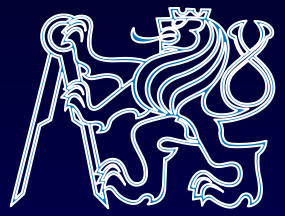
- ©Jan Schmidt 2011
  Katedra číslicového návrhu
  Fakulta informačních technologií
  České vysoké učení technické v Praze
- Zimní semestr 2013/14



EVROPSKÝ SOCIÁLNÍ FOND
PRAHA & EU: INVESTUJENE
DO VAŠÍ BUDOUCNOSTI

MI-PAA

4. Optimalizační problémy

- Třídy optimalizačních problémů
- Pseudopolynomiální algoritmy
- Aproximativní algoritmy
- Třídy aproximovatelných problémů
- Randomizované algoritmy

Třída NPO

- Definice: optimalizační problém *Π* patří do třídy NPO, jestliže splňuje následující podmínky:
  - velikost výstupu instance je omezena polynomem ve velikosti instance
  - problém, zda daná konfigurace je řešením, patří do P
  - hodnotu optimalizačního kritéria pro každé řešení každé instance lze vypočíst v polynomiálním čase

výstup lze zapsat v polynomiálním čase

omezující podmínky lze vyhodnotit v polynomiálním čase

optimalizační kritérium lze vyhodnotit v polynomiálním čase

Třída PO

- Definice (třída PO):
  optimalizační problém Π patří do třídy PO, jestliže splňuje následující podmínky:
    - patří do NPO
    - existuje program pro Turingův stroj, který každou instanci vyřeší v polynomiálním čase.
- Příklad:
  problém nejkratší cesty v grafu $G=(V,E)$ patří do PO
  - velikost výstupu $O(|E|)$
  - ověření omezení $O(|E| \log |E|)$
  - optimalizační kritérium $O(|E|)$
  - existuje polynomiální algoritmus (Dijkstra)

Příklad NPO problému: optimalizační TSP

Dána množina n měst $C=\{c_1,c_2,\ldots,c_n\}$. Pro každá dvě města c_i,c_j je dána vzdálenost $d(c_i,c_j)$. Nalezněte uzavřenou túru, která prochází každým městem právě jednou a má nejmenší délku.

- Velikost výstupu instance: $O(|C|)$
- Kontrola túry: $O(|C|)$ - viz Hamiltonova kružnice
- Výpočet optimalizačního kritéria: $O(|C|)$

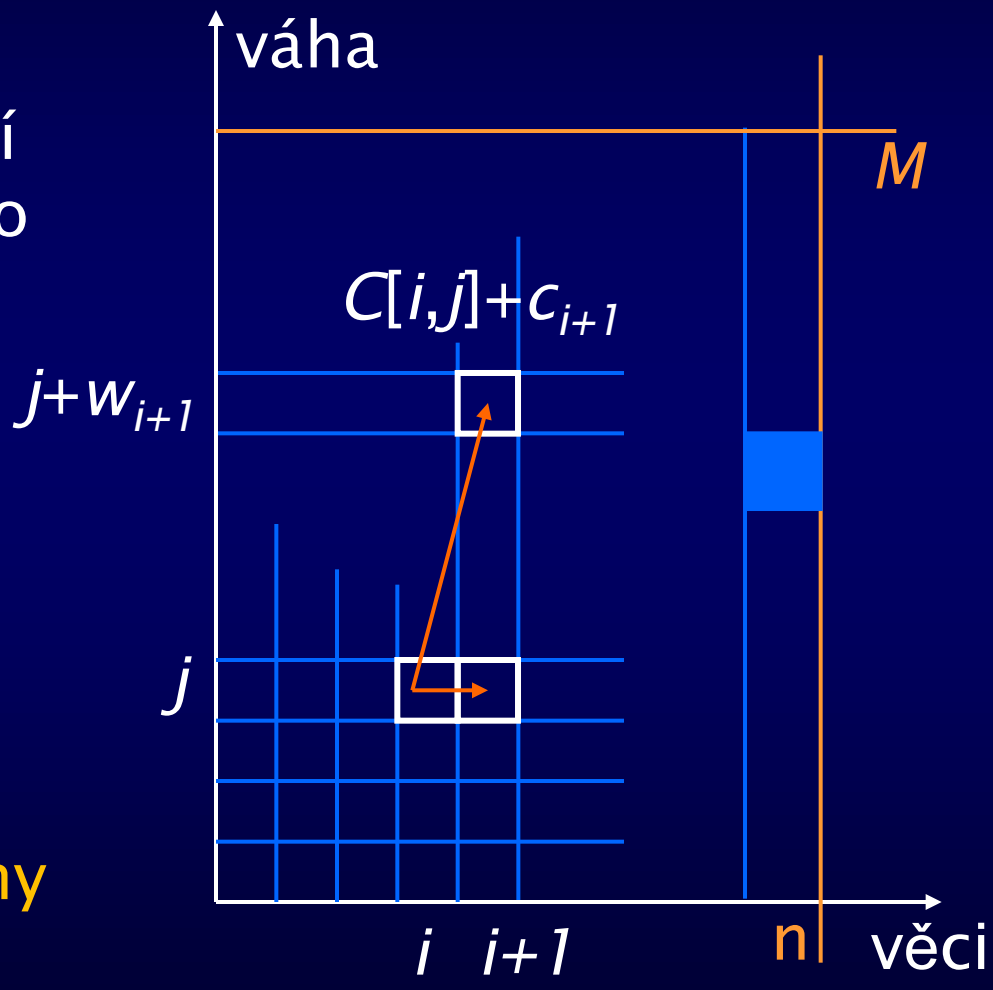
Pseudopolynomiální algoritmy

Pseudopolynomiální algoritmus pro problém batohu

$C[i,j]$:
cena optimálního plnění batohu s kapacitou j pro prvých i věcí

Výsledek:
maximální $C[n,j]$

Dekompozice podle váhy



Složitost

- Pole má $n.M$ prvků
- Každý prvek lze vypočíst v konstantním čase
- Složitost $O(n.M)$

