Časová složitost

Definition 14.1 (časová složitost)

Mějme Turingův stroj M, který zastaví na každém vstupu. **Časová složitost** M je funkce $f : \mathbb{N} \to \mathbb{N}$, kde $f(n)$ je maximální počet kroků výpočtu M nad vstupy délky n.

Definition 14.2 ((Asymptotická) horní hranice $O(g(n))$)

Mějme funkce $f, g : \mathbb{N} \to \mathbb{R}^+$. Říkáme, že $f(n) = O(g(n))$, pokud existují $c, n_0 \in \mathbb{N}^+$ taková, že:

$$\forall n \geq n_0 \text{ platí } f(n) \leq c \cdot g(n).$$

V takovém případě říkáme, že $g(n)$ je (asymptotická) **horní hranice** pro $f(n)$. (Slovo asymptotická vyjadřující ignorování prvních n_0 i konstanty c se zpravidla vynechává.)

Reálná čísla jsou tam kvůli logaritmu.

- Např. $f(5n^3 + 2n^2 + 22n + 6) = O(n^3)$ s $n_0 = 10$, $c = 6$.

Definition 14.3 (třída časové složitosti)

Mějme funkci $t : \mathbb{N} \to \mathbb{R}^+$. Definujeme **třídu časové složitosti** $TIME(t(n))$ jakožto množinu všech jazyků, které jsou rozhodnutelné Turingovým strojem v čase $O(t(n))$ tj. vždy zastaví, pro vstup délky n nejpozději po $O(t(n))$ krocích, a vydá správnou odpověď.

Example 14.1 ($\{0^i1^i|i \in \mathbb{N}_0\}$ je $O(n^2)$)

Jazyk $\{0^i1^i|i \in \mathbb{N}_0\}$ je $O(n^2)$

1. Zkontroluj vstup 0^i1^j, pokud za 1 je 0, nepřijmi (čas $O(n)$)
2. návrat na začátek se schová v konstantě, $O(2n) = O(n)$
3. procházej postupně 0 v čase $O(n^2)$
   1. přepiš 0 na X
   2. najdi 1 a přepiš na X
   3. vrať se na začátek
4. Když už není 0, ověř že není ani 1 a přijmi (s 1 nepřijmi). (čas $O(n)$)

Jde to rychleji?

Example 14.2 ($\{0^i1^i|i \in \mathbb{N}_0\}$ je $O(n \log n)$)

Jazyk $\{0^i1^i|i \in \mathbb{N}_0\}$ je $O(n \log n)$

1. Zkontroluj vstup 0^i1^j, zkontroluj sudou délku (čas $O(n)$)
2. procházej dokud najdeš 0 v čase $O(n \log n)$
   1. přepiš každou druhou 0 na X
   2. přepiš každou druhou 1 na X
   3. zkontroluj sudost počtu nul a jedniček dohromady a vrať se na začátek
3. Když už není 0, ověř že není ani 1 a přijmi (s 1 nepřijmi). (čas $O(n)$)

Regulární jazyky - jen pro zajímavost

- O moc rychleji to nejde.

Definition ($o()$)

Mějme funkce $f, g : \mathbb{N} \to \mathbb{R}^+$. Říkáme, že $f(n) = o(g(n))$, pokud

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0.$$

Tj. pro každé $c > 0$ existuje n_0 takové, že $(\forall n > n_0)\ (f(n) < cg(n))$.

- Malé o má význam 'ostře menší', velké O menší nebo rovno.

Theorem (just for info)

Každý jazyk rozhodnutelný v čase $o(n \log n)$ na jednopáskovém Turingově stroji je regulární.

