NP úplnosť

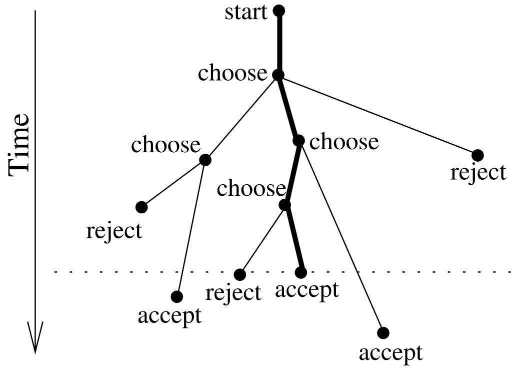
- Rozhodovacie vs. optimalizačné problémy.
- Trieda problémov P.
- Nedeterministické výpočty a trieda problémov NP.
- Polynomiálne transformácie a NP-úplnosť.
- Cookova veta: Existuje NP-úplný problém.
- Ako ukázať, že problém je NP-úplný?
- Základné "portfólio" NP-úplných problémov.

Nedeterministické výpočty

- `accept` – ukonči výpočet s odpoveďou **áno**
- `reject` – ukonči výpočet s odpoveďou **nie**
- `choose k between i and j` – nastav k na hodnotu medzi i a j tak, aby sa program dostal k príkazu `accept` najkratším spôsobom.

Nedeterministické výpočty

- `accept` – ukonči výpočet s odpoveďou **áno**
- `reject` – ukonči výpočet s odpoveďou **nie**
- `choose k between i and j` – nastav k na hodnotu medzi i a j tak, aby sa program dostal k príkazu `accept` najkratším spôsobom.



Nedeterministický algoritmus pre riešenie TSP-D

```
function TSP-D
  visited[i]:=false for all vertices;
  last_visited:=1; visited[1]:=true;
  length:=0;
  repeat n-1 times
    choose next_visited between 1 and n;
    if visited[next_visited] then reject;
    //we cannot visit a single vertex twice
    visited[next_visited]:=true;
    length:=length+w(last_visited,next_visited);
    last_visited:=next_visited;
  length:=length+w(last_visited,1);
  if length<=B then accept;
             else reject;
```

Časová zložitosť nedeterministických algoritmov

Akceptujúci výpočet je výpočet, ktorý skončí príkazom `accept`.

Čas nedeterministického algoritmu A na vstupe x je čas najkratšieho akceptujúceho výpočtu pre vstup x.

Časová zložitosť nedeterministického algoritmu A je funkcia veľkosti vstupu $T_A(n)$, pričom pre veľkosť vstupu n je to najhorší čas nedeterministického algoritmu A spomedzi všetkých vstupov veľkosti n.

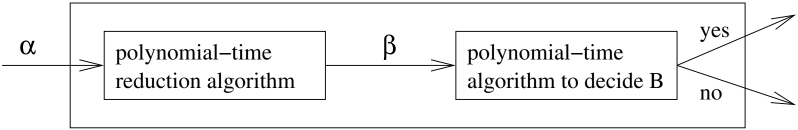
Rozhodovací problém Q patrí do triedy NP vtt ak existuje polynomiálny nedeterministický algoritmus pre Q.

Redukcie

Definícia

Rozhodovací problém A je polynomiálne redukovateľný na problém B ($A \leq_p B$) ak existuje funkcia f vypočítateľná v polynomiálnom čase taká, že:

- zobrazuje každý vstup A na nejaký vstup B
- A odpovedá na x **áno** práve vtedy keď B odpovedá na $f(x)$ **áno**



polynomial–time algorithm to decide A

NP-ťažké a NP-úplné problémy

Problém Q je NP-ťažký akk každý problém $R \in Q$ platí $R \leq_p Q$.

Ak NP-ťažký problém Q patrí do triedy NP, hovoríme, že je NP-úplný.

SAT

Def: $\boxed{\text{SAT}}$ (splniteľnosť) Uvažujme booleovské premenné $(u_1, \ldots, u_m)$ a logickú formulu f.
Problém: Existuje priradenie hodnôt premenných také, aby f bola splnená?

Veta (Cook)

SAT je NP-úplný

Náčrt dôkazu:

Potrebujeme dokázať:

1. SAT$\in$NP –alebo–
   Existuje nedeterministický polynomiálny algoritmus, ktorý rieši SAT.
2. SAT je NP-ťažký –alebo–
   pre ľubovoľný problém $Q \in NP$: $Q \leq_p \mathrm{SAT}$

1 SAT∈NP

```
for i:=1 to m do
  choose u[i] between 0 and 1; // 0 means false,
                               // 1 means true

evaluate formula f with assignment
(u[1],u[2],...,u[m])

if f is satisfied then ACCEPT
                  else REJECT
```

$\boxed{2}$ SAT je NP-ťažký

Uvažujme $Q \in \mathrm{NP}$
$\Longrightarrow$existuje polynomiálny nedeterministický algoritmus, ktorý rieši Q

Ako taký algoritmus zapíšeme?

