

Směrová konzistence

Směrová hranová konzistence (pro binární CSP)

- CSP je **směrově hranově konzistentní** vzhledem k uspořádání proměnných (x_1, \dots, x_n) , právě když je každá hrana (x_j, x_i) hranově konzistentní pro každé $j \leq i$.

- procedure

DAC($G, (x_1, \dots, x_n)$) *Directed arc consistency* **DAC**

for $i = n$ to 1 by -1 do

 for $\forall j < i$ takové, že existuje hrana (x_j, x_i) do
 revise $((j, i))$

end DAC

- Příklad: $x_1 = x_2, x_1 = x_3, x_3 = x_4$

● $D_1 = \{\text{red, white, black}\}, D_2 = \{\text{green, white, black}\}, D_3 = \{\text{red, white, blue}\}, D_4 = \{\text{white, blue, black}\}$

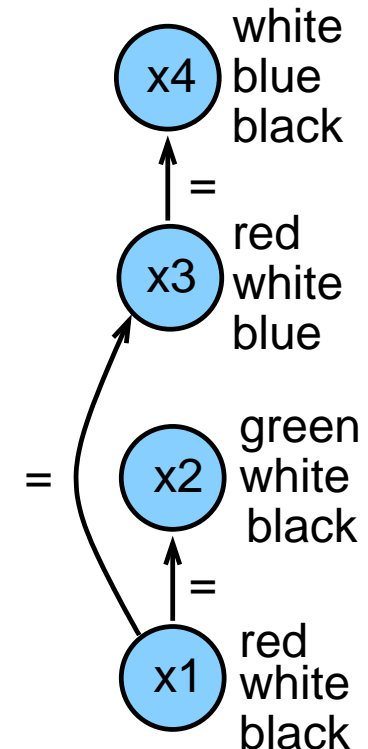
● Po DAC: $D_4 = \{\text{white, blue, black}\}, D_3 = \{\text{white, blue}\}, D_2 = \{\text{green, white, black}\}, D_1 = \{\text{white}\},$

není AC, ale máme řešení bez navracení vzhledem k (x_1, x_2, x_3, x_4)

- Opakování revizí není nutné, doména revidované proměnné se v dalších cyklech nemění

- Složitost $O(ek^2)$

každá hrana se reviduje jednou se složitostí $O(k^2)$



Směrová i -konzistence

- Algoritmus pro DAC byl pouze pro binární CSP
 - i -konzistence vyžaduje uvažování omezení s větším rozsahem proměnných
 - musíme aktualizovat relace až nad $(i - 1)$ proměnnými
- ⇒ uvažujeme obecné CSP (tedy i **nebinární podmínky**)
- CSP je **směrově i -konzistentní** vzhledem k uspořádání proměnných (x_1, \dots, x_n) právě tehdy, když každých $(i - 1)$ proměnných je i -konzistentní vzhledem ke všem proměnným, které je následují v uspořádání.
 - CSP je **silně směrově i -konzistentní (DIC- i)** právě tehdy, když je směrově j -konzistentní pro každé $j \leq i$

Vlastnosti DIC-*i*

● Opakování

- AC = silná 2-konzistence
- PC = silná 3-konzistence

● DIC-2 je ekvivalentní DAC

- u obou uvažujeme unární a binární omezení
- DAC je definován pouze na binární CSP
- DIC-2 je sice definován pro obecné CSP, ale je pouze schopen k dané hodnotě proměnné hledat konzistentní hodnotu v doméně další proměnné, nezachytí tedy omezení s více než dvěma proměnnými

● DIC-3 lze podobně srovnat

se směrovou konzistencí po cestě (directed path consistency, DPC)