Vícepáskový Turingův stroj

Vícepáskový Turingův stroj pro $\{0^n1^n\}$

1: **procedure** $0^N1^N(\ w \in \{0,1\}^*\ )$
2: Kopíruj nuly na pomocnou pásku.
3: Na první jedničce přepni do nového stavu, maž nulu i jedničku.
4: **return** Vymazaly se zároveň nuly i vstupní páska?
5: **end procedure**

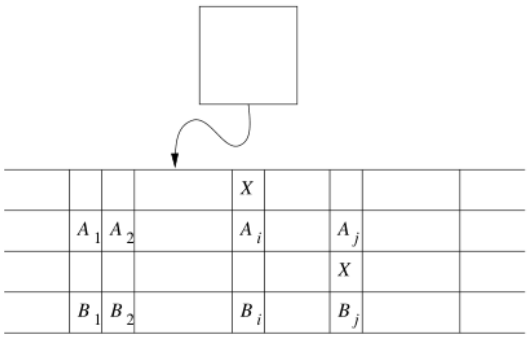
Lemma

Mějme funkci $t : \mathbb{N} \to \mathbb{R}^+$, $t(n) \geq n$. Každý vícepáskový Turingův stroj s časem $t(n)$ má jednopáskový ekvivalent $O(t^2(n))$.

Proof: opakování

- Simulaci výpočtu k–páskového stroje o n krocích lze provést v čase $O(n^2)$ (simulace jednoho kroku z prvních n trvá $4n + 2k$, hlavy nejvýš $2n$ daleko, přečíst, zapsat, posunout značky). □

Simulace 2–páskového TM na jedné pásce



Nedeterministický Turingův stroj

Definition 14.4 (doba běhu nedeterministického TM)

Mějme **nedeterministický** Turingův stroj M, který zastaví na každém vstupu. **Doba běhu** M je funkce $f : \mathbb{N} \to \mathbb{N}$, kde $f(n)$ je maximální počet kroků který M potřebuje v jakékoli větvi výpočtu nad jakýmkoli vstupem délky n.
O takovém nedeterministickém Turingovu stroji M říkáme, že **rozhoduje** jazyk $L(M)$ v čase $f(n)$.

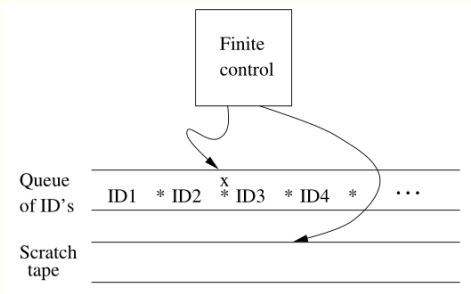
Theorem 14.1

Mějme funkci $t : \mathbb{N} \to \mathbb{R}^+$, $t(n) \geq n$. Každý nedeterministický Turingův stroj s časem $t(n)$ má deterministický ekvivalent $2^{O(t(n))}$.

- Pokud pro dvojici $Q \times \Gamma$ máme maximálně d variant, pak po k krocích se TM může dostat maximálně do d^k konfigurací.
- Jeden krok zvládneme 'schovat do konstanty' k $t(n)$, logaritmus d pro převod také, proto simulace je v čase $O(t(n)d^{t(n)}) = 2^{O(t(n))}$.
- Máme přidat převod simulace více pásek, ale $(2^{O(t(n))})^2 = 2^{O(2t(n))} = 2^{O(t(n))}$.

Proof: opakování - idea důkazu

- páska nekonečná – nelze použít podmnožinovou konstrukci
- prohledáváme do šířky všechny výpočty M_N
- modelujeme TM se dvěma páskami
  - první páska: posloupnost konfigurací
    - aktuální označena (křížkem na obrázku)
    - vlevo už prozkoumané, můžeme zapomenout
    - vpravo aktuální a pak další čekající
  - druhá páska: pomocný výpočet



- zpracování jedné konfigurace obnáší
  - přečti stav a symbol aktuální konfigurace ID
  - je–li stav přijímající $\in F$, přijmi a skonči
  - napiš konfiguraci ID na pomocnou pásku
  - pro každý možný krok δ (uložený v hlavě M_D)
    - proveď krok a napiš novou ID na konec první pásky
  - vrať se k označené ID, značku vymaž a posuň o 1 doprava
  - opakuj

Opakování - Jednosměrná páska

X_0	X_1	X_2	...
*	X_{-1}	X_{-2}	...

Lemma (Jednosměrná páska)

Pro každý Turingův stroj M_2 existuje Turing. stroj M_1, který přijímá stejný jazyk a

- *M_1 nikdy nejde vlevo od počáteční pozice*
- *M_1 nikdy nepíše blank B.*

Proof.

