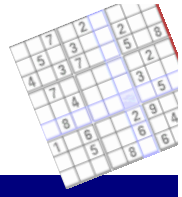


Programování
s omezujícími podmínkami

Roman Barták
Katedra teoretické informatiky a matematické logiky

`roman.bartak@mff.cuni.cz`
`http://ktiml.mff.cuni.cz/~bartak`

Co je to Sudoku?

- **Logická hádanka,** jejímž cílem je doplnit chybějící čísla 1-9 do tabulky 9×9 tak, aby se čísla neopakovala v žádném řádku, sloupci ani v malých čtvercích 3×3.

9	6	3	1	7	4	2	5	8
1	7	8	3	2	5	6	4	9
2	5	4	6	8	9	7	3	1
8	2	1	4	3	7	5	9	6
4	9	6	8	5	2	3	1	7
7	3	5	9	6	1	8	2	4
5	8	9	7	1	3	4	6	2
3	1	7	2	4	6	9	8	5
6	4	2	5	9	8	1	7	3

Trochu historie

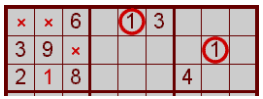
1979: poprvé publikováno v New Yorku
„Number Place" – „Umísti číslice"

1986: populární v Japonsku
Sudoku – zkráceno z japonštiny "**Sú**dži wa **doku**šin ni kagiru"
"Čísla musí být jedinečná"

2005: populární mezinárodně

Začínáme se Sudoku

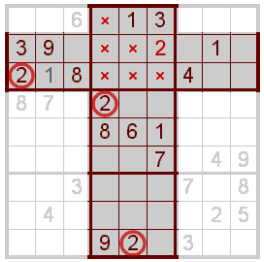
Jak poznáme, kam které číslo patří?



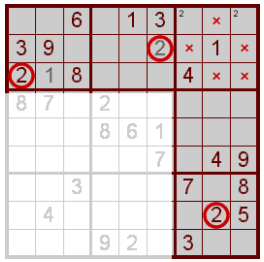
- Využijeme informace, že každé číslo je v řádku maximálně jedenkrát.

A co když to nestačí?

- Podíváme se do sloupců resp. zkombinujeme informaci ze sloupců a řádků.

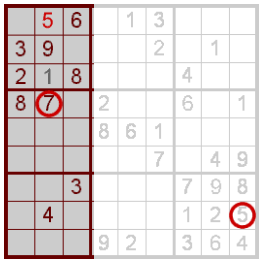


Sudoku o krok dál



- Pokud řádky ani sloupce neposkytnout přesnou informaci, alespoň si můžeme poznamenat, kde dané číslo může ležet.

- Poloha čísla může být odvozena i z přítomnosti jiných čísel a z vlastnosti, že každé číslo se musí objevit alespoň jednou.



Sudoku obecně

Na každé políčko se podíváme jako na **proměnnou** nabývající hodnoty z **domény** {1,…,9}.

Mezi všemi dvojicemi proměnných v řádku, sloupci či malém čtverci je **podmínka** nerovnosti.

Takto zadaný problém se nazývá **problém splňování podmínek.**

O čem bude přednáška?

Algoritmy splňování podmínek

- Algoritmy lokálního prohledávání
  - HC, MC, RW, Tabu, GSAT, Genet
- Prohledávací algoritmy
  - GT, BT, BJ, BM, DB, LDS
- Konzistenční (filtrační) techniky
  - NC, AC, DAC, PC, DPC, RPC, SC
- Konzistenční techniky v prohledávání
  - FC, PLA, LA
- Řešení optimalizačních problémů
  - B&B
- Řešení příliš omezených problémů
  - PCSP, ProbCSP, FuzzyCSP, VCSP, SCSP, hierarchie podmínek