● Algoritmus pro silnou směrovou konzistenci

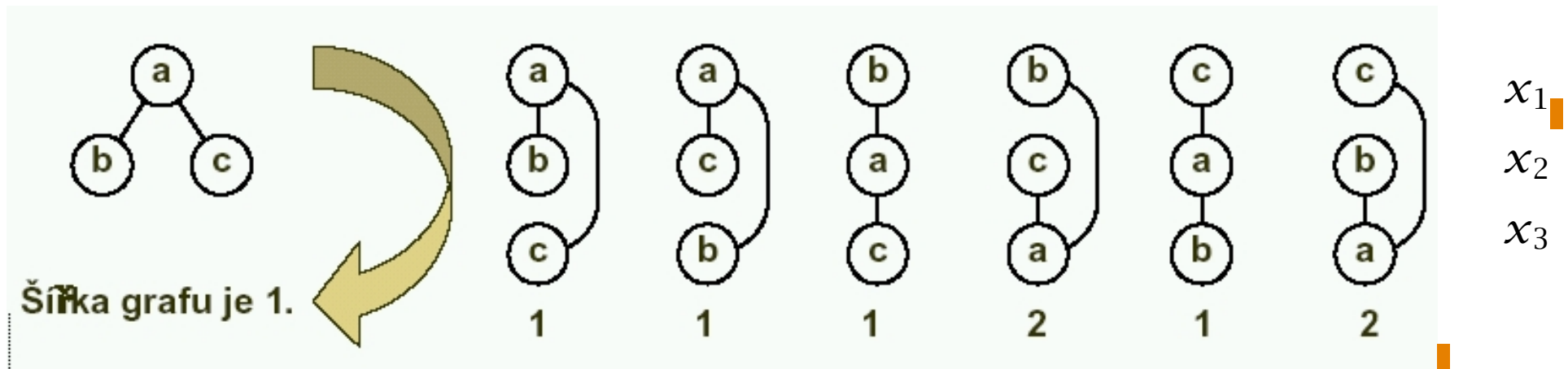
viz [Dechter, Constraint Processing]

Řešení bez navracení pomocí DIC- i

- Opakování:
CSP vyřešíme bez navracení vzhledem k uspořádání proměnných (x_1, \dots, x_n) , jestliže pro každé $i \leq n$ může být každé častečné řešení (x_1, \dots, x_i) konzistentně rozšířeno o proměnnou x_{i+1} . ■
- Proměnná musí být kompatibilní s již ohodnocenými proměnnými, tj. s tolika proměnnými, kolik má „zpětných“ hran ■
- Pro i zpětných hran potřebujeme směrovou $(i + 1)$ -konzistenci
- Je-li m maximum počtu zpětných hran pro všechny vrcholy, stačí nám silná směrovou $(m + 1)$ -konzistence
- Při různém uspořádání vrcholů je počet zpětných hran různý
⇒ hledáme **uspořádání vrcholů s nejmenším počtem zpětných hran m**

Šířka grafu

- **Uspořádaný graf** je graf s lineárním uspořádáním vrcholů.
- **Šířka vrcholu** v uspořádaném grafu je počet hran vedoucích z tohoto vrcholu do předchozích vrcholů.
- **Šířka uspořádaného grafu** je maximum z šířek jeho vrcholů.
- **Šířka grafu** je minimum z šířek všech jeho uspořádaných grafů.



- `procedure MinWidthOrdering((V,E))` (konstruuujeme od konce)
`Q := {}` (do vybraného uzlu povede nejméně zpětných hran)
`while V not empty do N := vyber a smaž uzel s nejmenším počtem hran z (V,E)`
`zařad' N do Q`
`return Q`

Graf podmínek s šířkou 1 = strom

● *Tvrzení:* Graf je strom právě tehdy, když má šířku 1.■

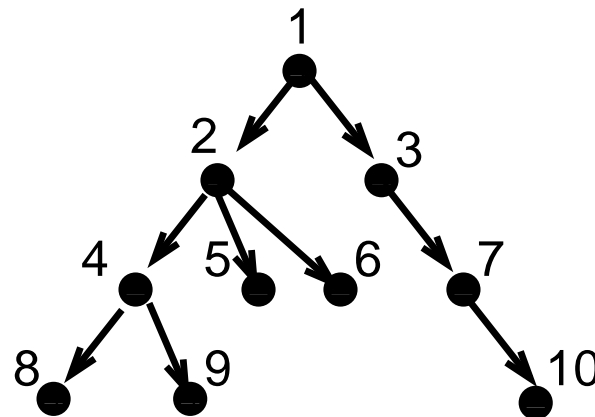
● *Důkaz:*

● Předpoklad: Strom neobsahuje cyklus.■

⇐ Mějme graf šířky 1. Pokud by obsahoval cyklus, tak bychom pro libovolné uspořádání proměnných měli proměnnou se dvěma rodiči, tj. spor. ■

⇒ Mějme graf bez cyklů. Pak lze vytvořit orientovaný strom s kořenem tak, že všechny hrany směřují z kořenového uzlu. ■ V tomto **stromu má každý uzel nejvýše jednoho rodiče.**

Libovolné uspořádání, ve kterém rodič předchází potomky ve stromě, má šířku 1.■



Konzistence pro stromové CSP

● *Tvrzení:* Necht' $d = (x_1, \dots, x_n)$ je uspořádání stromového grafu podmínek T pro daný CSP. Jestliže T je směrově hranově konzistentní vzhledem k d , pak má CSP řešení bez navracení vzhledem k d .■

● *Důkaz:*

● Uvažujme uspořádání proměnných $d = (x_1, \dots, x_n)$ s šířkou 1.