- Místo B blank zavedeme nový páskový symbol, B_1.
  - Místo každého psaní B píšeme B_1 a všechny instrukce pro čtení B zkopírujeme též pro čtení B_1.
  - Takto modifikovaný TM nikdy nepíše B (píše B_1).
- Pro jednosměrnou pásku
  - Nejdřív přepíšeme vstup, aby byl v horní stopě dvoustopé pásky.
  - Pod nejlevější symbol dáme nový znak $*$, abychom věděli, že jsme na levém okraji, a máme přepnout z horní stopy do dolní. Ve stavu ('hlavě') si pamatujeme, jestli čteme horní stopu ('normálně') nebo spodní stopu (kde L znamená doprava a R doleva). Pokud vidíme $*$, odpovídající instrukce přepíšeme na změnu

Definition 14.5 (třída P)

Definujeme P ($PTIME$) **třídu jazyků rozhodnutelných v polynomiálním čase** jednopáskovým deterministickým Turingovým strojem. Tedy:

$$P = \bigcup_k TIME(n^k).$$

Theorem 14.2 ($CFL \subseteq P$)

Každý bezkontextový jazyk patří to P.

Proof.

- Na vstup si vyžádáme gramatiku v ChNF.
- CYK algorithmus je polynomiální ($O(n^3)$).

□

Example 14.3

- Cesta v grafu *PATH*
- Nesoudělná čísla *RELPRIME*
  - Repeat until $y = 0$

Verifikátory, třída *NP*

Definition 14.6 (Verifikátor, NP)

- **Verifikátor** jazyka L je algoritmus V, kde:

$$L = \{w \mid V \text{pro nějaké } c \in \Sigma^* \text{ přijímá } \langle w, c \rangle\}.$$

- Časová složitost verifikátoru se měří pouze vzhledem k délce w, **polynomiální verifikátor** rozhoduje v čase polynomiálním vzhledem k $|w|$.
- Jazyk L je **polynomiálně verifikovatelný**, pokud má polynomiální verifikátor. Pak i certifikát musí být polynomiální, aby o verifikátor stihl přečíst.
- **Třída jazyků rozhodnutelných v polynomiálním čase** *NP* je tvořena jazyky s polynomiálním verifikátorem.
- Nápověda c pro snadné ověření se nazývá **certifikát**.

Hamiltonovská cesta

Example 14.4

$HAMPATH = \{\langle G, s, t\rangle | G$ je orientovaný graf s hamiltonovskou cestou z s do $t\}$.

Hamiltonovská cesta je taková (orientovaná) cesta, která obsahuje každý vrchol grafu právě jednou.

- Složitost u grafů bereme vůči počtu uzlů, počet hran je max. kvadratický, tj. polynomiální.
- Náš certifikát je posloupnost vrcholů cesty.
- Algoritmus v polynomiálním čase ověří, že jde o hamiltonovskou cestu.
- $\overline{HAMPATH}$, množina orientovaných grafů, pro které neexistuje hamiltonovská cesta, neznáme verifikátor (jen umíme ověřit v exponenciálním čase).

Třída *NP*

Definition 14.7 (*NP*)

Mějme funkci $t : \mathbb{N} \to \mathbb{R}^+$. Definujeme třídu

$$NTIME(t(n)) = \{L \mid L \text{ jazyk rozhodnutelný nedeterminist. TM v čase } O(t(n)).\}$$

Definition (část definice 14.6)

Třída jazyků rozhodnutelných v polynomiálním čase *NP* je tvořena jazyky s polynomiálním verifikátorem.

Theorem 14.3

$$NP = \bigcup_k NTIME(n^k).$$

- Idea důkazu: převedeme verifikátor na NTM a opačně.
- NTM uhodne certifikát a simuluje verifikátor.
- Verifikátor bere přijímající větev NTM jakožto certifikát.

verif. $\Rightarrow \bigcup$.

- Předpokládáme $L \in NP$.
- Hledáme nedeterministický TM M.
- Vezmeme verifikátor V z definice NP. Nechť rozhoduje L v čase n^k.
- M na vstupu w délky n:
  - Nedeterministicky uhodne řetězec c délky nanejvýš n^k.
  - Spustí V na vstupu $\langle w, c \rangle$.
  - Pokud V přijme, M také přijme, jinak nepřijímá.

