... Nie mogła być błędna tylko dlatego, że była prosta ...
Nie było dokładnie wiadomo, do czego prowadzi, jednak było dość oczywiste, że powinna być udostępniona. W zasadzie chciałem opublikować ten pomysł i powiedzieć: Oto zgrabna koncepcja - wyjaśnia na czym polega ten problem, wyjaśnia to, że jego rozwiązanie jest osiągalne i staje się jasno zdefiniowanym problemem badawczym. Teraz niech wkroczą inni i zobaczymy, co jeszcze będziemy mogli znaleźć. ...

[cytat słów Ralpha Merkle - Steven Levy, "Rewolucja kryptografii", WNT, W-wa, 2002, s. 89-90]

PRZEDMOWA

Od autora [manifest]

Istota rodzącej się w latach 50-tych ubiegłego stulecia tytułowej tematyki, której jestem rówieśnikiem, jest ściśle związana z pojęciem *ZUPEŁNOŚCI*. Gdy kilka lat temu zetknąłem się z teorią złożoności, byłem nie tylko jej rówieśnikiem – byłem w tym względzie zupełnym laikiem, mimo trwałych związków z informatyką od 1971 roku.

Lektura książki Davida Harela „*Algorytmika. Rzecz o istocie informatyki*", będąc źródłem inspiracji do prób znalezienia odpowiedzi na formułowane w niej pytania, zaowocowała szeregiem pomysłów. Przybrały one postać konkretnych modeli i koncepcji rozwiązania kilku problemów charakteryzujących się luką algorytmiczną.

Książka, którą zdecydowałem się zredagować, powstała na podstawie prac prowadzonych poza nurtem życia naukowego. Nie był to jednak świadomy wybór. Koncepcje odwołujące się do prostych, elementarnych modeli, abstrahujące istniejący zaawansowany aparat matematyczny i formalizmy wysokiego poziomu, były zapewne zbyt szokujące, by je traktować poważnie.

Teraz, gdy szkice tych idei mają bardziej dojrzały kształt, jest czas by stały się jednak przedmiotem weryfikacji i oceny ich wartości także poza środowiskiem akademickim. Dlatego zdecydowałem, że oprócz ograniczonego nakładu książki w tradycyjnej postaci, prace udostępnione zostaną także w postaci publikacji elektronicznej. W szczególności wydanie elektroniczne ma sprzyjać przełamaniu bariery w swobodzie wymiany i udostępniania informacji. W ten sposób nawiązuję w pewnym sensie do idei Tima Gowersa.

Wybór takich sposobów prezentacji rezultatów moich prac nie jest wyrazem kontestowania przyjętych w kręgach akademickich norm

i zwyczajów. Jest raczej formą ich wzbogacenia o swoisty mechanizm demokracji, który nie należy jednak utożsamiać z ustalaniem prawdy naukowej metodą głosowania. Zamieszczone motto wystarczająco komentuje wybór tej formy demokracji.

Prezentowane wyniki, nie są zapewne wolne od błędów i nieścisłości. Mimo to powinny być zrozumiałe i czytelne. Liczę więc, że znajdą zainteresowanie. Wyrażone oczekiwania wynikają z kilku przesłanek.

Po pierwsze, uważam podobnie jak Ralph Merkle, że zarówno koncepcje jak i same rozwiązania warte są dalszej pracy i „… *nie mogą być błędne tylko dlatego, że są proste*". Otwierając nową perspektywę, mimo destrukcyjnego charakteru, „ … *jest oczywiste, że powinny być udostępnione*".

Po drugie, jeden z dyskutowanych w książce problemów autoryzowali Panowie Adleman, Rivest i Shamir. Zakładam więc, że każdy, dla którego tematyka książki nie jest obca, ma świadomość znaczenia rozwiązania tego i pozostałych dyskutowanych problemów. Pisze o tym Pan Schneier w swojej książce „*Cryptography*". Także Panowie Arora i Barak piszą o tym w przeglądanym przeze mnie drafcie ich książki „*Computational Complexity: A Modern Approach*".

Po trzecie, moje rozwiązania wykorzystują z jednej strony znane już podstawowe wyniki badań w teorii złożoności. Z drugiej strony wykorzystują zupełnie nowe pomysły odwzorowane przez elementarne modele.