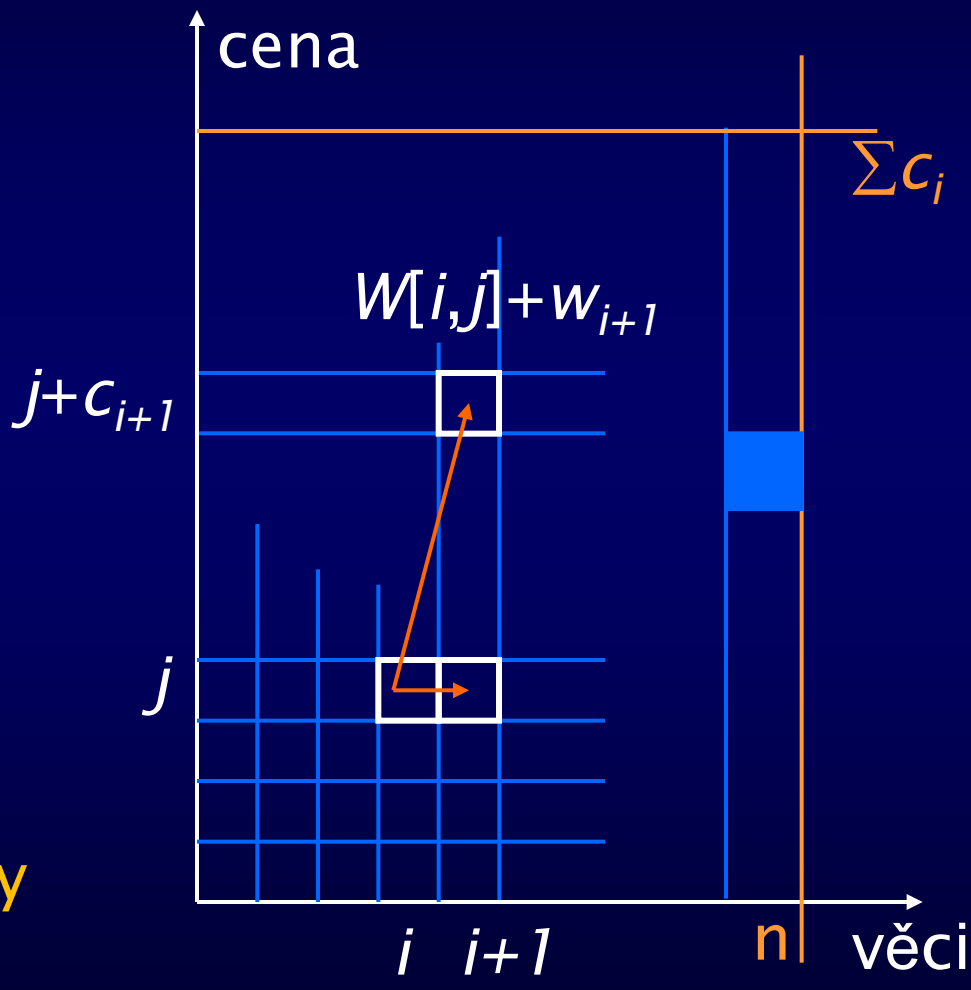
M nesouvisí s velikostí instance (měřené jakýmkoli rozumným způsobem)

Varianta

$W[i,j]$:
váha optimálního (nejlehčího) plnění batohu s cenou j pro prvých i věcí

Výsledek:
$W[n,j] \leq M$
s maximálním j

Dekompozice podle ceny



Složitost

- Velikost pole $n.\sum c_i$
- Každý prvek lze spočítat v konstantním čase
- Nechť $C_M = \max \{c_1, c_2, \ldots, c_n\}$.
- Pak $\sum c_i \leq n.C_M$
- Složitost $O(n^2.C_M)$

C_M nesouvisí s velikostí instance (měřené jakýmkoli rozumným způsobem)

Pseudopolynomiální algoritmus

- Definice:
  Algoritmus, jehož počet kroků závisí polynomiálně na velikosti instance, ale závisí dále na parametru, který s velikostí instance nesouvisí, nazýváme pseudopolynomiálním.

Aproximativní algoritmy, aproximovatelné problémy a jejich třídy

Aproximativní algoritmus pro problém batohu

Algoritmus APR-KNAP:

- Věci seřaďte podle klesajícího poměru cena/hmotnost
- V tomto pořadí vkládejte do batohu, pokud není překročena nosnost batohu
- Výsledné řešení porovnejte s řešením, které se skládá pouze z jediné, nejcennější věci

- Polynomiální složitost
- Výsledné řešení má cenu ≥50% optimálního řešení

Měření kvality

$C(S)$ hodnota opt. kritéria řešení S
APR(I) aprox. řešení instance I
OPT(I) optimální řešení instance I

Definice:
Algoritmus APR má relativní kvalitu R, jestliže

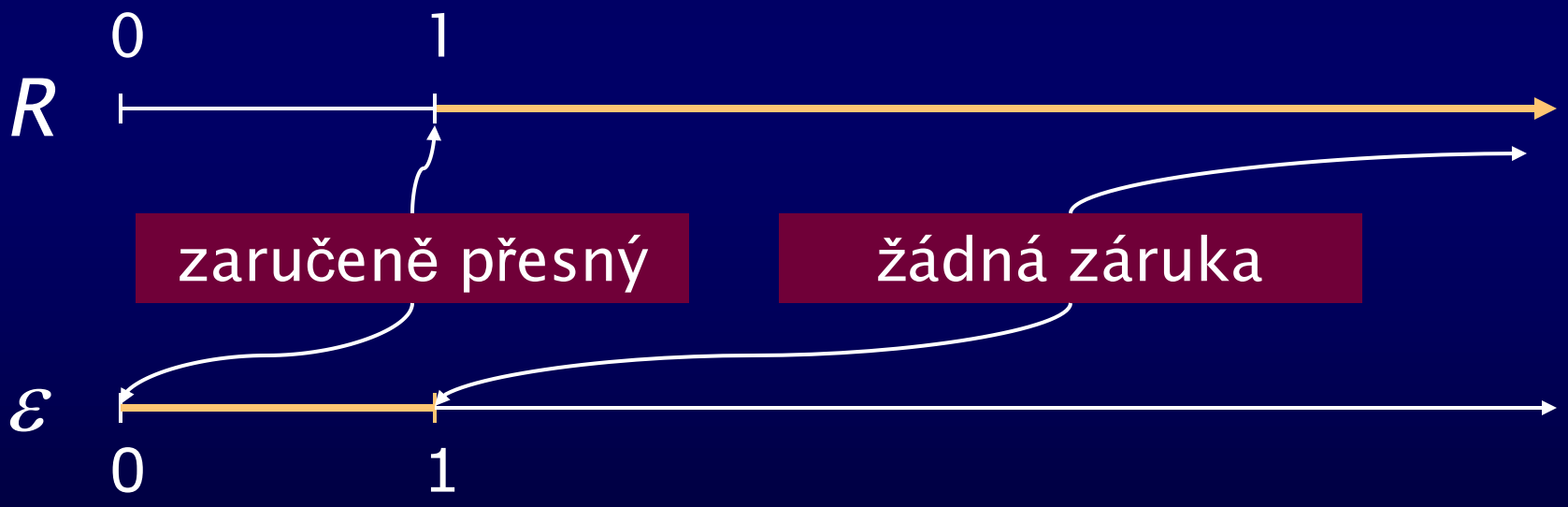
$$R \geq \max_{\forall I} \left\{ \frac{C(\mathrm{APR}(I))}{C(\mathrm{OPT}(I))}, \frac{C(\mathrm{OPT}(I))}{C(\mathrm{APR}(I))} \right\}$$

Algoritmus APR má relativní chybu ε, jestliže

$$\varepsilon \geq \max_{\forall I} \left\{ \frac{|C(\mathrm{APR}(I)) - C(\mathrm{OPT}(I))|}{\max\{ C(\mathrm{OPT}(I)), C(\mathrm{APR}(I)) \}} \right\}$$

Vlastnosti

$$\varepsilon = 1 - \frac{1}{R}$$



Aproximativní algoritmus, třída APX

- Definice:
  Algoritmus APR pro problém Π je *R*-aproximativní (ε-aproximativní), jestliže každou instanci Π vyřeší v polynomiálním čase s relativní kvalitou *R* (relativní chybou ε).
- Definice:
  Optimalizační problém Π je *R*-aproximativní (ε-aproximativní), jestliže pro něj existuje *R*-aproximativní (ε-aproximativní) polynomiální algoritmus. Číslo *R* (ε) nazveme aproximačním prahem problému Π.
- Definice:
  Optimalizační problém Π patří do třídy APX, jestliže je *R*-aproximativní pro konečné *R*.