□

$\bigcup \Rightarrow$ verifikátor .

- Předpokládáme $L = L(M)$ rozhodnutelný nějakým NTM M v polynomiálním čase.
- Hledáme verifikátor V.
- V na vstupu $\langle w, c \rangle$:
  - Simuluje M na vstupu w, v bodech větvení vybere větev podle c.
  - Pokud tato větev NTM přijme, V přijme, jinak nepřijímá.

□

Example 14.5 (*CLIQUE* je NP)

Problém k-kliky *CLIQUE* je v daném grafu G určit, jestli existuje klika velikosti k, tj.

$$CLIQUE = \{\langle G, k\rangle | G \text{ je neorientovaný graf s klikou velikosti } k\}.$$

Verifikátor pro *CLIQUE*

1: **procedure** CLIQUE_VERIFIER($\langle\langle G, k\rangle, c\rangle$)
2: Otestuj, jestli c je podgraf G o k uzlech.
3: Otestuj, jestli je c klika, tj. úplný podgraf, má všechny hrany.
4: **return** Oba předchozí testy TRUE?
5: **end procedure**

Nedeterministický Turingův stroj pro *CLIQUE*

1: **procedure** CLIQUE_NTM($\langle G, k\rangle$)
2: Nedeterministicky vyber k prvkovou podmnožinu c vrcholů G.
3: **return** Je c klika? tj. úplný podgraf, má všechny hrany.
4: **end procedure**

Převoditelnost v polynomiálním čase

Definition 14.8 (polynomiálně vyčíslitelná funkce)

Funkci $f : \Sigma^* \to \Sigma^*$ nazveme **polynomiálně vyčíslitelnou**, pokud existuje Turingův stroj M, který pro každý vstup w v polynomiálním čase zastaví s $f(w)$ na pásce.

Jazyk A je **převoditelný v polynomiálním čase** na jazyk B, $A \leq_P B$, pokud existuje funkce $f : \Sigma^* \to \Sigma^*$ vyčíslitelná v polynomiálním čase a pro každé $w \in \Sigma^*$

$$w \in A \Leftrightarrow f(w) \in B.$$

Funkci f pak nazýváme **polynomiální redukci A do B**.

Definition 14.9 (SAT, 3SAT)

Formuli ϕ nazveme **3-cnf formule**, pokud je formule výrokové logiky v CNF, kde v každé klauzuli jsou právě tři literály.
Formule **je splnitelná**, pokud existuje takové ohodnocení výrokových proměnných, že je hodnota formule TRUE.
Problém **3SAT** je pro každou 3-cnf formuli rozhodnout, zda je splnitelná, tj.

$$3SAT = \{\phi \mid \phi \text{ je splnitelná 3-cnf formule}\}.$$

Problém **SAT** je pro každou booleovskou formuli v rozhodnout, zda je splnitelná

$$SAT = \{\phi \mid \phi \text{ je splnitelná formule}\}.$$

Theorem 14.4

3SAT je polynomiálně převoditelný na CLIQUE.

Proof.

- Každý výskyt proměnné - uzel grafu.
- Hrany všude, jen ne:
  - mezi uzly z téže klauzule
  - mezi proměnnou a její negací x a $\neg x$.

NP úplnost

Definition 14.10 (*NP* úplnost)

Jazyk B je *NP* **úplný**, pokud je NP a každý jazyk $A \in NP$ je na B polynomiálně převeditelný.

Theorem 14.5

Pokud B je NP-úplný a $B \in P$, pak $P = NP$.

Proof.

Přímý důsledek definice polynomiální převoditelnosti a NP. □

Theorem 14.6

Pokud B je NP-úplný a $B \leq_P C$ pro nějaké $C \in NP$, pak C je NP-úplný.

Proof.

Převod problému na B dále převedeme na C, stačí polynomiální čas. □

Cook-Levin-ova věta

Theorem 14.7 (Cook-Levin-ova věta)

SAT je NP-úplný.

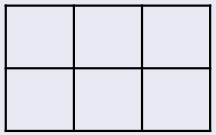
- idea důkazu úplnosti: převedeme výpočet Turingova stroje na SAT.

