Problemy NP-zupełne, co to takiego i jak je przezwyciężać

Andrzej BARCZAK, Alexey TRIETYAKOV, Leszek ZAKRZEWSKI, Siedlce

Problemy NP-zupełne (NPC) to takie problemy klasy NP, że każdy inny problem klasy NP może zostać do nich zredukowany w czasie wielomianowym – rozwiązanie jednego takiego problemu w czasie wielomianowym oznaczałoby, że P = NP. Przyjmuje się między innymi, że problem jest NP zupełny, gdy dowolny inny problem NP-zupełny można sprowadzić do danego problemu za pomocą obliczeń, które można wykonać w czasie wielomianowym. Taka definicja problemów NP zupełnych implikuje fakt, że jeśli tylko potrafimy rozwiązać jakikolwiek problem NP zupełny w czasie wielomianowym, to potrafimy rozwiązać w czasie wielomianowym wszystkie problemy NP zupełne.

Pytanie, czy problemy NP zupełne można rozwiązywać w czasie wielomianowym, jest największą zagadką informatyki teoretycznej – matematyki obliczeniowej. Ciągle nie udowodniono nierówności P ≠ N (nie udowodniono także przeciwnie – że P = NP), która jednoznacznie stwierdzałaby, że jest to niemożliwe. Rozwiązanie tego problemu znalazło się na liście siedmiu słynnych nierozwiązanych problemów matematycznych trzeciego tysiąclecia, razem ze słynną hipotezą Goldbacha. Udowodnienie nierówności P ≠ N (lub też przeciwnie – że P = N) nazywane jest niekiedy problemem Cooka.

Twierdzenie Cooka mówi że: problem spełnialności formuł logicznych jest problemem NP-zupełnym. (**Twierdzenie Cooka** – jedno z najważniejszych twierdzeń teorii złożoności obliczeniowej. Podaje ono pierwszy znany problem NP-zupełny. Od momentu jego udowodnienia można było stosować transformacje wielomianowe do dowodzenia NP-zupełności innych problemów decyzyjnych.) Możemy podać listę najważniejszych (z różnych dziedzin) problemów NP-zupełnych:

- Logika
  - Problem trójspełnialności (3SAT)
- Problemy grafowe
  - Problem znajdowania cyklu Hamiltona
  - Problem znajdowania kliki w grafie
  - Problem komiwojażera (COMI) (Salesman Problem)
  - Problem znajdowania pokrycia wierzchołkowego
  - Problem trójkolorowalności (3COL)
  - Kolorowanie grafu
  - Cykliczne pokrycie krawędziowe
- Problemy podziału zbioru
  - Problem plecakowy
  - Problem trójpodziału

Jeżeli wymiar modelu matematycznego zadania wynosi n (np. rozpatrujemy problem wyboru maksymalnej z n liczb, rozwiązujemy układ równań liniowych o n zmiennych, rozpatrujemy graf o n wierzchołkach itd.), to liczba podstawowych operacji potrzebnych do rozwiązania go (złożoność) określana jest wielomianem zależnym od n: $P(n)$ oznaczamy przez P, a algorytmy o złożoności wykładniczej, $\sim e^n$ (albo $n!$) oznaczamy przez NP (Non Polynomial).

Do takich problemów zalicza się też zadanie Cooka o spełnialności – zadana funkcja boolowska (Boolean) $f(x)$, zmiennej logicznej (boolowskiej) $x = (x_1, \ldots, x_n)$ przyjmuje wartość 1 dla danego – jedynego zbioru $x^* = (x_1^*, \ldots, x_n)^*$: $f(x^*) = 1$ dla innych przyjmuje wartość 0, ($f(x) = 0$, $x \neq x^*$), określić ten zbiór x^*. Olbrzymia wada takich zadań tkwi w tym, iż liczba operacji w algorytmach je rozwiązujących katastrofalnie rośnie wraz ze wzrostem wymiaru n. Na przykład przy $n = 60$ liczba ta byłaby równa ilości ziaren w workach ustawionych rzędem od Ziemi do Słońca.