- Každý register má v sebe uložené číslo konštantnej veľkosti (registre označíme $R_1, R_2, \dots$)
- Program je nemeniaca sa postupnosť príkazov s konštantným počtom očíslovaných riadkov
- Na začiatku je vstup uložený v prvých n registroch (n je veľkosť vstupu)
- Program beží nanajvýš $p(n)$ krokov a pristupuje najviac ku $q(n)$ prvým registrom ($p(n)$ a $q(n)$ sú polynómy závisiace od n)

- Sada inštrukcií:
  - ACCEPT
  - REJECT
  - GOTO m
  - IF $R_\ell = 0$ THEN GOTO m
  - CHOOSE R_l BETWEEN 0 AND 1
  - základné aritmetické operácie
    (napr. $R_\ell := R_u + R_v$, $R_\ell := R_u * R_v$)
  - nejaký mechanizmus na adresáciu prvých $q(n)$ registrov
    (detaily sú mierne komplikované, ale dá sa)

$\boxed{2}$ SAT je NP-ťažký: $Q \leq_p \mathrm{SAT}$

Chceme:

- Daný je program A, ktorý rieši problém Q v polynomiálnom čase
  a inštancia $x = x_1, x_2, \ldots, x_n$.
- Vyrobíme veľkú logickú formulu f, ktorá "simuluje" program A na vstupe x;
- A dosiahne ACCEPT $\iff$ f je splniteľná

Premenné formuly f:

- $Q[i, k]$ – v čase i program vykonáva riadok k
- $S[i, j, k]$ – v čase i má register R_j hodnotu k

Formula f bude konjunkcia ("AND") niekoľkých menších formúl t.j. všetky tieto menšie formuly musia byť splnené, aby formula f bola splnená

$\boxed{1}$ "V každom čase i program vykonáva práve jeden riadok."

$$\boxed{\neg(Q[i,k] \wedge Q[i,\ell])} \quad \text{pre všetky } i \text{ a } k \neq \ell$$

$\boxed{2}$ "V každom čase i každý register obsahuje práve jednu hodnotu."

$$\boxed{\neg(S[i,j,k] \wedge S[i,j,l])} \quad \text{pre všetky } i,j \text{ a } k \neq \ell$$

$\boxed{3}$ V čase 0:

- Program vykonáva riadok 1: $\boxed{Q[0,1]}$
- Prvých n registrov má hodnoty $x_1, \ldots, x_n$:
  $\boxed{S[0,1,x_1] \wedge S[0,2,x_2] \wedge \cdots \wedge S[0,n,x_n]}$
- Ostatné registre majú hodnotu 0:
  $\boxed{S[0,n+1,0] \wedge S[0,n+2,0] \wedge \cdots \wedge S[0,q(n),0]}$

$\boxed{4}$ "Po $p(n)$ krokoch program dosiahne riadok s inštrukciou ACCEPT"

$$\boxed{Q[p(n),k]} \quad k \text{ je riadok s inštrukciou "ACCEPT"}$$

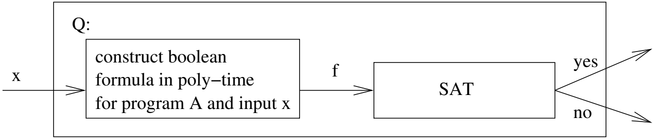
$\boxed{5}$ "Stav počítača sa mení v čase v súlade s programom."

k**-ty riadok**	**Formula**
ACCEPT alebo REJECT	$Q[i,k] \Rightarrow Q[i+1,k]$
GOTO ℓ	$Q[i,k] \Rightarrow Q[i+1,\ell]$
IF $R_\ell = 0$ THEN GOTO m	$Q[i,k] \wedge S[i,\ell,0] \Rightarrow Q[i+1,m]$ $Q[i,k] \wedge \neg S[i,\ell,0] \Rightarrow Q[i+1,k+1]$
CHOOSE R_ℓ	$Q[i,k] \Rightarrow Q[i+1,k+1] \wedge$ $(S[i+1,\ell,0] \vee S[i+1,\ell,1])$
atď. pre ďalšie inštrukcie	

SAT je NP-ťaždký: zhrnutie

Vyššieuvedeným postup skonštruujeme pre daný algoritmus A a vstup x formulu f:

- Postup možno zrealizovať v polynomiálnom čase v závislosti od n.
- Výsledná formula má polynomiálnu veľkosť v závislosti od n.
- f je splniteľná $\iff$ A akceptuje x



$\Longrightarrow$**Ukázali sme:** $Q \leq_p \mathrm{SAT}$ **pre ľubovoľné** $Q \in NP$