Příklad: algoritmus A^+

- Problém uzlového pokrytí: dán graf $G=(V,E)$; sestrojit $V' \subseteq V$ takovou, že $|V'| = \min$ a $\forall (u,v) \in E$, $u \in V'$ nebo $v \in V'$.
- Algoritmus:

  1. $V' = \emptyset$
  2. dokud $E \neq \emptyset$
     a. zvol hranu $(u,v) \in E$
     b. $V' = V' \cup \{u,v\}$
     c. odstraň z E hrany incidentní s u nebo v

- Vlastnost:
  $R = 2$

Důkaz

- Algoritmus:
  1. $V' = \emptyset$
  2. dokud $E \neq \emptyset$
     a. zvol hranu $(u,v) \in E$
     b. $V' = V' \cup \{u,v\}$
     c. odstraň z E hrany incidentní s u nebo v

- podle konstrukce: V' reprezentuje $|V'|/2$ hran, které neincidují
- $\Rightarrow V'_{OPT}$ musí mít nejméně $|V'|/2$ uzlů
- $\Rightarrow \varepsilon_{A+} \leq \dfrac{|V'| - |V'_{OPT}|}{|V'|} \leq ½.$
- na grafu o 1 hraně se toho dosáhne
- $\Rightarrow \varepsilon_{A+}=½,\ R_{A+}=2$

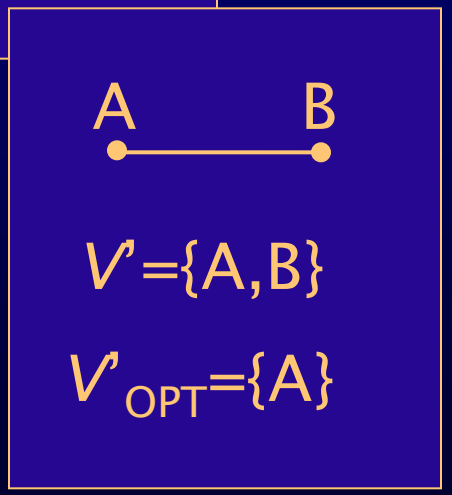
$V'=\{A,B\}$

$V'_{OPT}=\{A\}$

Příklad: A^{++}

- Algoritmus:
  1. $V' = \emptyset$
  2. dokud $E \neq \emptyset$
     a. zvol hranu $(u,v) \in E$ *tak,* že $\deg(u) + \deg(v) = \max$.
     b. $V' = V' \cup \{u,v\}$
     c. odstraň z E hrany incidentní s u nebo v

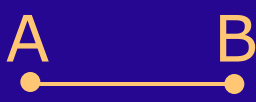
Analýza nejhoršího případu je stejná, ale průměrná kvalita (na náhodných instancích) je lepší.



Příklad: B^+

- Algoritmus:
  1. $V' = \emptyset$
  2. dokud $E \neq \emptyset$
     a. zvol uzel $v \in V - V'$, tak, že $\deg(v) = \max$.
     b. $V' = V' \cup \{v\}$
     c. odstraň z E hrany incidentní s v

Jaká je relativní kvalita?



$V' = \{A\}$

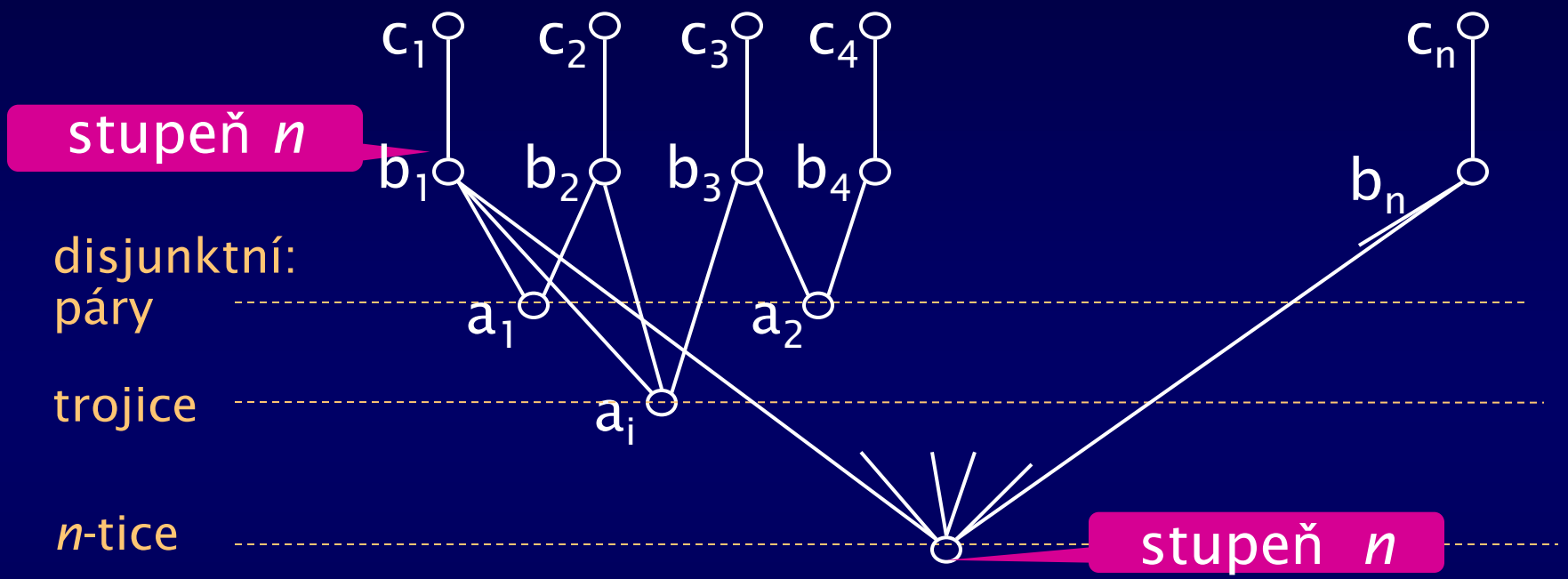
$V'_{OPT} = \{A\}$

Protipříklad

předpokládáme horší případ

$V'_{OPT}=\{b_i\}$

$V'=\{a_i,c_i\}$



$|V'_{OPT}|=n$ $\quad |V'|=\sum_{j=1}^{n}\left\lfloor\frac{n}{j}\right\rfloor \geq 1+n.\ln n$