Modelování reálných úloh

- tipy a triky, programování v CLP

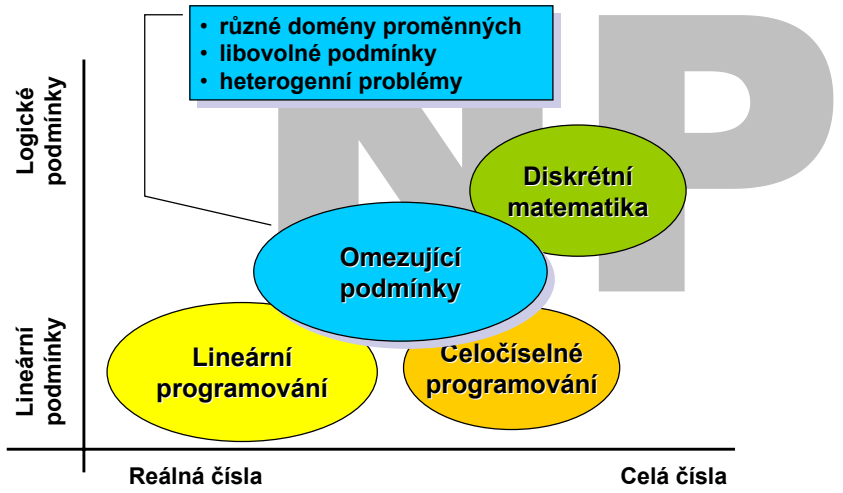
Zdroje a literatura

- Knížky
  - **P. Van Hentenryck:** Constraint Satisfaction in Logic Programming**, MIT Press, 1989**
  - **E. Tsang:** Foundations of Constraint Satisfaction**, Academic Press, 1993**
  - **K. Marriott, P.J. Stuckey:** Programming with Constraints: An Introduction**, MIT Press, 1998**
  - **R. Dechter:** Constraint Processing**, Morgan Kaufmann, 2003**
  - Handbook of Constraint Programming**, Elsevier, 2006**
- Časopisy
  - Constraints, An International Journal. **Springer Verlag**
  - Constraint Programming Letters**, free electronic journal**
- On-line zdroje
  - Web přednášky **(slajdy)** **http://ktiml.mff.cuni.cz/~bartak/podminky/**
  - On-line Guide to Constraint Programming **(tutorial)** **http://ktiml.mff.cuni.cz/~bartak/constraints/**
  - Constraints Archive **(archive and links)** **http://4c.ucc.ie/web/archive/index.jsp**
  - Constraint Programming online **(community web)** **http://www.cp-online.org/**

Pohled do historie

- **Umělá inteligence**
  - Ohodnocování scény (Waltz 1975)
  - Jak pomoci prohledávání?
- **Interaktivní grafika**
  - Sketchpad (Sutherland 1963)
  - ThingLab (Borning 1981)
- **Logické programování**
  - unifikace → řešení podmínek (Gallaire 1985, Jaffar, Lassez 1987)
- **Operační výzkum a diskrétní matematika**
  - NP-těžké kombinatorické problémy

