



Budapesti Műszaki és Gazdaságtudományi Egyetem Méréstechnika és Információs rendszerek Tanszék

Mesterséges intelligencia – VIMIAC10

Kényszerkielégítési problémák

Előadó: Dr. Hullám Gábor

Előadás anyaga: Dr. Pataki Béla
Dr. Dobrowiecki Tadeusz

Témakörök



Ajánló
rendszerek

Kényszerkielégítési (korlátkielégítési) problémák Constraint Satisfaction Problems (CSP):

Állapot: Az állapot a leíró változók és a hozzájuk rendelt értékek által definiált.

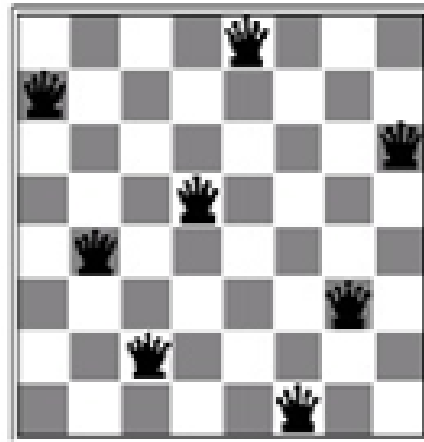
- Az x_k változók egy-egy D_k érték-tartományból (halmazból) veszik fel értékeiket .

Célállapotteszt:

1. Az összes állapotot leíró változóhoz rendeltünk számára megengedett értéket
2. Az adott korlátok teljes halmazát kielégítettük



		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	



T W O
+ T W O

F O U R

Órarend készítés:

Szerda 10-12, HG,MI-BSc5szem, IB.028

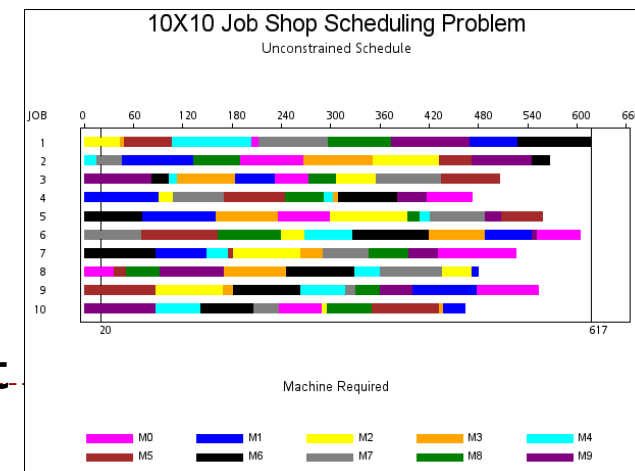
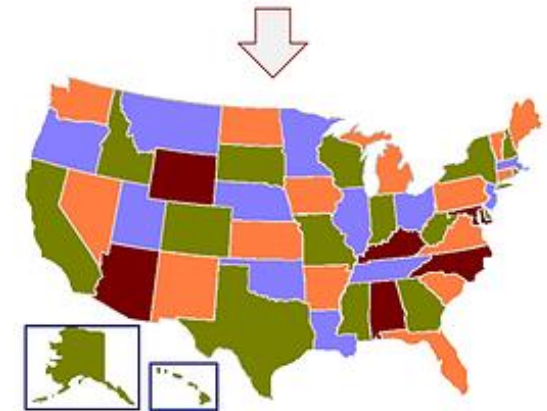
...

1. egyszerre egy teremben csak egy évfolyam
 2. egyszerre egy teremben csak egy oktató,
 3. senki se lehet egyszerre két helyen
 4. egy évfolyam egyszerre csak egy helyen lehet
- stb.

Térképszínezés

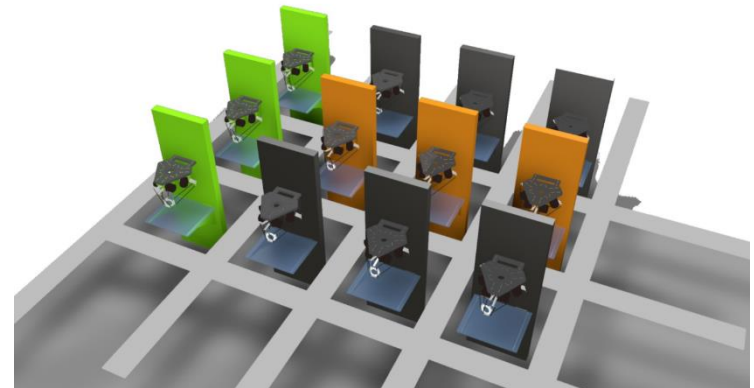
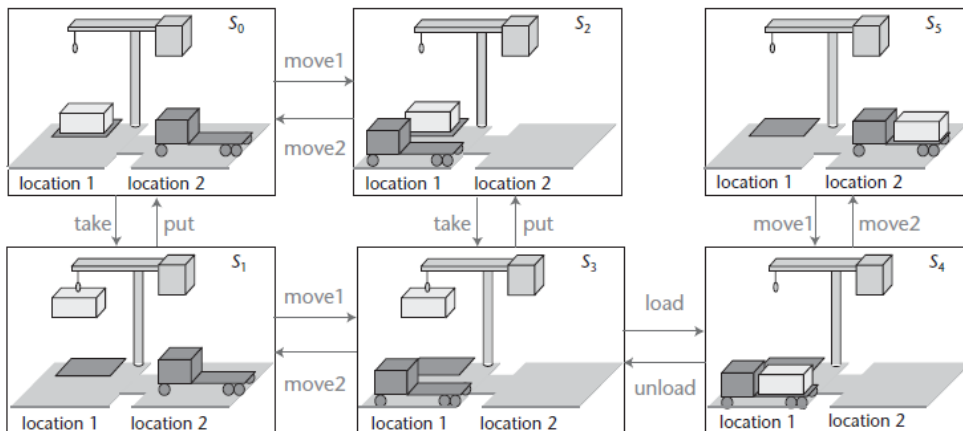
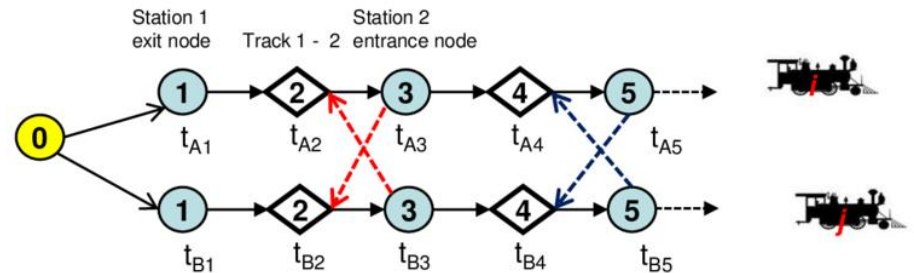


Satisfaction



Gyakorlati kényszerkielégítési problémák

- Hozzárendelési problémák, pl. ki, mit, hol tanítson
- Menetrendi problémák, pl. vasút
- Szállításütemezés
- Gyári ütemezés



Gyakorlati kényszerkielégítési problémák

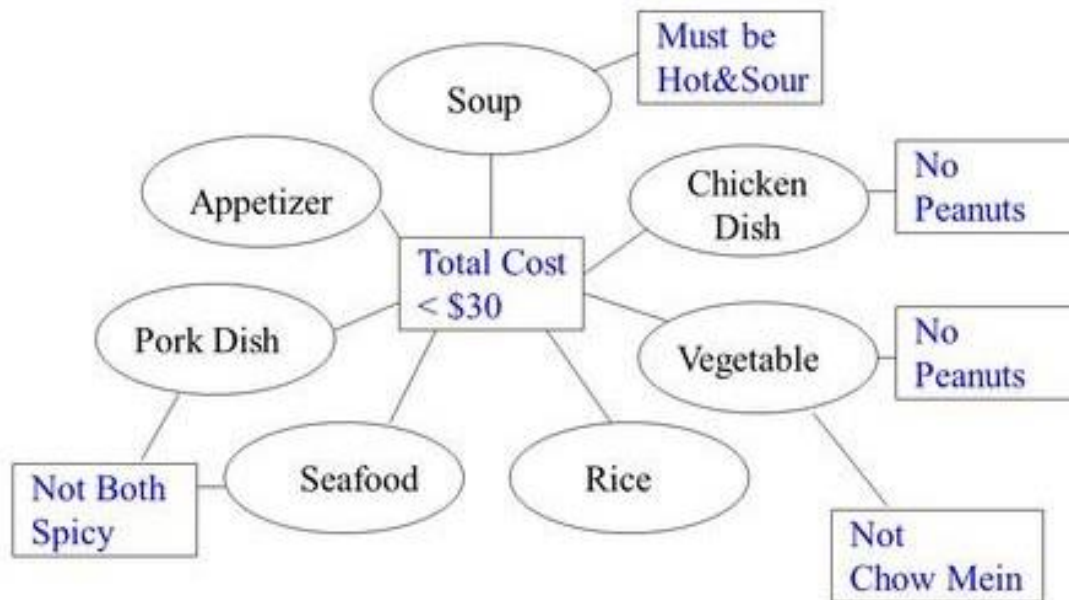
Raktár-tervezési probléma

Egy supermarket-lánc raktárakat szeretne telepíteni a bolti hálózat kiszolgálására. Mit tudunk?

