Úvod do programování s omezujícími podmínkami

Obsah přednášky

- Ukázkový příklad: sudoku
- Základní pojmy: omezení, ...
- Jak řešíme problémy s omezujícími podmínkami
- Příklady a aplikace
- Složitost a úplnost

Sudoku: problém

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

Přiřaď prázdným polím čísla tak, že:

čísla odlišná na řádku, ve sloupci a v bloku

Příklad převzat z přednášky Ch.Schulte, University of Uppsala

Sudoku: řádky

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

Přiřaď prázdným polím čísla tak, že:

čísla odlišná **na řádku**, ve sloupci a v bloku

Sudoku: sloupce

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

Přiřaď prázdným polím čísla tak, že:

čísla odlišná na řádku, **ve sloupci** a v bloku

Sudoku: bloky

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

Přiřaď prázdným polím čísla tak, že:

čísla odlišná na řádku, ve sloupci a **v bloku**

Propagace v bloku: vstup

	8	
	6	3

Propagace v bloku: vstup

	8	
	6	3

- Žádné pole v bloku nemůže obsahovat čísla 3,6,8

Propagace v bloku: promazání hodnot

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

- Žádné pole v bloku nemůže obsahovat čísla 3,6,8
  - propagace do dalších polí v bloku
- Řádky a sloupce: podobně

Propagace: jedno pole

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

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

Odstranění čísel z polí tak, že:

čísla odlišná na řádku, ve sloupci a v bloku

Propagace: jedno pole a řádek

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

1,3,5,6,7,8

Odstranění čísel z polí tak, že:

čísla odlišná **na řádku**, ve sloupci a v bloku

Propagace: jedno pole a sloupec

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

1,3,6,7

Odstranění čísel z polí tak, že:

čísla odlišná na řádku, **ve sloupci** a v bloku

Propagace: jedno pole a blok

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

1,3,6

Odstranění čísel z polí tak, že:

čísla odlišná na řádku, ve sloupci a **v bloku**

Iterativní propagace

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

- **Iterování propagací** pro řádky, sloupce, bloky
- **Pokud stále zůstává více možností:** prohledávání

Sudoku a programování s omezujícími podmínkami

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

- **Proměnné**: pro každé pole
  - mají **hodnoty**: čísla
  - udržování množiny možných čísel
- **Omezení**: vyjadřují odlišnosti
  - relace mezi proměnnými
- **Modelování**: proměnné, hodnoty, omezení
- **Řešení**: propagace, prohledávání

Omezení (*constraint*)

- Dána
  - množina **(doménových) proměnných** $Y = \{y_1, \ldots, y_k\}$
  - konečná množina hodnot (**doména**) $D = D_1 \cup \ldots \cup D_k$

Omezení (podmínka) c na Y je podmnožina $D_1 \times \ldots \times D_k$ **tj. relace**

- omezuje hodnoty, kterých mohou proměnné nabývat současně

Omezení (*constraint*)

- Dána
  - množina **(doménových) proměnných** $Y = \{y_1, \ldots, y_k\}$
  - konečná množina hodnot (**doména**) $D = D_1 \cup \ldots \cup D_k$

Omezení (podmínka) c na Y je podmnožina $D_1 \times \ldots \times D_k$ **tj. relace**

  - omezuje hodnoty, kterých mohou proměnné nabývat současně

- Příklad:
  - proměnné: A,B
  - domény: {0,1} pro A {1,2} pro B
  - omezení: A$\neq$B nebo (A,B) $\in$ {(0,1),(0,2),(1,2)}

Omezení (*constraint*)

- Dána
  - množina **(doménových) proměnných** $Y = \{y_1, \ldots, y_k\}$
  - konečná množina hodnot (**doména**) $D = D_1 \cup \ldots \cup D_k$

  Omezení (podmínka) c na Y je podmnožina $D_1 \times \ldots \times D_k$ **tj. relace**

