ZŁO* — ćwiczenia 10

Alternacja, zupełność

Maszyna alternująca to maszyna Turinga o stanach podzielonych na $Q_\forall$ i $Q_\exists$. Przejścia albo prowadzą z jednej konfiguracji do drugiej, albo wywołują instrukcję akceptacji/odrzucenia. Maszyna akceptuje z konfiguracji ze stanem z $Q_\exists$ jeśli *istnieje* przejście, które albo natychmiast akceptuje, albo prowadzi do konfiguracji, z której maszyna akceptuje. Analogicznie dla konfiguracji ze stanami z $Q_\forall$ wymagamy by *każde* przejście miało tę własność. Maszyna działa w czasie $t(n)$ jeśli dla każdego słowa wielkości n można określić, czy maszyna akceptuje czy odrzuca na podstawie konfiguracji osiągalnych w co najwyżej $t(n)$ krokach. Maszyna używa pamięci $s(n)$ jeśli dla każdego słowa wielkości n można określić, czy maszyna akceptuje czy odrzuca na podstawie konfiguracji o pamięci roboczej $s(n)$. Klasy $\mathsf{ATIME}(t(n))$ oraz $\mathsf{ASPACE}(s(n))$ są zdefiniowane naturalnie. Oznaczamy

$$\mathsf{AL} = \bigcup_{c\in\mathbb{N}} \mathsf{ASPACE}(c\log n) \qquad \text{oraz} \qquad \mathsf{AP} = \bigcup_{c\in\mathbb{N}} \mathsf{ATIME}(n^c).$$

Zadanie 1. Wykaż, że jeśli język L jest w klasie $\mathsf{ATIME}(t(n))$, to również dopełnienie L jest w tej klasie. Udowodnij również taki sam fakt dla klasy $\mathsf{ASPACE}(s(n))$.

Zadanie 2. Wykaż, że $\mathsf{AP} = \mathsf{PSPACE}$.

Zadanie 3. Wykaż, że $\mathsf{AL} = \mathsf{P}$.

Zadanie 4. Wskaż algorytm rozwiązujący Directed Reachability w klasie $\mathsf{DSPACE}(\log^2 n)$. W jakim czasie działa ten algorytm?

Niech $t(n), s(n)$ to funkcje obliczalne w L. Klasa $\mathsf{DTiSp}[t(n), s(n)]$ obejmuje problemy rozwiązywalne przez deterministyczną maszynę Turinga działającą *zarówno* w czasie $\mathcal{O}(t(n))$ jak i pamięci $\mathcal{O}(s(n))$. Klasa $\mathsf{NTiSp}[t(n), s(n)]$ jest zdefiniowana analogicznie, tylko dla niedeterministycznych maszyn. Skrót poly i polylog oznaczają odpowiednio, że sumujemy n^k oraz $(\log n)^k$ po wszystkich naturalnych k. Definiujemy *klasę Scotta*:

$$\mathsf{SC} = \mathsf{DTiSp}[\mathrm{poly}, \mathrm{polylog}].$$

Zadanie 5. Udowodnij, że klasa SC jest zamknięta pod L-redukcjami. Udowodnij też, że jeśli $s(n) \geqslant \log n$, to klasa $\mathsf{NTiSp}[\mathrm{poly}, s]$ jest zamknięta pod L-redukcjami.

Zadanie 6. Udowodnij następujące zawierania się klas:

$$\mathsf{NTiSp}[\mathrm{poly}, s] \subseteq \mathsf{DTIME}(2^{\mathcal{O}(s(n))} \cdot \mathrm{poly}(n)) \qquad \text{oraz} \qquad \mathsf{NTiSp}[\mathrm{poly}, s] \subseteq \mathsf{DSPACE}(s(n) \cdot \log n).$$

Rozważmy następujący problem CNF-SAT/pathwidth. Dana jest formuła logiczna φ w postaci CNF, oraz następująca *dekompozycja ścieżkowa* tej formuła. Dekompozycja składa się z ciągu worków $(B_1, B_2, \ldots, B_p)$, gdzie każdy worek jest podzbiorem zmiennych, oraz spełnione są następujące warunki:

(P1) Dla każdej klauzuli C, istnieje worek zawierający wszystkie zmienne C.

(P2) Dla każdej zmiennej x, zbiór worków zawierających x jest przedziałem w dekompozycji.

Szerokość dekompozycji to maksymalna wielkość worka. W problemie mamy daną dekompozycję ścieżkową φ o szerokości co najwyżej s; możemy założyć, że $p = \mathcal{O}(n)$.

Zadanie 7. Wykaż, że problem CNF-SAT/PATHWIDTH da się rozwiązać:

- w czasie $2^{\mathcal{O}(s)} \cdot \mathrm{poly}(n)$;
- w czasie $n^{\mathcal{O}(s)}$ i pamięci $\mathcal{O}(s \cdot \log n)$;
- w czasie $2^{\mathcal{O}(s^2)} \cdot \mathrm{poly}(n)$ i pamięci $\mathrm{poly}(n)$.

Zadanie 8. Udowodnij, że problem CNF-SAT/PATHWIDTH dla $s = s(n) \geqslant \log n$ jest zupełny dla klasy $\mathsf{NTiSp}[\mathrm{poly}, s(n)]$ pod L-redukcjami.

Zadanie 9. Wykaż, że następujące warunki są równoważne:

(i) Problem CNF-SAT/PATHWIDTH posiada algorytm rozwiązujący go w czasie $2^{\mathcal{O}(s)} \cdot \mathrm{poly}(n)$ i pamięci $\mathrm{poly}(\log n, s)$.

(ii) $\mathsf{NL} \subseteq \mathsf{SC}$.