- $L_1, \dots, L_n$ lokáció, ahol raktár megépíthető.
- Csak k db. raktárra van szükség ($k < n$).
- Minden lokációra jellemző CP_i kapacitás: hány boltot képes kiszolgálni
- Minden bolthoz rendelni kell egy raktárt.
- S_j bolt ellátása L_i lokációból $P_{i,j}$ -be kerül.
- Az összköltséget TC konstans alatt kell tartani.



Menüösszeállítás az étlap alapján



CSP problémák típusai

Változók alapján:

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

- megfigyeléseket határoló időpontok,
- fizikai állapotváltozók



CSP problémák típusai

Kényszerek, korlátok alapján:

Unáris korlát: egyetlen egy változóra vonatkozik, pl. $SA \neq \text{green}$

Bináris korlát: két változó viszonyára vonatkozik, pl. $SA \neq WA$

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

Preferecia-kényszer:

Egy érték jobb(an preferált), mint egy másik, pl. a vörös jobb, mint a zöld



Példa: Térképszínezés



Változók: WA, NT, Q, NSW, V, SA, T

Értéktartományok $D_i = \{\text{red, green, blue}\}$

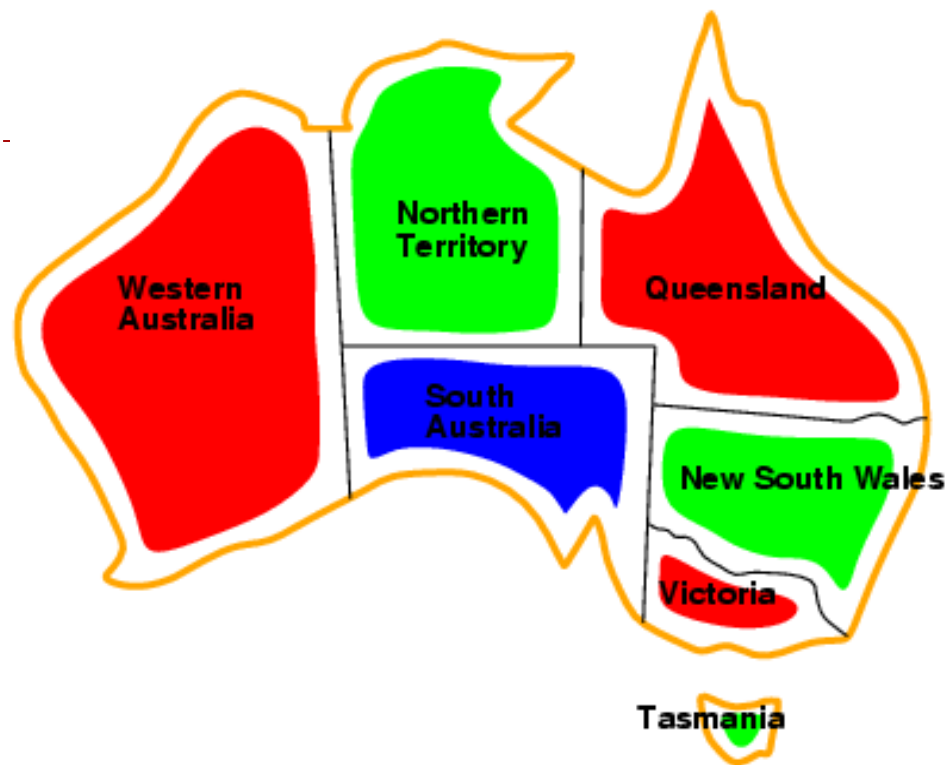
Korlátok: szomszédos területek színe legyen eltérő

pl. $WA \neq NT$, ill. más megfogalmazásban: (WA, NT) értékét csakis a $\{(\text{red, green}), (\text{red, blue}), (\text{green, red}), (\text{green, blue}), (\text{blue, red}), (\text{blue, green})\}$ halmazból vehet fel.

Lehetséges-e?



pl. Térképszínezés



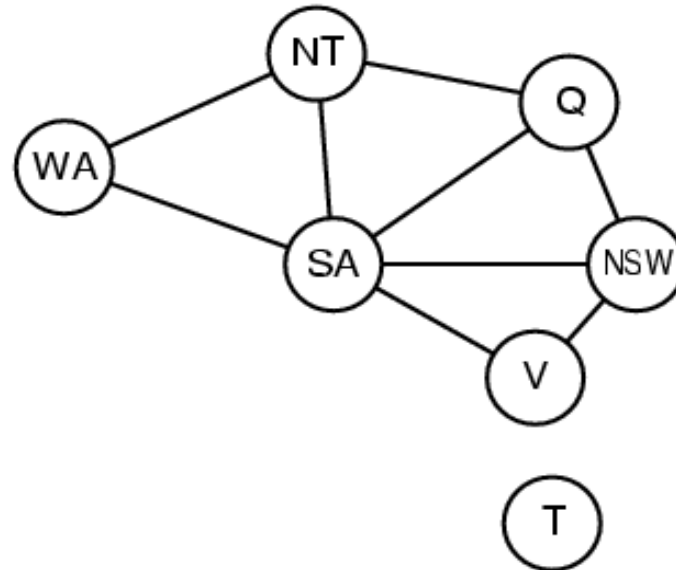
Megoldás: teljes és konzisztens változó-érték hozzárendelés

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

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



Korlátok gráfja



korlátgráf: csomópontjai a változók és élei a korlátok

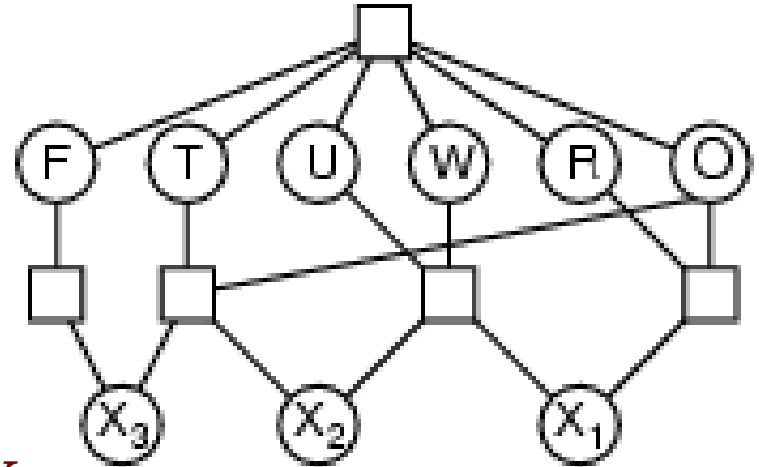
Itt csak **bináris CSP**-k: egy-egy korlát 2 változót köt össze

...



Másik példa: kriptoaritmetika

$$\begin{array}{r} T \ W \ O \\ + \ T \ W \ O \\ \hline F \ O \ U \ R \end{array}$$



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:

1. $\text{Mind-eltérő}(F, T, U, W, R, O)$
2. $O + O = R + 10 \cdot X_1$
3. $X_1 + W + W = U + 10 \cdot X_2$
4. $X_2 + T + T = O + 10 \cdot X_3$
5. $X_3 = F, T \neq 0, F \neq 0$



A probléma megfogalmazása

Kezdeti állapot: összes változó-hozzárendelés üres { }

Operátor: értéket hozzárendelni egy még nem lekötött változóhoz úgy, hogy az eddigi hozzárendelésekkel ne ütközzön (egyik kényszer se sérüljön)

→ kudarc, ha nincs megengedett hozzárendelés

Célállapotteszt: ha az aktuális hozzárendelés teljes és mindegyik kényszer teljesül



A probléma megfogalmazása

Keresési fa

n db változó esetén minden megoldás n mélységben fekszik

→ mi van, ha a **szélességi keresést** használjuk?

Elágazások száma L mélységben ($L=0, 1, 2, \dots$), ha minden változó d számú értéket vehet fel (ez a változó értékkészlete v. más néven doménje)

$b = (n - L) d$, (mert L változó már értéket kapott)

azaz $n! \cdot d^n$ levélcsomópont

(miközben d^n lehetséges hozzárendelés van)

(pl. 8 betűs számjegyaritmetika, $n=8$, $d=10$, levelek száma $4 \cdot 10^{12}$)



Keresés

Változó-hozzárendelés **kommutatív**, azaz például

(WA = red) majd (NT = green)
ugyanaz, mint (NT = green) majd (WA = red)

Egy-egy csomópontban csakis egyetlen egy változó hozzárendelése történhet meg, így:

$\rightarrow b = d$ és a fának d^n levele van



Keresés

- Változó-hozzárendelés **kommutatív**, azaz például
- Egy-egy csomópontban csakis egyetlen egy változó hozzárendelése történhet meg, így:
$$\rightarrow b = d \text{ és a fának } d^n \text{ levele van}$$
- Alapvető nem informált algoritmus (keresés) CSP problémák megoldására: **visszalépéses keresés**, azaz

mélyégi keresés

- minden szinten egyetlen egy változó-hozzárendeléssel
- ha sérül valamelyik kényszer, visszalép
(egyszer sérült kényszer mélyebben nem jöhet helyre)