Proof

- SAT is NP
  - Nedeterministický TM uhodne správné ohodnocení a v polynomiálním čase ověří, je je pro něj formule pravdivá.
- SAT je NP-úplný
  - Vezmeme libovolný $L \in NP$
  - nechť M je nedeterministický TM který rozhoduje jazyk L v čase $n^k - 3$ pro nějaké k.
  - Vytvoříme tabulku (tableau) $n^k \times n^k$, každý řádek odpovídá konfiguraci M na vstupu w
  - můžeme předpokládat (opatřit) každou konfiguraci ohraničenou zarážkami #.

#	q_0	w_1	w_2	...	w_n	_	_	...	#
#									#
#	⋮								#
#	⋮								#
#									#

  - Výpočet budeme skládat po okýnkách 2×3.



Proof:

- Vybraná dovolená okénka, $(a, b, c, d \in \Gamma)$

$\delta(q_1, b) \ni (q_2, c, L)$

a	q_1	b
q_2	a	c

$\delta(q_1, b) \ni (q_2, c, R)$

a	q_1	b
a	c	q_2

$\delta(q_1, b) \ni (q_2, c, R)$

d	a	q_1
d	a	c

přenos beze změny

#	a	b
#	a	b

$\delta(\_, \_) \ni (q_2, \_, L)$

a	b	c
a	b	q_2

$\delta(\_, \_) \ni (\_, c, L)$

a	b	d
c	b	d

  - Přenos beze změny všude, kde není v okolí stav (hlava)
  - na každém řádku nejvýše jeden stav
  - okénko musí být částí povoleného přechodu
  - rozbor technický, udělali za nás jiní.
  - **Tvrzení:** Pokud je první řádek tabulky počáteční konfigurace a každé okénko je legální, pak každý řádek odpovídá legální konfiguraci dosažitelné jedním krokem z předchozího řádku.
    - V horním řádku není stav, pak se prostřední symbol musí opsat beze změny.
    - V horním řádku stav vprostřed: okénko garantuje korektnost přepisu obou stran.

□

- Z tabulky vytvoříme formuli $\phi = \phi_{cell}$ & ϕ_{start} & ϕ_{move} & ϕ_{accept}.
- pro každé políčko tabulky (i, j) a písmeno $a \in \Gamma \cup Q \cup \{\#\}$ vytvoříme výrokovou proměnnou $x_{i,j,a}$

$$\phi_{cell} = \bigwedge_{1 \le i,j \le n^k} \left[\left(\bigvee_{a \in \Gamma \cup Q \cup \{\#\}} x_{i,j,a} \right) \& \bigwedge_{s \neq t \in \Gamma \cup Q \cup \{\#\}} \left(\overline{x_{i,j,s}} \vee \overline{x_{i,j,t}} \right) \right] \quad \#\text{právě jedno}$$

$$\phi_{start} = x_{1,1,\#} \& x_{1,2,q_0} \& x_{1,3,w_1} \& x_{1,4,w_2} \& \ldots \& x_{1,n+2,w_n} \& x_{1,n+3,\_} \& \ldots \& x_{1,n^k,\#}$$

$$\phi_{accept} = \bigvee_{1 \le i,j \le n^k} x_{i,j,q_{accept}}$$

- Celková formule ϕ_{move} bude konjunkce, že každé okénko s horním středem na pozici i, j je legální

$$\phi_{move} = \bigwedge_{1 \le i < n^k, 1 < j < n^k} \phi_{i,j} \qquad \# \text{ okénko } (i,j) \text{ je legální}$$

- legálnost okénka (i, j) zajistíme disjunkcí legálních okének

$$\phi_{i,j} = \bigvee_{(a_1,\ldots,a_6) \in LEGAL} (x_{i,j-1,a_1} \& x_{i,j,a_2} \& x_{i,j+1,a_3} \& x_{i+1,j-1,a_4} \& x_{i+1,j,a_5} \& x_{i+1,j+1,a_6})$$

- **Tvrzení:** Převod má polynomiální složitost, konkrétně $O(n^{2k} \log n)$.
  - $\phi_{cell} \in O(n^{2k})$, procházíme dvojice buněk
  - $\phi_{start} \in O(n^2)$, procházíme první řádek
  - $\phi_{move}, \phi_{accept} \in O(n^{2k})$,
    - $|\delta|$ je pro nás konstanta, protože nezávisí na délce vstupu
    - počet buněk n^{2k}, pro každou konstantní velikost formule.
- pro šťouraly $\log n$ pro zápis indexu proměnných, jehož délka závisí na n.

