11 NP-trudność problemów niesprzeczności

11.1 Dane dla maszyny Turinga

Będziemy teraz badać algorytmy dotyczące równań liniowych posługując się jako modelem obliczeń maszynami Turinga. Najpierw ustalimy sposób przekazywania maszynie Turinga informacji o liczbach całkowitych lub wymiernych, wektorach i macierzach. Sposób przekazywania takich informacji będzie zawierać nietypowy element. Będziemy rozróżniać liczbę zero i cyfrę zero (zero będzie oznaczane dwoma znakami w zależności od tego, czy oznacza liczbę zero, czy też jest jedną z cyfr w przedstawieniu liczby różnej od zera).

Wiemy już, w jaki sposób będziemy zapisywać liczbę zero. Liczby całkowite różne od zera zapisujemy jako zwykłe przedstawienie dwójkowe modułu liczby (w tym przedstawieniu występują tylko cyfry oznaczające zero) poprzedzone znakiem liczby. Jeżeli n jest liczbą całkowitą, to $\langle n \rangle$ będzie oznaczać długość opisanego przedstawienia liczby n. Nietrudno zauważyć, że $\langle n \rangle = 1 + \lceil \log_2(| n | + 1) \rceil$. Wartość $\langle n \rangle$ nazywamy rozmiarem liczby n.

Aby przekazać maszynie Turinga informację o liczbie wymiernej x, najpierw przedstawiamy ją w postaci ułamka $\frac{p}{q}$ takiego, że q jest liczbą dodatnią (byłoby najlepiej, gdyby ułamek był nieskracalny, a zero przedstawiamy zawsze w postaci $\frac{0}{1}$), a następnie podajemy konkatenację przedstawień liczb p i q. Rozmiar $\langle x \rangle$ przedstawienia liczby wymiernej $x = \frac{p}{q}$ (czyli długość przedstawienia) jest równa $\langle p \rangle + \langle q \rangle$.

Będziemy rozważać dwa rodzaje wektorów (ciągów) i macierzy: o wyrazach całkowitych i wymiernych. Zakładamy, że takie dynamiczne struktury danych (o zmiennej wielkości) są jednorodne, tzn. ich przedstawienia składają się albo tylko z przedstawień liczb całkowitych, albo tylko wymiernych. Dla obu struktur danych przedstawienia definiujemy jako konkatenacje przedstawień elementów tych struktur. Konkatenacje te tworzymy biorąc przedstawienia elementów w pewnym, ustalonym porządku. Łatwo podać wzory na rozmiar przedstawień rozważanych struktur.

Teraz możemy wyjaśnić dlaczego posługujemy się dwoma znakami oznaczającymi zera. Dzięki temu uzyskaliśmy spełnienie podanego wzoru na rozmiar przedstawienia liczby całkowitej oraz możliwość jednoznacznego rozkładu przedstawienia ciągu liczb na przedstawienia poszczególnych wyrazów tego ciągu.

Opisane zasady pozwalają na przedstawianie wielu skomplikowanych struktur danych, a więc np. 5 liczb, albo struktury złożonej z 3 liczb i ciągu liczb o nieokreślonej długości, albo też jednej liczby i macierzy o 3 kolumnach i o wierszach nieznanej długości. W ten sposób jednak nie możemy przedstawiać struktury danych złożonej z dwóch ciągów, albo macierzy o nieznanych wymiarach. Aby umożliwić przedstawianie takich struktur, będziemy posługiwać się dwoma rodzajami znaków $+$, znaków - oraz znaków oznaczających liczbę 0. Te dwa rodzaje znaków będziemy określać jako „duże" i „małe" przez analogię do dużych i małych liter. Duże znaki, oprócz zwykłej, będą przekazywać także dodatkową informację, że znajdują się na początku pewnej struktury. Znaki małe przekazują także informację, że nie znajdują się początku odpowiedniej struktury. Mając małe i duże znaki możemy przedstawiać np. dwa ciągi podając konkatenację przedstawień obu ciągów (złożonych z małych liter) zmodyfikowanych przez zastąpienie pierwszego znaku każdego ciągu odpowiednim dużym znakiem. W podobny sposób możemy przedstawiać macierze, ciągi ciągów i inne, podobne struktury danych.

11.2 Własności rozmiaru danych

Pojęcie rozmiaru danych zostało wprowadzone w rozdziale 11.1. Przypomnijmy, że dla dowolnej liczby całkowitej n, rozmiar tej liczby jest dany wzorem $\langle n \rangle = 1 + \lceil \log_2(|n| + 1) \rceil$.