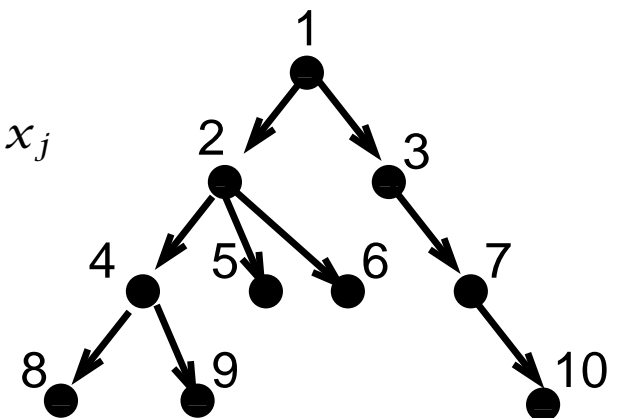
● Předpokládejme, že x_1, \dots, x_i jsou konzistentně nainstanciovány. ■

● Snažíme se nainstanciovat x_{i+1} :

● d je uspořádání šířky 1, tedy existuje pouze jeden rodič x_j ($j \leq i$) proměnné x_{i+1} , který může omezovat x_{i+1} :wq

● (x_j, x_{i+1}) je hranově konzistentní (z DAC), tedy existuje hodnota konzistentní se současným přiřazením x_j

● tuto hodnotu přiřadíme x_{i+1}



Algoritmus zajištění DAC pro stromové CSP

- procedure TreeSolving(T)

načtení uspořádání (x_1, \dots, x_n) s šířkou 1 pomocí stromu T

(rodič předchází potomky)

necht' $x_{p(i)}$ označuje rodiče x_i v uspořádaném stromu

for $i = n$ to 1 by -1 do

 revise($(x_{p(i)}, x_i)$)

 if $D_{p(i)} = \emptyset$ exit (řešení neexistuje)

end TreeSolving

Cvičení: x_1, x_2, x_3, x_4 in $1..5$, $x_1 = x_2 + 1$, $x_1 < x_3$, $x_3 < x_4$

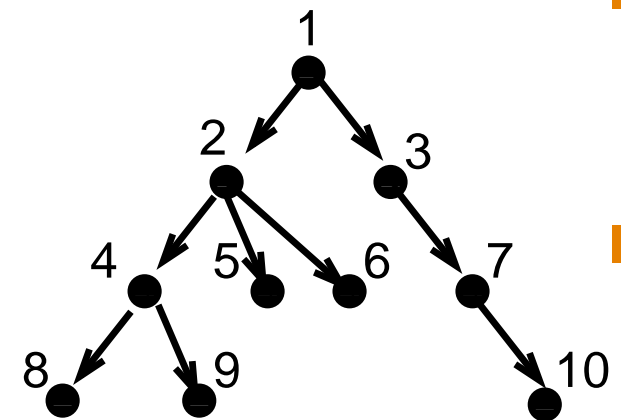
⇒ Složitost algoritmu $O(nk^2)$

- Algoritmus zajistí DAC pro stromové CSP,

tj. bude možné nalézt řešení bez navracení

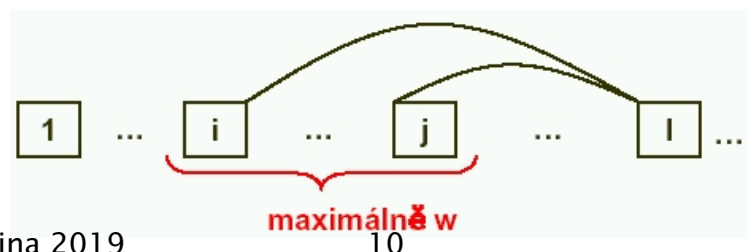
- Pokud aplikujeme DAC vzhledem k uspořádání šířky 1, a pak v opačném směru, tak dosáhneme plné hranové konzistence.