Tabela 1. Czasy działania algorytmu wykładniczego na dwóch komputerach

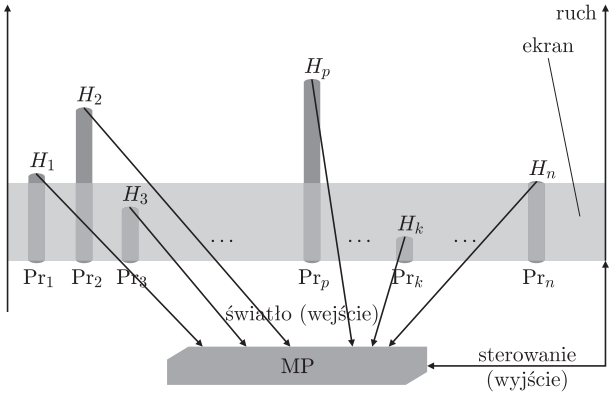
Wielkość n	20	50	100	200
Czas działania $(2^n/10^6)$	1,04 s	35,7 lat	$4 \cdot 10^{14}$ wieków	$5 \cdot 10^{14}$ wieków
Czas działania $(2^n/10^9)$	0,001 s	13 dni	$4 \cdot 10^{11}$ wieków	$5 \cdot 10^{41}$ wieków

Już przy $n > 50$ żadna obecnie znana technika obliczeniowa (komputerowa) nie może rozwiązać problemów NP-zupełnych w przeciągu (ramach) realnego czasu (patrz tabela 1). Widzimy również, iż 1000-krotne przyspieszenie szybkości działania komputera niewiele pomaga w przypadku algorytmów o złożoności wykładniczej. Do takich problemów doskonale pasuje (zalicza się) problem komiwojażera (Salesman Problem) (odwiedzenie wszystkich n punktów, każdego jednokrotnie i przebycie najkrótszej drogi). Ponadto w otaczającym nas świecie, w zasadzie przychodzi się nam stykać właśnie z takimi problemami i brak jest algorytmów szybko je rozwiązujących, co istotnie hamuje proces poznania.

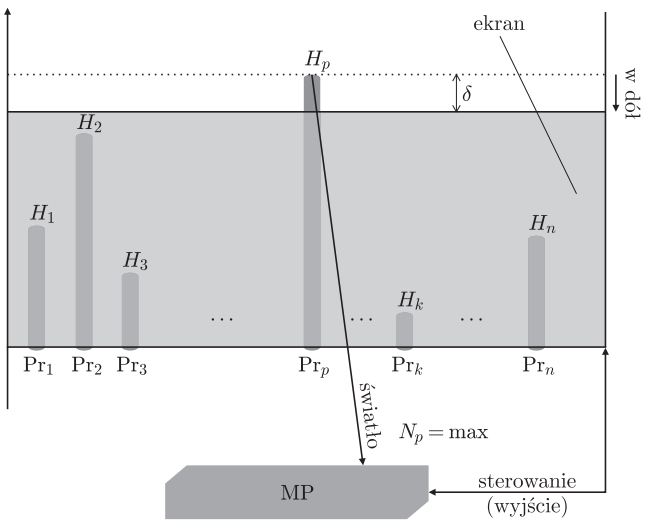
Jednakże, tak jak potwierdzają badania naukowe, istnieją systemy, które mogą nieporównanie szybciej rozwiązać szereg zadań, niż współczesne komputery, w tym niektóre problemy z klasy problemów NP-zupełnych. Taki system to nasz Mózg! W tym czasie kiedy „najmocniejsze" komputery potrzebują długotrwałego wykorzystania zasobów obliczeniowych dla rozwiązania problemu, nasz mózg czasem daje natychmiastową odpowiedź. Szkoda, iż praca naszego mózgu nie jest jeszcze prawie wcale zgłębiona i zrozumienie mechanizmów takich „zwariowanych" prędkości to zadanie przyszłości. Ale niektóre dostrzegalne właściwości pojmowania i reakcji mózgu można wykorzystać przy konstruowaniu nowej klasy algorytmów, w tym do rozwiązywania problemów NP-zupełnych. Objaśnimy to na przykładzie.