Wreszcie po czwarte, traktując tematykę złożoności obliczeniowej jako hobby, chciałbym poznać popełnione błędy, a wśród nich być może błędy kardynalne, jeśli je popełniłem.

Niezależnie od możliwych błędów, wyrażam nadzieję, że udostępnione przez publikację tej książki prace, wnoszą pewien nowy element w rozwój nauki i w obszary praktycznych zastosowań informatyki.

Także niezależnie od tego jak ta wiara zostanie potraktowana, pozostanie mi satysfakcja z potyczek z piękną, acz trudną algorytmiką, Zaś Wszystkim zwolennikom platonizmu w matematyce, dedykuję krótki esej.

Mam nadzieję, że odebrany zostanie jako przemyślenia ***nie-ZUPEŁNEGO*** laika.

Pojęcie ZUPEŁNOŚCI, *zdefiniowane w teorii złożoności obliczeniowej, jest bytem obiektywnie występującym w realnym świecie, niezależnym od czasu i przestrzeni. Jego istota polega na tym, że „coś nie jest gorsze od czegoś" z jednej strony i „to coś nie jest lepsze od czegoś innego". Do*

wykazania takich relacji wykorzystywany jest mechanizm redukcji wielomianowej.

Rok 1736 – Euler formułuje podwaliny teorii grafów, definiuje problem **EULER-PATH** *oraz* **EULER-CYCLE**. *Dla obu podaje jedynie algorytmy wykładnicze. Jednocześnie definiuje problem* **P** *versus* **NP** *z pierwotnym problemem* **NP-***zupełnym, czyli spełnialnością. Wykorzystując pojęcie* ZUPEŁNOŚCI *pokazuje istnienie redukcji wielomianowej* **EULER-PATH**
w obie strony do i z **3SAT** *i tym samym jego przynależność do klasy problemów* **NP-***zupełnych.*

Rok 1971 – Cook pokazuje wielomianowe rozwiązanie problemu **EULER-PATH** *(dany graf G jest grafem Eulera wtedy i tylko wtedy, gdy wszystkie wierzchołki G są stopnia parzystego). Tym samym pokazał relację* **P = NP**.

Zawartość i układ książki

Książka jest zbiorem opracowanych w różnym czasie rezultatów prowadzonych prac. Przedmiotem tych prac był problem spełnialności **SAT** i jego warianty, wariant problemu faktoryzacji **MULT(n)** rozkładu iloczynu dwóch liczb pierwszych na czynniki proste oraz problem znajdywania największego wspólnego dzielnika **NWD**. W szczególności dwa pierwsze problemy (**SAT** i **MULT(n)**) rozważane są w modelu maszyny Turinga oraz równoważnym modelu sieci logicznych. Dla obu problemów i w obu modelach, uzyskane wyniki prowadzą ogólnie do „nieprawdopodobnego" rozwiązania w postaci równości klas ***NC = P = NP***.

Całość książki podzielona jest na dwie części. Część pierwsza, zatytułowana *„W kręgu spełnialności"* poświęcona jest problemowi spełnialności **SAT**. W części drugiej, zatytułowanej *„W kręgu faktoryzacji"* zostały umieszczone rozdziały poświęcone dyskusji problemów **NWD** i **MULT(n)**. Taki podział powinien ułatwić zaznajomienie się z prezentowanymi treściami, a przy tym czytać je niezależnie.

Na ogólny wynik w postaci równości klas ***NC = P = NP*** składają się rozwiązania przedstawione w poszczególnych rozdziałach. I tak w części pierwszej pokazuję, że:
- dla problemu **3SAT** istnieje wielomianowa jednostajna sieć logiczna, co rozstrzyga kwestię ***P versus NP*** (rozdział 1);
- przy analizie problemów **3SAT** wygodnie jest posłużyć się modelem referencyjnym, który uwzględnia wszystkie możliwe przypadki i warianty

problemów **3SAT** oraz, że wprowadzony model referencyjny dla dowolnego *k>1* jest strukturalnie identyczny (rozdział 2).
- dla elementarnych struktur modeli referencyjnych istnieją różnorodne odwzorowania pozwalające konstruować algorytmy wielomianowe dla **3SAT**, co rozstrzyga równość klas ***P = NP***, a z wykorzystaniem równoległości, zarówno dla **2SAT** jak i **3SAT,** konstruować algorytmy charakteryzujące się czasem polilogarytmicznym, co rozstrzyga równość klas ***NC = P*** oraz, że przy pomocy tych przekształceń możliwe jest rozwiązanie problemu **MAX2SAT** (rozdział 3 i 4).