špatná zpráva: $R_{B+}> 1/n + \ln n$

není aproximativní

Zhodnocení B^+

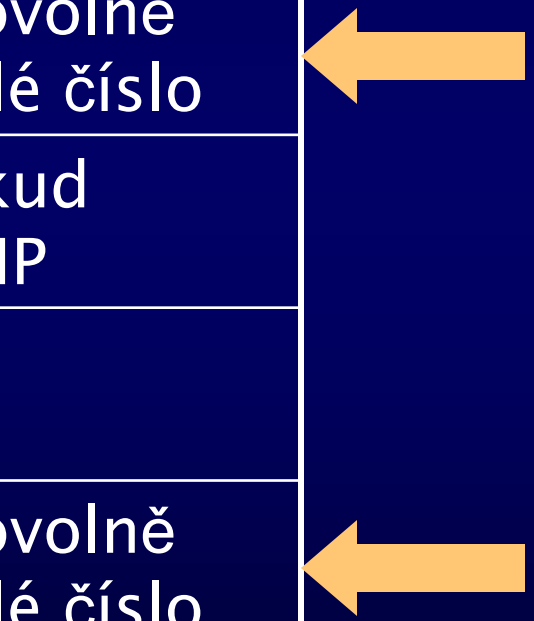
špatná zpráva: $R_{B+} > 1/n + \ln n$

S rostoucím n roste chyba na předloženém protipříkladu nade všechny meze
$\Rightarrow$ nelze dát žádnou záruku
$\Rightarrow$ B^+ není aproximativní

dobrá zpráva (odjinud): $R_{B+} < 1 + \ln n$

Aproximační prahy

uzlové pokrytí	$\varepsilon_{VC} \leq 1/2$	
batoh	$\varepsilon_{K} > 0$	libovolně malé číslo
TSO	$\varepsilon_{TSO} = 1$	pokud P≠NP
ΔTSO (metrický)	$\varepsilon_{\Delta TSO} \leq 1/3$	
TSO geometrický	$\varepsilon_{gTSO} > 0$	libovolně malé číslo



PTAS
(Polynomial Time Approximation Scheme)

- Definice:
  Algoritmus APR, který pro každé $1 > \varepsilon > 0$ vyřeší každou instanci problému Π s relativní chybou nejvýše ε v čase polynomiálním v $|I|$ nazýváme polynomiální aproximační schéma problému Π.

- Definice:
  Problém Π patří do třídy PTAS, jestliže pro Π existuje polynomiální aproximační schéma.

Polynomiální aproximační schéma pro problém batohu

- Dáno: $0<\varepsilon\leq 1$
- Algoritmus PTAS-KNAP:
  - nechť S je množina konfigurací batohu, které obsahují $\lceil 1/\varepsilon \rceil$ věcí nebo méně
  - každou konfiguraci z S doplnit algoritmem APR-KNAP
- Složitost:
  - $O(|I| \log |I|)$ ve velikosti instance
  - $O(2^{1/\varepsilon})$ v převrácené hodnotě relativní chyby

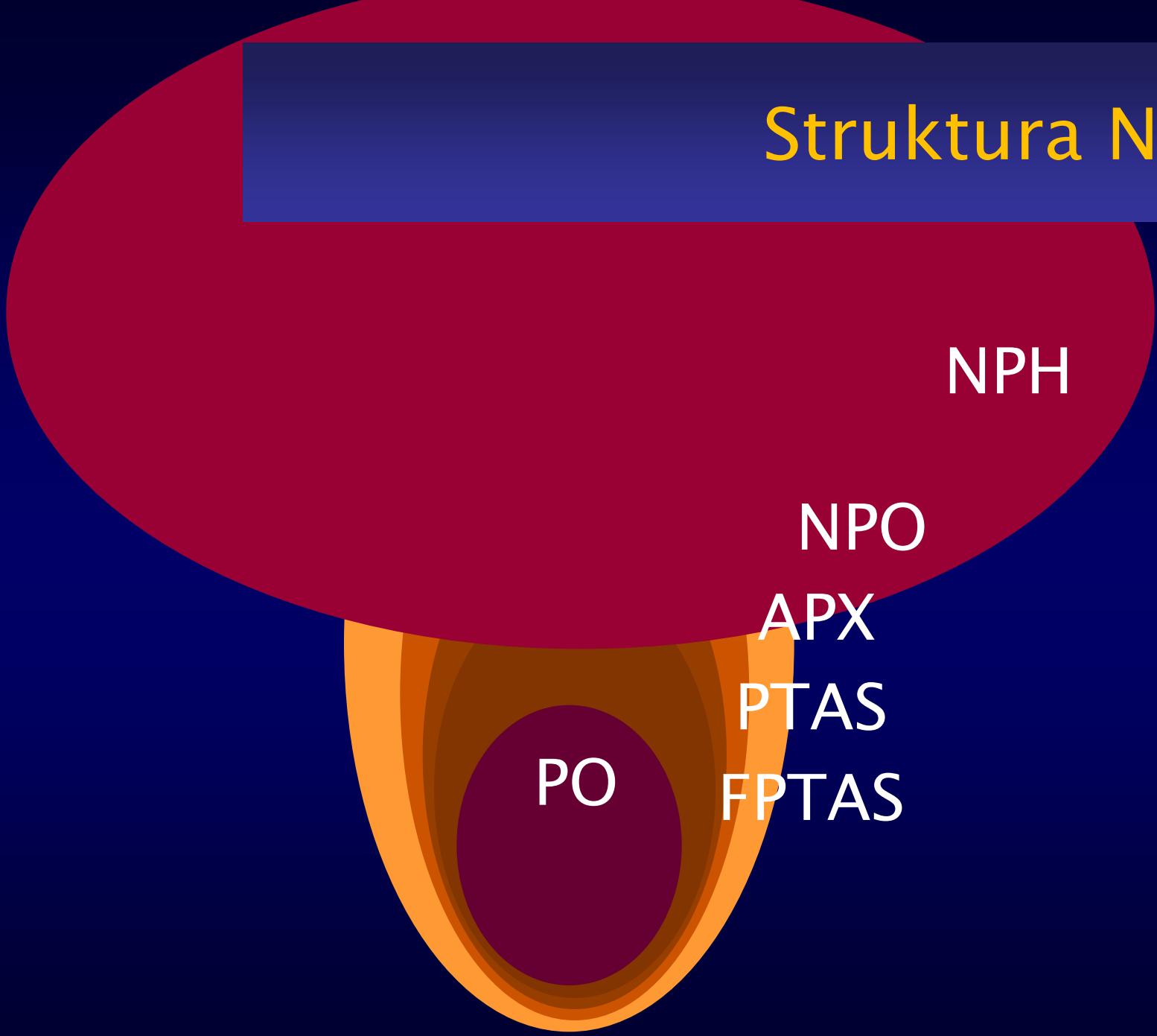
půjde to lépe?