Megfogalmazás (modell) hatása a problémamegoldásra

Az n -királynő probléma tanulsága: hasonlítsunk össze három modellt

- 1. modell**, a változók: x_{ij} (*a sakktáblamezők pozíciója*)
értékkészlet: $\{0, 1\}$ (*van rajta királynő vagy nincs*)
kényszerek (sorok, oszlopok, átlók)
- 2. modell**, a változók: $x_1, \dots, x_n$ (*egy királynő által elfoglalt mező pozíciója*)
értékkészlet: $\{0, 1, 2, \dots, n^2-1\}$ (*mezőindex*)
kényszerek (sorok, oszlopok, átlók)
- 3. modell**, a változók: $x_1, \dots, x_n$ (*az 1., 2. stb. sorban álló királynő sorindexe*)
értékkészlet: $\{1, 2, \dots, n\}$ (*oszlop-index*)
kényszerek (sorok, oszlopok, átlók)



Megfogalmazás (modell) hatása a problémamegoldásra

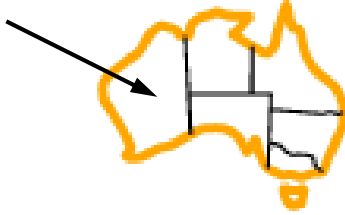
- 1. modell, a változók: x_{ij} (*a sakktáblamezők pozíciója*)
értékkészlet: $\{0, 1\}$ (*van rajta királynő vagy nincs*)
- 2. modell, a változók: $x_1, \dots, x_n$ (*egy királynő által elfoglalt mező pozíciója*)
értékkészlet: $\{0, 1, 2, \dots, n^2-1\}$ (*mezőindex*)
- 3. modell, a változók: $x_1, \dots, x_n$ (*az 1., 2. stb. sorban álló királynő sorindexe*)
értékkészlet: $\{1, 2, \dots, n\}$ (*oszlop-index*)

$n \times n$ -es sakktáblán az n -királynő probléma néhány n -re

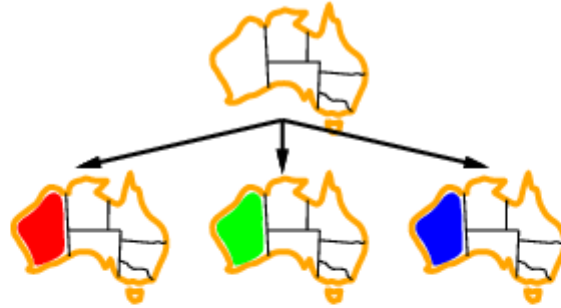
modell	változó	értékkészl.(d)	levelek sz. (d^n)	$n = 4$	$n = 8$	$n = 20$
1.	n^2	2	$(2)^{(n^2)}$	$6.6 \cdot 10^4$	$1.8 \cdot 10^{19}$	$2.6 \cdot 10^{120}$
2.	n	n^2	$(n^2)^n$	$6.6 \cdot 10^4$	$2.8 \cdot 10^{14}$	$1.1 \cdot 10^{52}$
3.	n	n	n^n	256	$1.6 \cdot 10^7$	$1.0 \cdot 10^{26}$