W części drugiej pokazuję, że:
- **NWD** należy do problemów *P-zupełnych* i posiada równoległy algorytm *log Time*, co rozstrzyga równość klas ***NC = P*** (rozdział 5);
- dla problemu **MULT(n)** istnieje model umożliwiający skonstruowanie równoległego algorytmu *log Time*, co przy założeniu *NP-zupełności* problemu **MULT(n)** bezpośrednio rozstrzyga równość klas **NC = NP**, (rozdział 6);
- istnieje redukcja problemu **MULT(n)** do **CIRCUIT SAT**, a tym samym **MULT(n)**
należy do klasy problemów *NP - zupełnych* oraz, że istnieje wielomianowa jednostajna sieć logiczna dla **MULT(n)** (rozdział 7).

Każdy rozdział zredagowany został w formie niezależnego opracowania, co z jednej strony pozwala czytać je niezależnie. Taki sposób redakcji sprawił jednak, że w części opracowań pojawiają się powtórzenia pewnych treści. Prezentowane opracowania, zawierają opis koncepcji rozwiązania tytułowego problemu, niekiedy krytyczną analizę stosowanych metodologii oraz polemikę z niektórymi twierdzeniami, a wreszcie rozwiązania dyskutowanych problemów.

Część pierwsza

W kręgu spełnialności

R o z d z i a ł p i e r w s z y

Dyskusja sieci logicznej dla 3SAT

Marek Malinowski

Technologie Teleinformatyczne

marmal.pl@home.pl

Dyskusję poprzedza przegląd rezultatów badań złożoności obliczeniowej prowadzonych z wykorzystaniem modelu sieci logicznych. Zaprezentowane wyniki upoważniają do stwierdzenia, że dla problemu **3SAT** istnieje jednostajna sieć wielomianowa. W szczególności pokazano: - że problem **3SAT** może być traktowany jako koniunkcja elementarnych problemów **3SAT**; - że dla dowolnego elementarnego problemu **3SAT** istnieje uniwersalna sieć logiczna o stałym rozmiarze (ilości bramek); - i ostatecznie, że dla koniunkcji elementarnych problemów **3SAT** istnieje sieć wielomianowa, i jest to sieć jednostajna.

1

Zrób dobry początek

[z teorii rozwiązywania problemów]

1.1 Wstęp – sformułowanie tytułowej tezy.

Rozstrzyganie charakteru relacji zawierania klas $P \subseteq NP$ bazuje na parametryzacji klas złożoności poprzez model obliczeń (maszyna Turinga), tryb obliczeń (determinizm i niedeterminizm), zasoby (czas i pamięć) i ograniczenia (notacja $O(.)$).

Przyjęty sposób parametryzacji stał się podstawą sformułowania twierdzenia Cooka-Levina. Określa ono pierwotny problem ***NP-zupełny***, a pośrednio wskazuje, że jeśli pokazany zostanie wielomianowy algorytm dla problemu **3SAT** (lub innego problemu ***NP-zupełnego***), to $N = NP$. Z kolei, jeśli udowodnione zostanie wykładnicze dolne ograniczenie, to $P \neq NP$.

W toku prowadzonych badań, określony na wstępie sposób parametryzacji klas złożoności był uzupełniany o inne proste modele, będące odpowiednikiem modelu maszyny Turinga. Wiadomo na przykład, że maszyny Turinga można symulować za pomocą sieci logicznych. Dowód twierdzenia wiążącego złożoność sieci logicznych ze złożonością czasową można znaleźć w książce Michaela Sipsera [1].

Przydatność modelu sieci logicznych w rozstrzyganiu problemu ***P versus NP*** wynika z bliskich związków sieci logicznych z problemem spełnialności **SAT**.