FPTAS
(Fully Polynomial Time Approximation Scheme)

- Definice:
  Polynomiální aproximační schéma APR, jeho čas výpočtu závisí polynomiálně na $1/\varepsilon$, nazýváme plně polynomiální aproximační schéma.

- Definice:
  Problém Π patří do třídy FPTAS, jestliže pro Π existuje plně polynomiální aproximační schéma.

Struktura NPO



Plně polynomiální aproximační schéma pro problém batohu

- Instance: n, M, c_1, c_2, …, c_n, w_1, w_2, …, w_n.
- Nechť $C_M = \max\{c_1, c_2, \ldots, c_n\}$.
- Existuje pseudopolynomiální algoritmus se složitostí $O(n^2 C_M)$.
- Zanedbáme b nejméně významných bitů v cenách.

hlavní fígl

- Složitost ..... $O(\frac{n^2}{2^b} C_M)$.
- Relativní chyba nejhůře ..... $\varepsilon = \frac{n.2^b}{C_M}$
- Pro $0 < \varepsilon \leq 1$ volíme ..... $b = \lfloor \log \frac{\varepsilon C_M}{n} \rfloor$
- Složitost $O(\frac{n^3}{\varepsilon}) \Rightarrow$ plně polynomiální aproximační schéma.
- Aproximativní algoritmus = přesný algoritmus nad zjednodušeným modelem

FPTAS algoritmus pro batoh

- Instance: n, M, c_1, c_2, …, c_n, w_1, w_2, …, w_n.
- Dána požadovaná relativní chyba ε.

Algoritmus:

- Nechť $C_M = \max\{c_1, c_2, \ldots, c_n\}$.
- Nechť $K = \dfrac{\varepsilon C_M}{n}$
- Pro $i = 1 \ldots n$, nechť $c_i' = \left\lfloor \dfrac{c_i}{K} \right\rfloor$
- Řešení je výstup instance
  n, M, c_1', c_2', …, c_n', w_1, w_2, …, w_n
  získaný pomocí dynamického programování s dekompozicí podle váhy

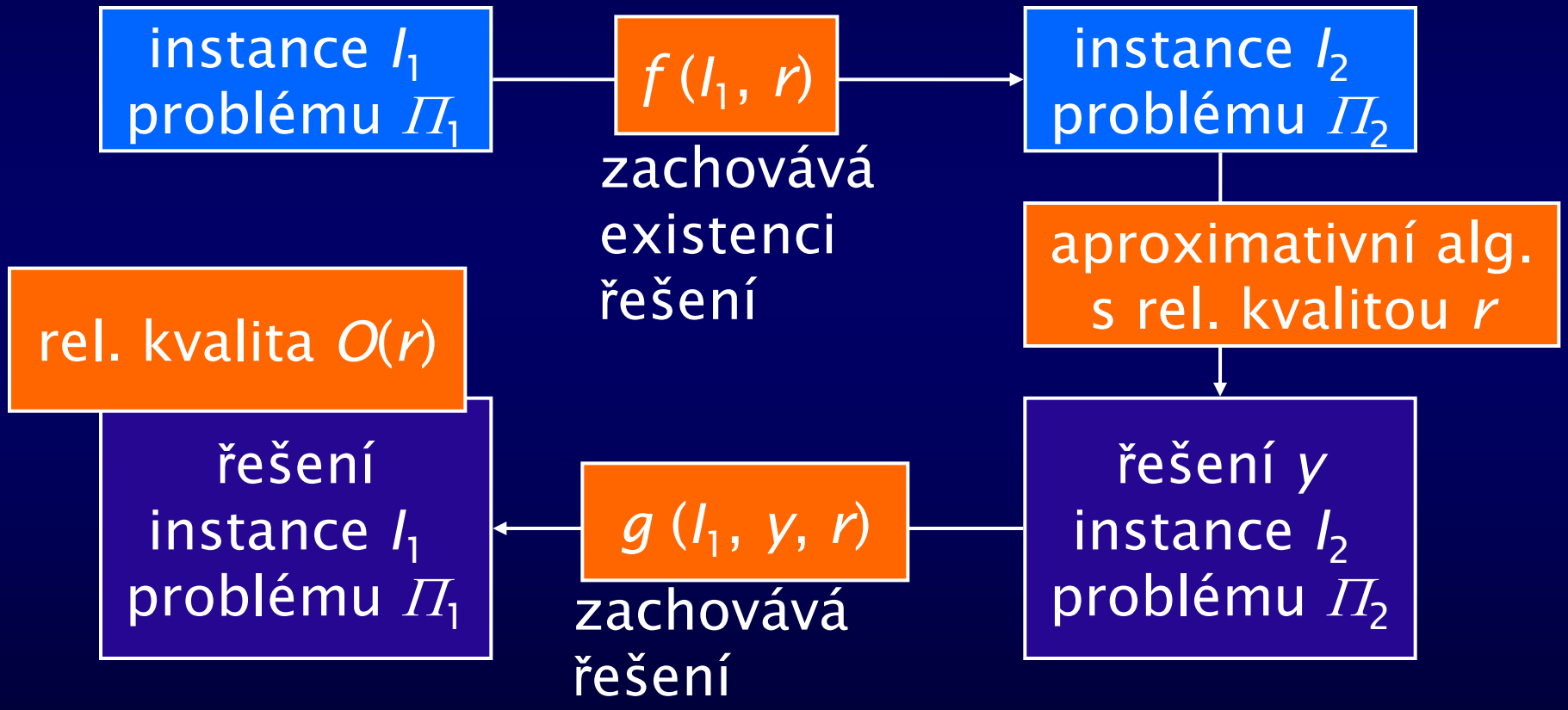
Aproximačně nejtěžší problémy

Úplnost

- Problém *Π* je X-těžký, jestliže se efektivní řešení všech problémů z třídy X dá zredukovat na efektivní řešení problému *Π*.
- Problém *Π* je X-úplný, jestliže je X-těžký a sám patří do třídy X.
- Efektivní řešení: zde: aproximovatelné řešení
- Redukce: zachovává efektivitu, zde: zachovává aproximaci

APX redukce $\Pi_1 \overset{\text{APX}}{\propto} \Pi_2$

Nechť $\Pi_1, \Pi_2 \in$ NPO.