  - omezuje hodnoty, kterých mohou proměnné nabývat současně
- Příklad:
  - proměnné: A,B
  - domény: {0,1} pro A {1,2} pro B
  - omezení: A≠B nebo (A,B) ∈ {(0,1),(0,2),(1,2)}
- Omezení c definováno na proměnných $y_1, \ldots y_k$ je **splněno**, pokud pro hodnoty $d_1 \in D_1, \ldots d_k \in D_k$ platí $(d_1, \ldots d_k) \in c$
  - příklad (pokračování): omezení splněno pro $(0,1), (0,2), (1,2)$, není splněno pro $(1,1)$

Problém splňování podmínek (CSP)

- Dána
  - konečná množina **proměnných** $X = \{x_1, \ldots, x_n\}$
  - konečná množina hodnot (**doména**) $D = D_1 \cup \ldots \cup D_n$
  - konečná množina **omezení** $C = \{c_1, \ldots, c_m\}$
    - omezení je definováno na podmnožině X

Problém splňování podmínek je **trojice** (X, D, C)

(constraint satisfaction problem)

Problém splňování podmínek (CSP)

- Dána
  - konečná množina **proměnných** $X = \{x_1, \ldots, x_n\}$
  - konečná množina hodnot (**doména**) $D = D_1 \cup \ldots \cup D_n$
  - konečná množina **omezení** $C = \{c_1, \ldots, c_m\}$
    - omezení je definováno na podmnožině X

Problém splňování podmínek je **trojice** (X, D, C)

(constraint satisfaction problem)

- Příklad:
  - proměnné: A, B, C
  - domény: $A \in \{0, 1\}$ $B = 1$ $C \in \{0, 1, 2\}$
  - omezení: $A \neq B$ $B \neq C$

Řešení CSP

- **Částečné ohodnocení (přiřazení) proměnných** $(d_1, \ldots, d_k), k < n$
  - některé proměnné mají přiřazenu hodnotu
- **Úplné ohodnocení (přiřazení) proměnných** $(d_1, \ldots, d_n)$
  - všechny proměnné mají přiřazenu hodnotu

Řešení CSP

- **Částečné ohodnocení (přiřazení) proměnných** $(d_1, \ldots, d_k), k < n$
  - některé proměnné mají přiřazenu hodnotu
- **Úplné ohodnocení (přiřazení) proměnných** $(d_1, \ldots, d_n)$
  - všechny proměnné mají přiřazenu hodnotu
- **Řešení CSP**
  - úplné ohodnocení proměnných, které splňuje všechna omezení
  - $(d_1, \ldots, d_n) \in D_1 \times \ldots \times D_n$ je **řešení** (X, D, C)
    - pro každé $c_i \in C$ na $x_{i_1}, \ldots x_{i_k}$ platí $(d_{i_1}, \ldots d_{i_k}) \in c_i$

Řešení CSP

- **Částečné ohodnocení (přiřazení) proměnných** $(d_1, \ldots, d_k), k < n$
  - některé proměnné mají přiřazenu hodnotu
- **Úplné ohodnocení (přiřazení) proměnných** $(d_1, \ldots, d_n)$
  - všechny proměnné mají přiřazenu hodnotu
- **Řešení CSP**
  - úplné ohodnocení proměnných, které splňuje všechna omezení
  - $(d_1, \ldots, d_n) \in D_1 \times \ldots \times D_n$ je **řešení** (X, D, C)
    - pro každé $c_i \in C$ na $x_{i_1}, \ldots x_{i_k}$ platí $(d_{i_1}, \ldots d_{i_k}) \in c_i$
- Hledáme: **jedno řešení** nebo

  všechna řešení nebo

  optimální řešení vzhledem k účelové funkci, tj. řešíme

  optimalizační problém s podmínkami (constraint optimization problem)

Přístup CP k programování

- Formulace daného problému pomocí omezení: **modelování**
- Řešení vybrané reprezentace pomocí
  - **doménově specifických metod**
  - **obecných metod**

