ldots, x_n$ (királynő sorindexe)
   domén: {1, 2, …, n} (oszlop-index)
   kényszerek (sorok, oszlopok, átlók)

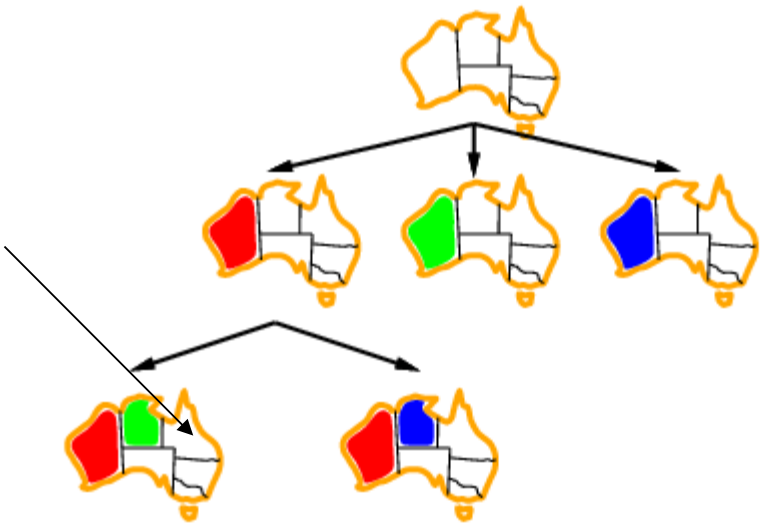
modell	változó	domén	keresési tér	n = 4	n = 8	n = 20
1.	n^2	2	(2)^n^2	6.6e4	1.8e19	2.6e120
2.	n	n^2	$(n^2)^n$	6.6e4	2.8e14	1.1e52
3.	n	n	n!	24	4e4	2.4e18

Visszalépéses keresés

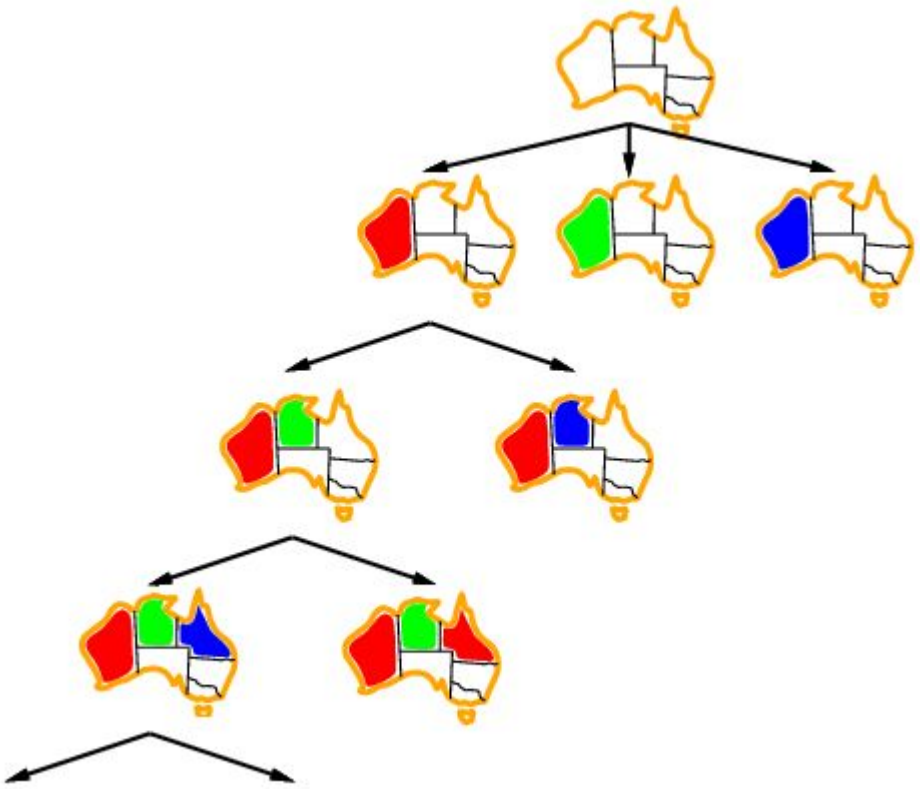


Visszalépéses keresés

Visszalépéses keresés



Visszalépéses keresés



Visszalépéses keresés hatékonyságának növelése

Általános módszerekkel is komoly gyorsítást el lehet érni:

- Melyik változóval foglalkozunk a legközelebb?
- Milyen sorrendben vizsgáljuk az értékeit?
- Érzékelhetjük-e jól előre a kudarcokat?
  (korai nyesés, mint az alfa-béta)

ezek az un. domain-független heurisztikák

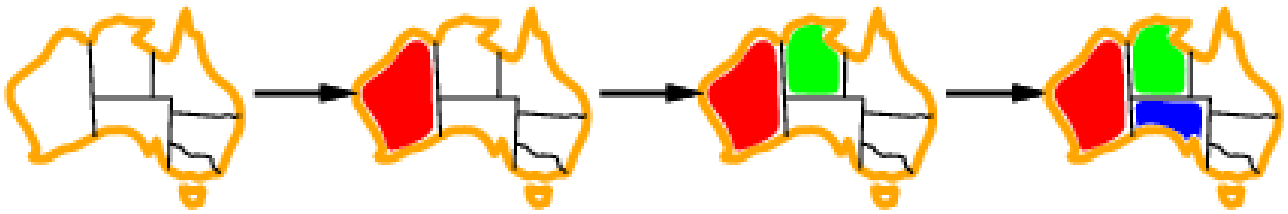
„A legkevesebb fennmaradó érték” ötlete

A leginkább korlátozott változó:

a legkisebb számú megengedett értékkel rendelkező változóval kezdjünk, ill. folytatjuk

= **legkevesebb fennmaradó érték** heurisztika

(min remaining variables, MRV)

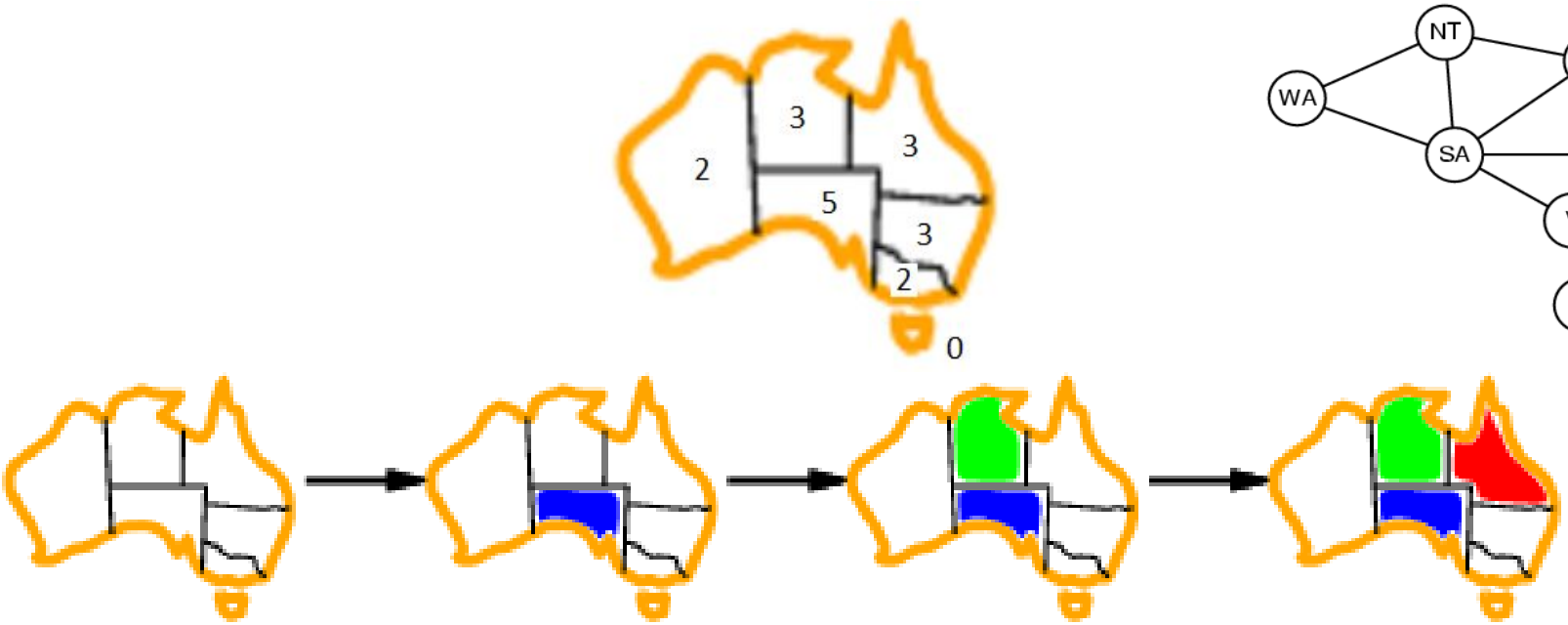


(NT ill. SA csak 2 megengedett érték (piros már nem lehet), minden más 3)