Vlastnosti

- $\Pi_1 \overset{\text{APX}}{\propto} \Pi_2$, $\Pi_2 \in \text{APX} \Rightarrow \Pi_1 \in \text{APX}$
- $\Pi_1 \overset{\text{APX}}{\propto} \Pi_2$, $\Pi_2 \in \text{PTAS} \Rightarrow \Pi_1 \in \text{PTAS}$

Pozor: $\overset{APX}{\propto}$ nikoliv $\propto$ (podle obtížnosti aproximace)

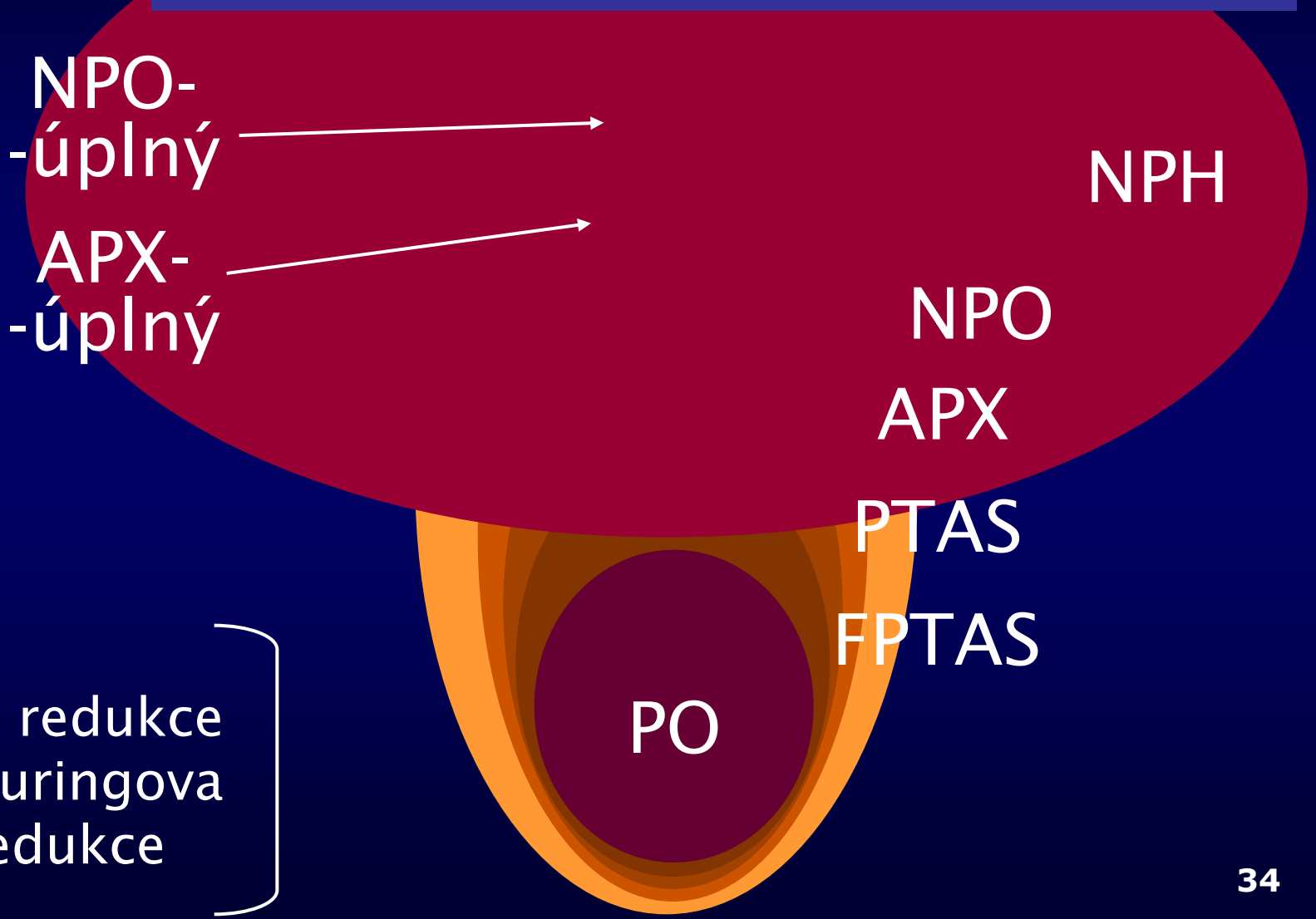
NPO-úplný, -těžký

- Problém Π je NPO-těžký, jestliže $\forall \Pi' \in$NPO, $\Pi' \overset{APX}{\propto} \Pi$.
- Problém Π je NPO-úplný, jestliže je NPO-těžký a $\Pi \in$NPO.

APX-úplný, -těžký

- Problém Π je APX-těžký, jestliže $\forall \Pi' \in$APX, $\Pi' \overset{APX}{\propto} \Pi$.
- Problém Π je APX-úplný, jestliže je APX-těžký a $\Pi \in$APX.

Struktura NPO



APX redukce je Turingova redukce

Problém pokrytí

- Dáno:
  kolekce C podmnožin konečné množiny S.
- Zkonstruovat:
  - podmnožinu $C' \subseteq C$ takovou, že každý prvek S patří do alespoň jedné podmnožiny z C', a dále $|C'| = \min$. (unátní pokrytí)
  - podmnožinu $C' \subseteq C$ takovou, že každý prvek S patří do právě jedné podmnožiny z C', a dále $|C'| = \min$. (binátní pokrytí)

Optimalizační SAT-folklór

		MAX WEIGHTED SAT	MAX SAT	MAX WEIGHTED SAT(!)
vstup	F, X	F, X, W	F, X	F, X, W
konfigurace	Y	Y	Y	Y
výstup	Y	Y	Y	Y
omezení	$F(Y) = 1$	$F(Y) = 1$	---	---
opt. kritérium	max. počet jedniček	max. vážený počet jedniček	max. počet splněných termů	max. vážený počet splněných termů

Dobré a špatné zprávy

Problém	Dobré (*R*)	Špatné
Min. uzlové pokrytí	2-log log |V| / 2log|V| 1985	APX-úplný 1991
Min. (unátní) pokrytí množiny S	1+ln |S| 1974	Není v APX 1993
MAX SAT (počet klauzulí)	1.2987 1997	APX-úplný 1991
MAX WEIGHTED SAT (váha proměnných v 1)		NPO-úplný
Batoh	FPTAS 1975	

Dobré a špatné zprávy

Problém	Dobré	Špatné
TSO		NPO-úplný 1987
ΔTSO (metrický TSO)	R=1.5 Christofides 1976	APX-úplný 1993
Geom. TSO souřadnice $\in \mathbf{Z}$	PTAS 1996	
Geom. TSO souřadnice $\in \mathbf{Q}$		APX-úplný 1997

Randomizované algoritmy

Algoritmus pro MAX *k* SAT

- Booleovská formule *F* v konjunktivní formě proměnných *X*, *k* literálů v každé klauzuli, nalézt ohodnocení *Y* proměnných *X* tak, aby bylo splněno co nejvíce klauzulí.
- Algoritmus: každou proměnnou ohodnotíme 0 nebo 1 se stejnou pravděpodobností.
- Vlastnosti:
  pro splnitelné formule o *c* klauzulích, kde má každá klauzule alespoň *k* literálů, je očekávaná hodnota optimalizačního kritéria

$$\left(1 - \frac{1}{2^k}\right).c$$

maximum je *c*

Důkaz

- Pravděpodobnost, že klauzule o k literálech není splněna ---------- 2^{-k}
- Pravděpodobnost, že klauzule o k literálech je splněna ---------- $1-2^{-k}$
- Očekávaný počet splněných klauzulí ---------- $c.(1-2^{-k})$

