ZLOŻONOŚĆ OBLICZENIOWA - WYK. 10

1. Dwa rodzaje algorytmów randomizowanych:

   - **Monte Carlo**: zawsze działamy szybko, zwykle udzielamy dobrej odpowiedzi
   - **Las Vegas**: zawsze udzielamy dobrej odpowiedzi, zwykle działamy szybko, a pętlimy się z prawdopodobieństwem 0

   Klasa **ZPP** (*zero-error probabilistic polynomial time*): te problemy, które dadzą się rozstrzygnąć w oczekiwanym czasie wielomianowym. *Uwaga:* musimy to ostrożnie zdefiniować, bo po czym jest brana ta wartość oczekiwana? Formalnie, rozważamy (deterministyczne) maszyny z dostępem do nieskończonej taśmy bitów, po której jedziemy tylko w prawo (bity losowe). Każde obliczenie (tzn. obliczenie na każdej zawartości taśmy losowej), które się zatrzymuje po pewnej liczbie k odczytów taśmy losowej, dostaje prawdopodobieństwo 2^{-k}. Oczekiwaną długość obliczenia bierzemy po tym rozkładzie, przy czym prawdopodobieństwo zapętlenia powinno być 0.

   Ćwiczenie: $\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}$ (czyli algorytmy Las Vegas dają się tłumaczyć na algorytmy Monte Carlo)

2. **Twierdzenie** o niejednorodnej derandomizacji: $\mathbf{BPP} \subseteq \mathbf{P/poly}$. (Adleman'78, ten Adleman od szyfru RSA, Rivest-Shamir-Adleman)

   Uwaga: to twierdzenie mówi, że dla każdego języka w **BPP** istnieje ciąg obwodów o rozmiarze wielomianowym, który go rozpoznaje. Z drugiej strony, gdyby taki ciąg był jednorodny, to język byłby w **P** i każdy język w **BPP** dałby się derandomizować. To jest jednak otwarty problem, więc jasne że ciąg obwodów, który wyniknie z naszego dowodu, będzie "dziwny", tj. trudny do obliczenia.

   Dowód: Niech język L będzie rozpoznawany maszyną M w czasie $p(n)$, z prawdopodobieństwem błędu $\frac{1}{4}$. Dla $(2m+1)$-krotnie dłuższych świadków, prawdopodobieństwo błędu spada do $(\frac{3}{4})^m$.

   Weźmy $m = 3n$. Prawdopodobieństwo, że losowo wybrany świadek długości $(2m+1) \cdot p(n)$ myli się dla ustalonego słowa w to $(\frac{3}{4})^{3n}$. Wobec tego prawdopodobieństwo, że losowo wybrany świadek myli się dla *jakiegoś* słowa w długości n to najwyżej $2^n \cdot (\frac{3}{4})^{3n}$. Tymczasem

   $$2^n \cdot (\frac{3}{4})^{3n} = (2 \cdot \frac{27}{64})^n < 1$$

   więc istnieje świadek, który nie myli się dla żadnego słowa długości n. Ciąg takich świadków to porada rozpoznająca L.

3. Nie znamy żadnego niewydumanego problemu, który byłby w **BPP** ale o którym nie byłoby wiadomo czy jest w **RP** lub w **coRP**. Kiedyś było

tam sprawdzanie pierwszości, ale wpadło do **P**. Natomiast **BPP** pozostaje dobrym kandydatem na klasę tych problemów, które umiemy szybko rozwiązywać "w praktyce". *Ćwiczenie:* Jeśli **NP** $\subseteq$ **BPP** to **NP** $\subseteq$ **RP**.

4. Teraz coś o derandomizacji algorytmów Monte Carlo. Ogólnie wiemy tylko, że **BPP** $\subseteq$ **PSPACE**, ale czasami można powiedzieć więcej i są do tego różne techniki. Weźmy przykład MAX-CUT: dla nieskierowanego grafu $G = (V, E)$ (bez pętli), obliczyć podzbiór $S \subseteq V$ taki że

$$cut(S) = \{\{u, v\} \in E \mid u \in S, v \notin S\}$$

jest maksymalnie duży. (Jest też wersja decyzyjna: czy maksymalny cut jest większy niż dane k?)