Wyobraźmy sobie, iż naprzeciwko nas stoi ogrodzenie – płot, w którym wszystkie deski (n – sztuk) są „ciemne", a jedna deska została zamieniona na nową – „białą". Jak człowiek i komputer rozwiązuję to zadanie – brzmiące: określić (znaleźć) nową – „białą" deskę. Komputer w ten czy inny sposób będzie porównywał wszystkie deski według kryterium koloru dla określenia „białej" deski. W przypadku najlepszym aby wykonać zadanie komputer potrzebuje wykonać $\lg n$ operacji.

Człowiek (mózg) gdy rozwiązuje problem – błyskawicznie (za jedną iteracją) – od razu pokaże – oto ona – „biała" deska !!! To genialność Stwórcy, ustanawiającego tak uniwersalny system jakim jest człowiek, tak mało wiedzący o samym sobie. Ale spróbujmy realizować prosty model pracy mózgu (w sensie algorytmu) na przytoczonym wyżej przykładzie w postaci matematycznej. Niech istnieje zbiór n wartości $N_1, N_2, \ldots$. Potrzebujemy znaleźć maksymalną liczbę $N_P = \max N_k$ z tych liczb $k = \overline{1,n}$. Zakładamy, że wszystkie liczby są różne i $|N_i - N_j| \geq \delta > 0$ dla $i \neq j$. Przy tym wspominamy, że technika obliczeniowa wyrosła z urządzeń analogowych, a w modelu będziemy wykorzystywać operacje analogowe. Dla zorganizowania struktury obliczeniowej (algorytmicznej), zakładamy, że posiadamy n jednakowych procesorów elementarnych P_{i_k}, $k = \overline{1,n}$ realizujących jedna i ta samą operację – przekształcanie wartości N_k w odpowiednią wysokość słupa świetlnego (lub wartość napięcia elektrycznego), wysokość którego to $H_k = N_k$. Posiadamy, także procesor centralny – główny procesor (Main Processor – MP), który reaguje na istnienie światła i steruje ruchem nieprzepuszczalnego ekranu, a zasadniczo – jeżeli jest światło, to realizowany jest ruch ekranu do góry; – jeżeli nie ma światła to ekran zatrzymuje się i opuszcza się do dołu o wielkość δ – dokładność rozwiązania zadania (patrz rys. 1).



Rys. 1. Model wielopoziomowej struktury obliczeniowej



Rys. 2. Działanie wielopoziomowej struktury obliczeniowej

W tym momencie pojawia się sygnał świetlny tylko od jednego źródła – odpowiadającego maksymalnej wartości (patrz rys. 2).

Sygnał ten jednocześnie przenosi informację o numerze P, wielkości N_P i inne. Oczywiste jest, iż tak zorganizowany model obliczeniowy daje rezultat w wyniku dwóch operacji analogowych:
1 – podnoszenie ekranu do momentu zaprzestania detekcji światła,
2 – opuszczenie ekranu o wielkość δ.
Złożoność nie zależy od n !!!

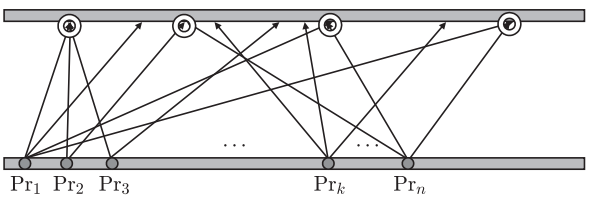
Czas obliczeń – fizyczny odpowiednik ilości operacji – zależy tylko od wielkości maksymalnej liczby N_P.

W wielu realnych zadaniach wielkość N_P może być niewielka. Na przykład dla zadania (problemu) komiwojażera (Salesman Problem) wielkość N_P równa jest długości maksymalnej drogi przechodzącej przez wszystkie wierzchołki (punkty). W związku z tym obliczenie (znalezienie) wielkości maksymalnej z liczb nie zależy od wielkości n, równe jest czasowi podnoszenia ekranu – T_P na wysokość $H_P = N_P$ i proporcjonalnie $N_P : T_P = = \alpha \cdot N_P$. Przy czym współczynnik α może być równy 1/c, gdzie c – prędkość światła.