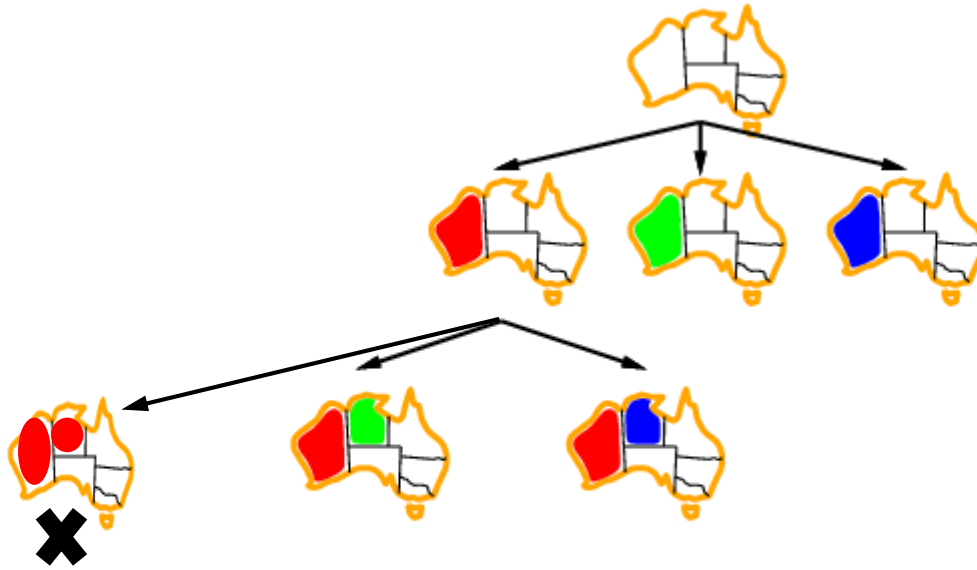
Visszalépéses keresés



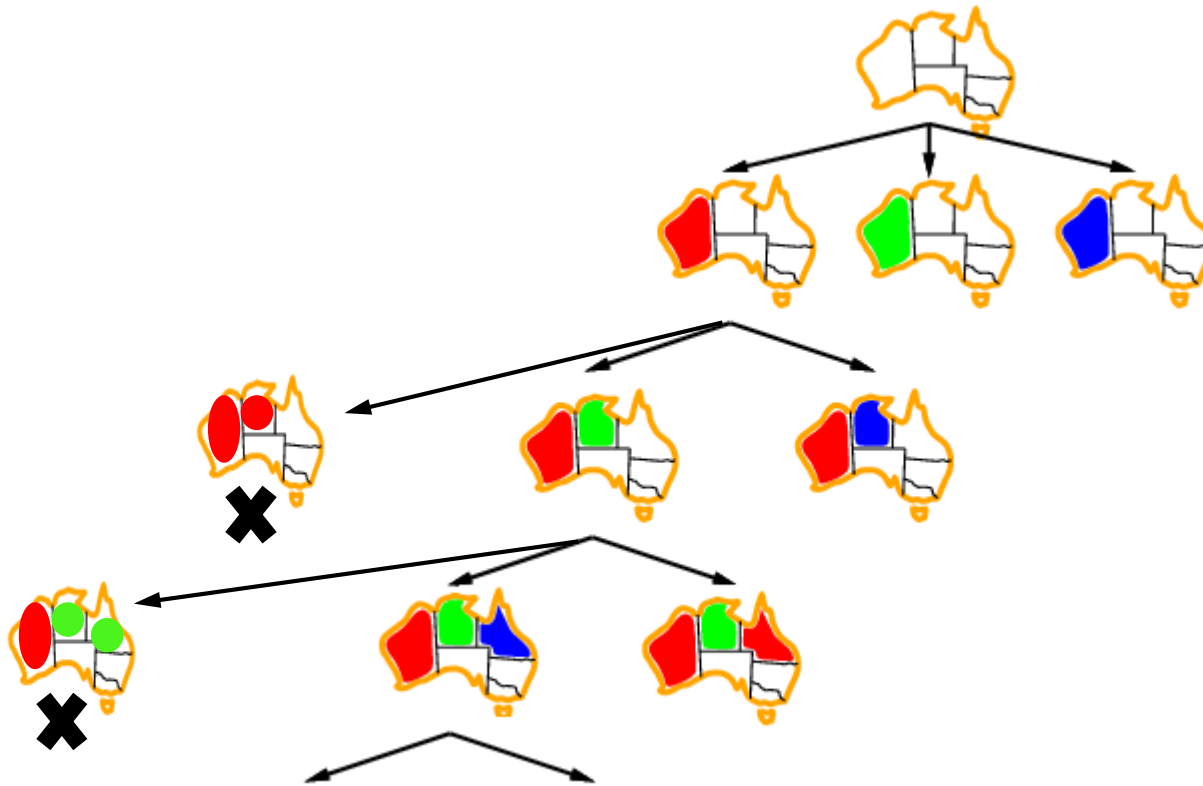
Visszalépéses keresés



Visszalépéses keresés

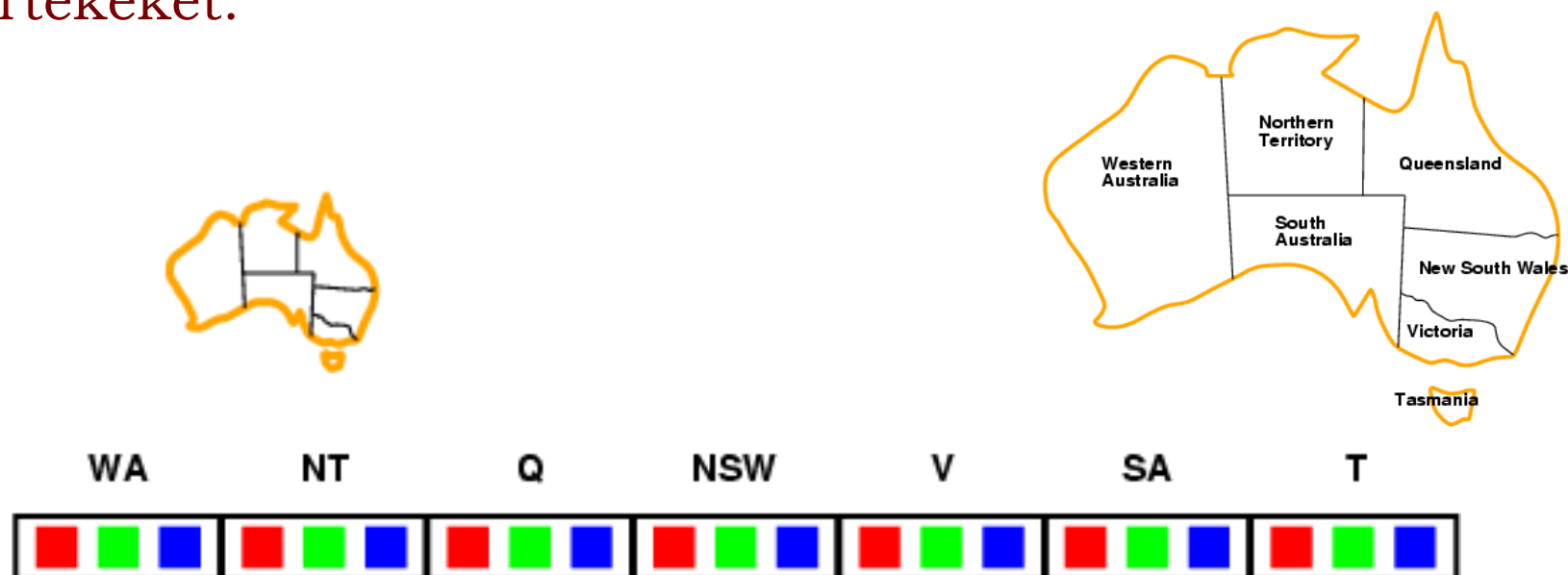


Visszalépéses keresés



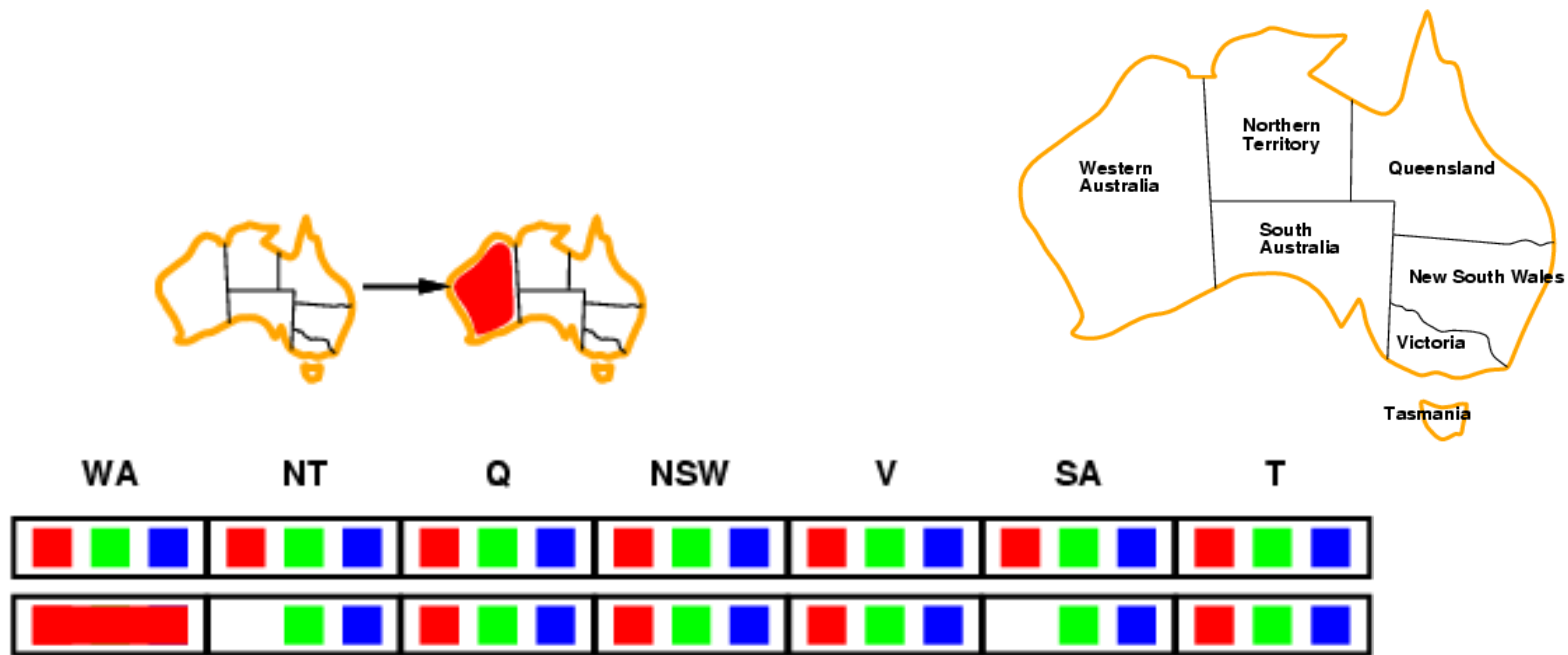
Előrettekintő ellenőrzés (keresés)

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



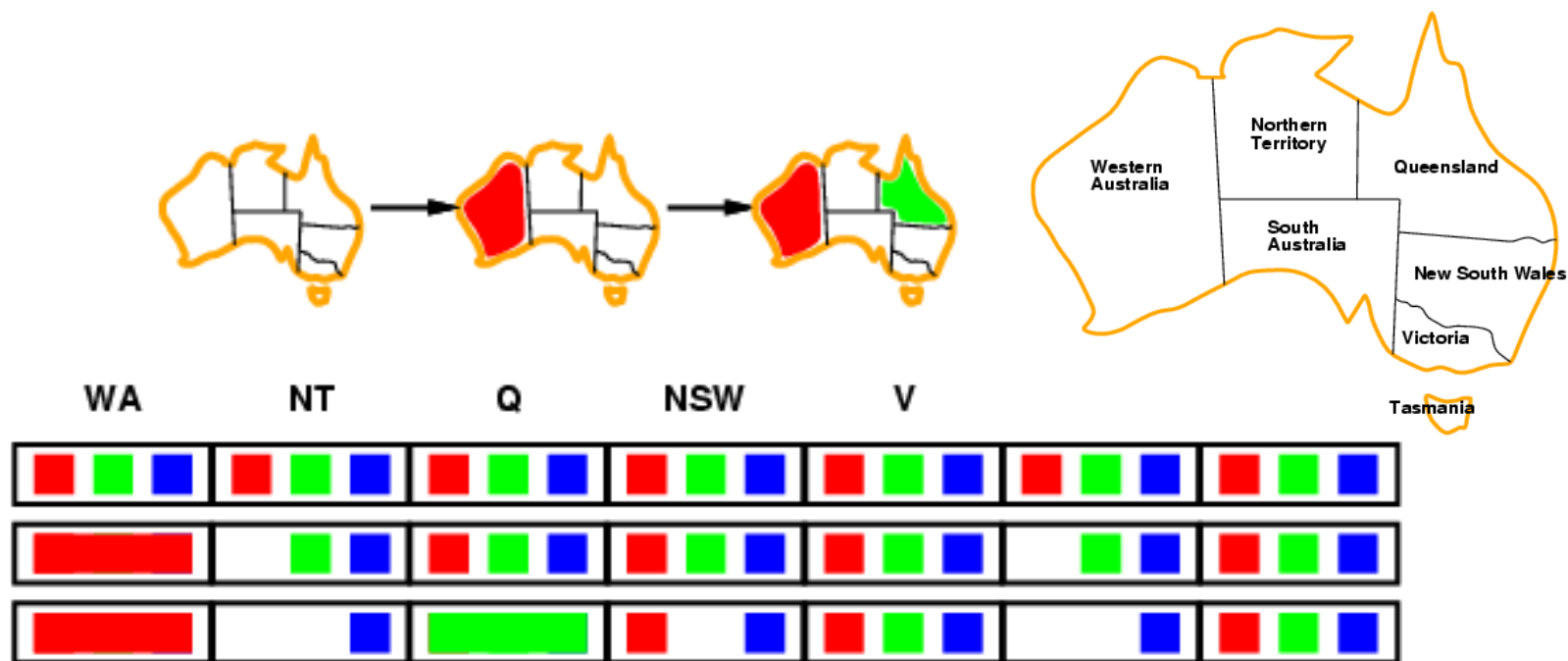
Előretekintő ellenőrzés (keresés)