viz Barták, přednáška



Šířka grafu a stupeň konzistence

- *Tvrzení:* Necht' je dán CSP, jehož uspořádaný graf podmínek s uspořádáním d má šířku $i - 1$. Jestliže je problém silně směrově i -konzistentní, pak je problém řešitelný bez navracení vzhledem k d .■
- *Důkaz:*
 - existuje uspořádaný graf s uspořádáním d a šířkou $i - 1$
 - speciálně, počet zpětných hran pro každou proměnnou je maximálně $i - 1$
 - proměnné ohodnocujeme v pořadí uspořádání grafu
 - nyní, pokud ohodnocujeme proměnnou:
 - musíme najít hodnotu kompatibilní se všemi již ohodnocenými proměnnými, které jsou s proměnnou spojené podmínkou (hranou)
 - necht' takových proměnných je m , potom $m \leq (i - 1)$ (\Leftrightarrow graf má šířku $(i - 1)$)
 - graf je směrově i -konzistentní, tedy taková hodnota musí existovat (\Leftrightarrow z definice)



Adaptivní konzistence

- *Původní intuice:* proměnná musí být kompatibilní s již ohodnocenými proměnnými, tj. s tolika proměnnými, kolik má „zpětných“ hran.
 - Je tedy nutná směrová i -konzistence odpovídající šířce grafu?
- *Problém:* algoritmy silné směrové i -konzistence pro $i \geq 3$ mění graf podmínek přidáváním hran, nutná úroveň i se může zvětšovat
- Algoritmus **adaptivní konzistence ADC** pro nalezení řešení bez navracení
 - uvažujme uspořádání proměnných d
 - rekurzivně vytváříme směrovou i -konzistenci (od poslední k první proměnné v d)
 - měníme úroveň i od uzlu k uzlu tak, abychom se **adaptovali aktuální šířce uzlu** v momentě zpracování
- Proč algoritmus funguje?
 - v momentě zpracování uzlu je určena finální šířka uzlu
 - víme tedy, jakou úroveň směrové i -konzistence musíme dosáhnout

Polynomiální CSP

- w šířka grafu, n počet proměnných, k horní mez velikosti domén
- Složitost DIC- i $O(nw^i \cdot (2k)^i)$ důkaz viz Dechter, Constraint Processing
- Časová a prostorová složitost algoritmu adaptivní konzistence je $O(n \cdot (2k)^{w+1})$ a $O(n \cdot k^w)$ důkaz viz Dechter, Constraint Processing
- **Věta:** Třída CSP problémů, jejichž šířka grafu je ohraničena konstantou b je řešitelná v polynomiálním čase a prostoru.
- Použitelnost algoritmu ADC
 - ADC není jen procedura pro rozhodnutí konzistence
 - ADC může sloužit i ke kompilaci
 - ADC transformuje problém na ekvivalentní graf, ze kterého může být každé řešení odvozeno v lineárním čase
 - prostorová složitost ale zůstává

Obecná hranová konzistence, konzistence mezí

Pojmy a značení

- **Rozsah omezení** $scope(c)$

množina proměnných, na kterých je c definováno

- příklad: $A, B, C \in \{0, 1, 2\}$

$$scope(A < B) = \{A, B\}$$

- **k-tice** \vec{t} hodnot patřících do c : $\vec{t} \in c$

- příklad (pokračování): $(0, 1) \in (A < B)$

- Proměnná $x \in scope(c)$, k-tice $\vec{t} \in c$

$\vec{t}[x]$ je hodnota proměnné x v \vec{t}

- příklad (pokračování): $(0, 1)[A] = 0$

Definice obecné hranové konzistence (GAC)

- **Generalized arc consistency** (někdy nazývána **doménová konzistence**)
 - pro každou proměnnou z podmínky a každou její hodnotu existuje ohodnocení zbylých proměnných v podmínce tak, že podmínka platí
 - $A + B = C$, $A \in \{1, 2, 3\}$, $B \in \{2, 3, 4\}$, $C \in \{3, \dots, 7\}$ je obecně hranově konzistentní
- **Omezení c je obecně hranově konzistentní**, jestliže má každá hodnota a každé proměnné $x \in scope(c)$ doménovou podporu v c
- Hodnota a proměnné $x \in scope(c)$ má **doménovou podporu** \vec{t} v c , jestliže
 - $\vec{t} \in c$ a platí $a = \vec{t}[x]$
 - pro každé $y \in scope(c)$ platí $\vec{t}[y] \in D_y$
- Příklad: $A \in \{0, 1\}$, $B \in \{0, 1\}$, $C \in \{1, 2\}$, $A = B + C$,
0 u A nemá podporu, 1 u A má podporu $(1, 0, 1)$
- **CSP je obecně hranově konzistentní** \iff
všechna jeho omezení jsou obecně hranově konzistentní
- Příklad (pokračování): po GAC dostaneme $A=1$, $B=0$, $C=1$