Lemat 11.1 *Dla dowolnej liczby całkowitej lub wymiernej r mamy*

$$|\, r \,| \leq 2^{\langle r \rangle - 1} - 1.$$

Dowód. Dla liczby całkowitej n z podanej nierówności otrzymujemy, że

$$\langle n \rangle - 1 = \lceil \log_2(|\, n \,| + 1) \rceil \geq \log_2(|\, n \,| + 1)$$

i w konsekwencji

$$2^{\langle n \rangle - 1} \geq 2^{\log_2(|\, n \,| + 1)} = |\, n \,| + 1.$$

Dla liczby wymiernej $r = \frac{p}{q}$, wobec oczywistej nierówności $\langle r \rangle \geq \langle p \rangle$, mamy

$$|\, r \,| \leq |\, p \,| \leq 2^{\langle p \rangle - 1} - 1 \leq 2^{\langle r \rangle - 1} - 1.$$

□

Lemat 11.2 *Jeżeli $\vec{x} = (x_1, \ldots, x_m)$ ma współrzędne wymierne lub całkowite, to*

$$|\, \vec{x} \,| \leq \sum_{j=1}^{m} |\, x_j \,| \leq 2^{\langle \vec{x} \rangle - m} - 1.$$

Dowód. Pierwsza z podanych nierówności wynika z oczywistej nierówności

$$\sum_{j=1}^{m} x_j^2 \leq (\sum_{j=1}^{m} |\, x_j \,|)^2.$$

Aby dowieść drugą, zauważmy, że

$$1 + \sum_{j=1}^{m} |\, x_j \,| \leq \prod_{j=1}^{m} (1 + |\, x_j \,|) \leq \prod_{j=1}^{m} 2^{\langle x_j \rangle - 1} = 2^{\sum_{j=1}^{m} (\langle x_j \rangle - 1)} = 2^{\langle x \rangle - m}.$$

□

Lemat 11.3 *Jeżeli C jest macierzą o wymiarach $m \times m$ i o wyrazach całkowitych lub wymiernych, to*

$$|\, |\, C \,| \,| \leq 2^{\langle C \rangle - m^2} - 1.$$

Dowód. Przypuśćmy, że wierszami macierzy C są wektory $\vec{c}_1, \ldots, \vec{c}_m$. Dowodzony lemat wynika z nierówności Hadamarda. Zauważmy, że

$$1 + |\, |\, C \,| \,| \leq 1 + \prod_{j=1}^{m} |\, \vec{c}_j \,| \leq \prod_{j=1}^{m} (1 + |\, \vec{c}_j \,|) \leq \prod_{j=1}^{m} 2^{\langle \vec{c}_j \rangle - m} = 2^{\sum_{j=1}^{m} (\langle \vec{c}_j \rangle - m)} = 2^{\langle C \rangle - m^2}.$$

□

Lemat 11.4 *Dla dowolnych liczb wymiernych r i s mamy*

$$\langle rs \rangle \leq \langle r \rangle + \langle s \rangle.$$

Lemat 11.5 *Jeżeli C jest macierzą o wyrazach całkowitych i o wymiarach $m \times m$, to*

$$\langle |\, C \,| \rangle \leq \langle C \rangle - m^2 + 1.$$

Dowód. Jest to oczywista konsekwencja lematu 11.3. □

Lemat 11.6 *Jeżeli C jest macierzą o wyrazach wymiernych i o wymiarach $m \times m$, to*

$$\langle | C | \rangle \leq 2\langle C \rangle - m^2.$$

Dowód. Przypuśćmy, że elementy macierzy C mają postać $\frac{p_{i,j}}{q_{i,j}}$ i niech q będzie iloczynem mianowników elementów macierzy C. Tak, jak w dowodzie lematu 11.3 pokazujemy, że

$$| \, | C | \, | \leq 2^{\sum_{i,j=1}^{n} \langle p_{i,j} \rangle - m^2} - 1.$$

Zauważmy, że $q| C |$ jest liczbą całkowitą oraz $| C | = \frac{q| C |}{q}$. Wobec tego

$$\begin{aligned}
\langle | C | \rangle &\leq \langle q| C | \rangle + \langle q \rangle = 1 + \lceil \log_2(| \, q| C | \, | + 1) \rceil + \langle q \rangle \\
&\leq 1 + \lceil \log_2(1 + q(2^{\sum_{i,j=1}^{n} \langle p_{i,j} \rangle - m^2} - 1)) \rceil + \langle q \rangle \\
&\leq 1 + \lceil \textstyle\sum_{i,j=1}^{n} \langle p_{i,j} \rangle - m^2 + \log_2 q \rceil + \langle q \rangle \\
&\leq 2\langle q \rangle + \textstyle\sum_{i,j=1}^{n} \langle p_{i,j} \rangle - m^2 \leq 2\langle C \rangle - m^2.
\end{aligned}$$