Po pierwsze, problem **SAT** został sformułowany w kategoriach rachunku zdań. Ponieważ formy zdaniowe można wyrazić w postaci funkcji logicznych, to **SAT** może być definiowany poprzez funkcje logiczne, a te wygodnie jest reprezentować w postaci sieci logicznych.

Po drugie, dowód twierdzenia Cooka-Levina można alternatywnie przeprowadzić przy pomocy właśnie sieci logicznych [1, s. 401-408]. Wreszcie, sieci logiczne znajdują praktyczną realizację w postaci układów cyfrowych obliczających funkcje logiczne [2].

Wiadomo także, że każdy problem z klasy ***P*** ma sieć wielomianową. Niestety, ponieważ istnieją sieci wielomianowe dla problemów nierozstrzygalnych, stwierdzenie odwrotne nie jest prawdziwe. Wprowadza się więc pojęcie jednostajnych sieci wielomianowych i poprzez stwierdzenie, że jednostajna sieć wielomianowa dla reprezentowanego przez nią pewnego problemu istnieje wtedy i tylko wtedy, gdy problem ten należy do klasy ***P***, wiąże się je z obliczeniami wielomianowymi.

Powyższe stwierdzenie można traktować jako równoważne twierdzeniu Cooka-Levina. Wskazuje ono, że jeśli pokazana zostanie jednostajna sieć wielomianowa dla pewnego problemu ***NP*** - *zupełnego*, to ***P = NP***. Z kolei, jeśli udowodnione zostanie, że problemy ***NP*** - *zupełne* nie mają jednostajnych sieci wielomianowych, to ***P*** ≠ ***NP***.

Wbrew pewnym przesłankom (np. twierdzenie Razborowa) przemawiającym za hipotezą, że problemy ***NP*** - *zupełne* nie mają jednostajnych ani niejednostajnych sieci wielomianowych (**Hipoteza B** [3 s. 287], w dalszej części dyskutowany będzie schemat dowodu istnienia jednostajnej sieci wielomianowej dla pierwotnego problemu ***NP*** - *zupełnego* jakim jest **3SAT**.

Na niekorzyść **Hipotezy B** przemawia samo określenie problemu ***NP***, według którego weryfikacja (sprawdzenie) certyfikatu rozwiązania problemu odbywa się w deterministycznym czasie wielomianowym. Można więc założyć, że proces weryfikacji będzie zrealizowany przez sieć wielomianową. Ta sytuacja koresponduje z przywoływanymi dalej stwierdzeniami i określeniami dotyczącymi sieci i ich związków z problemami **CIRCUIT SAT** i **CIRCUIT VALUE** i definicją sieci jednostajnej [1] (fakt 2.7).

W kolejnych etapach wnioskowania wykorzystane zostaną, przedstawione w następnym punkcie, znane już rezultaty badań w modelu sieci logicznych.

1 W istocie, zgodnie z definicją, sieć jednostajna ma rozstrzygać „specjalny" rodzaj certyfikatu – słowo, w którym wszystkie zmienne mają ustaloną wartość logiczną *true*.

1.2 Funkcje logiczne i sieci logiczne

Podstawowe stwierdzenia dotyczące modelu sieci logicznych można przedstawić w postaci faktów, których zdecydowana większość została zaczerpnięta z książki Christosa Papadimitriou [3].

Fakt 2.1: Ekwiwalentność formuł logicznych i funkcji logicznych [3, s. 95].

Każda formuła φ ze zmiennymi $x_1, \ldots, x_n$ wyraża n-argumentową funkcję logiczną f i odwrotnie, dowolna n-argumentowa funkcja logiczna f może być wyrażona jako formuła φ zawierająca zmienne $x_1, \ldots, x_n$.

Fakt 2.2: Reprezentacja funkcji logicznych przez sieci logiczne [3, s. 95-96].

Każda funkcja logiczna może być przedstawiona jako sieć logiczna, zdefiniowaną jako graf $C = (V, E)$, w którym $V = \{1, \ldots, n\}$ są bramkami grafu C, zaś E zbiorem krawędzi (i, j) przy $i < j$ (warunek acykliczności).