Fokszám heurisztika ötlete

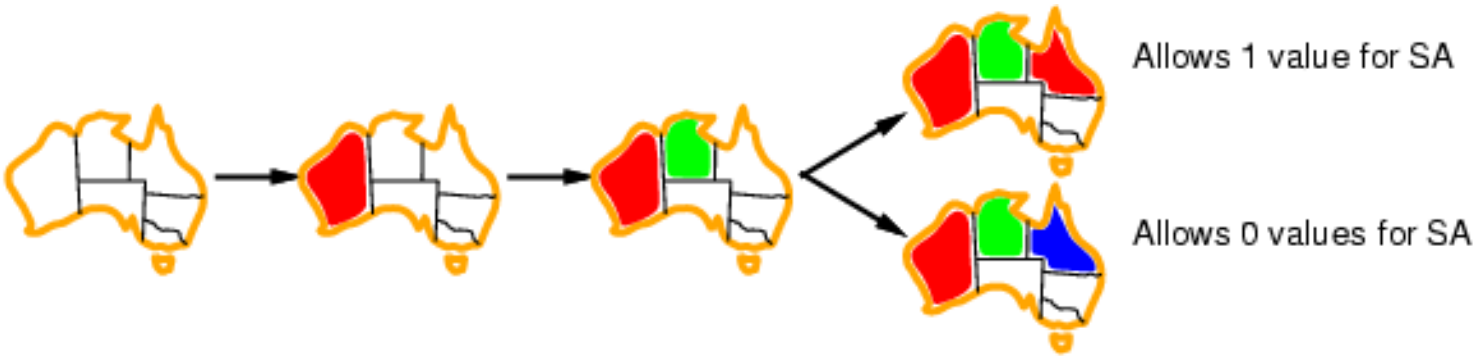
Az MRV-heurisztika semmit sem segít abban, hogy melyik régiót válasszuk ki elsőként Ausztrália kiszínezésekor, mert a kiinduláskor mindegyik régiónak három megengedett színe van.

A későbbi választások elágazási tényezőjét csökkenteni, hogy azt a változót választja ki, amely a legtöbbször szerepel a hozzárendeletlen változókra vonatkozó kényszerekben.



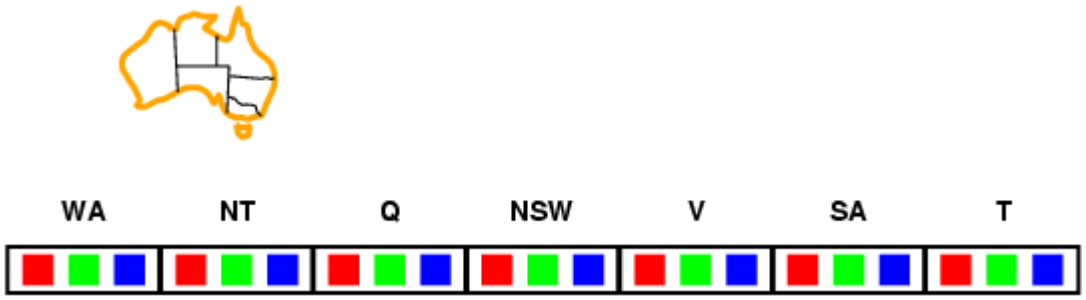
A legkevésbé korlátozott érték ötlete

Előnyben részesíteni azt az értéket, amely a legkevesebb választást zárja ki a kényszergráfban a szomszédos változóknál.