Obecné metody

- **Algoritmy propagace omezení (konzistenční algoritmy)**
  - umožňují odstranit nekonzistentní hodnoty z domén proměnných
  - zjednodušují problém
  - udržují ekvivalenci mezi původním a zjednodušeným problémem
  - používají se pro výpočet **lokální konzistence**
  - aproximují tak **globální konzistenci**

$$A \in \{0,1\},\ B = 1,\ C \in \{0,1,2\},\ A \neq B,\ A \neq C$$

Obecné metody

- **Algoritmy propagace omezení (konzistenční algoritmy)**
  - umožňují odstranit nekonzistentní hodnoty z domén proměnných
  - zjednodušují problém
  - udržují ekvivalenci mezi původním a zjednodušeným problémem
  - používají se pro výpočet **lokální konzistence**
  - aproximují tak **globální konzistenci**

$$A \in \{0,1\},\ B = 1,\ C \in \{0,1,2\},\ A \neq B,\ A \neq C$$

po propagaci: $A = 0,\ B = 1,\ C \in \{1,2\}$

Obecné metody

- **Algoritmy propagace omezení (konzistenční algoritmy)**
  - umožňují odstranit nekonzistentní hodnoty z domén proměnných
  - zjednodušují problém
  - udržují ekvivalenci mezi původním a zjednodušeným problémem
  - používají se pro výpočet **lokální konzistence**
  - aproximují tak **globální konzistenci**

$$A \in \{0,1\},\ B = 1,\ C \in \{0,1,2\},\ A \neq B,\ A \neq C$$

po propagaci: $A = 0,\ B = 1,\ C \in \{1,2\}$

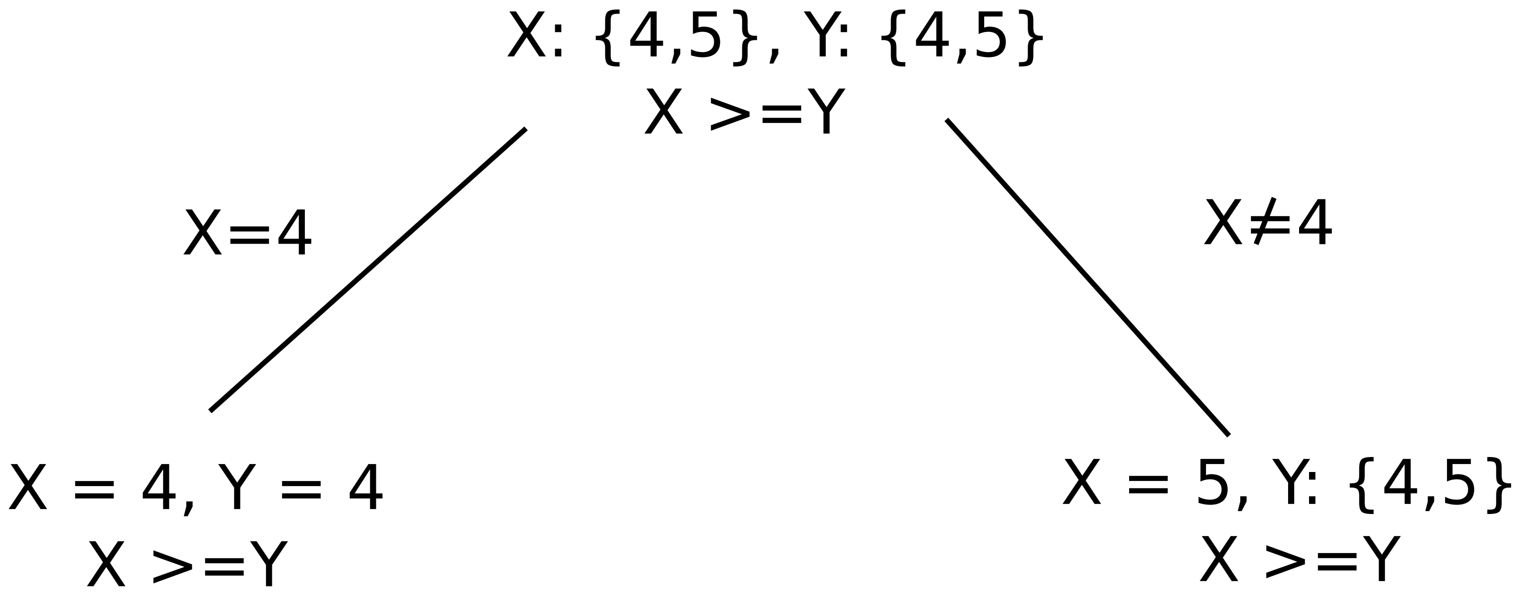
- **Prohledávací algoritmy**
  - prohledávání stavového prostoru řešení
  - příklady: backtracking, metoda větví a mezí