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



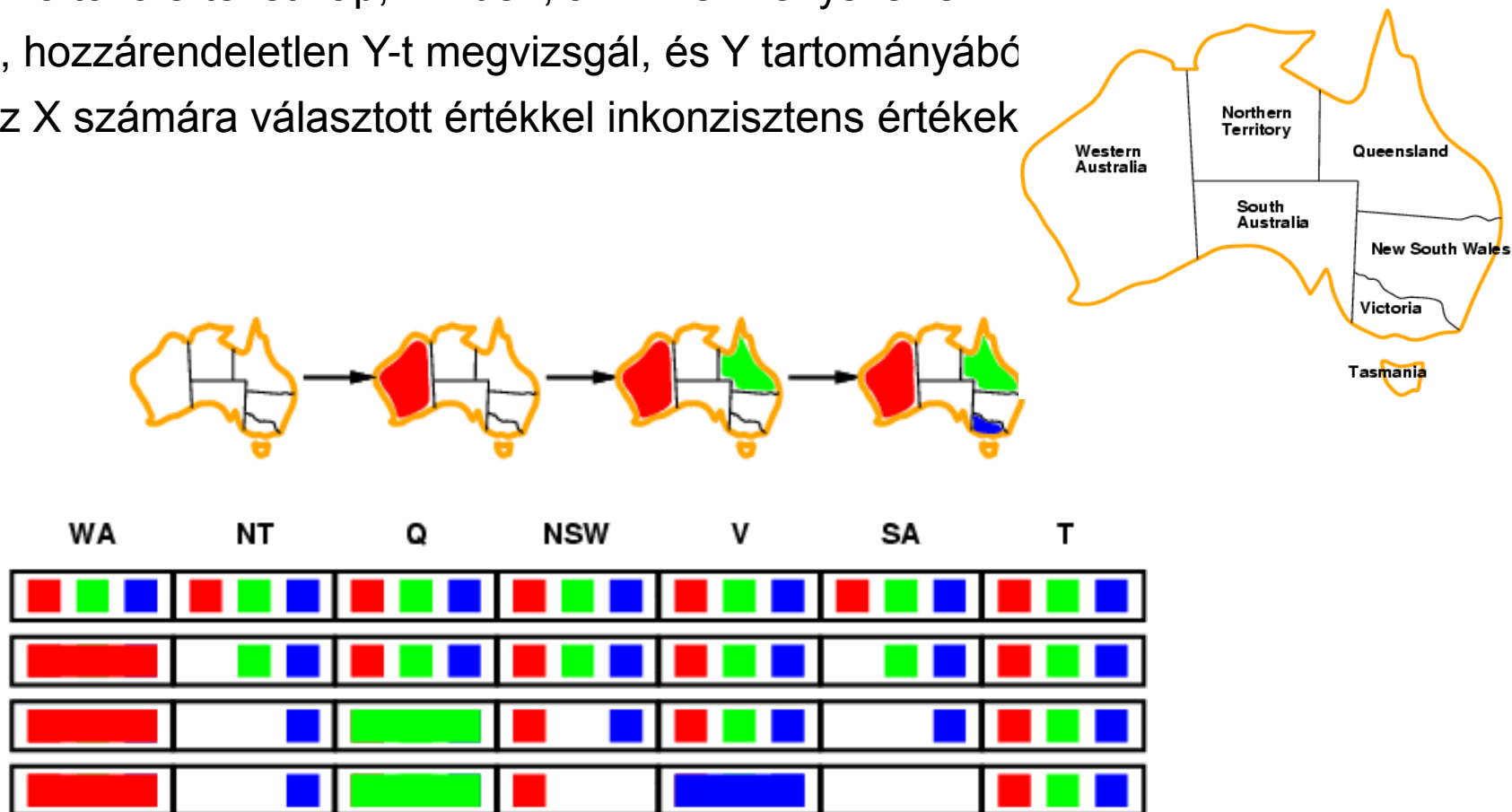
Előrettekintő ellenőrzés (keresés)

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.



Előrettekintő ellenőrzés (keresés)

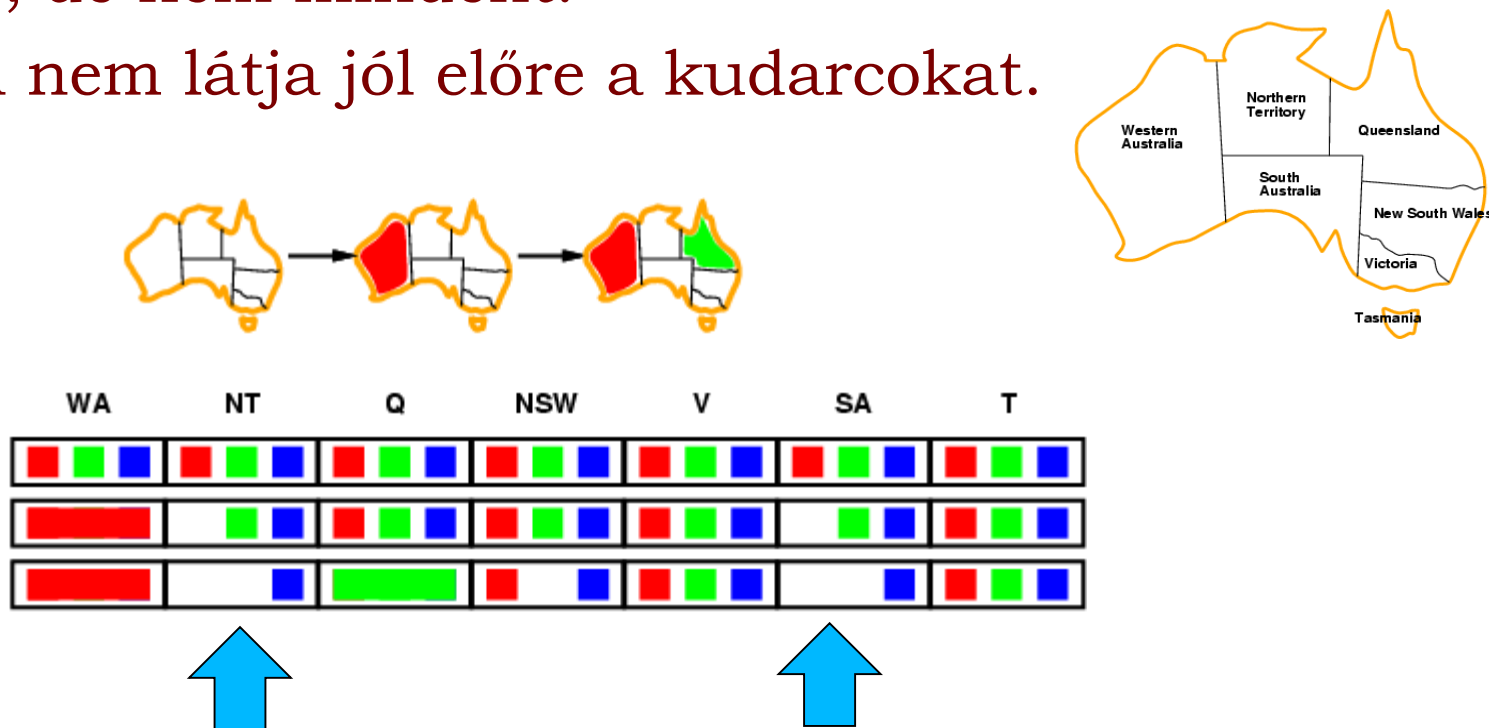
Az előrenéző ellenőrzés minden egyes alkalommal, amikor egy X változó értéket kap, minden, az X-hez kényszerrel kötött, hozzárendeletlen Y-t megvizsgál, és Y tartományából törli az X számára választott értékkel inkonzisztens értékek



Korlátozás előreterjesztése

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

Ráadásul nem látja jól előre a kudarcokat.



NT és SA egyszerre nem lehet kék.

Visszalépéses keresés hatékonyságának növelése (általános heurisztikák CSP-khez)

Általános módszerekkel is komoly gyorsítást el lehet érni:

- Melyik változóval foglalkozzunk a legközelebb?
- Milyen sorrendben vizsgáljuk az értékeit?
- Érzékelhetjük-e jó előre a kudarokat?
(korai nyesés)

ezek az ún. tárgyterület-független heurisztikák



Melyik változóval foglalkozunk először, és milyen értéket rendeljünk hozzá?

Kényszerek:

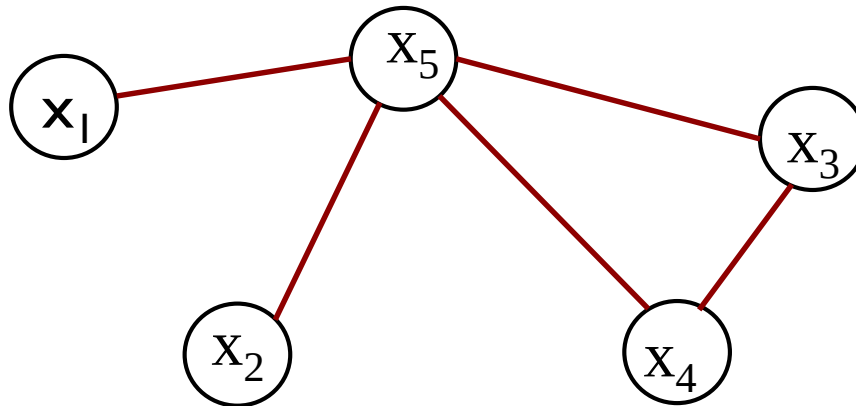
$$x_1 > x_5$$

$$x_2 > 1 + x_5$$

$$x_3 > 2 \cdot x_5$$

$$x_4 > x_5 - 1$$

$$x_3 = 2 + x_4$$



Mindegyik változó értékkészlete az egyjegyű nem negatív egészek halmaza: $\{0, 1, 2, \dots, 9\}$

- A. $x_1 \Leftarrow 5$
- B. $x_2 \Leftarrow 1$
- C. $x_3 \Leftarrow 2$
- D. $x_5 \Leftarrow 0$

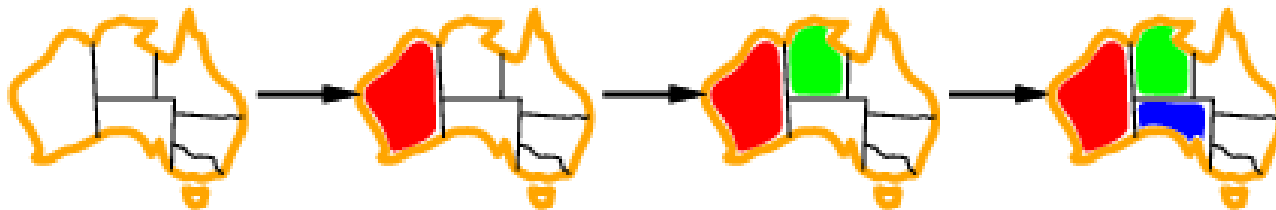


1. A *legkevesebb fennmaradó érték* ötlete (melyik változót válasszuk?)

A leginkább korlátozott változó:

a legkisebb számú megengedett értékkel rendelkező
változóval kezdünk, ill. folytassunk (lokálisan kicsi az
elágazási tényező!)

