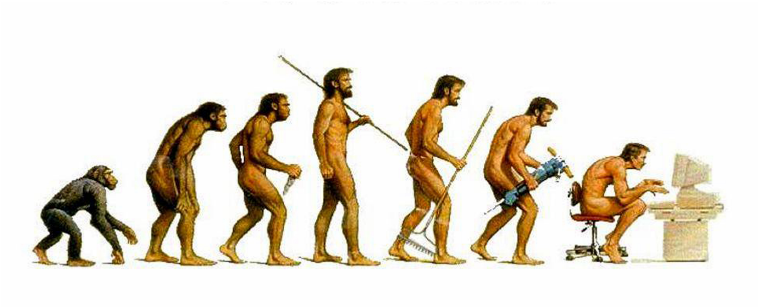


Teoria Obliczalności

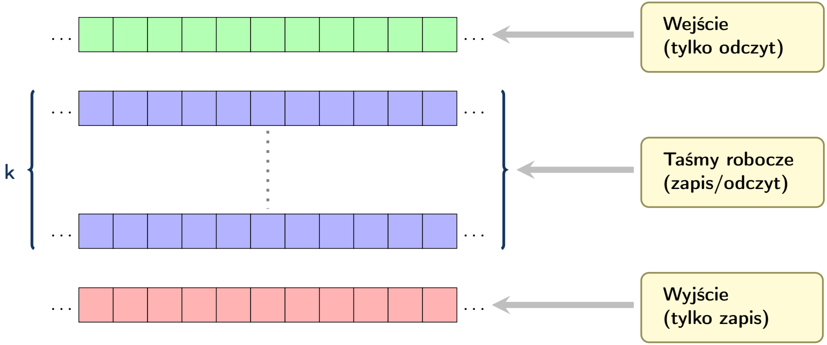
Marcin Piątkowski

Wykład 10

Pamięciowa złożoność obliczeniowa

Model

Maszyna (k+2)-taśmowa



Pamięciowa złożoność obliczeniowa

Złożoność deterministyczna

Złożonością pamięciową **deterministycznej** maszyny Turinga M nazywamy funkcję $f : \boldsymbol{N} \to \boldsymbol{N}$, gdzie $f(n)$ jest równa **maksymalnej** liczbie komórek taśm roboczych wykorzystywanych (odczyt/zapis) przez maszynę M uruchomioną dla **dowolnego** słowa długości n.

Złożoność niedeterministyczna

Złożonością pamięciową **niedeterministycznej** maszyny Turinga M nazywamy funkcję $f : \boldsymbol{N} \to \boldsymbol{N}$, gdzie $f(n)$ jest równa **maksymalnej** liczbie komórek taśm roboczych wykorzystywanych (odczyt/zapis) przez maszynę M na **dowolnej** ścieżce obliczeń po uruchomieniu na **dowolnym** słowie długości n.

Klasy pamięciowej złożoności obliczeniowej

$\mathbf{SPACE}(\mathbf{f}(\mathbf{n}))$ — zbiór wszystkich problemów rozstrzyganych w pamięci $O(f(n))$ przez **deterministyczne** maszyny Turinga

$\mathbf{NSPACE}(\mathbf{f}(\mathbf{n}))$ — zbiór wszystkich problemów rozstrzyganych w pamięci $O(f(n))$ przez **niedeterministyczne** maszyne Turinga

Przykład

Wyrażenie składające się ze zmiennych oraz operacji $\neg$, $\vee$, $\wedge$

$$SAT = \{\phi : \ \phi - \text{spełnialna formuła logiczna}\}$$

✓ $\phi_1 = (\neg x \vee y) \wedge (z \vee \neg y) \wedge (x \vee \neg z)$

✓ $\phi_2 = (x \vee \neg x) \wedge (y \vee \neg y) \wedge (z \vee \neg z)$

✗ $\phi_3 = (x \vee \neg y) \wedge (\neg x \vee y) \wedge (x \vee y) \wedge (\neg x \vee \neg y)$

Przykład

$$SAT = \{\phi : \ \phi - \text{spełnialna formuła logiczna}\}$$

Dla formuły ϕ rozmiaru n zawierającej m różnych zmiennych:

- ☞ Generuj kolejno wszystkie wartościowania zmiennych $x_1, \ldots, x_m$
- ☞ Dla każdego wartościowania oblicz wartość formuły ϕ
- ☞ Jeśli ϕ ma wartość **true** $\Longrightarrow$ **akceptuj**
- ☞ Jeśli dla żadnego wartościowania ϕ nie miała wartości **true** $\Longrightarrow$ **odrzuć**

Przykład

$$SAT \;=\; \{\phi :\; \phi - \text{spełnialna formuła logiczna}\}$$

Dla formuły ϕ rozmiaru n zawierającej m ró…

- Generuj kolejno wszystk… $x_1, \ldots, x_m$
- Dla każd… ość formuły ϕ
- Jeśli ϕ … $\Longrightarrow$ **akceptuj**
- Jeśli dla …ego wartościowania ϕ nie miała wartości **true** $\Longrightarrow$ **odrzuć**

!

$$\left.\begin{array}{ll}\textbf{Czas:} & O(2^n) \\ \textbf{Pamięć:} & O(n)\end{array}\right\} \Longrightarrow SAT \in SPACE(n)$$

Twierdzenie (Savitch – 1970)

Niech $f : \mathbb{N} \longrightarrow \mathbb{R}^+$ będzie funkcją spełniającą warunek $f(n) \geq log(n)$.

Wówczas zachodzi inkluzja:

$$NSPACE\big(f(n)\big) \subseteq SPACE\big(f^2(n)\big).$$

Osiągalność konfiguracji

REACH(c_1, c_2, t)

t = 0

$$\begin{cases} \textbf{true} & \text{jeśli} \quad c_1 = c_2 \\ \textbf{false} & \text{jeśli} \quad c_1 \neq c_2 \end{cases}$$

t = 1

$$\begin{cases} \textbf{true} & \text{jeśli} \quad c_1 = c_2 \;\; \text{lub} \;\; c_1 \to c_2 \\ \textbf{false} & \text{w przeciwnym przypadku} \end{cases}$$

t > 1

Dla każdej poprawnej konfuguracji c_m:

- Wywołaj REACH$\left(c_1,\ c_m,\ \left\lfloor \frac{t}{2} \right\rfloor\right)$
- Wywołaj REACH$\left(c_m,\ c_2,\ \left\lceil \frac{t}{2} \right\rceil\right)$
- Zwróć **true** jeśli oba wywołania zwróciły **true**

Zwróć **false** (jeśli dla żadnej konfiguracji nie było **true**)

Osiągalność konfiguracji

REACH(c_1, c_2, t)

t = 0

$$\begin{cases} \textbf{true} & \text{jeśli} \quad c_1 = c_2 \\ \textbf{false} & \text{jeśli} \quad c_1 \neq c_2 \end{cases}$$

t = 1

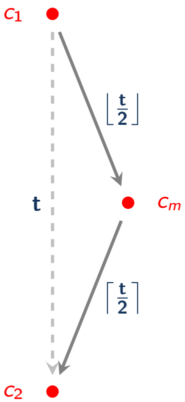
$$\begin{cases} \textbf{true} & \text{jeśli} \quad c_1 = c_2 \ \text{ lub } \ c_1 \to c_2 \\ \textbf{false} & \text{w przeciwnym przypadku} \end{cases}$$

t > 1

Dla każdej poprawnej konfuguracji c_m:

- Wywołaj REACH$\left(c_1, c_m, \left\lfloor \frac{t}{2} \right\rfloor\right)$
- Wywołaj REACH$\left(c_m, c_2, \left\lceil \frac{t}{2} \right\rceil\right)$
- Zwróć **true** jeśli oba wywołania zwróciły **true**

Zwróć **false** (jeśli dla żadnej konfiguracji nie było **true**)



M_1 – maszyna niedeterministyczna działająca w pamięci $O(f(n))$

- Zakładamy, że M_1 ma dokładnie jedną konfigurację akceptującą c_{ACC}
- c_0 – konfiguracja początkowa maszyny M_1
- Każda ścieżka w drzewie obliczeń M_1 wykorzystuje pamięć rozmiaru $O(f(n))$ oraz może składać się z $2^{O(f(n))}$ kroków obliczeń
- Górne ograniczenie liczby konfiguracji $M_1 \implies 2^{d \cdot f(n)}$ $(d \in \mathbb{N})$