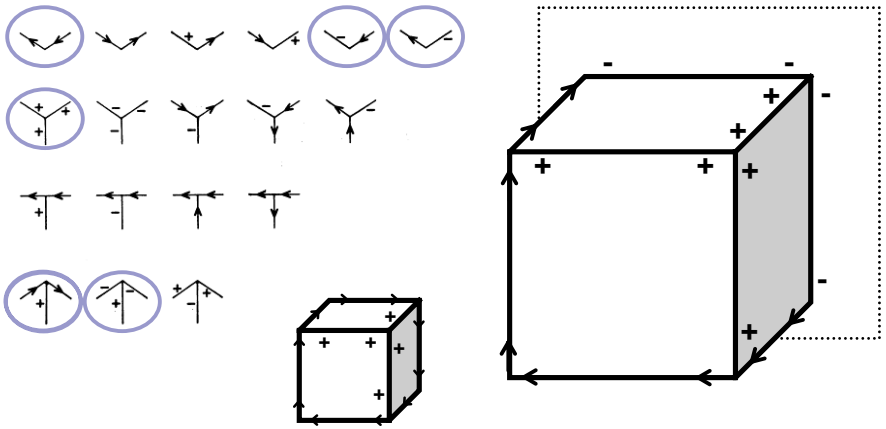
Vztah k ostatním



Ohodnocování scény

3D interpretace čar v 2D obrázku

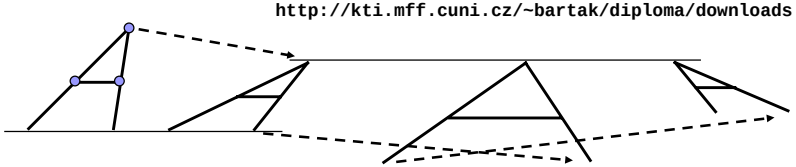
- rozdělení na konvexní (+), konkávní (-) a okrajové (←) hrany
- hledáme ohodnocení hran, které je fyzikálně možné

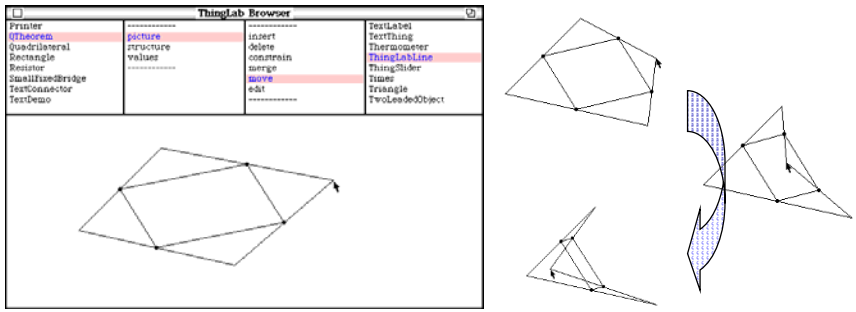


Interaktivní grafika

manipulace s geometrickými objekty popsanými podmínkami

http://kti.mff.cuni.cz/~bartak/diploma/downloads.html

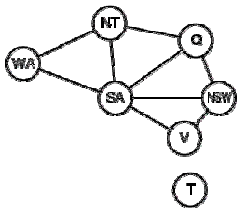




http://www.cs.washington.edu/research/constraints/

Barvení grafů

- Najděte obarvení států (červená, modrá, zelená) takové, že sousední státy nemají stejnou barvu.



- **Reprezentace formou CSP**
  - proměnné: {WA, NT, Q, NSW, V, SA, T}
  - superdoména: {č, m, z}
  - podmínky: WA ≠ NT, WA ≠ SA atd.
- Lze také popsat formou **sítě podmínek** (vrcholy=proměnné, hrany=podmínky)

- **Řešení problému**
  WA = č, NT = z, Q = č, NSW = z,
  V = č, SA = m, T = z

Algebrogramy

Demo

Přiřaďte cifry 0,…,9 písmenům S,E,N,D,M,O,R,Y tak, aby platilo:

- SEND + MORE = MONEY
- různá písmena mají přiřazena různé cifry
- S a M nejsou 0

Model 1:

```
E,N,D,O,R,Y::0..9, S,M::1..9
              1000*S + 100*E + 10*N + D
+             1000*M + 100*O + 10*R + E
= 10000*M + 1000*O + 100*N + 10*E + Y
```

Model 2:

použití „přenosových" 0-1 proměnných

```
E,N,D,O,R,Y::0..9, S,M::1..9, P1,P2,P3::0..1
   D+E = 10*P1+Y
P1+N+R = 10*P2+E
P2+E+O = 10*P3+N
P3+S+M = 10*M +O
```

```
all_different(S,E,N,D,M,O,R,Y)
```