Prohledávání: příklad

Prohledávání pomocí větvení

- vytvoření podproblému s dodatečnou informací

  umožní další propagaci omezení



Doménově specifické metody

- Specializované algoritmy
- Nazývány **řešiče omezení (*constraint solvers*)**
- Příklady:
  - program pro řešení systému lineárních rovnic
  - knihovny pro lineární programování
  - implementace unifikačního algoritmu

Doménově specifické metody

- Specializované algoritmy
- Nazývány **řešiče omezení (*constraint solvers*)**
- Příklady:
  - program pro řešení systému lineárních rovnic
  - knihovny pro lineární programování
  - implementace unifikačního algoritmu
- Programování s omezujícími podmínkami
  - široký pojem zahrnující řadu oblastí
  - lineární algebra, globální optimalizace, lineární a celočíselné programování, . . .
- **Existence doménově specifických metod**

  $\Longrightarrow$ použití místo obecných metod
  - hledání doménově specifických metod tak, aby mohly být použity místo obecných metod

Algebrogram

- Přiřaďte cifry 0, ... 9 písmenům S, E, N, D, M, O, R, Y tak, aby platilo:
  - $$\begin{array}{r} \texttt{SEND} \\ + \ \texttt{MORE} \\ \hline \texttt{MONEY} \end{array}$$
  - různá písmena mají přiřazena různé cifry
  - S a M nejsou 0
- Jediné řešení:

$$\begin{array}{r} 9567 \\ + \ 1085 \\ \hline 10652 \end{array}$$

- **Proměnné:**

Algebrogram

- Přiřaďte cifry 0, . . . 9 písmenům S, E, N, D, M, O, R, Y tak, aby platilo:
  - $$\begin{array}{r} \text{SEND} \\ +\ \text{MORE} \\ \hline \text{MONEY} \end{array}$$
  - různá písmena mají přiřazena různé cifry
  - S a M nejsou 0
- Jediné řešení:

$$\begin{array}{r} 9567 \\ +\ 1085 \\ \hline 10652 \end{array}$$

- **Proměnné:** S,E,N,D,M,O,R,Y
- **Domény:** 1..9 pro S,M   0..9 pro E,N,D,O,R,Y

Algebrogram: alternativy pro omezení rovnosti

- 1 omezení rovnosti

Algebrogram: alternativy pro omezení rovnosti

- 1 omezení rovnosti

```
                     1000*S + 100*E + 10*N + D                    SEND
+                    1000*M + 100*O + 10*R + E                  + MORE
= 10000*M + 1000*O + 100*N + 10*E + Y                            MONEY
```

Algebrogram: alternativy pro omezení rovnosti

- 1 omezení rovnosti

```
                 1000*S + 100*E + 10*N + D            SEND
+                1000*M + 100*O + 10*R + E          + MORE
= 10000*M + 1000*O + 100*N + 10*E + Y               MONEY
```

- 5 omezení rovnosti

  použití „přenosových” proměnných P1,P2,P3,P4 s doménami 0..1

```
     D + E = 10*P1 + Y,
P1 + N + R = 10*P2 + E,
P2 + E + O = 10*P3 + N,
P3 + S + M = 10*P4 + O
        P4 = M
```