= **legkevesebb fennmaradó érték** heurisztika
(min remaining variables, MRV)

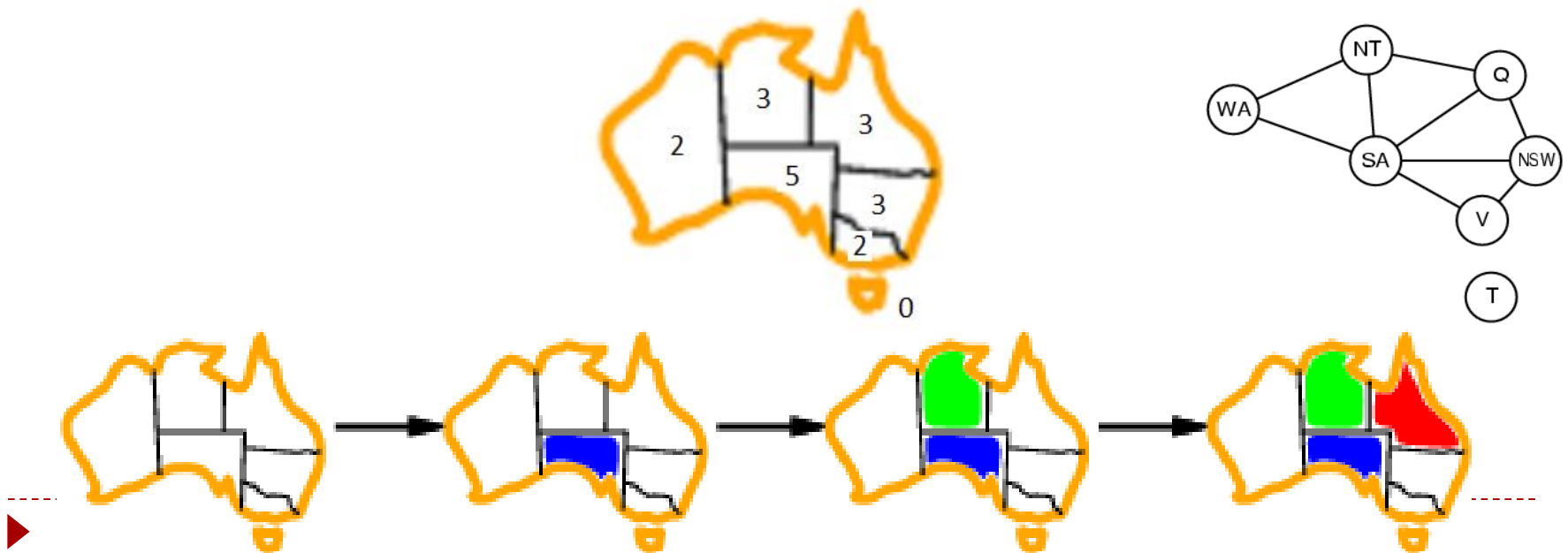


(NT ill. SA csak 2 megengedett érték (piros már nem lehet),
▶ minden más 3)

2. *Fokszám heurisztika* ötlete (melyik változót válasszuk?)

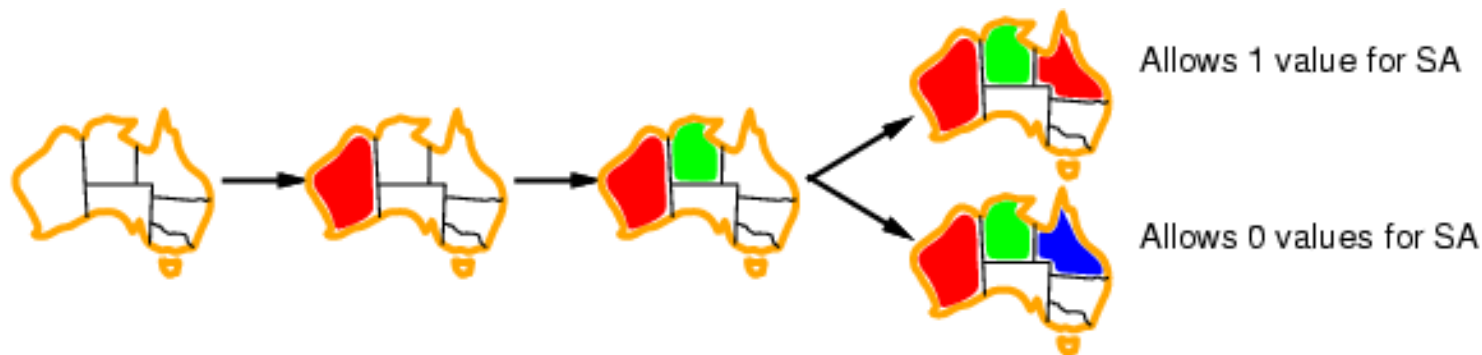
Az MRV-heurisztika semmit sem segít abban, hogy melyik régiót válasszuk ki elsőként Ausztrália kiszínezésekor, mert a kiinduláskor mindegyik régiónak három megengedett színe van.

A *későbbi választások* elágazási tényezőjét csökkentheti, ha azt a változót választjuk ki, amely a legtöbbször szerepel a még hozzárendeletlen változókra vonatkozó kényszerekben.