□

11.3 Obliczenia wielomianowe

Przyjmujemy, że pojęcie maszyny RAM jest znane, ale musimy je sprecyzować. Zwykle zakłada się, że maszyna RAM składa się z nieskończonego ciągu rejestrów i programu. Rejestry mogą pamiętać dowolne liczby całkowite i są ponumerowane liczbami naturalnymi. Numery rejestrów nazywamy adresami. Program dla maszyny RAM jest ciągiem rozkazów z etykietami, jest wykonywany sekwencyjnie, jedynymi rozkazami sterującymi są warunkowe rozkazy skoków. Wykonanie tych rozkazów polega na porównaniu zawartości wskazanego rejestru z liczbą 0 i wykonaniu odpowiedniego skoku, jeżeli wynik porównywania jest pozytywny. Maszyna RAM wykonuje także rozkazy umożliwiające przekazywanie danych z rejestru do rejestru, wczytywanie danych do rejestrów i wyprowadzanie danych. Maszyny RAM wykonują jeszcze rozkazy arytmetyczne. Czasem są to tylko rozkazy dodawania, odejmowania i zerowania. Dopuszcza się dwa sposoby wskazywania rejestrów (czyli adresowania): pośrednie i bezpośrednie. Maszynę RAM spełniającą podane warunki będziemy określać jako wykonującą dodawanie. Będziemy także rozważać maszyny RAM z bogatszą listą rozkazów arytmetycznych, wykonujące także mnożenie. Będziemy je nazywać wykonującymi dodawanie i mnożenie. Maszyny te mogą też wykonywać rozkazy dzielenia całkowitego.

Będziemy wyobrażać sobie, że maszyny RAM wykonują także obliczenia na liczbach wymiernych. Wtedy będziemy zakładać, że liczby wymierne są pamiętane w postaci ułamka o danym liczniku i mianowniku. Te dwie liczby są pamiętane w parze rejestrów. Działania na liczbach wymiernych są wykonywane zgodnie z oczywistymi i znanymi wzorami, i dodatkowo może być wykonywane skracanie. W zależności od potrzeb będziemy rozważać maszyny RAM, które wykonują skracanie wtedy, gdy mianownik dzieli licznik, lub na życzenie programisty.

Rozmiar danych definiujemy w podany już sposób. Czas wykonania programu dla ustalonych danych definiujemy jako liczbę rozkazów wykonanych po uruchomieniu programu z tymi danymi. Korzystając z pojęcia czasu wykonania programu definiujemy w zwykły sposób pojęcie złożoności czasowej programu. Jak zwykle, utożsamiamy maszyny RAM z ich programami.

Zachodzi następujące twierdzenie:

Twierdzenie 11.7 *Problemy, które można rozwiązać na wielomianowych maszynach RAM wykonujących dodawanie, można też rozwiązać na wielomianowych maszynach Turinga.*
□

Rozważmy następujący algorytm:

$k := \langle n \rangle$:

$m := 2$;

for $i := 1$ **to** k **do** $m := m * m$.

Symbol n oznacza tu daną liczbę, a interesuje nas wartość zmiennej m po wykonaniu tego algorytmu (sposób obliczania rozmiaru danej $\langle n \rangle$ nie ma znaczenia). Jest oczywiste, podane obliczenia można wykonać na wielomianowej maszynie RAM wykonującej dodawanie i mnożenie. Nietrudno zauważyć, że po wykonaniu podanego algorytmu zmienna m przechowuje wartość

$$2^{2^{\langle n \rangle}}.$$

Rozmiar tej liczby jest równy $2^{\langle n \rangle} + 2$ (pamiętajmy, że zgodnie z przyjętymi zasadami przedstawienie liczby całkowitej zawsze zawiera znak tej liczby). Stąd wynika, że rozważanego algorytmu nie można zrealizować na wielomianowej maszynie Turinga. Maszyna Turinga realizująca ten algorytm musi bowiem zapisać wykładniczo wiele komórek.

Maszyna RAM korzysta z wielomianowej pamięci, jeżeli dla pewnego wielomianu w i dla dowolnych danych d podczas pracy maszyny po uruchomieniu z danymi d, w dowolnym momencie rozmiar zawartości dowolnego rejestru nie przekracza $w(\langle d \rangle)$.

Twierdzenie 11.8 *Problemy, które można rozwiązać na wielomianowych maszynach RAM wykonujących dodawanie i mnożenie, korzystających z wielomianowej pamięci, można też rozwiązać na wielomianowych maszynach Turinga.*

Idea dowodu. Będziemy symulować pracę maszyny RAM na maszynie Turinga M. Na jednej z taśm będzie pamiętana zawartość wszystkich wykorzystywanych rejestrów, na pozostałych pomocnicze informacje, np. adresy rejestrów wykorzystywanych podczas wykonywania aktualnie symulowanego rozkazu. Program maszyny RAM będzie pamiętany w stanach maszyny M.