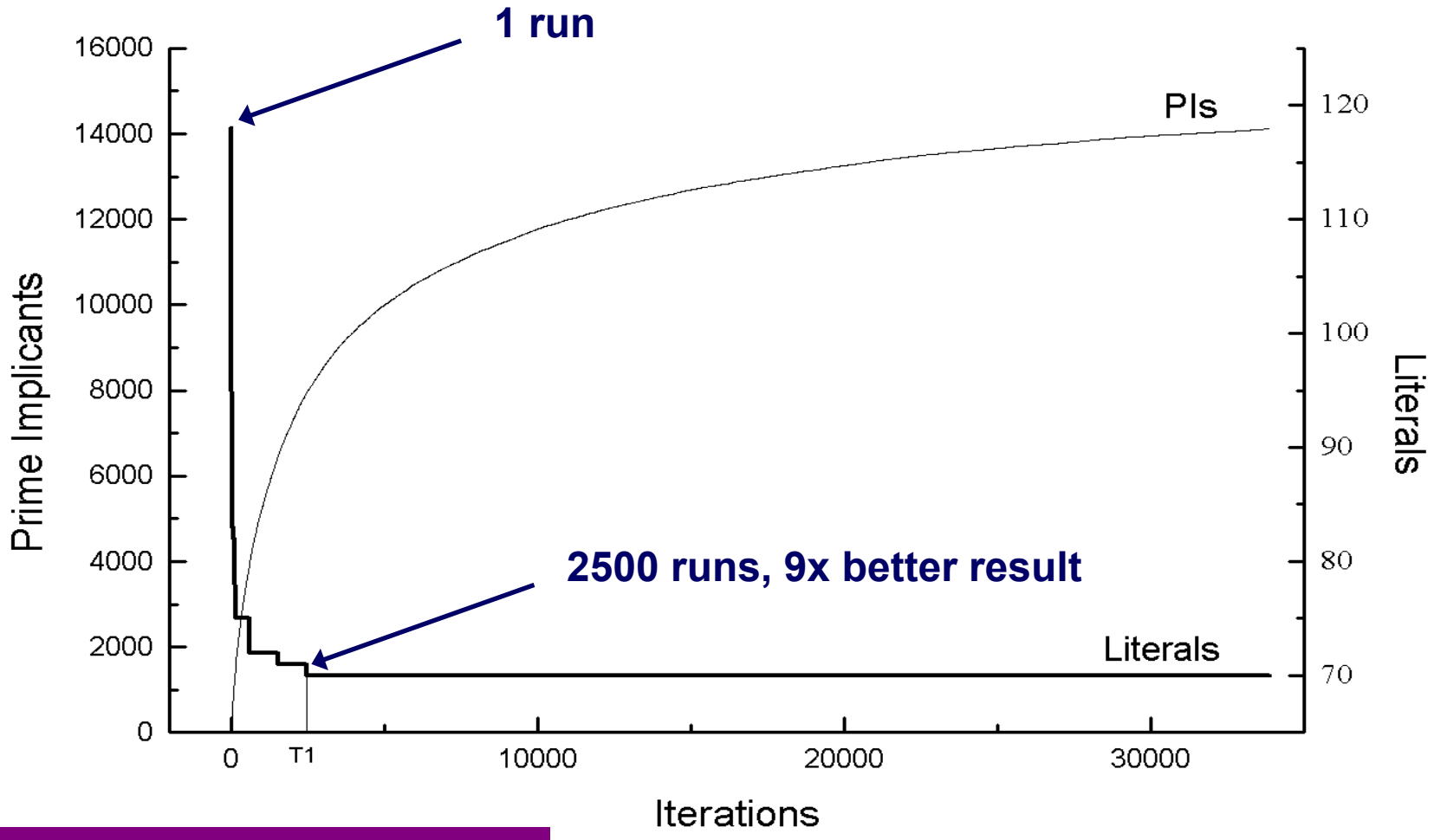
Q.E.D.

$$\left(1-\frac{1}{2^k}\right).c$$

Miller-Rabinův test prvočíselnosti

- Dáno číslo n, zjistit, zda je prvočíslem.
- Malá věta Fermatova:
  Jestliže n je prvočíslo a k přirozené číslo,
  $1 \leq k < n$, pak $k^{n-1} \equiv 1 \pmod{n}$.
- Jestliže pro čísla n, k tvrzení malé věty Fermatovy neplatí, k nazveme svědkem složenosti čísla n.
- Jestliže n je složené číslo, pak ¾ přirozených čísel menších než n jsou svědky složenosti n.
- Otestujeme-li 100 náhodných potenciálních svědků, pak pravděpodobnost, že o n neprávem tvrdíme, že je prvočíslem, je $¼^{100}$.
- Tuto pravděpodobnost můžeme tedy libovolně snižovat.

Minimalizace booleovských výrazů



Petr Fišer

Randomizovaný algoritmus

- Založen na náhodné volbě
- Jeho vlastnosti jsou vyjádřeny statisticky
  - dosažený výsledek (optimalizační kritérium) je náhodná proměnná, čas běhu pevný pro danou instanci → Monte Carlo algoritmy
  - čas běhu je náhodná proměnná, výsledek vždy správný → Las Vegas algoritmy

Monte Carlo: z herny se vypotácíte ráno, ale nevíte, kolik peněz vám zůstane
Las Vegas: vždycky vás oberou na kost, otázka je do kdy

Quicksort

- Výsledek vždy správně (seřazený)
- Čas běhu záleží na volbě pivotu
- Pivot musí být alespoň přibližně správně zvolen
- Typický Las Vegas algoritmus

Hra „Quicksort je k ničemu“

Hráč č. 1 předloží implementaci quicksortu.
Hráč č. 2 se snaží strefit do instance, pro kterou implementace volí pivoty špatně a čas je $O(n^2)$.

Když se to podaří do k pokusů, vyhrává č. 2

Pokud implementace má randomizovanou volbu pivotu, č. 2 může vyhrát jen s velmi malou pravděpodobností

Analýza randomizovaných algoritmů

- Poskytuje očekávanou (střední) hodnotu charakteristické veličiny (kvality, času)
- Platí pro jakýkoli vstup
- Pravděpodobnostní analýza poskytuje průměrné hodnoty při předpokládaném rozložení charakteristik vstupních instancí
- Když se ví jakých

Randomizovaný alg. 3 SAT

Booleovská formule F v konjunktivní formě,
vektor n proměnných X,
3 literály v každé klauzuli, nalézt ohodnocení Y proměnných X tak, aby $F(Y)=1$.

- Algoritmus:
  1. Počáteční ohodnocení Y: každou proměnnou ohodnoť 0 nebo 1 se stejnou pravděpodobností.
  2. Pokud existuje ohodnocení Y', které se liší od Y v právě jedné proměnné a má více splněných klauzulí, pak $Y \leftarrow Y'$ a opakuj 2
- Vlastnosti:
  pro každé $0<\varepsilon<½$ se dá vyjádřit pravděpodobnost, že algoritmus nalezne řešení, jako funkce n a ε.
  Výrok pak platí pro všechny splnitelné formule až na jistou část, jejíž velikost je opět funkcí n a ε.