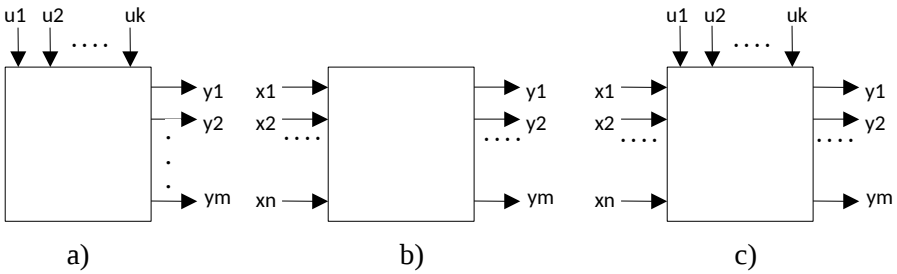
Składnia sieci ponadto wiąże z każdą bramką $i \in V$ rodzaj $s(i) \in \{true, false\} \cup \{x_1, \ldots, x_n\} \cup \{\vee, \wedge, \neg\}$ (gdzie $\{true, false\}$ są bramkami stałych, $\{x_1, \ldots, x_n\}$ są bramkami zmiennych oraz $\{\vee, \wedge, \neg\}$ są bramkami OR, AND, NOT odpowiednio) i charakteryzuje je stopniem wejściowym (liczba wchodzących krawędzi) równym 0, 1 lub 2. Wierzchołek, który nie ma krawędzi wychodzących, nazywany bramką wyjściową sieci oblicza funkcję logiczną.

Jeśli będziemy rozważać sieci, które mają kilka wyjść, to obliczają one kilka funkcji jednocześnie. W oparciu o tak zdefiniowaną sieć, możliwe jest konstruowanie trzech typów sieci:

- sieć bez zmiennych (rys.1 a), tj. z bramkami $V \in \{true, false\} \cup \{\vee, \wedge, \neg\}$;
- sieć ze zmiennymi i bez stałych (rys. 1 b), tj. z bramkami $V \in \{x_1, \ldots, x_n\} \cup \{\vee, \wedge, \neg\}$;
- sieć ze zmiennymi i stałymi (rys. 1 c), tj. z bramkami $V \in \{true, false\} \cup \{x_1, \ldots, x_n\} \cup \{\vee, \wedge, \neg\}$.

Uwaga 2.1: Każda sieć wielowyjściowa w elementarny sposób daje się sprowadzić przy pomocy sieci rozszerzającej C_{ext}, do sieci z jednym wyjściem. Na przykład, dla sieci m wyjściowej, sieć rozszerzająca C_{ext} może być zbudowana z nie więcej niż $(m-1)$ bramek AND w układzie sekwencyjnym lub równoległym.

Dopuszczając sieci wielowyjściowe, każdy z typów sieci można poprzez analogię z układami kombinacyjnymi traktować jako wielobiegunnik. Schematy ideowe wyróżnionych typów sieci przedstawia rysunek 1.1.



Rys. 1.1 Schematy ideowe typów sieci logicznych, gdzie:
X – wektor sygnałów wejściowych (zmienne $x_1, x_2, \ldots, x_n$);
U – wektor stałych $u_i \in \{true, false\}$ $i=1, 2, \ldots, k$;
Y – wektor sygnałów wyjściowych, $y_j \in \{true, false\}$ $j=1, 2, \ldots, m$ jest obliczoną wartością pewnej funkcji (niekoniecznie funkcji wszystkich *n* zmiennych).

Kolejne stwierdzenia wiążą sieci logiczne z problemami różnych klas złożoności obliczeniowej.

Fakt 2.3: Sieć logiczna jako problem **CIRCUIT VALUE** [3, s.97].

Sieć *C* bez zmiennych (tj. sieć z bramkami $V(i) \in \{true, false\} \cup \{\vee, \wedge, \neg\}$) (schemat ideowy rys. 1.1 a) definiuje problem **CIRCUIT VALUE** $\in$ ***P***.

Fakt 2.4: Sieć logiczna jako problem **CIRCUIT SAT** [3, s. 97].