Ponieważ maszyna RAM wykonuje wielomianowo wiele rozkazów, także wykorzystuje wielomianowo wiele rejestrów. Zawartość każdego rejestru można zapisać w ograniczonym, dającym się z góry przewidzieć, wielomianowym fragmencie taśmy. To samo możma powiedzieć o zawartości wszystkich wykorzystywanych rejestrów.

Przypuśćmy że maszyna M znajduje się w stanie, w którym zaczyna symulować wykonanie pewnego rozkazu, np. arytmetycznego. W tym stanie maszyna M wypisze na odpowiednich taśmach wszystkie dane zawarte w symulowanym rozkazie, a więc zawarte tam stałe i adresy (np. rejestrów z argumentami rozkazu). Dane powinny być pamiętane w programie maszyny M. Następnie przeglądając wielomianowy fragment odpowiedniej taśmy ustali wartości argumentów i przepisze je. Mając wartości argumentów wykona stosowne działanie, a w końcu jego wynik wpisze do odpowiedniego rejestru i przejdzie do stanu, w którym rozpocznie wykonywanie następnego rozkazu. □

Zauważmy jeszcze, że oba sformułowane wyżej twierdzenia można odwrócić. Aby odwrócić pierwsze twierdzenie wystarczy w naturalny sposób symulować maszynę Turinga na maszynie RAM.

11.4 Problemy NP-trudne i NP-zupełne

Symbolem **NP** oznaczamy klasę problemów rozwiązywanych na wielomianowych maszynach Turinga, także niedeterministycznych, a symbolem **P** – klasę problemów rozwiązywanych na deterministycznych, wielomianowych maszynach Turinga.

Będziemy mówić, że problem L_1 jest wielomianowo redukowalny do problemu L_2, jeżeli dane $\vec{x}$ dla problemu L_1 można przekształcić w dane $f(\vec{x})$ dla problemu L_2 tak, aby dane $\vec{x}$ spełniały warunek z definicji problemu L_1 wtedy i tylko wtedy, gdy dane $f(\vec{x})$ spełniają warunek z definicji problemu L_2, i dodatkowo, gdy sposób przekształcania danych f można obliczać w czasie wielomianowym (a więc dane $f(\vec{x})$ można obliczyć na maszynie Turinga działającej w czasie $w(\langle\vec{x}\rangle)$ dla pewnego wielomianu w). Dowody wielomianowej redukowalności często będziemy ograniczać do zdefiniowania odpowiedniego sposobu przekształcania danych pozostawiając sprawdzenie wymaganych własności przekształcenia jako ćwiczenie.

Lemat 11.9 *Jeżeli problem L_1 jest wielomianowo redukowalny do problemu L_2 i problem L_2 jest wielomianowo redukowalny do problemu L_3, to problem L_1 jest wielomianowo redukowalny do problemu L_3 (a więc relacja wielomianowej redukowalności jest przechodnia).* □

Problem L jest **NP**-trudny, jeżeli do niego można wielomianowo zredukować wszystkie problemy z klasy **NP**. Problem L jest **NP**-zupełny, jeżeli należy do klasy **NP** i jest **NP**-trudny. Główny sposób dowodzenia **NP**-zupełności problemów wynika z następującego lematu:

Lemat 11.10 *Jeżeli do problemu L można wielomianowo zredukować pewien problem* **NP***-zupełny, to L jest problemem* **NP***-trudnym.* □

Lemat 11.11 *Jeżeli pewien problem* **NP***-zupełny jest wielomianowy (należy do klasy* **P***), to* **P**=**NP**. □