Problém N královen

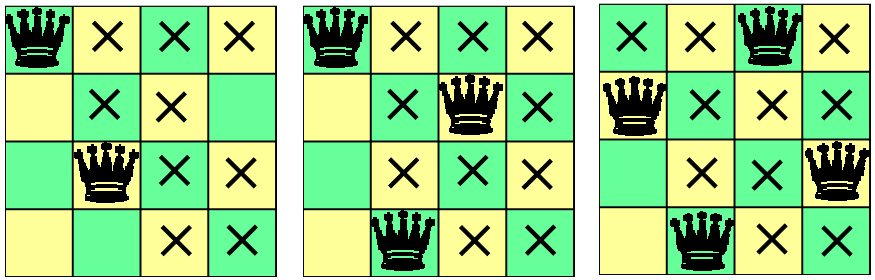
Rozmístěte N královen na šachovnici N×N tak, aby se neohrožovaly

modelové rozhodnutí: královny jsou v různých sloupcích

proměnné: N proměnných r(i) s doménou {1,…,N}

podmínky: žádné dvě královny se neohrožují

$\forall i \neq j \quad r(i) \neq r(j) \;\&\; |i-j| \neq |r(i)-r(j)|$



To si jenom hrajete?

Bioinformatika

- sekvencování DNA (Celera Genomics)
- určení 3D struktury proteinů z posloupnosti aminokyselin

Plánování a rozvrhování

- autonomní plánování aktivit kosmické lodi (Deep Space 1)
- rozvrhování výroby

Problém splňování podmínek

Problém splňování podmínek (CSP – Constraint Satisfaction Problem) se skládá z:

- konečné množiny **proměnných**
  - popisují „atributy" řešení, o kterých je potřeba rozhodnout například pozici dámy na šachovnici
- **domén** – konečné množiny hodnot pro proměnné
  - popisují možnosti, které máme při rozhodování např. čísla řádků pro umístění dámy
  - někdy se definuje jedna superdoména společná pro všechny proměnné a individuální domény se v ní vytyčí přes unární podmínky
- konečné množiny **podmínek**
  - podmínka je libovolná relace nad množinou proměnných například poziceA ≠ poziceB
  - může být definována extenzionálně (jako množina kompatibilních n-tic) nebo intenzionálně (formulí)

Řešení CSP

Přípustné řešení problému splňování podmínek je úplné konzistentní ohodnocení proměnných.

- **úplné** = každé proměnné přiřazena hodnota z domény
- **konzistentní** = všechny podmínky jsou splněny

Někdy se hledají všechna přípustná řešení (nebo se zjišťuje jejich počet).

Optimální řešení problému splňování podmínek je přípustné řešení, které minimalizuje/maximalizuje hodnotu objektivní funkce.

- **objektivní funkce** = funkce z množiny přípustných řešení do reálných čísel

Vlastnosti podmínek

- vyjadřují **částečnou informaci**
  - X je větší než 3, hodnota X není určena jednoznačně
- poskytují **lokální pohled** na problém
  - svazují vždy jen několik proměnných (ne všechny najednou)
- mohou být **heterogenní**
  - domény proměnných mohou být různé
- **nejsou směrové** (funkce)
  - X = Y+2 lze použít pro výpočet X i Y
- jsou **deklarativní**
  - neurčují výpočtovou proceduru pro své splnění
- jsou **aditivní**
  - pořadí podmínek nehraje roli, důležitá je jejich konjunkce
- jsou **zřídkakdy nezávislé**
  - podmínky sdílejí proměnné

Co nás bude zajímat?

- **Modelování problémů**

  Jak popsat reálný problém pomocí sady omezujících podmínek?

- **Řešící algoritmy**

  Jak nalézt hodnoty proměnných tak, aby všechny podmínky byly splněny?