Analogiczne opierając się na tych zasadach można zbudować model rozwiązujący zadanie komiwojażera (Salesman Problem) – znajdowanie drogi o najmniejszej długości, przechodzącej przez n punktów (każdy odwiedzając jeden raz). Pokażemy jak to wykonać.

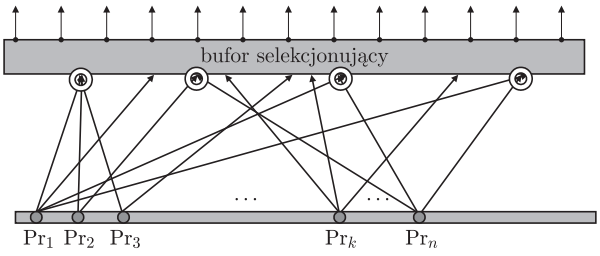
Weźmy n punktów $a_1, \ldots, a_n$ o współrzędnych $N_1, \ldots, N_n$. Potrzebujemy znaleźć minimum funkcji $f(i_1, i_2, \ldots, i_n)$ – długość drogi, przechodzącej przez punkty $a_{i_1}, \ldots, a_{i_n}$. Przy czym, $i_j \neq i_k$ jeśli $j \neq k$ i $i_j, i_k \in \{1, \ldots, n\}$, $j, k \in \{1, \ldots, n\}$. Wykonujemy obliczenia aby określić $\min f(i_1, \ldots, i_n) = = f(i_1^*, \ldots, i_n^*)$. Musimy wybrać więc minimalną wartość z $n!$ wartości $f(i_1, \ldots, i_n)$. Liczba zestawów wartości $(i_1, \ldots, i_n)$ równa jest $n!$. Można to zadanie rozwiązać analogicznie do poprzedniego , ale aby to uczynić potrzebujemy $n!$ elementarnych procesorów. Pokażemy jak można zorganizować strukturę obliczeniową przy wykorzystaniu $n(n-1)/2$ procesorów elementarnych, tak aby liczba obliczeń była funkcją liniową zależną od n.

Niech kolejny elementarny procesor otrzymuje od procesora głównego MP współrzędne wszystkich N_j, oblicza tylko odległość pomiędzy dwoma punktami a_{i_1} i a_{i_2} przypisaną na stałe do tego procesora. Jeżeli wszystkie $n(n-1)/2$ elementarne procesory przekażą (wyliczą) wszystkie możliwe odległości pomiędzy parami punktów a_{i_1} i a_{i_2} z pośród n punktów $a_1, \ldots, a_n$. Niech, ponadto, każdy elementarny procesor Pr_i określa odpowiednią do dystansu (odległości) swojej pary (a_{i_1}, a_{i_2}) długość fali elektromagnetycznej i odpowiadającej jej częstotliwości. Przy czym szybkość rozprzestrzeniania się sygnału V_i jest odwrotnie proporcjonalna do długości $\mathrm{dist}(a_{i_1}, a_{i_2})$: $V_i \sim \mathrm{dist}(a_{i_1}, a_{i_2})^{-1}$. Jednocześnie przekazywana jest informacja o numerach i_1 i i_2. Przy tym zachodzi interferencja fal (sygnałów) od różnych nadajników (patrz rys. 3).

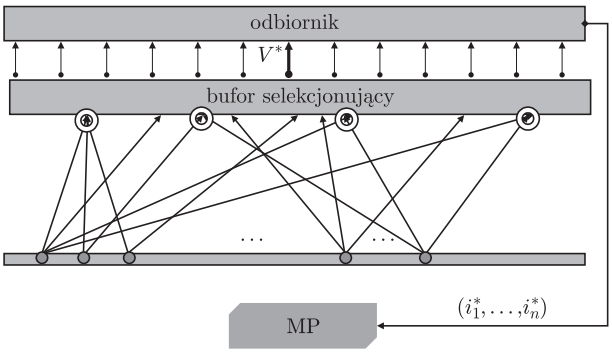


Rys. 3. Model wielopoziomowej struktury obliczeniowej – generowanie sygnałów świetlnych.