Konzistence mezí

- **Bounds consistency BC**: slabší než obecná hranová konzistence
 - propagace pouze při **změně minimální nebo maximální hodnoty** v doméně proměnné
 - tedy došlo ke **změně mezí domény proměnné**

Konzistence mezí pro nerovnice

- $A > B \Rightarrow \min(A) \geq \min(B)+1, \max(B) \leq \max(A)-1$
- příklad: $A \in \{4, \dots, 10\}, B \in \{6, \dots, 18\}, A > B$
 $\min(A) \geq 6+1 \Rightarrow A \in \{7, \dots, 10\}$
 $\max(B) \leq 10-1 \Rightarrow B \in \{6, \dots, 9\}$

Cvičení: napište pravidla pro konzistenci mezí pro uvedená omezení

- $A < B, A \geq B, A \leq B$
- a aplikujte je v případě domén $A \in \{1, \dots, 10\}, B \in \{0, \dots, 5\}$

Jak je to tedy pro nebinární omezení?

Konzistence mezi a aritmetická omezení

● $A = B + C \Rightarrow \min(A) \geq \min(B) + \min(C), \max(A) \leq \max(B) + \max(C)$
 $\min(B) \geq \min(A) - \max(C), \max(B) \leq \max(A) - \min(C)$
 $\min(C) \geq \min(A) - \max(B), \max(C) \leq \max(A) - \min(B)$ ■

● změna $\min(A)$ vyvolá pouze změnu $\min(B)$ a $\min(C)$

● změna $\max(A)$ vyvolá pouze změnu $\max(B)$ a $\max(C)$, ... ■

● Příklad: $A \in \{1, \dots, 6, 9, 10\}, B \in \{1, \dots, 10\}, A = B + 2$ ■

$A = B + 2 \Rightarrow \min(A) \geq 1 + 2, \max(A) \leq 10 + 2 \Rightarrow A \in \{3, \dots, 6, 9, 10\}$

$\Rightarrow \min(B) \geq 1 - 2, \max(B) \leq 10 - 2 \Rightarrow B \in \{1, \dots, 8\}$ ■

tj. doména B má pouze změněny meze a hodnoty 5, 6 zůstanou v doméně

$A \in \{3, \dots, 6, 9, 10\}, B \in \{1, \dots, 8\}, A = B + 2$ je BC, není GAC, není AC ■

● **Cvičení:** napište pravidla pro konzistenci mezí pro uvedená omezení

● $A = B - 3, A = B - C, A = B + C, A < B + C,$

● a aplikujte je v případě domén $A \in \{1, \dots, 10\}, B \in \{0, \dots, 5\}, C \in \{2, 3, 4\}$

Definice konzistence mezí

- Hodnota a proměnné $x \in scope(c)$ má **intervalovou podporu** \vec{t} v c , jestliže
 - $\vec{t} \in c$ a platí $a = \vec{t}[x]$
 - příklad: $a = 1, \vec{t} = (1, 5, 7)$ ■
 - pro každé $y \in scope(c)$ platí $\vec{t}[y] \in [min(D_y), max(D_y)]$
 - př. (pokrač.) $y = 5, z = 7, D_y = D_z = \{1, 2, 4, 5, 10\}$ ■
 $5 = \vec{t}[y]$ i $7 = \vec{t}[z]$ mají intervalovou podporu ■
- Srovnání: doménová podpora GAC
 - pro každé $y \in scope(c)$ platí $\vec{t}[y] \in D_y$ ■
 - př. (pokrač.) $7 = \vec{t}[z]$ nemá doménovou podporu ■
- **Omezení má konzistentní meze**, jestliže každá hodnota a proměnné $x \in scope(c)$ má intervalovou podporu v c
- **CSP má konzistentní meze** \iff všechna jeho omezení mají konzistentní meze