Randomizovaný 3 SAT

- Kombinace randomizované a deterministické fáze
- Z hlediska heuristických algoritmů, kombinace
  - náhodné konstruktivní fáze
  - deterministické iterativní fáze
- Praktičtější podoba, obecný SAT:
  - algoritmus GSAT
  - zaujatá náhodná procházka

GSAT
(Selman, Levesque, and Mitchell 1992)

- Algoritmus:

1. Počáteční ohodnocení *Y*: každou proměnnou ohodnoť 0 nebo 1 se stejnou pravděpodobností.
2. Najdi ohodnocení *Y'*, které se liší od *Y* v právě jedné proměnné a poskytne nejvíce splněných klauzulí
3. *Y*←*Y'*. Pokud nejsou všechny klauzule splněny nebo vyčerpán stanovaný počet kroků, opakuj 2.
4. Pokud stále nejsou všechny klauzule splněny, opakuj s jiným náhodným počátečním ohodnocením.

Náhodná procházka
(Random Walk)

- Algoritmus:
  1. Počáteční ohodnocení *Y*: každou proměnnou ohodnoť 0 nebo 1 se stejnou pravděpodobností.
  2. Najdi ohodnocení *Y'*, které se liší od *Y* v právě jedné proměnné některé nesplněné klauzule
  3. *Y*←*Y'*. Pokud nejsou všechny klauzule splněny nebo vyčerpán stanovaný počet kroků, opakuj 2.
- Řeší instance 2 SAT V čase $O(n^2)$ (Papadimitriou 1992)
- Nepracuje dobře na 3 SAT

Zaujatá náhodná procházka
(Biased Random Walk)

- Algoritmus:
  1. Počáteční ohodnocení *Y*: každou proměnnou ohodnoť 0 nebo 1 se stejnou pravděpodobností.
  2. S pravděpodobností $0<q<1$ proveď 3 jinak proveď 4.
  3. Najdi ohodnocení *Y'*, které se liší od *Y* v právě jedné proměnné některé nesplněné klauzule
  4. Najdi ohodnocení *Y'*, které se liší od *Y* v právě jedné proměnné a poskytne nejvíce splněných klauzulí
  5. $Y \leftarrow Y'$. Pokud nejsou všechny klauzule splněny nebo vyčerpán stanovaný počet kroků, opakuj 2.

Výhody randomizovaných algoritmů

- strukturní jednoduchost
- očekávaná kvalita výsledku může být lepší než zaručená kvalita aproximativních algoritmů
- nezávislým opakováním se dá zlepšit kvalita

Kombinace s deterministickými prvky: poskytuje nestrannost při vzorkování libovolného souboru prvků:

- každý náhodný start je stejně pravděpodobný
- každý náhodně vybraný sousední stav je stejně pravděpodobný
- každý náhodně vybraný krok z množiny, pro které dává heuristická funkce stejnou hodnotu atd.

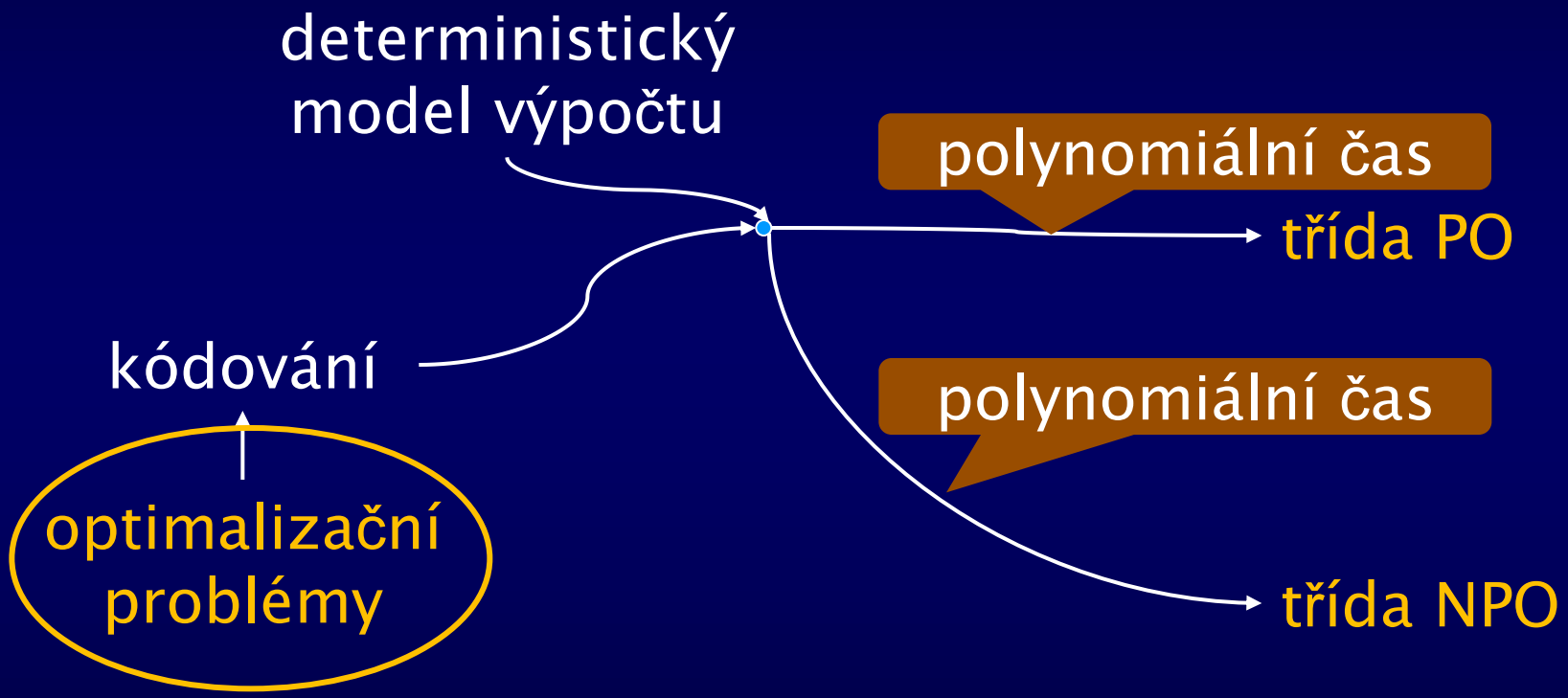
Meze randomizace

- Nechť je čas výpočtu randomizovaným algoritmem $T(n)$.
- Je možno (deterministicky) zkonstruovat posloupnost čísel takovou, že ji žádný algoritmus v čase $T(n)$ nerozezná od náhodné posloupnosti.
- Tudíž, původní randomizovaný algoritmus bude fungovat stejně.

$\Rightarrow$ randomizované paradigma výpočtu není silnější než deterministické

$\Rightarrow$ derandomizace algoritmů

Třídy optimalizačních problémů



Aproximativní algoritmus

Třídy aproximovatelnosti