W ten sposób realizujemy tworzenie wszystkich fal różnych częstotliwości, w tym także wszystkich możliwych fal odpowiadających odległościom tras (dróg) przechodzących przez różne punkty $a_{i_1}, a_{i_2}, \ldots, a_{i_n}$ w postaci sumy częstotliwości $V_{i_1} + V_{i_2} + \ldots + V_{i_n}$, przy czym najmniejszej długości odpowiada maksymalna prędkość rozchodzenia się fali. Na drodze przebiegu (przechodzenia) fali stoi bufor selekcjonujący (Selection Bar) (patrz rys. 4).



Rys. 4. Model wielopoziomowej struktury obliczeniowej z buforem selekcjonującym.



Rys. 5. Model wielopoziomowej struktury obliczeniowej z buforem selekcjonującym i odbiornikiem.

Bufor ten wygasza sygnały posiadające mniej niż $2n$ par punktów, a także posiadające 3 lub więcej razy powtarzające się punkty $a_{i_k}, a_{i_k}, a_{i_k}, \ldots$.

W ten sposób przepuszczane są tylko sygnały składające się z sumy długości różnych n par, każdy punkt wchodzi równo 2 razy, to znaczy realizowane są wszystkie możliwe długości tras, przechodzących przez każdy punkt dokładnie jeden raz.

Na trzecim etapie, za buforem selekcjonującym, stoi odbiornik (Receiver) (patrz rys. 5), który reaguje na pierwszy przychodzący sygnał odpowiadający fali o maksymalnej szybkości $V^* = V^*_{i_1} + V^*_{i_2} + \ldots + V^*_{i_n}$, a to oznacza falę o minimalnej długości $f(i^*_1, \ldots, i^*_n) = \min f(i_1, \ldots, i_n)$. Po czym, sygnał ten wraz z informacją o numerach $i^*_1, \ldots, i^*_n$, po których przechodząc realizowane jest minimum drogi, przekazywane jest dla procesora głównego MP.

Przy tym czas obliczenia drogi minimalnej (trasy) równy jest czasowi przechodzenia sygnału (fali) stanowiącego sumę n sygnałów i proporcjonalnego do $\sim n * \max(\mathrm{dist}(a_i, a_j))$, $i, j \in \{1, \ldots, n\}$ tj. liniowy względny rozmiar zadania wynosi n. W przedstawionym modelu wykorzystaliśmy $n(n-1)/2$ elementarnych procesorów i otrzymaliśmy liniowy stopień złożoności struktury algorytmicznej w ocenionym pierwszym modelu ze skończoną złożonością struktury obliczeniowej.

Przytoczmy jeszcze jeden przykład, tym razem realizujący wielopoziomową strukturę obliczeniową symulowaną w oparciu o porównywanie napięć elektrycznych w celu realizowania algorytmu sortowania. Sortowanie nie jest problemem klasy NP, ale w przypadku dużej ilości danych czas sortowania ma zasadnicze znaczenie. Sortowanie jest jedną z najczęściej wykonywanych czynności podczas przetwarzania danych.

Przyjmijmy model wykonany zgodnie z przedstawianą powyżej architekturą. Model interpretowany jest na klasycznym przykładzie wyboru maksymalnej liczby z n podanych liczb $\varphi_0 = \varphi(x_0)$, $\varphi_1 = \varphi(x_1)$, ..., $\varphi_n = \varphi(x_n)$. Rozpatrzmy złożoność algorytmiczną obliczeń. Oznaczmy przez c złożoność algorytmiczną. Korzystając z opisanej architektury globalnej mamy złożoność niezależną od ilości danych n i wynoszącą $c_1 = 1$ (algorytm niezależnie od ilości danych wykonuje pewną stała liczbę operacji). W przypadku wykorzystania obliczeń równoległych, złożoność algorytmiczna wynosi $c_2 = \log n$. Sortowanie zbioru danych zawierającego n wartości φ_i, $i \in \{0, 1, 2, \ldots, n-1, n\}$ jest wykonywane według następującego opisanego poniżej algorytmu (wykorzystując wielopoziomową architekturę obliczeniową). Zakładamy, iż posiadamy n procesorów elementarnych Pr_i, $i \in \{1, \ldots, n\}$, traktujemy każdy z procesorów jako komparator posiadający dwa wejścia analogowe A i B oraz wyjście cyfrowe $F(A, B)$ (patrz rys. 6). Wartość na wyjściu cyfrowym określona jest wzorem