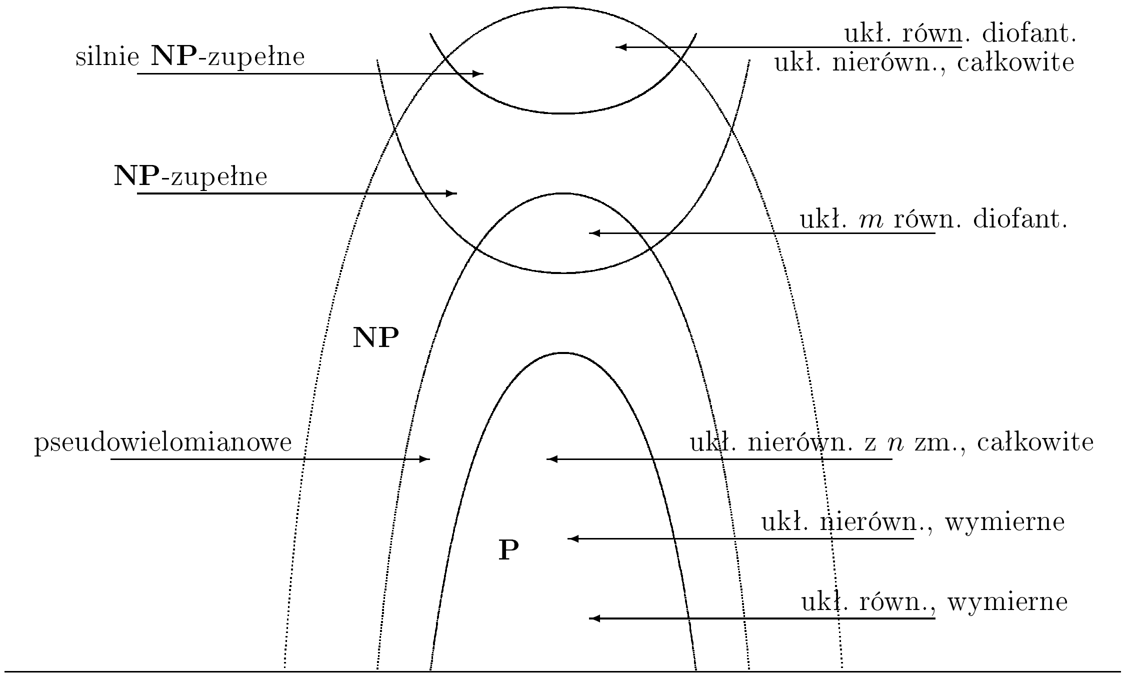
Ponieważ prawdopodobnie okaże się, że $\mathbf{P} \neq \mathbf{NP}$, więc na podstawie podanego lematu, należy oczekiwać, że problemy **NP**-zupełne nie są wielomianowe.

W dalszym ciągu będziemy posługiwać się następującym problemem spełnialności:

Problem 11.1 (spełnialności form zdaniowych)
Dane: forma zdaniowa φ będąca koniunkcją trójczłonowych alternatyw, o członach będących zmiennymi zdaniowymi lub negacjami zmiennych zdaniowych,
Zadanie: sprawdzić, czy istnieje wartościowanie zmiennych zdaniowych, przy którym forma φ jest spełniona.

Twierdzenie Cooka stwierdza, że jest to problem NP-zupełny. Wobec tego wszystkie problemy, do których można go wielomianowo zredukować są problemami NP-trudnymi.

11.5 Przedstawiane rezultaty



Złożoność problemów niesprzeczności układów nierówności lub równań liniowych o rozwiązaniach należących do różnych zbiorów liczb.

11.6 Problemy niesprzeczności układów równań liniowych

Będziemy rozważać następujące problemy:

Problem 11.2 (niesprzeczności układu równań liniowych)
Dane: macierz A i wektor $\vec{b}$, obie struktury o wyrazach całkowitych,
Zadanie: zbadać, czy układ równań $A\vec{x} = \vec{b}$ ma rozwiązanie (będące ciągiem liczb wymiernych).

Problem 11.3 (niesprzeczności układu m diofantycznych równań liniowych)
Dane: macierz A o m wierszach i wektor $\vec{b}$ długości m, o wyrazach całkowitych,
Zadanie: sprawdzić, czy układ równań $A\vec{x} = \vec{b}$ ma rozwiązanie złożone z liczb naturalnych.

Problem 11.4 (niesprzeczności układu diofantycznych równań liniowych)
Dane: macierz A i wektor $\vec{b}$, o wyrazach całkowitych,
Zadanie: sprawdzić, czy układ równań $A\vec{x} = \vec{b}$ ma rozwiązanie złożone z liczb naturalnych.

Powyższe problemy zostały wymienione od najłatwiejszego w sensie złożoności do najtrudniejszego. Pierwszy z nich jest wielomianowy. O pozostałych najpierw pokażemy, że są **NP**-trudne.

11.7 Selektory

Rozważamy następujący pomocniczy problem:

Problem 11.5 (istnienia selektora)
Dane: rodzina $\mathcal{A}$ zbiorów dwu- i trójelementowych,
Zadanie: sprawdzić, czy istnieje zbiór S taki, że dla dowolnego $A \in \mathcal{A}$ przekrój $A \cap S$ jest zbiorem jednoelementowym?

Lemat 11.12 *Problem spełnialności jest wielomianowo redukowalny do problemu istnienia selektora.*

Dowód. Zdefiniujemy funkcję redukującą problem spełnialności do problemu istnienia selektora. Weźmy więc formę zdaniową φ będącą koniunkcją trójczłonowych alternatyw zmiennych i negacji zmiennych zdaniowych.