Algebrogram: alternativy pro omezení nerovnosti

Algebrogram: alternativy pro omezení nerovnosti

- **28 omezení nerovnosti:** $X \neq Y$ pro $X,Y \in \{S,E,N,D,M,O,R,Y\}$

Algebrogram: alternativy pro omezení nerovnosti

- **28 omezení nerovnosti:** `X` $\neq$ `Y pro X,Y` $\in$ `{S,E,N,D,M,O,R,Y}`
- **1 omezení pro nerovnost**

  pro proměnné $x_1, \ldots, x_n$ s doménami $D_1, \ldots, D_n$:

  `all_different` $(x_1, \ldots, x_n) := \{(d_1, \ldots, d_n)\ ,\ d_i \neq d_j$ pro $i \neq j\}$

Optimalizační problém s podmínkami (COP)

- Problém splňování podmínek (X, D, C)
- **Účelová funkce** $obj : Sol \to W$
- **Optimalizační problém s podmínkami (*constraint optimization problem*)**
  - nalezení řešení d pro (X, D, C) takové, že $obj(d)$ je optimální
    - optimální $\equiv$ maximální nebo minimální

Problém batohu (*knapsack problem*)

Je dán batoh velikosti m a n předmětů různé velikosti a ceny. Vyberte takové předměty, aby se vešly do batohu a jejich celková cena byla maximální.

- Batoh velikosti m
- Předměty velikosti $v_1, \ldots, v_n$ a ceny $c_1, \ldots, c_n$
- **Proměnné:**

Problém batohu (*knapsack problem*)

Je dán batoh velikosti m a n předmětů různé velikosti a ceny. Vyberte takové předměty, aby se vešly do batohu a jejich celková cena byla maximální.

- Batoh velikosti m
- Předměty velikosti $v_1, \ldots, v_n$ a ceny $c_1, \ldots, c_n$
- **Proměnné:** $x_1, \ldots, x_n$
- **Domény:**

Problém batohu (*knapsack problem*)

Je dán batoh velikosti m a n předmětů různé velikosti a ceny. Vyberte takové předměty, aby se vešly do batohu a jejich celková cena byla maximální.

- Batoh velikosti m
- Předměty velikosti $v_1, \ldots, v_n$ a ceny $c_1, \ldots, c_n$
- **Proměnné:** $x_1, \ldots, x_n$
- **Domény:** $0..1$
- **Omezení:**

Problém batohu (*knapsack problem*)

Je dán batoh velikosti m a n předmětů různé velikosti a ceny. Vyberte takové předměty, aby se vešly do batohu a jejich celková cena byla maximální.

- Batoh velikosti m
- Předměty velikosti $v_1, \ldots, v_n$ a ceny $c_1, \ldots, c_n$
- **Proměnné:** $x_1, \ldots, x_n$
- **Domény:** $0..1$
- **Omezení:** $\sum_{i=1}^{n} v_i.x_i \leq m$
- **Účelová funkce:**

Problém batohu (*knapsack problem*)