Rys. 6. Komparator analogowo – cyfrowy.

$$F(A,B) = \begin{cases} 0 & \text{jeżeli } A \neq B \\ 1 & \text{jeżeli } A = B \end{cases}$$

Wszystkie procesory wykonawcze Pr_i, $i \in \{0, 1, 2, \ldots, n-1, n\}$ kontrolowane są przez centralny procesor – procesor główny MP. Pomiędzy procesorami wykonawczymi nie ma wymiany danych. Dane są wymieniane tylko pomiędzy procesorem centralnym MP a procesorami wykonawczymi.

Założenia:
• Układ sortuje wartości liczbowe $\varphi_i \in \mathbb{R}^+$
• Maksymalna wartość sortowana wynosi $\varphi_{\max}$ i jest zależna od realizacji sprzętowej i jej ograniczeń.

Sortowanie realizowane jest według następującego algorytmu:

1. Procesory wykonawcze Pr_i otrzymują od procesora dane (każdy z procesorów otrzymuje i-ty wyraz sortowanego ciągu danych) – operacja jednokrotna – wszystkie procesory otrzymują dane jednocześnie. Wartość tej danej podawana jest na wejście A procesora wykonawczego.
2. Na wejścia B procesorów wykonawczych procesor podaje sygnał ciągły o wartości φ_B rosnącej od 0 (zera) do $\varphi_{|max}$. Sygnał podawany jest na wszystkie procesory jednocześnie.
3. Procesory w tym samym czasie, w którym podawany jest sygnał, zwracają procesorowi centralnemu wartość $F(A, B)$. Jeżeli $F(A, B) = 1$ procesor centralny dodaje wartość 1 do indeksu w (przed rozpoczęciem wykonywania algorytmu $w = 0$) oraz wpisuje wartość podawaną na wejście B jako kolejny element nowego ciągu. Jeżeli $w = n - i$, i – numer iteracji, $i \in \langle 0, n - 1\rangle$, procesor centralny zaprzestaje zwiększania wartości na wejściach A procesorów wykonawczych.
4. Otrzymany nowy ciąg jest posortowany niemalejąco.

Korzystając z tego algorytmu opartego o wielopoziomową architekturę obliczeniową możemy więc rozwiązywać problem sortowania dużych zbiorów danych.

Wracając do problemu Cooka, można potwierdzić, że ilość obliczeń i liczba elementarnych procesorów potrzebna do otrzymania optymalnych rezultatów zależy od konkretnej postaci funkcji $f(x_1, \ldots, x_2)$. Analogicznie do problemu komiwojażera (Salesman Problem) dla każdej konkretnej funkcji $f(x_1, \ldots, x_2)$ może być realizowany własny model obliczeniowy, dający wielomianowy stopień złożoności.

Literatura

[1] Cook S., *The* P *versus* NP *Problem*, April, 2000. Manuscript prepared for the Clay Mathematics Institute for the Millennium Prize Problems (revised November, 2000).

[2] Trietyakov A.A. (2004), *Rachunek globalny jako synteza nowej wiedzy*, w: Rachunek Globalny a Przyszłość Informatyki, Olchowik J.M. (red.), PWSZ Biała Podlaska, 67–72.

[3] Zakrzewski, L. (2005), *Wielopoziomowa globalnie zorientowana architektura obliczeniowa*, w: Hołubiec J., Analiza Systemowa w Finansach i Zarządzaniu, Tom 7, IBS PAN, 261–270.