Najpierw zdefiniujemy przestrzeń zawierającą zbiory z rodziny, którą zamierzamy zdefiniować. Przestrzeń ta będzie zawierać cztery rodzaje elementów:

1) zmienne występujące w φ (w postaci zenegowanej lub nie),

2) negacje tych zmiennych,

3) wszystkie wystąpienia w formie φ zmiennych i ich negacji

4) oraz kopie wystąpień zmiennych i negacji zmiennych.

Jeżeli p jest zmienną występującą w φ, to symbolem p_i będziemy oznaczać i-te wystąpienie tej zmiennej (bez negacji), a symbolem $\neg p_i$ – i-te wystąpienie negacji zmiennej p. Symbole p'_i oraz $\neg p'_i$ będą odpowiednio kopiami i-tego wystąpienia zmiennej p i jej negacji.

Formie φ przyporządkowujemy rodzinę $\mathcal{A}$ o następujących elementach:

1) zbiory $\{p, \neg p\}$ dla każdej zmiennej występującej w φ,

2) zbiory $\{p, p_i, p'_i\}$ dla każdej zmiennej p występującej w φ i dla każdego wystąpienia p_i tej zmiennej,

3) zbiory $\{\neg p, \neg p_i, \neg p'_i\}$ dla każdej zmiennej p występującej w φ i dla każdego wystąpienia $\neg p_i$ negacji tej zmiennej,

4) zbiory postaci $\{x, y, z\}$ wyznaczone przez alternatywy będące członami koniunkcji φ, gdzie x, y i z są tymi wystąpieniami zmiennych zdaniowych bądź negacji zmiennych zdaniowych, z których składa się odpowiednia alternatywa.

Przypuśćmy, że forma φ jest spełnialna. Niech h będzie wartościowaniem spełniającym φ. Pokażemy, że rodzina $\mathcal{A}$ ma selektor. Ten selektor S będzie się składać

1) ze zmiennych występujących w φ, fałszywych przy wartościowaniu h

2) negacji zmiennych prawdziwych przy tym wartościowaniu (a więc do S zaliczamy fałszywe literały),

3) z wybranych po jednym z każdej alternatywy wystąpień zmiennych bądź negacji zmiennych, ale wystąpień spełnionych przy wartościowaniu h

4) oraz z kopii tych spełnionych wystąpień zmiennych bądź negacji zmiennych, które nie zostały wybrane.

Przekroje S ze zbiorami członów alternatyw są oczywiście jednoelementowe. To samo dotyczy przekrojów S ze zbiorami postaci $\{p, \neg p\}$. Będziemy teraz analizować przekrój S ze zbiorem postaci $\{p, p_i, p_i'\}$. Jeżeli p jest zmienną fałszywą przy wartościowaniu h, to p należy do tego zbioru i do S. I jest to jdyny element tego zbioru i S, gdyż do S należą wyłącznie wystąpienia spełnione i kopie takich wystąpień. Jeżeli p jest zmienną prawdziwą przy wartościowaniu h, to do S należy p_i lub p_i' zależnie od tego, czy wystąpienie to zostało wybrane, i jeden z tych dwóch elementów. Podobnie obliczamy przekrój z S zbiorów postaci $\{\neg p, \neg p_i, \neg p_i'\}$. Wobec tego S jest selektorem zdefiniowanej rodziny zbiorów.

Załózmy teraz, że rodzina $\mathcal{A}$ ma selektor S. Definiujemy wartościowanie h przyjmując, że $h(p) = 1$ wtedy i tylko wtedy, gdy $p \notin S$. Weźmy dowolną alternatywą, będącą członem φ. Zbiór $\{x, y, z\}$ członów tej alternatywy jest elementem $\mathcal{A}$. Jeden z elementów tego zbioru należy do S. Pokażemy, że ten element jest spełniony przy wartościowaniu h. Przypuśćmy, że jest to wystąpienie p_i zmiennej p. Ze zbioru $\{p, p_i, p_i'\} \in \mathcal{A}$ do S należy p_i, a ponieważ S jest selektorem, więc $p \notin S$. Wobec tego $h(p) = 1$. Jeżeli elementem S jest wystąpienie $\neg p_i$ negacji zmiennej p, to analogicznie pokazujemy, że $\neg p \notin S$. Ponieważ $\{p, \neg p\} \in \mathcal{A}$, więc $p \in \mathcal{A}$. W tej sytuacji $h(p) = 0$ i w konsekwencji wystąpienie $\neg p_i$ jest spełnione przy wartościowaniu h. $\Box$

Wniosek 11.13 *Problem istnienia selektora jest NP-zupełny.* $\Box$

11.8 NP-trudność problemów niesprzeczności układów diofantycznych

Łatwo dowodzi się, że problem 11.4 jest **NP**-trudny.