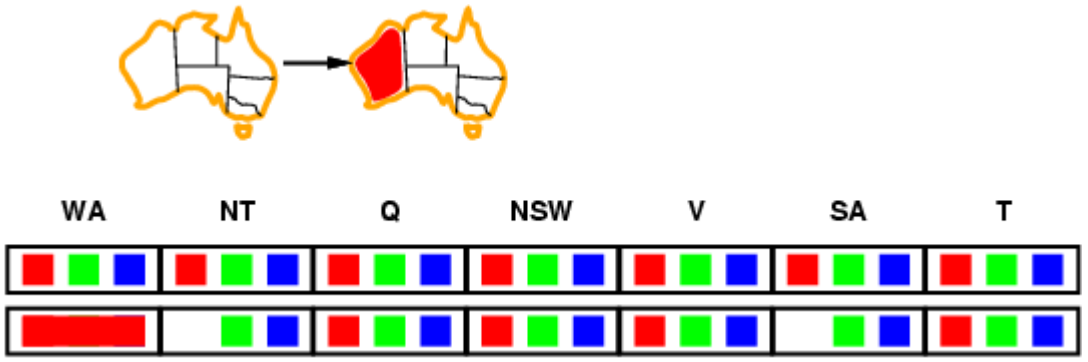
Előrenéző ellenőrzés

Az előrenéző ellenőrzés minden egyes alkalommal, amikor egy X változó értéket kap, minden, az X-hez kényszerrel kötött, hozzárendeletlen Y-t megvizsgál, és Y tartományából törli az X számára választott értékkel inkonzisztens értékeket.



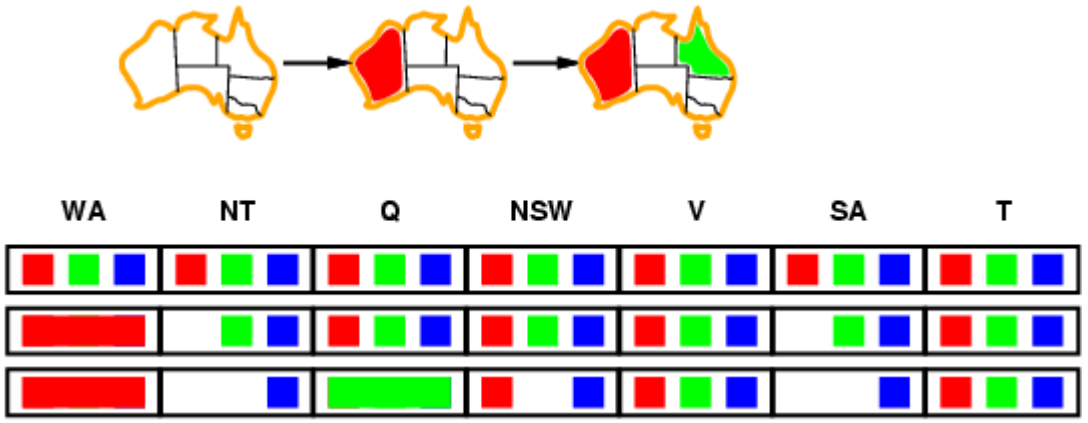
Előrenéző ellenőrzés

Az előrenéző ellenőrzés minden egyes alkalommal, amikor egy X változó értéket kap, minden, az X-hez kényszerrel kötött, hozzárendeletlen Y-t megvizsgál, és Y tartományából törli az X számára választott értékkel inkonzisztens értékeket.



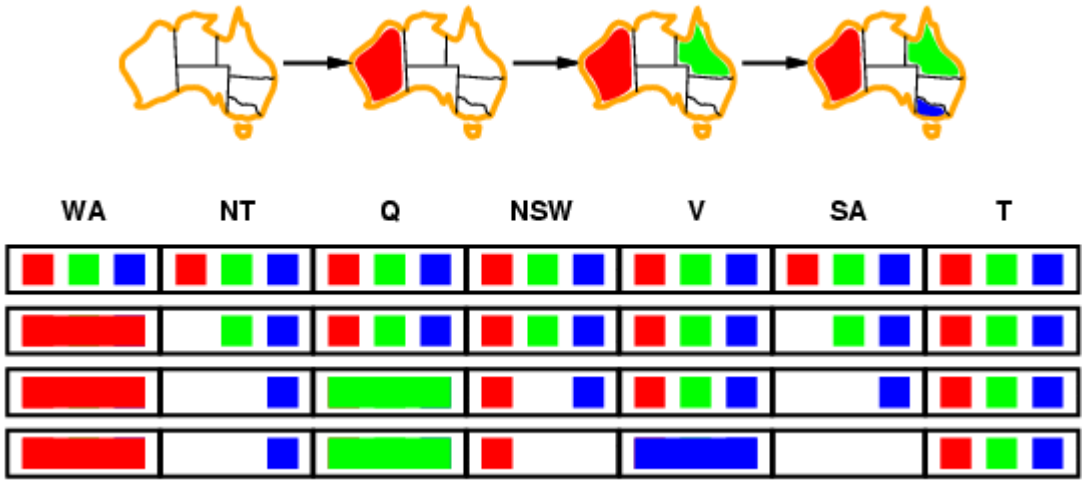
Előrenéző ellenőrzés

Az előrenéző ellenőrzés minden egyes alkalommal, amikor egy X változó értéket kap, minden, az X-hez kényszerrel kötött, hozzárendeletlen Y-t megvizsgál, és Y tartományából törli az X számára választott értékkel inkonzisztens értékeket.