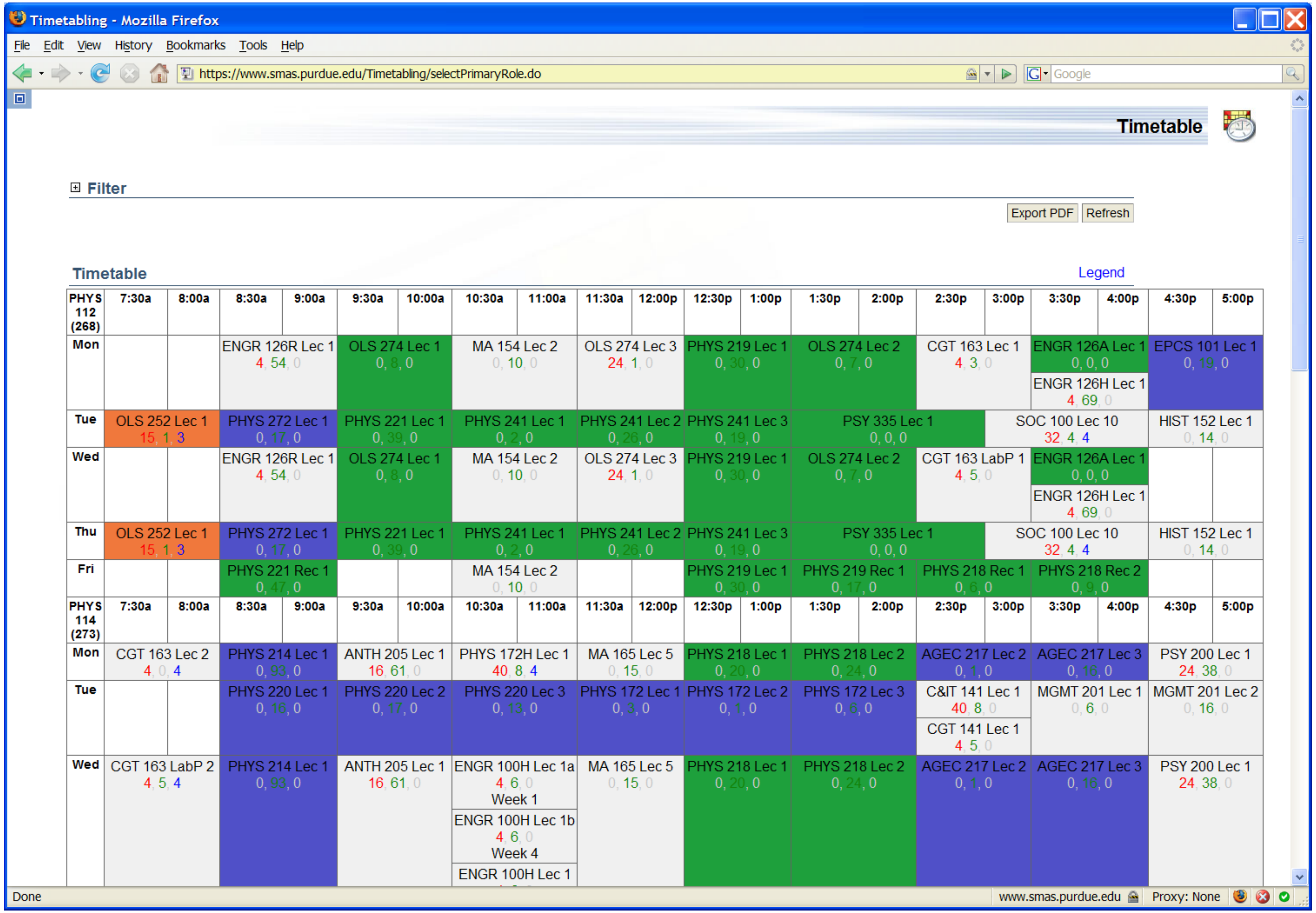
Je dán batoh velikosti m a n předmětů různé velikosti a ceny. Vyberte takové předměty, aby se vešly do batohu a jejich celková cena byla maximální.

- Batoh velikosti m
- Předměty velikosti $v_1, \dots, v_n$ a ceny $c_1, \dots, c_n$
- **Proměnné:** $x_1, \dots, x_n$
- **Domény:** $0..1$
- **Omezení:** $\sum_{i=1}^{n} v_i.x_i \leq m$
- **Účelová funkce:** maximize $\sum_{i=1}^{n} c_i x_i$

Aplikace: prehled

- Operační výzkum
  - optimalizační problémy
  - rozvrhování, plánování, alokace zdrojů
- Zpracování přirozeného jazyka
  - konstrukce parserů
- Počítačová grafika
  - geometrické vztahy při analýze scény
- Databáze
  - obnovení/zajištění konzistence dat
- Molekulární biologie
  - DNA sekvencování
- ...