Twierdzenie 11.14 *Problem istnienia selektora redukuje się wielomianowo do problemu 11.4 niesprzeczności układu diofantycznych równań liniowych.*

Dowód. Przypuśćmy, że mamy rodzinę zbiorów złożoną ze zbiorów dwu- i trójelementowych. Dla takiej rodziny zbiorów zdefiniujemy odpowiadający jej układ równań. Równania tego układu będą wyznaczone przez zbiory z rozważanej rodziny, a zmienne występujące w tych równaniach będziemy utożsamiać z elementami tych zbiorów. Jeżeli w danej rodzinie jest zbiór $\{x, y, z\}$, to odpowiada mu równanie $x + y + z = 1$, a zbiorowi $\{x, y\}$ odpowiada równanie $x + y = 1$.

Wymagane własności zdefiniowanego w ten sposób przekształcenia danych są łatwe do udowodnienia. $\Box$

Trudniej dowodzi się analogiczne twierdzenie dla problemu 11.3, w którym rozważamy układy o określonej liczbie równań. Zauważmy, że twierdzenie to wystarczy dowieść dla układów złożonych z jednego rówania.

Twierdzenie 11.15 *Problem istnienia selektora redukuje się wielomianowo do problemu 11.3 niesprzeczności pojedyńczego diofantycznego równania liniowego (układu m diofantycznych równań liniowych dla $m = 1$).*

Dowód. Przypuśćmy, że mamy rodzinę zbiorów $\mathcal{A} = \{A_0, \ldots, A_{n-1}\}$ stanowiącą dane dla problemu istnienia selektora. Niech $X = \{x_0, \ldots, x_{m-1}\}$ będzie sumą zbiorów z rodziny $\mathcal{A}$.

Zdefiniujemy macierz M, której wiersze odpowiadają odpowiadają elementom zbioru X (są indeksowane liczbami od 0 do $m - 1$), a kolumny są wyznaczone przez zbiory należące do $\mathcal{A}$ (są indeksowane liczbami od 0 do $n - 1$). W i-tym wierszu (odpowiadającemu elementowi a_i), w j-tej kolumnie (wyznaczonej przez zbiór A_j) umieszczamy liczbę $a_{i,j} = 1$, jeżeli $x_i \in A_j$, a w przeciwnym razie umieszczamy tam $a_{i,j} = 0$.

Niech $p = m+1$. Wiersze macierzy M będziemy interpretować jako przedstawienia o podstawie p pewnych liczb. Tak więc informację w i-tym wierszu M interpretujemy jako liczbę $u_i = \sum_{k=0}^{n-1} a_{i,k} \cdot p^k$. Przyjmijmy, że

1) $V = \sum_{k=0}^{2m+n-1} p^k$,

2) $w_i = p^i + p^{2m+n-1-i}$ dla $i < m$,

3) $v_i = w_i + p^m \cdot u_i$ dla $i < m$.

Przyjmujemy, że rodzina $\mathcal{A}$ wyznacza równanie $f(\mathcal{A})$ postaci

$$\sum_{k=0}^{m-1} v_i \cdot x_i + \sum_{k=0}^{m-1} w_i \cdot y_i = V.$$

Przypuśćmy, że $X_0, \ldots, X_{m-1}, Y_0, \ldots, Y_{m-1}$ są liczbami spełniającymi równanie $f(\mathcal{A})$, a więc zachodzi równość

$$\sum_{k=0}^{m-1} v_i \cdot X_i + \sum_{k=0}^{m-1} w_i \cdot Y_i = V. \tag{37}$$

Zauważmy, że dla wszystkich j takich, że $1 \leq j \leq m$ oraz $i < m$ zachodzą równości

$$v_i \bmod p^j = w_i \bmod p^j = \begin{cases} 0 & \text{gdy } j \leq i \\ p^i & \text{gdy } j > i \end{cases} \quad \text{oraz} \quad V \bmod p^j = \sum_{k=0}^{j-1} p^k.$$

Wynika z nich, że

$$\sum_{k=0}^{j-1} p^k (X_k + Y_k) \bmod p^j = \sum_{k=0}^{j-1} p^k. \tag{38}$$

Prawdziwa jest także nierówność

$$\sum_{k=0}^{j-1} p^{2m+n-1-k}(X_k + Y_k) < \sum_{k=0}^{2m+n-1} p^k. \tag{39}$$

Z równości (38) dla $j = 1$ wynika, że $(X_0 + Y_0) \bmod p = 1$, a więc albo $X_0 + Y_0 = 1$, albo $X_0 + Y_0 > p$. W drugim przypadku, z nierówności (39) dla $j = 1$ otrzymujemy, że

$$p^{2m+n} < p^{2m+n-1}(X_0 + Y_0) < \sum_{k=0}^{2m+n-1} p^k < p^{2m+n},$$

co jest niemożliwe. Stąd $X_0 + Y_0 = 1$.