V čem má CP navrch

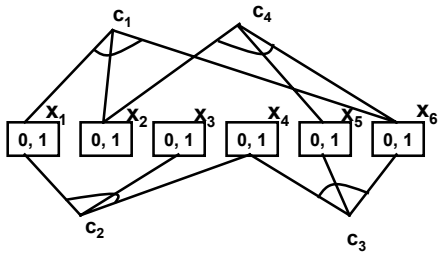
- **blízko reálnému problému**
  - všichni používáme omezení při řešení problémů
  - pomocí podmínek lze zachytit mnoho reálných vlastností
- **deklarativní charakter**
  - soustředí se na popis problému spíše než na způsob, jak problém řešit
- **kooperativní řešení problémů**
  - jednotný rámec pro integraci různých řešících technik
  - jednoduché (prohledávání) i sofistikované (inference) techniky
- **sémantické základy**
  - čisté a elegantní jazyky
  - kořeny v logickém programování
- **aplikace**
  - nejedná se jen o akademickou hříčku, ale o reálně používané systémy

Známe své meze

- **efektivita**
  - kombinatorická exploze
  - řešíme koneckonců NP-těžké problémy
- **nepředvídatelné chování**
  - dokud to nezkusíme, nevíme jaký bude výsledek
- **stabilita modelů**
  - nová data = nový problém
- **příliš lokální**
  - přes jednotlivé podmínky nevidíme problém v celku
  - globální podmínky
  - distribuované výpočty
- **slabá spolupráce řešičů**
  - integrace různých řešících technologií není jednoduchá
  - často jen sdílení proměnných

Reprezentace CSP

- **Reprezentace podmínek:**
  - intenzionální (matematická/logická formule)
  - extenzionálně (výčet k-tic kompatibilních hodnot, 0-1 matice)
- **Reprezentace CSP** problému pomocí (hyper)grafu
  - vrcholy = proměnné
  - (hyper)hrany = podmínky
- **Příklad:**
  - proměnné $x_1,\ldots,x_6$ s doménou {0,1}
  - c_1: $x_1+x_2+x_6=1$
  - c_2: $x_1-x_3+x_4=1$
  - c_3: $x_4+x_5-x_6>0$
  - c_4: $x_2+x_5-x_6=0$



Binární podmínky

Svět není binární ...
ale lze na binární tvar transformovat!

Binární CSP
CSP + všechny podmínky jsou binární

Poznámka: unární podmínky jsou nezajímavé, lze je kódovat v doméně proměnné

Ekvivalence CSP
Dva CSP problémy jsou ekvivalentní, pokud mají stejnou množinu řešení.

Rozšířená ekvivalence
Řešení problémů lze mezi sebou vzájemně „syntakticky" převést.

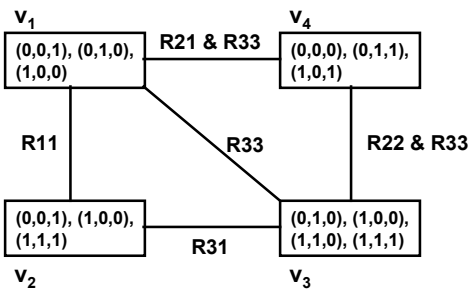
Lze každý CSP problém transformovat na (rozšířeně) ekvivalentní binární CSP?

Duální kódování

Prohoďme proměnné a podmínky.

- k- ární podmínku c převedeme na duální proměnnou v_c s doménou obsahující konzistentní k-tice
- pro každou dvojici podmínek c a c` sdílejících proměnné zavedeme binární podmínku mezi v_c a $v_{c'}$ omezující duální proměnné na k-tice, ve kterých mají sdílené proměnné stejnou hodnotu

Příklad:
- proměnné $x_1,\ldots,x_6$ s doménou {0,1}
- c_1: $x_1+x_2+x_6=1$
- c_2: $x_1-x_3+x_4=1$
- c_3: $x_4+x_5-x_6>0$
- c_4: $x_2+x_5-x_6=0$



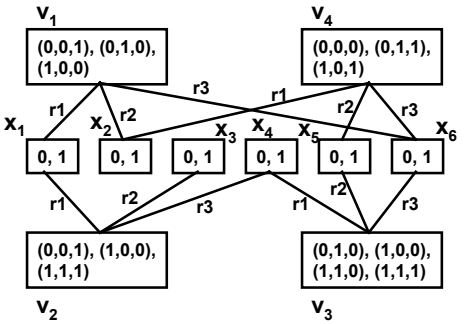
Kódování se skrytou proměnnou

Pro (nebinární) podmínky zaveďme nové proměnné.

