Складні класи, пов'язані з пам'яттю. Теорема Севіча. PSPACE-повнота. Обчислення на логарифмічній пам'яті. NL-повнота.

Питання:

1. Складні класи, пов'язані з пам'яттю.
2. Теорема Севіча.
3. PSPACE-повнота.
4. Обчислення на логарифмічній пам'яті.
5. NL-повнота

Складні класи, пов'язані з пам'яттю

- Функція s: N → N називається ***функцією, що конструктується за пам’яттю,*** якщо існує машина Тьюринга, яка по входу 1^n обчислює $s(n)$, використовуючи пам'ять $O(s(n))$.

- Нехай $s(n)$ – неспадна функція. Класом **DSPACE($s(n)$)** називається клас мов, які можна розпізнати на детермінованій машині Тьюринга, на будь-якому вході довжини n, що використовує $O(s(n))$ осередків на робочих стрічках.

- Нехай $s(n)$ - неспадна функція. Класом **NSPACE($s(n)$)** називається клас мов, які можна розпізнати на недетермінованій машині Тьюринга, на будь-якому вході довжини n, що використовує $O(s(n))$ осередків на робочих стрічках (при будь-яких результатах недетермінірованного вибору).

- Класом **PSPACE** називається $\bigcup_{c=0}^{\infty}$ DSPACE (n^c). Класом **NPSPACE** називається $\bigcup_{c=0}^{\infty}$ NSPACE (n^c).

- **Теорема 9.1.** Мають місце співвідношення

$$\mathbf{DTIME}(t(n)) \subset \mathbf{NTIME}(t(n)) \subset \mathbf{DSPACE}(t(n)) \subset \mathbf{NSPACE}(t(n)) \subset \mathbf{DTIME}(2O(t(n)))$$

- **Зауваження до теореми 9.1**. Розглядаючи клас DTIME($t(n)$), ми фактично говоримо про те, що $t(n) \geq n$. Втім, вкладення NSPACE($t(n)$) ⊂ DTIME ($2^{O(t(n))}$) виконано вже для $t(n) \geq \log n$.

$$\mathrm{P} \subset \mathrm{NP} \subset \mathrm{PSPACE} \subset \mathrm{NPSPACE} \subset \mathrm{EXP}.$$

- **Слідство 9.1:**

- **Теорема 9.2.** *Нехай f і g - зростаючі функції, що конструюються по пам'яті*, причому $f(n)=o(g(n))$ і $f(n)\geq\log n$. Тоді DSPACE($f(n)$) $\subsetneq$ (DSPACE($g(n)$)).

- **Теорема 9.3. Теорема Севіча:** Якщо $s(n)\geq\log n$, то NSPACE($s(n)$) $\subset$ DSPACE($s(n)^2$).

- **Слідство 9.2.** PSPACE = NPSPACE.

PSPACE-повнота

- **Мова $\boldsymbol{B}$** називається **PSPACE - складною**, якщо для будь-якої мови A з PSPACE виконано $A \leq_p B$.
- Мова називається **PSPACE-повною**, якщо вона PSPACE - складна і лежить в PSPACE.
- Якщо B є PSPACE-складною і $B \leq_p C$, то C також PSPACE - складна.
- Якщо B є PSPACE-складною і лежить в P (в NP), то P = PSPACE (відповідно, NP = PSPACE).

- **Мовою SPACETMSAT** називається множина $\{(M, x, 1^s) \mid M(x) = 1$ і $M(x)$ займає не більше s осередків пам'яті$\}$.
- **Теорема 9.4**. Мова SPACETMSAT є PSPACE-повною.

- **Мовою SUCCINCTPATH** називається множина $\{(\varphi, u, v) \mid$ в графі, побудованому за формулою φ, є шлях з u в $v\}$. Граф за формулою будується таким чином: u і v є слова з n бітів, а φ формула залежить від $2n$ змінних. Ребро між u і v проводиться в разі, коли $\varphi(u, v) = 1$.
- **Теорема 9.5**. *Мова SUCCINCTPATH є PSPACE-повною.*

Приклади повних задач

- **Мовою GG** називається множина $\{(G, x)$ | в узагальненій грі в міста на графі G з початковою вершиною x виграє перший гравець$\}$.
- *Гра відбувається наступним чином*: спочатку фішка ставиться в вершину x, потім двоє по черзі зрушують її по ребрах, при цьому заборонено зрушувати фішку в вершину, де вона вже була. Програє той, хто не може зробити хід.

- **Теорема 9.6.** *Мова GG є PSPACE-повною.*

Обчислення за логарифмічною пам'яті

- **Класом L** називається DSPACE $(\log n)$. Класом **NL** називається NSPACE $(\log n)$.
- **Класом coNL** називається множина мов A, таких що $\overline{A} \in$ NL.

Твердження1. **Мова LE** = $\{(x, y) \mid x \leq y\}$ належить L.

***Твердження2*.** Мова ADD = $\{(x, y, z) \mid x + y = z\}$ належить L.

***Твердження3*.** Мова TREE = $\{ G \mid$ неорієнтовані граф G є деревом $\}$ лежить в L .

Таким чином, алгоритм такий:

- Перевірити, що число ребер на одиницю менше числа вершин;
- Перевірити, що немає ізольованих вершин;
- Запустити обхід, починаючи з довільного ребра, перевірити, що він вперше повториться через 2 (n - 1) кроків.

NL-повнота

- *Функція f обчислюється на логарифмічній пам'яті* тоді і тільки тоді, коли мови $D_f = \{(x, k)$: $|f(x)| \le k\}$ і $E_f = \{(x, i) \mid f(x)_i = 1\}$ лежать в L, при цьому максимальне k, при якому $(x, k) \in D_f$, обмежена поліномом від $| x |$.

- Композиція функцій, обчислюваних на логарифмічній пам'яті, обчислювана на логарифмічній пам'яті.

- **Мова A логарифмічно зводиться до мови B**, якщо існує функція f, обчислювана на логарифмічній пам'яті, така що для всіх x виконано $x \in A$ тоді і тільки тоді, коли $f(x) \in B$. Позначення $A \le_l B$.

Твердження :

1. Логарифмічна зведеність *рефлексивна*: $A \le_l A$;
2. Логарифмічна зведеність *транзитивна*: якщо $A \le_l B$ і $B \le_l C$, то

$A \le_l C$;

3. Якщо $A \in$ L, а $B \neq \emptyset$ і $B \neq \{0, 1\}^*$, то $A \le_l B$;
4. Якщо $B \in$ L і $A \le_l B$, то $A \in$ L;
5. Якщо $B \in$ NL і $A \le_l B$, то $A \in$ NL.

- **Мова $\boldsymbol{B}$ називається NL-складною**, якщо для будь-якого $A \in$ NL виконана зведеність $A \leq_l B$. **Мова** називається **NL-повною**, якщо вона NL-складна і лежить в NL.

- Як завжди, виконані прості твердження:
- **Твердження 1.** Якщо B є NL-складною і $B \leq_l C$, то C також NL-складна.

- **Твердження 2.** Якщо B є NL-складною і лежить в L, то L = NL.