Előrenéző ellenőrzés

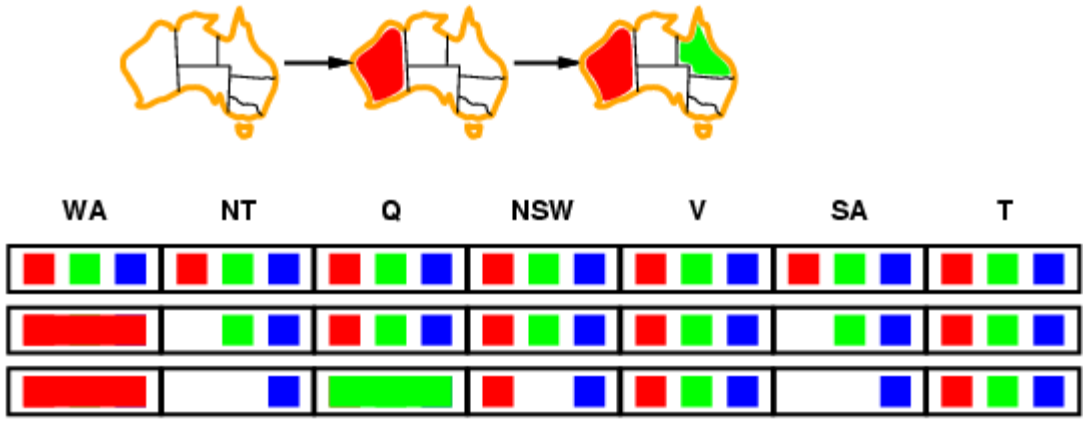
Az előrenéző ellenőrzés minden egyes alkalommal, amikor egy X változó értéket kap, minden, az X-hez kényszerrel kötött, hozzárendeletlen Y-t megvizsgál, és Y tartományából törli az X számára választott értékkel inkonzisztens értékeket.



Korlátozás előreterjesztése

Az előrenéző ellenőrzés ugyan sok inkonzisztenciát észrevesz, de nem mindent.

Ráadásul nem látja jól előre a kudarcokat.

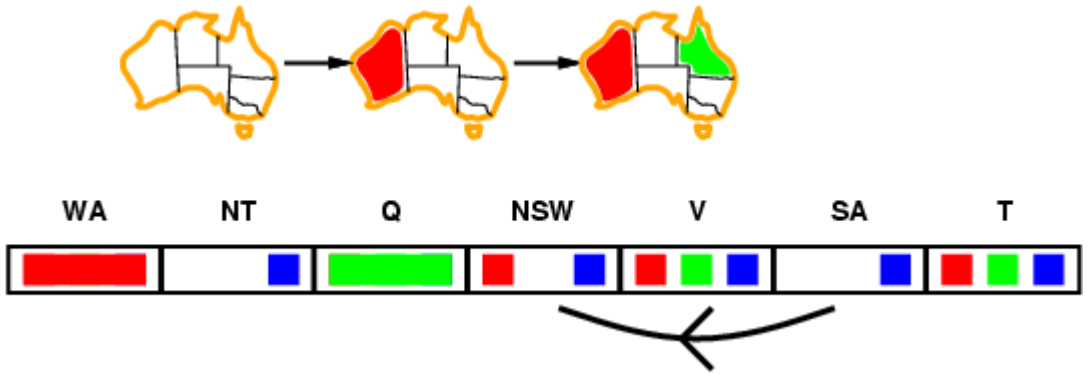


NT és SA egyszerre nem lehet kék.