To jest problem **NP**-zupełny, ale istnieje prościutki algorytm randomizowany, który oblicza S o wartości oczekiwanej $cut(S) \geq \frac{|E|}{2}$:

- dla kolejnych wierzchołków $v \in V$, wrzuć v do S z prawdopodobieństwem $\frac{1}{2}$.

Ponieważ wybory są niezależne, to dla każdej krawędzi prawdopodobieństwo, że będzie ona w *cut* jest $\frac{1}{2}$, a więc (z liniowości wartości oczekiwanej) oczekiwany rozmiar *cut* to $\frac{|E|}{2}$.

A jakie jest prawdopodobieństwo, że obliczony cut ma rozmiar co najmniej $\frac{|E|}{2}$? Ograniczmy to prawdopodobieństwo z dołu. Oznaczmy $p = |V|$. Najgorszy możliwy przypadek to ten, kiedy algorytm rzadko (w k przypadkach, dla pewnego k) zwraca cut o rozmiarze $|E|$, a często (w $2^p - k$ przypadkach) zwraca cut o rozmiarze $\frac{|E|}{2} - 1$. Liczymy:

$$\begin{aligned}
kE + (2^p - k)(\frac{1}{2}E - 1) &\geq 2^p \frac{E}{2} \\
kE + 2^{p-1}E - 2^p - \frac{1}{2}kE + k &\geq 2^{p-1}E \\
\frac{1}{2}kE &\geq 2^p - k \\
k(E-2) &\geq 2^{p+1} \\
\frac{k}{2^p} &\geq \frac{2}{E-2}
\end{aligned}$$

Z tego wynika, że prawdopodobieństwo tego, że obliczony cut ma rozmiar co najmniej $\frac{|E|}{2}$, wynosi co najmniej $\frac{1}{|E|}$. Powtarzając algorytm liniową liczbę razy można prawdopodobieństwo błędu zbić do stałej. Istotnie, zauważmy że

$$\lim_{n\to\infty} \left(1 - \frac{1}{n}\right)^n = \frac{1}{e}$$

a prawdopodobieństwo nieuzyskania cutu o rozmiarze co najmniej $\frac{|E|}{2}$ w k niezależnych losowaniach jest najwyżej $(1 - \frac{1}{|E|})^k$.

Jak to teraz zderandomizować, tzn. podać deterministyczny algorytm na obliczanie S takiego że $cut(S)$ ma rozmiar co najmniej $\frac{|E|}{2}$? Pokażemy dwa pomysły.

5. **Metoda warunkowych wartości oczekiwanych**: Aby zderandomizować algorytm, musimy umieć znaleźć "dobrego" świadka, w tym przypadku takiego, który da $cut(S) \geq \frac{|E|}{2}$.

   Dla ustalonego ciągu bitów (zgadnięć), $b_1 \dots b_k$, niech $E(b_1, \dots, b_k)$ wartością oczekiwaną rozmiaru cięcia dla maszyny, która na początku wylosuje takie właśnie bity. Jest jasne że

   $$E(b_1, \dots, b_k) = \frac{1}{2}E(b_1, \dots, b_k, b_{k+1}) + \frac{1}{2}E(b_1, \dots, b_k, \neg b_{k+1})$$

   więc któraś z wartości po prawej musi być nie mniejsza niż ta po lewej. Załóżmy, że umiemy deterministycznie policzyć $E(b_1, \dots, b_k)$. Wtedy możemy działać "zachłannie": zawsze wybierać taki b_{k+1}, który daje większą oczekiwaną wielkość cięcia. Zauważmy, że wtedy zawsze mamy

   $$E(b_1, \dots, b_k, b_{k+1}) \geq E(b_1, \dots, b_k) \geq \cdots \geq E(b_1) \geq E(\epsilon) = \frac{|E|}{2}$$

   czyli na końcu musimy dostać cięcie o odpowiedniej wielkości, bo ostatnia wartość $E(\cdots)$ to już konkretna wielkość cięcia.

   Nie zawsze umiemy szybko policzyć $E(b_1, \dots, b_k)$, ale np. w przypadku MAX-CUT umiemy. Pytanie: jaka jest oczekiwana wielkość *cut*, jeśli dotąd wybraliśmy wierzchołki z S i odrzuciliśmy wierzchołki z T? Ano taka:

   $$E(b_1, \dots, b_k) = |cut(S, T)| + \frac{|X|}{2}$$

   gdzie X to zbiór tych krawędzi, które mają co najmniej 1 wierzchołek poza $S \cup T$. Bity trzeba wybierać tak, żeby maksymalizować tę wartość.