Symulacja

M_1 – maszyna niedeterministyczna działająca w pamięci $\mathrm{O}(\mathrm{f}(\mathrm{n}))$

- Zakładamy, że M_1 ma dokładnie jedną konfigurację akceptującą c_{ACC}
- c_0 – konfiguracja początkowa maszyny M_1
- Każda ścieżka w drzewie obliczeń M_1 wykorzystuje pamięć rozmiaru $O(f(n))$ oraz może składać się z $2^{O(f(n))}$ kroków obliczeń
- Górne ograniczenie liczby konfiguracji $M_1 \implies 2^{d \cdot f(n)}$ $(d \in \mathbb{N})$

M_2 – maszyna deterministyczna symulująca działanie M_1

- Symulacja działania $M_1 \implies \mathrm{REACH}\left(c_0,\ c_{ACC},\ 2^{d \cdot f(n)}\right)$
- Każde wywołanie $\mathrm{REACH}(c_1, c_2, t)$ wymaga zapamiętania c_1, c_2 oraz t
  $\implies O(f(n))$ dodatkowej pamięci
- Głębokość rekurencji $\implies O\left(\log\left(2^{d \cdot f(n)}\right)\right) = O(f(n))$
- Pamięć wykorzystywana przez $M_2 \implies O(f^2(n))$

$$PSPACE = \bigcup_k SPACE(n^k)$$

$$NPSPACE = \bigcup_k NSPACE(n^k)$$

$$SAT = \{\phi : \ \phi - \text{spełnialna formuła logiczna}\}$$

$$PSPACE = \bigcup_k SPACE(n^k)$$

$NPSPA$ … $\bigcup_k NSPACE(n^k)$

! Na mocy twierdzenia Savitcha $PSPACE = NPSPACE$

$$SAT = \{\phi : \phi - \text{spełnialna formuła logiczna}\}$$

- **Formuła logiczna z kwantyfikatorami**
  wyrażenie składające się ze zmiennych, operatorów $\neg$, $\vee$, $\wedge$ oraz kwantyfikatorów $\forall$, $\exists$
- **Formuła bez zmiennych wolnych**
  każda zmienna jest w zasięgu pewnego kwantyfikatora
- **Formuła w preneksowej postaci normalnej**
  wszystkie kwantyfikatory występują na początku ϕ

$$\phi \;=\; \forall_x \, \exists_y \, \forall_z \, \Big(z \wedge (x \vee y) \wedge (\neg x \vee \neg y)\Big)$$

$$TBQF = \left\{\phi : \ \phi - \text{prawdziwa formuła logiczna bez zmiennych wolnych}\right\}$$

1. ϕ nie zawiera kwantyfikatorów

   Oblicz wartość ϕ i odpowiednio **akceptuj** lub **odrzuć**

2. $\phi \ = \ \exists_x \ \psi$

   $$\left.\begin{array}{l} \text{Oblicz wartość } \psi \text{ dla } x = \textbf{true} \\ \text{Oblicz wartość } \psi \text{ dla } x = \textbf{false} \end{array}\right\} \implies \textbf{akceptuj/odrzuć}$$

3. $\phi \ = \ \forall_x \ \psi$

   $$\left.\begin{array}{l} \text{Oblicz wartość } \psi \text{ dla } x = \textbf{true} \\ \text{Oblicz wartość } \psi \text{ dla } x = \textbf{false} \end{array}\right\} \implies \textbf{akceptuj/odrzuć}$$

$$TBQF = \left\{\phi :\ \phi - \text{prawdziwa formuła logiczna bez zmiennych wolnych}\right\}$$

1. ϕ nie zawiera kwantyfikatorów

   Oblicz wartość ϕ i odpowiednio **akceptu**…

2. $\phi = \exists_x \psi$

   Oblicz w… …**odrzuć**

3. …

   … ψ dla $x =$ **true**
   Oblicz wartość ψ dla $x =$ **false** $\Longrightarrow$ **akceptuj**/**odrzuć**

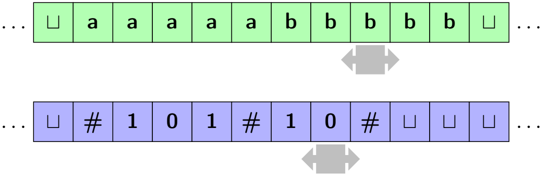
!