Teraz, przez indukcję pokażemy, że $X_k + Y_k = 1$ dla wszystkich $k < m$. Załóżmy, że te równości są prawdziwe dla wszystkich $k < j-1$. Równość (38) przyjmuje więc postać

$$\sum_{k=0}^{j-2} p^k + p^{j-1}(X_{j-1} + Y_{j-1}) \bmod p^j = \sum_{k=0}^{j-1} p^k,$$

i implikuje, że

$$p^{j-1}(X_{j-1} + Y_{j-1}) = p^{j-1} + c \cdot p^j$$

dla pewnej liczby naturalnej c. Po podzieleniu przez p^{j-1} i podstawieniu do wzoru (39) otrzymamy nierówność

$$\sum_{k=0}^{j-2} p^{2m+n-1-k} + p^{2m+n-j}(1 + c \cdot p) < \sum_{k=0}^{2m+n-1} p^k,$$

z której wynika

$$p^{2m+n-j}(1+c\cdot p) < \sum_{k=0}^{2m+n-j} p^k < p^{2m+n-j+1}.$$

Dla c większego od 0 otrzymana nierówność jest fałszywa. Tak więc $c=0$ oraz $X_{j-1}+Y_{j-1}=1$.

Przekształćmy równość (37) korzystając z definicji współczynników v_i i w_i. Otrzymamy wtedy

$$\sum_{k=0}^{m-1} p^m \cdot u_i \cdot X_i + \sum_{k=0}^{m-1} (p^i + p^{2m+n-1-i}) \cdot (X_i+Y_i) = V.$$

Przenosząc na drugą stronę powtarzające się potęgi p, a następnie dzieląc stronami przez p^m nadamy powyższej równości postać

$$\sum_{k=0}^{m-1} u_i \cdot X_i = \sum_{k=0}^{n-1} p^k.$$

Z równości $X_i+Y_i=1$ wynika, że X_i jest równe 0 lub 1. Niech $S=\{i<m : X_i=1\}$. Z udowodnionej równości wynika, że

$$\sum_{i\in S} u_i = \sum_{k=0}^{n-1} p^k.$$

Spróbujmy teraz wyliczyć sumę $\sum_{i\in S} u_i$ posługując się macierzą M. Najpierw z tej macierzy wykreślamy wiersze o indeksach nie należących do S. Wiersze otrzymanej macierzy M^S są przedstawieniami sumowanych liczb u_i. Będziemy sumować liczby stosując algorytm pisemnego dodawania. Ponieważ w każdej kolumnie występują tylko cyfry 0 lub 1 i jest ich mniej niż podstawa p systemu, którym się posługujemy, więc podczas dodawania nie występuje przeniesienie. Oznacza to, że

$$\sum_{i\in S} u_i = \sum_{i\in S}\sum_{k=0}^{n-1} a_{i,k}\cdot p^k = \sum_{k=0}^{n-1}(\sum_{i\in S} a_{i,k})\cdot p^k = \sum_{k=0}^{n-1} p^k$$

i wartości $\sum_{i\in S} a_{i,k} \le m < p$ możemy uważać za cyfry systemu o podstawie p. Z twierdzenia o jednoznaczności przedstawienia przy podstawie p wynika, że $\sum_{i\in S} a_{i,k} = 1$ dla wszystkich $k<n$. Udowodnione równości można odczytać jako stwierdzenie, że dla każdego $k<n$ istnieje dokładnie jedno $i\in S$ takie, że $a_{i,k}=1$, lub w równoważny sposób, że dla każdego $k<n$ istnieje dokładnie jedno $i\in S$ takie, że $x_i\in A_k$. Tak więc zbiór $\{x_i : i\in S\}$ jest selektorem rodziny $\mathcal{A}$.

Udowodniliśmy więc, że jeżeli równanie $f(\mathcal{A})$ ma rozwiązanie, rodzina $\mathcal{A}$ ma selektor. Prawdziwa jest też implikacja odwrotna. Jej dowód jest znacznie łatwiejszy i zostanie pominięty. □

Wniosek 11.16 *Problem 11.3 niesprzeczności układu złożonego z jednego liniowego równania diofantycznego jest* **NP**-*trudny.* □