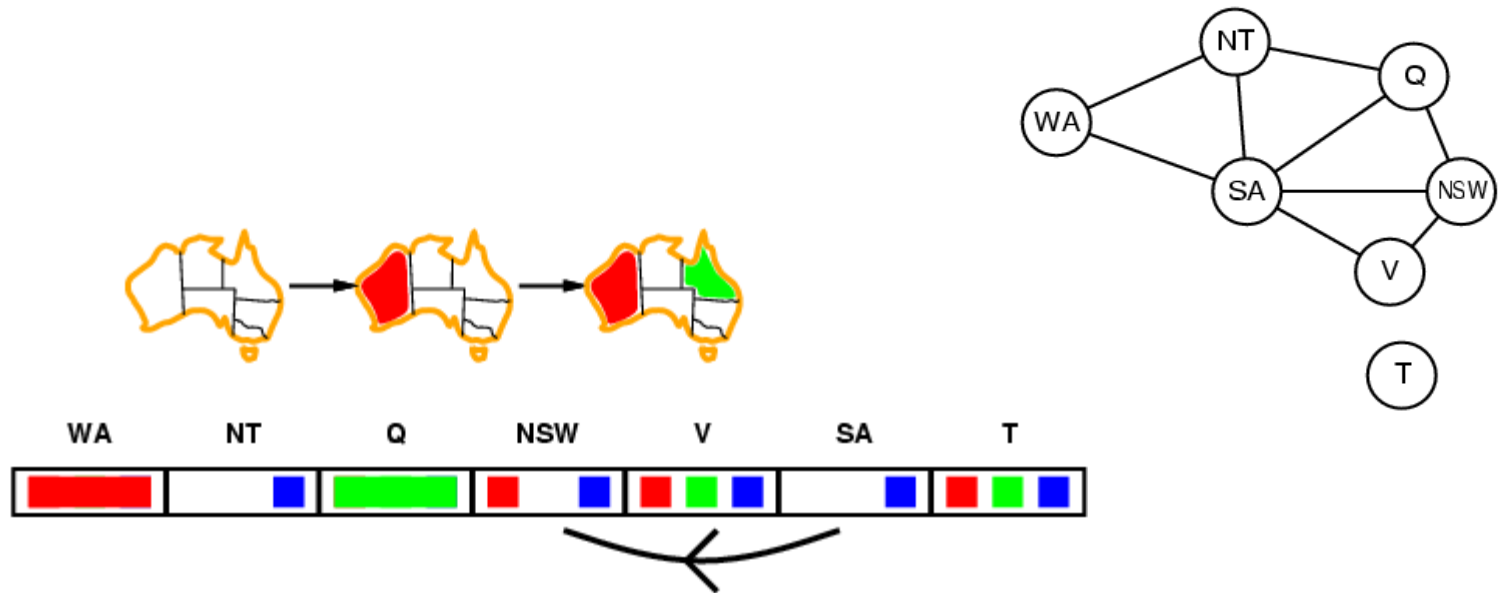
3. A legkevésbé korlátozó érték ötlete

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



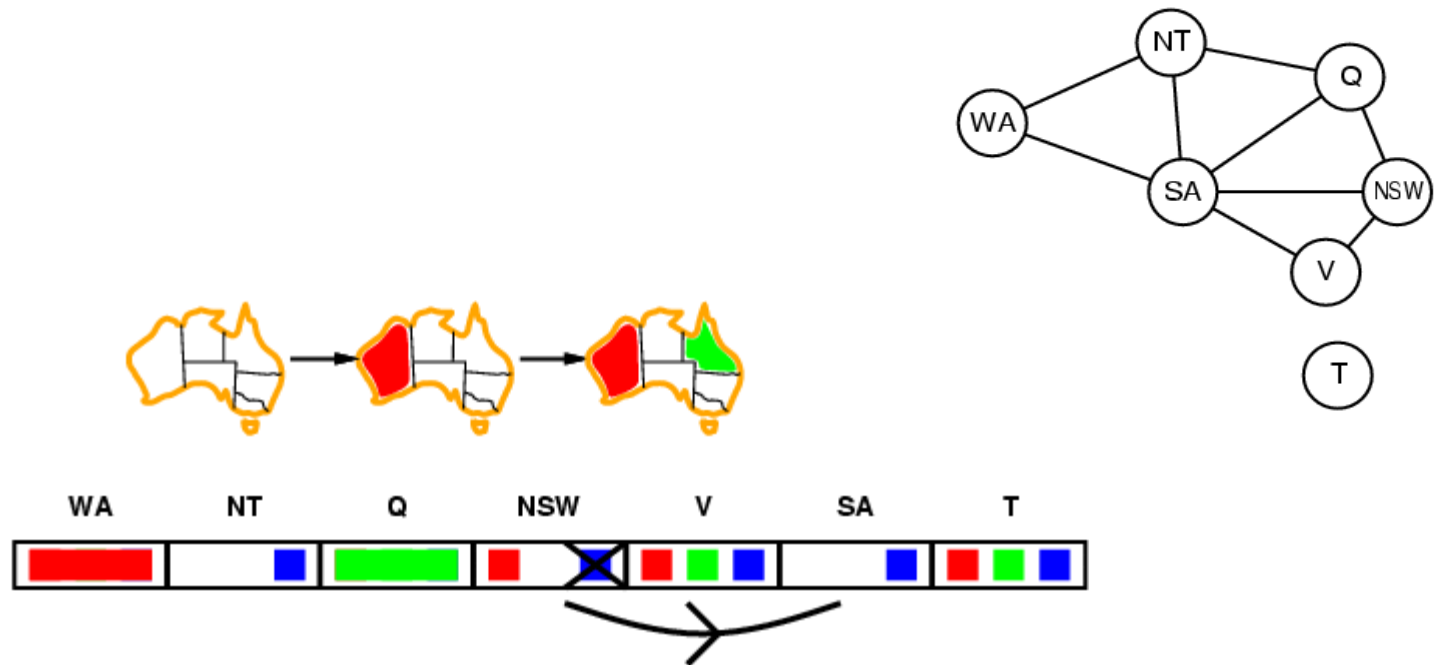
Élkonzisztencia

$X \rightarrow Y$ él konzisztens akkor és csak akkor, ha
 X minden x értékére létezik Y -nak valamilyen megengedett y értéke



Élkonzisztencia

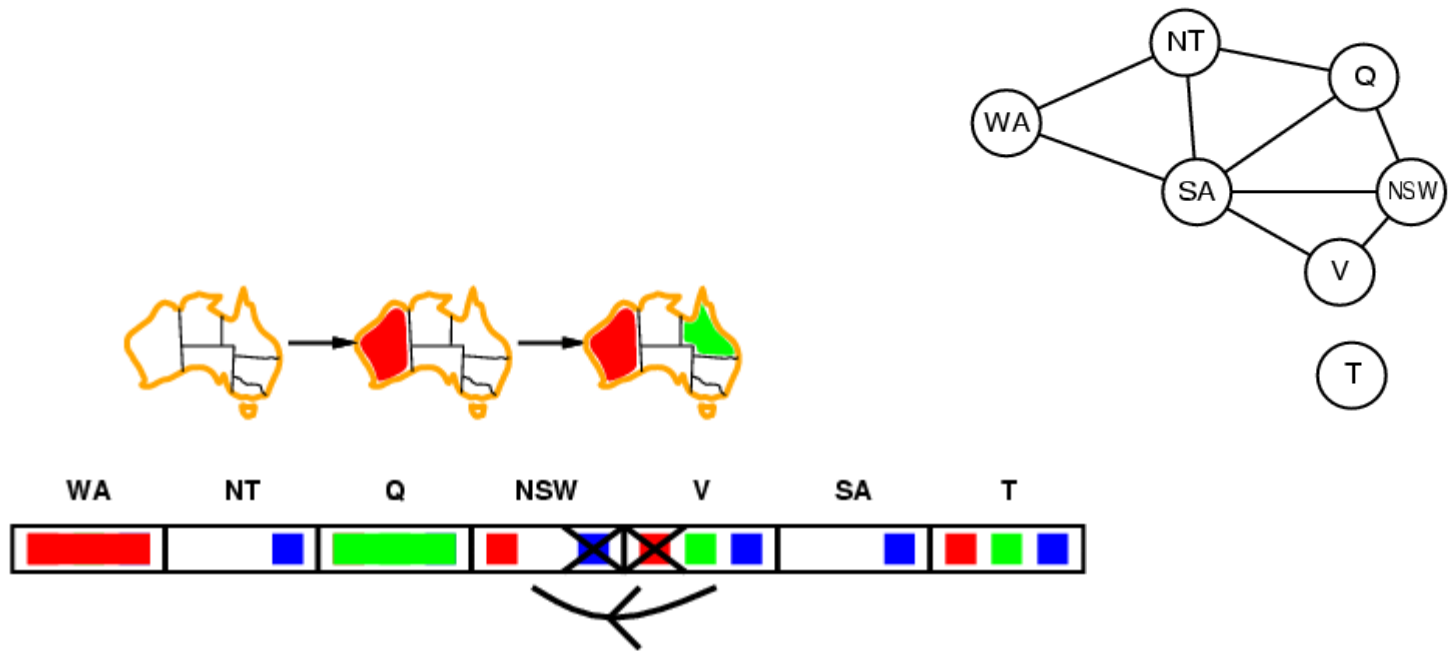
$X \rightarrow Y$ él konzisztens akkor és csak akkor, ha
 X minden x értékére létezik Y -nak valamilyen megengedett y értéke



Élkonzisztencia

$X \rightarrow Y$ éi konzisztens akkor és csak akkor, ha
 X minden x értékére létezik Y -nak valamilyen megengedett y értéke

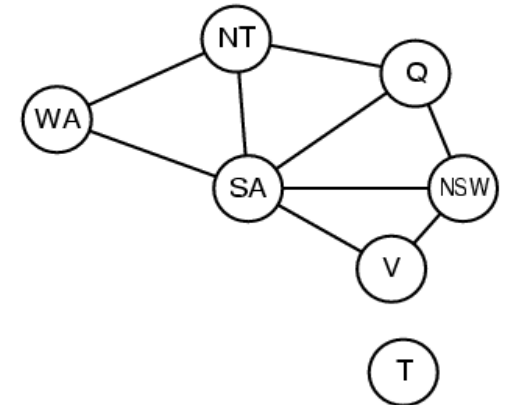
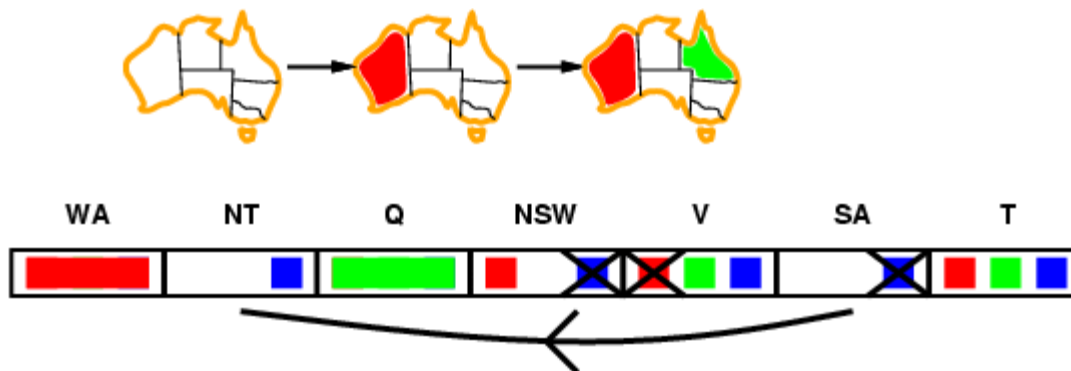
Ha X értéket veszít, a szomszédjait újra kell ellenőrizni



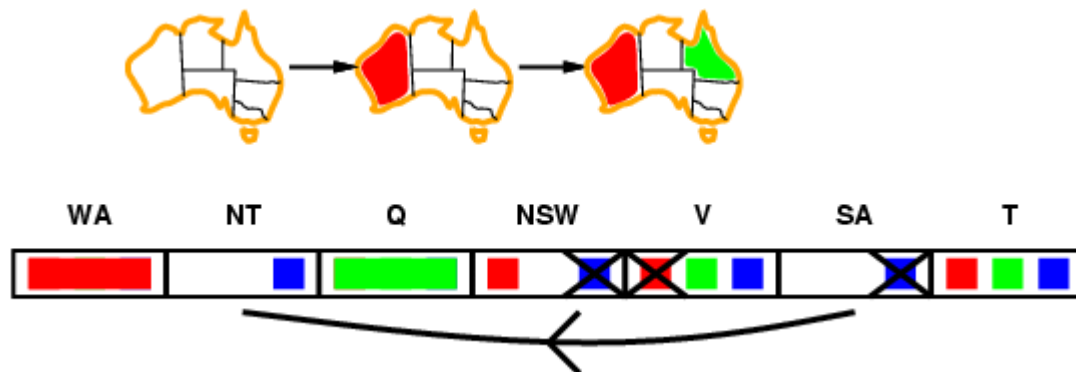
Élkonzisztencia

$X \rightarrow Y$ él konzisztens akkor és csak akkor, ha X minden x_j értékére létezik Y -nak valamilyen megengedett y_k értéke.