- ☞ Głębokość rekurencji ograniczona przez rozmiar ϕ
- ☞ Wywołanie rekurencyjne $\Longrightarrow$ zapamiętanie wartości zmieniej x
- ☞ Całkowity rozmiar dodatkowej pamięci $\Longrightarrow$ $O(n)$

Klasa L (LSPACE)

$$L = SPACE\big(log(n)\big)$$

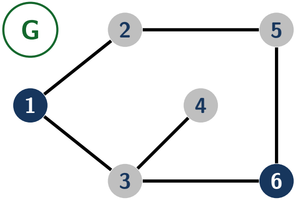
$$L \;=\; \left\{a^k b^k : \; k \geq 0\right\}$$

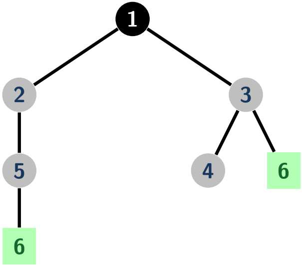


Klasa NL (NLSPACE)

$$NL = NSPACE(\log(n))$$

$$PATH = \Big\{(G, v_1, v_2) : \text{ w grafie } G \text{ istnieje ścieżka } v_1 \rightsquigarrow v_2\Big\}$$





Klasy EXPSPACE oraz NEXPSPACE

$$\mathit{EXPSPACE} \; = \; \bigcup_k \mathit{SPACE}\left(2^{n^k}\right)$$

$$\mathit{NEXPSPACE} \; = \; \bigcup_k \mathit{NSPACE}\left(2^{n^k}\right)$$

Klasy EXPSPACE oraz NEXPSPACE

$$EXPSPACE = \bigcup_k SPACE\left(2^{n^k}\right)$$

$$NEXPSPACE = \bigcup_k NSPACE\left(2^{n^k}\right)$$

! Na mocy twierdzenia Savitcha $EXPSPACE = NEXPSPACE$

Uogólnione wyrażenia regularne

$\varnothing \qquad \varepsilon \qquad \mathbf{R_1 \cdot R_2} \qquad \mathbf{R_1 | R_2} \qquad \mathbf{R^*} \qquad \mathbf{R^k} = \underbrace{\mathbf{R \cdot R \cdots R}}_{\mathbf{k}}$

$$EQREX\uparrow = \Big\{(R_1, R_2) : \text{równoważne uogólnione wyrażenia regularne}\Big\}$$

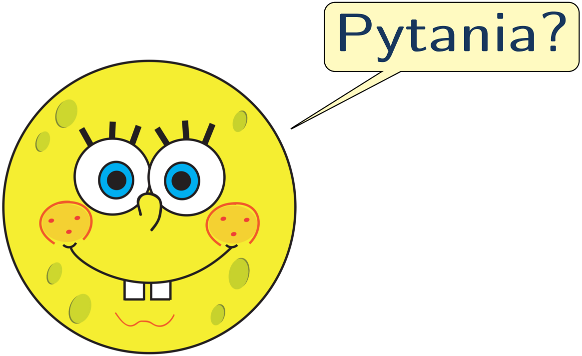
$L(R_1) = L(R_2)$

$$\left(a^k \mid (b \cdot c)^m\right)^n \implies \overbrace{\big(\underbrace{a \cdot a \cdots a}_{\mathbf{k}} \mid \underbrace{bc \cdot bc \cdots bc}_{\mathbf{m}}\big) \cdots \big(\underbrace{a \cdot a \cdots a}_{\mathbf{k}} \mid \underbrace{bc \cdot bc \cdots bc}_{\mathbf{m}}\big)}^{n}$$

Złożoność czasowa vs złożoność pamięciowa

$$DTIME\big(f(n)\big) \subseteq SPACE\big(f(n)\big)$$

$$NTIME\big(f(n)\big) \subseteq NSPACE\big(f(n)\big)$$


Pytania?