Élkonzisztencia

X → *Y* él konzisztens akkor és csak akkor, ha

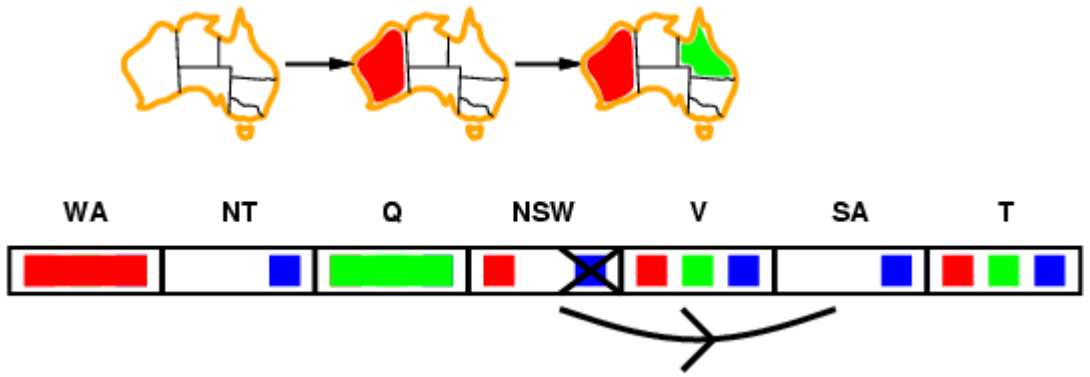
X minden x értékére létezik valamilyen megengedett y érték



Élkonzisztencia

X → *Y* él konzisztens akkor és csak akkor, ha

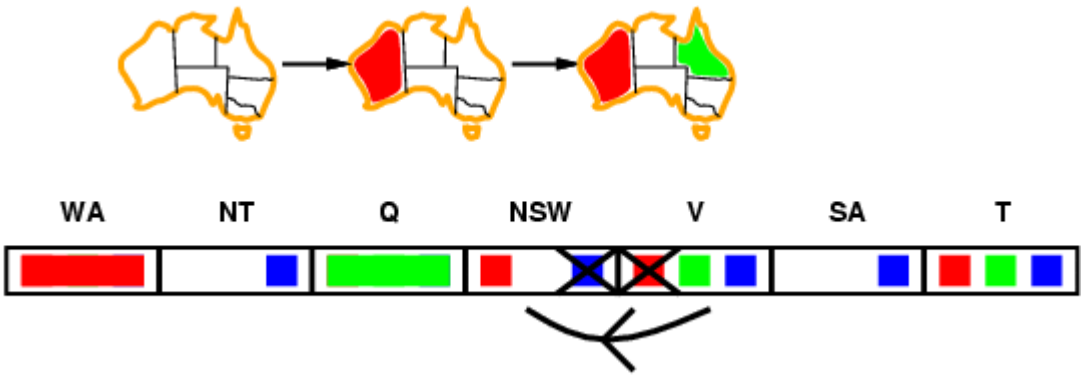
X minden x értékére létezik valamilyen megengedett y érték



Élkonzisztencia

X → *Y* él konzisztens akkor és csak akkor, ha
X minden x értékére létezik valamilyen megengedett y érték

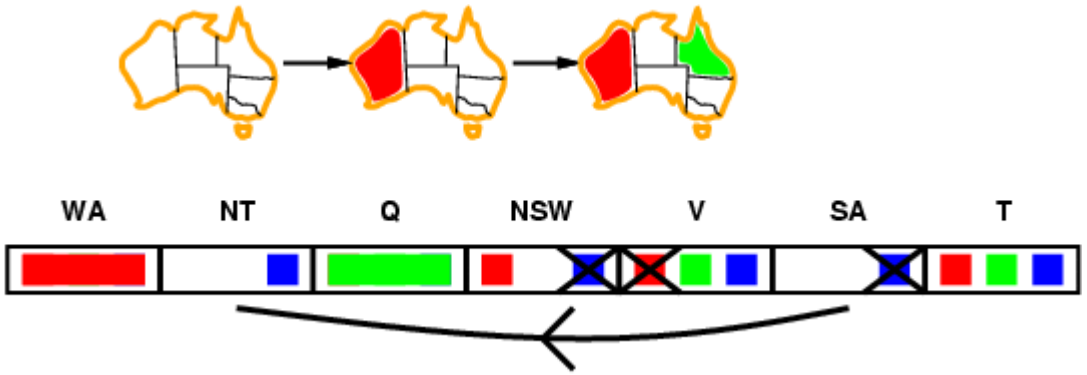
Ha X értéket veszít, a szomszédjait ujra kell ellenőrizni



Élkonzisztencia

X → *Y* él konzisztens akkor és csak akkor, ha
X minden x értékére létezik valamilyen megengedett y érték

Ha X értéket veszít, a szomszédjait ujra kell ellenőrizni



Az élkonzisztencia-ellenőrzés alkalmazható előfeldolgozó lépésként a keresés megkezdése előtt, vagy a keresési folyamat minden egyes hozzárendelését követő terjesztési lépésként