Ha X értéket veszít, akkor a szomszédjait (akikkel valamilyen kényszerkapcsolatban van) újra kell ellenőrizni



Élkonzisztencia



Az élkonzisztencia-ellenőrzés alkalmazható:

- **előfeldolgozó lépésként** a keresés megkezdése előtt, vagy a
- keresési folyamat minden egyes hozzárendelését követő **terjesztési lépésként**



CSP lokális kereséssel

Kiindulás: teljes állapotleírás = minden változónak van értéke (esetleg rossz, nem teljesült kényszerekkel)

Operátorok: megváltoztatják a változók hozzárendelését, hogy csökkenjen a sérült kényszerek száma

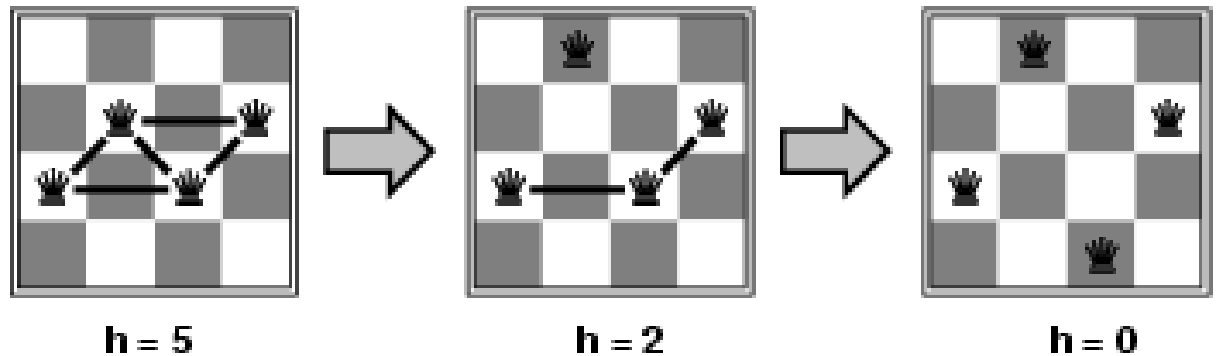
Változó szelekció: véletlen módon, bármely konfliktusban lévő (valamelyik kényszer sérül) változót választhatjuk



CSP lokális kereséssel

Min. konfliktus heurisztika:

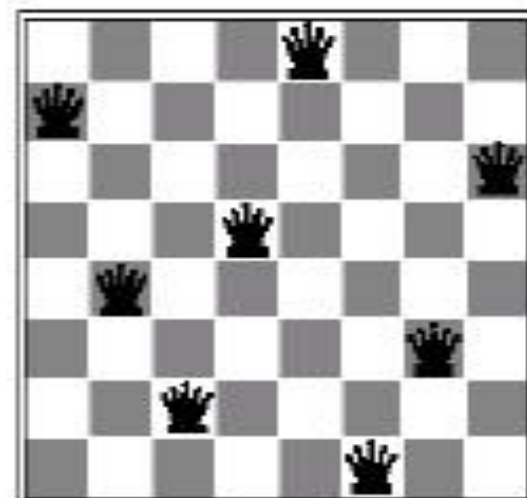
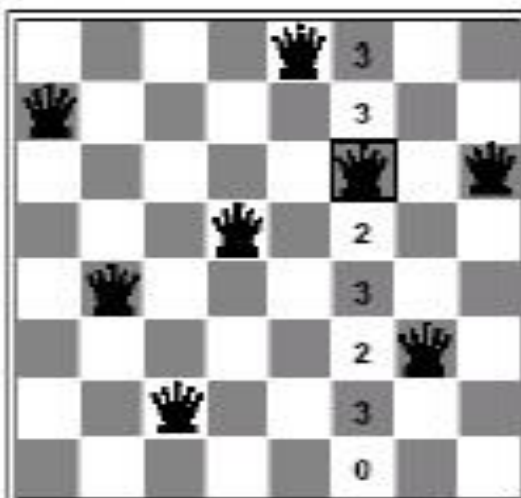
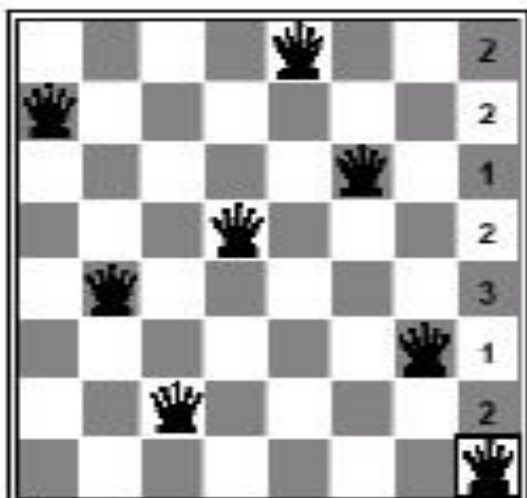
azt az értéket állítjuk be, amely a legkevesebb számú korlátot sérti, pl. hegymászó: $h(n)$ = sérült korlátok száma, $h(n)$ csökkentése a cél (itt most lefele mászunk a völgybe)



$h(n)$ = a támadások száma



CSP lokális kereséssel / Min. konfliktus heurisztika



Min. konfliktus heurisztika nagyon hatékony, nagy valószínűséggel gyorsan old meg nagyon nagy problémaeseteket (10 millió királynő!).

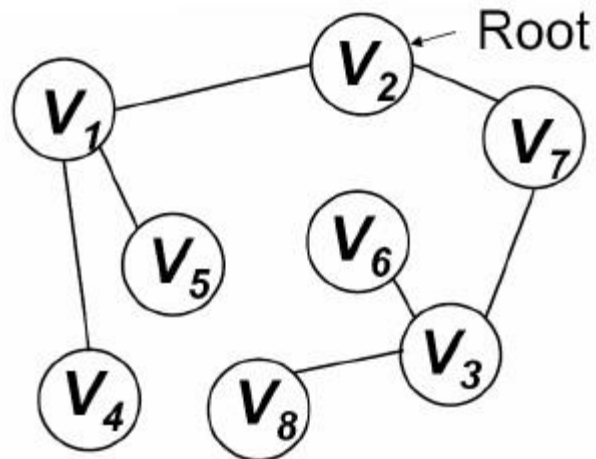


CSP struktúrája – miben segíthet?

CSP gráfja: független komponensek ... (komplexitás-számítás)

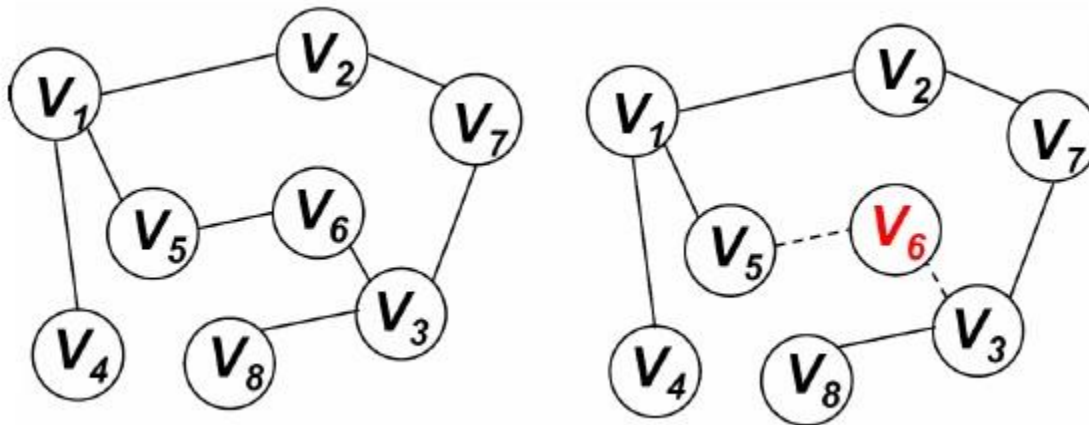
CSP fa gráf: megoldás könnyű, változószámban lineáris

válasszunk egy levelet gyökérnek (lokális keresés kiinduló áll.)
élkonzisztencia-nyesés gyerekektől a szülők felé



CSP struktúrája – miben segíthet?

CSP hurkos gráf: megoldás általánosságban NP nehéz



Gráf CSP konvertálása fába:
vágóhalmaz
fa-dekompozíció

Megoldáskeresés
 V_6 minden értékére