SAT je *NP*-úplný.

Theorem 14.8

SAT je NP-úplný.

Proof.

- Upravíme předchozí převod na formuli $3SAT$.
- V CNF je skoro vše, kromě disjunkce okének pohybu, která jsou konjunkcí,
  - převedeme do CNF
  - stačí polynomiální (konstantní) čas - velikost podformule závisí jen na stroji N, nikoli na délce vstupu
- v krátkých klauzulích zkopírujeme proměnnou
- dlouhé rozdělíme zavedením nových proměnných.

□

co-NP, Tautologičnost

Definition 15.1

Jazyk $L \subseteq \Sigma^*$ patří do třídy **co-NP** právě když jeho doplněk $\Sigma^* - L$ patří do NP.

- P je částí NP i co-NP.
- Domníváme se, že NP-úplné problémy nejsou v co-NP.
  - pokud $P = NP$, tak jsou.

Definition 15.2 (tautologičnost)

Problém, zda je daná formule výrokové logiky, nazýváme **tautologičnost TAUT**.

Theorem 15.1

Problém tautologičnosti TAUT je co-NP.

- Důkaz z pozorování, že doplněk TAUT je SAT a SAT je v NP.
- Doplněk SAT je otázka, jestli negace dané formule je tautologie.
- Doplněk SAT je co-NP.
- SAT rozhoduje všechny formule, tedy i jejich negace.

Prostorová složitost

- Podobně jako časovou složitost měříme prostor potřebný k výpočtu.
- Turingův stroj je jednoduchý a dost podobný reálným počítačům, proto je často používán k definici tříd složitosti.

Definition 15.3 (prostorová složitost)

Pro deterministický Turingův stroj M, který zastaví na každém vstupu, je **prostorová složitost** M funkce $f : \mathbb{N} \to \mathbb{N}$, kde $f(n)$ je maximální počet buněk pásky, které M přečte při jakémkoli vstupu délky n.
Pro nedeterministický Turingův stroj M, který všechy větve výpočtu zastaví na každém vstupu, je **prostorová složitost** M je funkce $f : \mathbb{N} \to \mathbb{N}$, kde $f(n)$ je maximální počet buněk pásky, které M přečte při jakémkolivstupu délky n na libovolné větvi výpočtu.

Třídy prostorové složitosti

Definition 15.4 (třídy prostorové složitosti)

Mějme funkci $t : \mathbb{N} \to \mathbb{R}^+$. Definujeme **třídy prostorové složitosti** $SPACE(f(n))$ a $NSPACE(f(n))$:

- $SPACE(f(n)) =$
  $\{L \mid L$je jazyk rozhodnutelný v prostoru $O(f(n))$ deterministickým TM$\}$,
- $NSPACE(f(n)) =$
  $\{L \mid L$je jazyk rozhodnutelný v prostoru $O(f(n))$ nedeterministickým TM$\}$.

Theorem 15.2

$$P \subseteq NP \subseteq PSPACE \subseteq NPSPACE \subseteq EXPTIME = \bigcup_k TIME(2^{n^k}).$$

Přehled kapitol

1. Úvod, Iterační lemma pro reg. jazyky
2. Redukovaný DFA a ekvivalence automatů, stavů
3. Nedeterministické ϵ–NFA, Operace zachovávající regularitu
4. Regulární výrazy, Kleeneova věta, Substituce, Homomorfizmus
5. Dvousměrné FA, Mealy a Moore stroje
6. Gramatiky, Chomského hierarchie, víceznačnost
7. Chomského NF, Pumping Lemma pro CFL,CYK – náležení do CFL
8. Zásobníkové automaty, Deterministické PDA
9. Uzávěrové vlastnosti, Dykovy jazyky
10. Turingův stroj, rozšíření
11. Lineárně omezené automaty, Univerzální TM, Diagonální jazyk
12. Nerozhodnutelné problémy, Postův korespondenční p.
13. Časová složitost
14. co-NP, Prostorová složitost