CSP lokális kereséssel

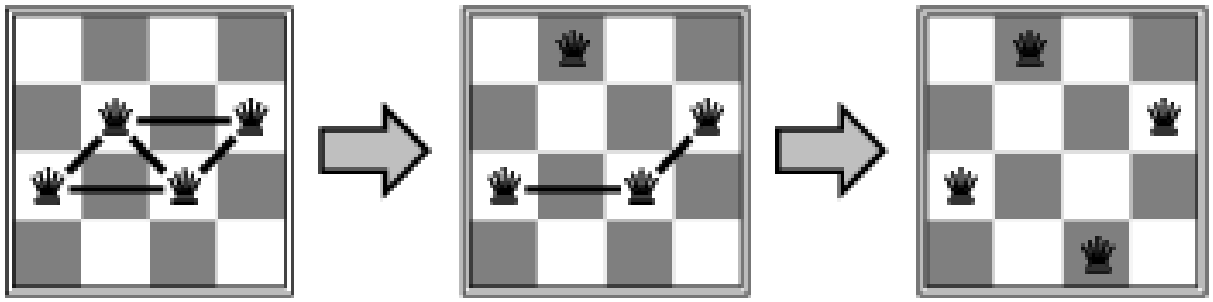
Hegymászó, szimulált lehűtés, …,
teljes állapotleírás = minden változónak van (esetleg rossz) értéke

CSP-re: - állapotok a nem teljesült kényeszerekkel is
- operátorok megváltoztatják a változók hozzárendelését

Változó szelekció: véletlen módon bármely konfliktusban lévő változó

Min. konfliktus heurisztika:

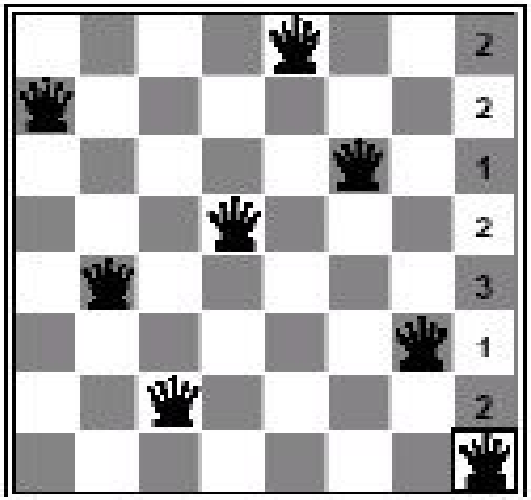
azt az értéket állítjuk be, amely a legkevesebb sz. korlátot sérti, pl.
hegymászó: *h(n)* = sérült korlátok száma

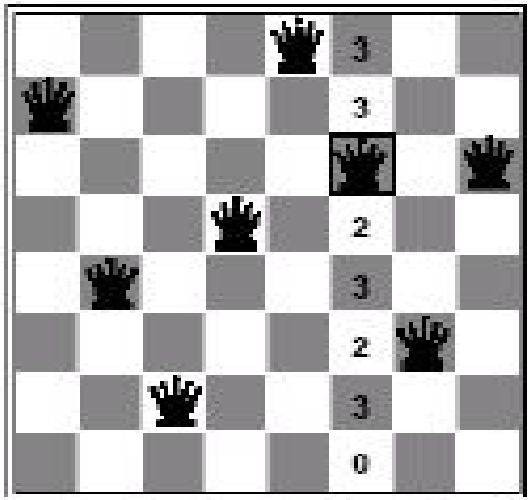


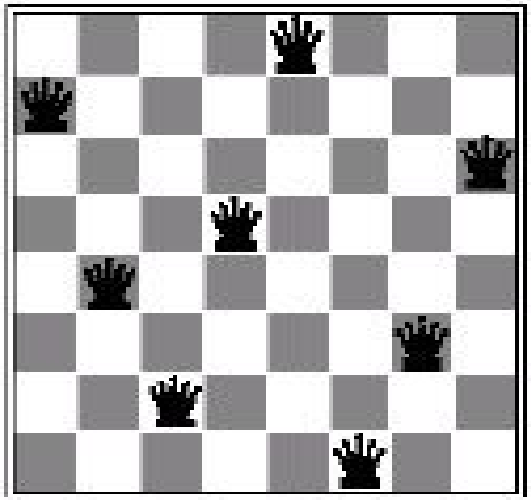
h(n) = a támadások száma

CSP lokális kereséssel

Min. konfliktus heurisztika:





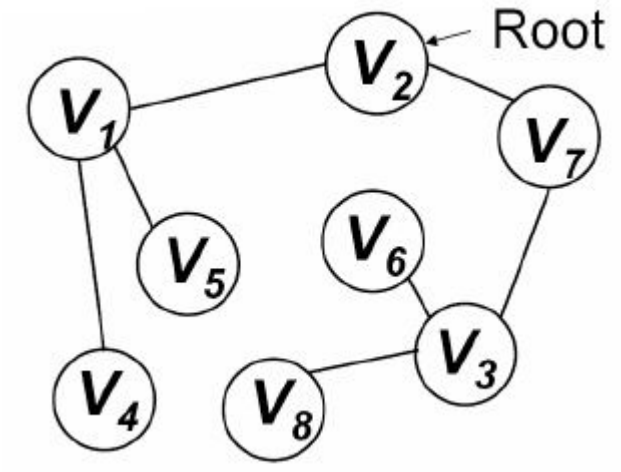


CSP struktúrája – miben segíthet

CSP gráfja: független komponensek … (komplexitás-számítás)

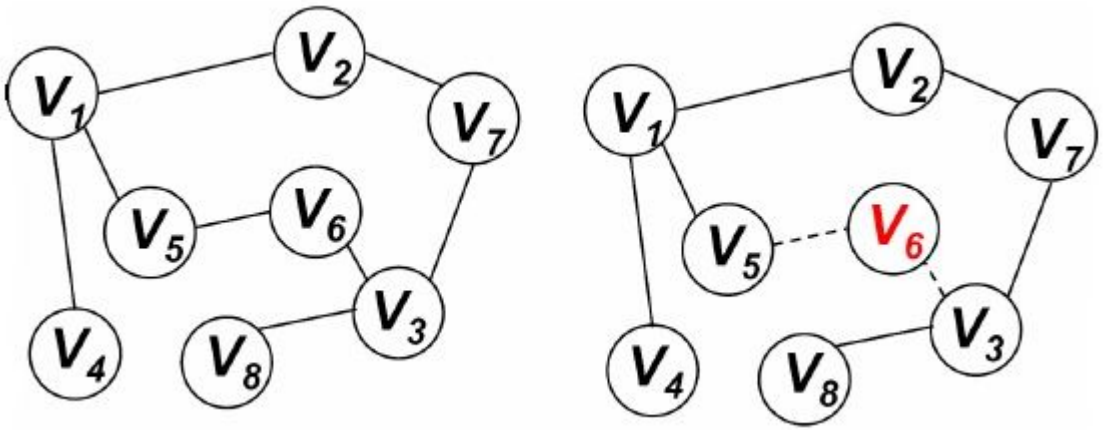
CSP fagráf: megoldás könnyű, változószámban lineáris

válasszunk egy levelet gyökérnek
élkonzisztencia-nyesés gyerekektől szülői felé
szülőkkel konzisztens értékek hozzárendelése gyerekeknek



CSP struktúrája

CSP hurkos gráf: megoldás általánosságban NP nehéz

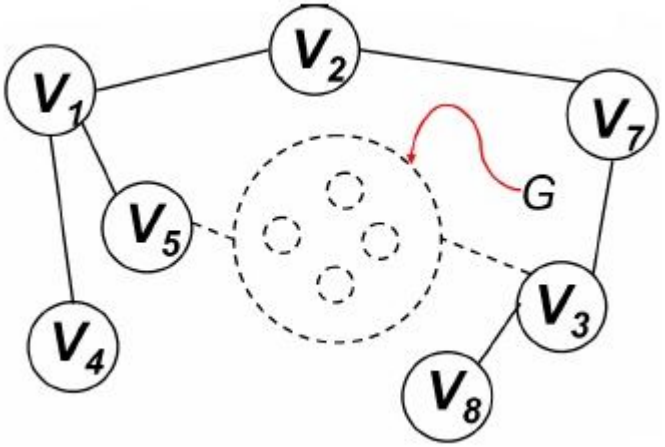


Gráf CSP konvertálása fába:
kondicionálás
vágóhalmaz
fa-dekompozíció

Megoldáskeresés
V_6 minden értékére

CSP struktúrája

CSP hurkos gráf: megoldás általánosságban NP nehéz



Megoldáskeresés
G minden értékére
CSP/G problémában

Gráf CSP konvertálása fába:
kondicionálás
vágóhalmaz
fa-dekompozíció
(részmegoldások „él-konzisztenciája”)