Příklad aplikace z MU: univerzitní rozvrhování



rozvrhovací systém Unitime `http://www.unitime.org`

International Timetabling Competition 2019 co-organized by FI MU: `https://www.itc2019.org`

Univerzitní rozvrhování: proměnné a omezení

Doménové proměnné

- čas výuky předmětu I: TimeI, hodnoty: možné časy
- místnost výuky předmětu I: RoomI, hodnoty: identifikátory místností

Univerzitní rozvrhování: proměnné a omezení

Doménové proměnné

- čas výuky předmětu I: TimeI, hodnoty: možné časy
- místnost výuky předmětu I: RoomI, hodnoty: identifikátory místností

Omezující podmínky

- zakázaný čas: TimeI $\neq$ Prohibited
- minimální velikost místnosti: RoomI $\geq$ ConstSize
  - identifikátory místností uspořádány podle velikosti
  - ConstSize: nejmenší identifikátor místnosti s velikostí Size

Univerzitní rozvrhování: proměnné a omezení

Doménové proměnné

- čas výuky předmětu I: TimeI, hodnoty: možné časy
- místnost výuky předmětu I: RoomI, hodnoty: identifikátory místností

Omezující podmínky

- zakázaný čas: TimeI $\neq$ Prohibited
- minimální velikost místnosti: RoomI $\geq$ ConstSize
  - identifikátory místností uspořádány podle velikosti
  - ConstSize: nejmenší identifikátor místnosti s velikostí Size
- v jedné místnosti v každém čase nejvýše jeden předmět
- jeden vyučující učí nejvýše jeden předmět v každém čase
- ...

Univerzitní rozvrhování: optimalizace

Optimalizační kritéria

- **výuka v preferovaných časech**
  - cena za výuku předmětu I ve vybraném čase: CostTimeI
  - CostTime = CostTime1 + CostTime2 + $\cdots$

Univerzitní rozvrhování: optimalizace

Optimalizační kritéria

- **výuka v preferovaných časech**
  - cena za výuku předmětu I ve vybraném čase: CostTimeI
  - CostTime = CostTime1 + CostTime2 + $\cdots$
- **výuka v preferovaných místnostech**
  - cena za výuku předmětu I ve vybraném čase: CostRoomI
  - CostRoom = CostRoom1 + CostRoom2 + $\cdots$

Univerzitní rozvrhování: optimalizace

Optimalizační kritéria

- **výuka v preferovaných časech**
  - cena za výuku předmětu I ve vybraném čase: CostTimeI
  - CostTime = CostTime1 + CostTime2 + $\cdots$
- **výuka v preferovaných místnostech**
  - cena za výuku předmětu I ve vybraném čase: CostRoomI
  - CostRoom = CostRoom1 + CostRoom2 + $\cdots$
- **dva předměty jednoho studenta by se neměly překrývat**
  - dva předměty I,J zároveň navštěvuje SIJ studentů

Univerzitní rozvrhování: optimalizace

Optimalizační kritéria

- **výuka v preferovaných časech**
  - cena za výuku předmětu I ve vybraném čase: CostTimeI
  - CostTime = CostTime1 + CostTime2 + $\cdots$
- **výuka v preferovaných místnostech**
  - cena za výuku předmětu I ve vybraném čase: CostRoomI
  - CostRoom = CostRoom1 + CostRoom2 + $\cdots$
- **dva předměty jednoho studenta by se neměly překrývat**
  - dva předměty I,J zároveň navštěvuje SIJ studentů
  - CostOverlap = $\sum_{\text{I,J: timeOverlap(I,J)}} \text{SIJ}$

Univerzitní rozvrhování: optimalizace

Optimalizační kritéria

- **výuka v preferovaných časech**
  - cena za výuku předmětu I ve vybraném čase: CostTimeI
  - CostTime = CostTime1 + CostTime2 + $\cdots$
- **výuka v preferovaných místnostech**
  - cena za výuku předmětu I ve vybraném čase: CostRoomI
  - CostRoom = CostRoom1 + CostRoom2 + $\cdots$
- **dva předměty jednoho studenta by se neměly překrývat**
  - dva předměty I,J zároveň navštěvuje SIJ studentů
  - CostOverlap = $\displaystyle\sum_{\text{I,J: timeOverlap(I,J)}} \text{SIJ}$
- minimize (WTime * CostTime + WRoom * CostRoom + WOverlap * CostOverlap)