6. **Metoda zmiennych parami niezależnych**: Dotychczas założyliśmy, że bity na taśmie losowej są niezależne. Ale w algorytmie na MAX-CUT wystarczy tylko, żeby $Pr[b_i = b_j] = \frac{1}{2}$ dla każdego $i \neq j$. A więc wystarczy, żeby bity były *parami* niezależne.

   Formalnie, w algorytmie randomizowanym można liczyć prawdopodobieństwa sukcesu tylko po tych ciągach bitów losowych, które są parami niezależne, i prawdopodobieństwo błędu nadal będzie małe.

   Fakt: mając do dyspozycji $\log(n)$ bitów niezależnych, można wyprodukować n bitów parami niezależnych. Konkretnie bierzemy wszystkie możliwe XOR-y podzbiorów tych bitów niezależnych.

   A wszystkie kombinacje $\log(n)$ bitów niezależnych można po kolei przejrzeć w czasie wielomianowym.

7. Ogólna idea dla derandomizacji: uogólnienie metody zmiennych parami niezależnych.

   Ciąg jest *pseudolosowy* jeśli żaden algorytm wielomianowy nie umie go odróżnić od losowego. Idea: jeśli można deterministycznie, w czasie wielomianowym, wygenerować ciąg pseudolosowy długości n z doskonale losowych $\log n$ bitów, to możemy deterministycznie zasymulować maszynę probabilistyczną!

   Dokładniej: *generator* (obliczalna rodzina funkcji obliczalnych w **P** względem n reprezentowanego unarnie) $G : \{0,1\}^{\log n} \to \{0,1\}^n$ jest ϵ-*pseudolosowy*, jeśli dla każdej rodziny funkcji $D : \{0,1\}^n \to \{0,1\}$ w klasie $\mathbf{P/poly}$ mamy własność, dla każdego dużego n:

   $$Pr_{x\in\{0,1\}^{\log n}}[D(G(x)) = 1] - Pr_{y\in\{0,1\}^n}[D(y) = 1] \leq \epsilon.$$

   Twierdzenie (Yao'82): Jeśli istnieje generator $\frac{1}{10}$-pseudolosowy, to $\mathbf{BPP} = \mathbf{P}$.

   Dowód: Weźmy maszynę M, która **BPP**-rozpoznaje pewien język L w czasie $p(n)$. Dla danego słowa w długości n,

   - wygeneruj po kolei wszystkie ciągi bitów x długości $\log(p(n))$,
   - dla każdego z nich oblicz $G(x)$,
   - zasymuluj M na słowie w z bitami $G(x)$.
   - zaakceptuj jeśli ponad połowa świadków zaakceptowała.

   Dowód poprawności: średnia uzyskanych wyników (zer i jedynek) to

   $$Pr_{x\in\{0,1\}^{\log n}}[M(w, G(x)) = 1]$$

   Dla każdego słowa w funkcja $D(y) = M(w, y)$ jest w klasie **P**, więc wiemy że obliczona wartość różni się o co najwyżej $\frac{1}{10}$ od

   $$Pr_{y\in\{0,1\}^n}[M(w, y) = 1]$$

   Czyli jeśli $w \in L$, to ta ostatnia wartość jest co najmniej $\frac{3}{4}$, czyli ta poprzednia jest większa niż $\frac{1}{2}$. Analogicznie dla $w \notin L$.

8. Na początku myślano, że takie generatory raczej nie istnieją, więc twierdzenie Yao nie wywołało wstrząsu. Ale z biegiem czasu okazywało się, że istnieją przy coraz słabszych założeniach. Najmocniejszy tego rodzaju wynik to:

   Twierdzenie (Impagliazzo-Widgerson'98): Jeśli dla SAT nie istnieje (niejednorodna) rodzina obwodów rozmiaru mniejszego niż $2^{\Theta(n)}$, to $\mathbf{BPP} = \mathbf{P}$.

   Dowód trudny.

   Zauważmy jaki to ciekawy wynik: z trudności pewnego problemu (tu: nieistnienia małych obwodów dla SAT) można wywnioskować wynik „łatwościowy", czyli równość dwóch klas złożoności.