- k- ární podmínku c převedeme na duální proměnnou v_c s doménou obsahující konzistentní k-tice
- pro každou proměnnou x v podmínce c zavedeme podmínku mezi x a v_c omezující k-tice duální proměnné na kompatibilní s x

Příklad:

- proměnné $x_1,\ldots,x_6$ s doménou {0,1}
- c_1: $x_1+x_2+x_6=1$
- c_2: $x_1-x_3+x_4=1$
- c_3: $x_4+x_5-x_6>0$
- c_4: $x_2+x_5-x_6=0$

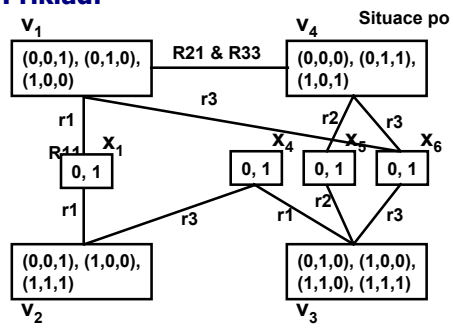


Převody mezi kódováními

Kódování se skrytou proměnnou lze převést na duální kódování:

- cesty délky 2 mezi každou dvojicí duálních proměnných nahraď binární podmínkou, která je spojením relací na smazaných cestách (Př. r1 a r1 přejde na R11); pozor na hrany sdílené více cestami!
- pokud se původní proměnná stane izolovanou nebo je spojena jen s jednou duální proměnnou, vyřaď ji

Příklad:



V každém kroku transformace máme ekvivalentní binární CSP
⇒ **„hybridní" kódování**

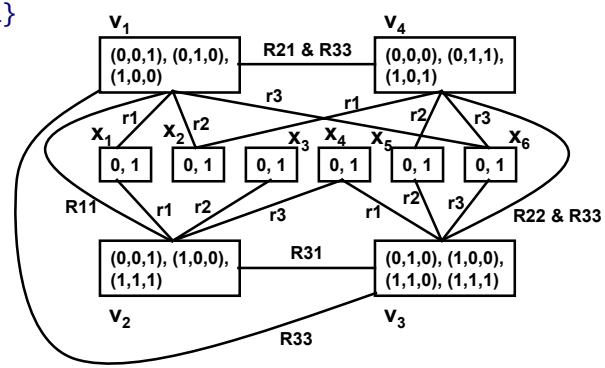
Transformaci lze provést i v obráceném směru.

Zdvojené kódování

Kódování se skrytou proměnnou obohaťme o podmínky duálního kódování.

Příklad:

- proměnné $x_1,\ldots,x_6$ s doménou {0,1}
- c_1: $x_1+x_2+x_6=1$
- c_2: $x_1-x_3+x_4=1$
- c_3: $x_4+x_5-x_6>0$
- c_4: $x_2+x_5-x_6=0$



Poznámky k binarizaci

- **Proč binarizujeme?**
  - unifikovaný tvar CSP problému
  - řada řešících algoritmů navržena pro binární CSP
  - tradice (historické důvody)
- **Jaký způsob binarizace je lepší?**
  - těžko říct
  - duální kódování:
    lepší propagace vs. extenzionální reprezentace podmínky
  - kódování se skrytou proměnnou:
    zachovává původní proměnné vs. oslabení propagace
- **Binární vs. nebinární podmínky**
  - složitější propagační algoritmy pro nebinární podmínky
  - využití sémantiky nebinárních podmínek pro lepší propagaci