Sieć *C* ze zmiennymi (rys. 1.1 b, c), kiedy należy stwierdzić czy istnieje takie wartościowanie zmiennych, że bramka wyjściowa zwraca wartość *true*, definiuje problem **CIRCUIT SAT**.

Pokazując [3, s.179] redukcję wielomianową **CIRCUIT SAT** do **3SAT** w pamięci *log(n)*, przyjmuje się, że **CIRCUIT SAT** należy do klasy problemów ***NP****-zupełnych*.

Kolejne przytaczane stwierdzenia są wynikiem prac wiążących złożoność obliczeniową ze złożonością sieci logicznych. Złożoność sieci jest określana jako rozmiar sieci wyrażany liczbą bramek w tej sieci.

Fakt 2.5: Sieci wielomianowe [3, s 285-286].

Sieć ma rozmiar wielomianowy, jeśli istnieje rodzina sieci $C=\{C_0, C_1, \ldots\}$ dla której prawdą jest po pierwsze, że rozmiar C_n jest równy co najwyżej $p(n)$ dla pewnego ustalonego wielomianu p i po drugie, że dla każdej kombinacji zmiennych $x_i \in \{0,1\}$ $i=1,2, \ldots, n$ spełniających formułę reprezentowaną przez tą sieć, wartością sieci jest *true* (w kategoriach języka maszyn Turinga powiedzielibyśmy, że dla słów wejściowych, które maszyna akceptuje – wartością sieci jest *true*).

Fakt 2.6: Każdy problem z klasy ***P*** ma sieć wielomianową [3, s. 286].

Niestety, stwierdzenie odwrotne nie jest prawdziwe. Nie każda sieć wielomianowa jest reprezentacją problemu z klasy ***P***.

Uwaga 2.2: Pokazano, że istnieją problemy nierozstrzygalne, które mają sieci wielomianowe. Sprawia to, że nie możemy wnioskować równości klas ***P = NP***, mimo że sieć *C* definiująca problem **CIRCUIT SAT** (tj. sieć ze zmiennymi, kiedy należy stwierdzić czy istnieje takie wartościowanie zmiennych, że bramka wyjściowa zwraca wartość *true*) musi mieć rozmiar wielomianowy by redukcja do **3SAT** mogła być przeprowadzona w czasie wielomianowym. A jeśli tak, to **CIRCUIT SAT** należałoby zaliczyć do klasy ***P***. Z drugiej strony, jeśli **CIRCUIT SAT** redukuje się do **3SAT**, i tym samym w aspekcie *ZUPEŁNOŚCI* nie jest gorszy od **3SAT**, to i **3SAT** należałoby zaliczyć do ***P***.

Związanie sieci wielomianowych z obliczeniami wielomianowymi dokonuje się poprzez wprowadzenie pojęcia jednostajności sieci.

Fakt 2.7: Wielomianowe sieci jednostajne [3, s. 287].

Rodzina sieci $C=\{C_0, C_1, \dots\}$ jest jednostajna, jeśli dla zmiennych $x_i \epsilon \{1\}$ $i=1, 2, \dots, n$ możliwe jest skonstruowanie sieci C_n w pamięci $\log n$.

Tak zdefiniowana jednostajna sieć wielomianowa prowadzi do stwierdzenia, że jednostajna rodzina sieci wielomianowych dla reprezentowanego przez nią problemu istnieje wtedy i tylko wtedy, gdy problem ten należy do klasy ***P***. Zatem, jeśli dla dowolnego problemu ***NP*** - *zupełnego* pokazana zostanie jednostajna sieć wielomianowa, to rozstrzygniecie kwestii ***P*** *versus* ***NP*** przyjmie postać ***P = NP***.

1.3 Jednostajna sieć wielomianowa 3SAT

Zaprezentowane w poprzednim punkcie stwierdzenia upoważniają do rozważenia schematu dowodu istnienia sieci jednostajnych dla **3SAT**.

Na omawiany schemat składają się kolejno następujące elementy: - poszczególne klauzule w zapisie *CNF* problemu **3SAT** stanowią elementarne problemy **3SAT**; - dla każdego elementarnego **3SAT** istnieje uniwersalna sieć o stałym rozmiarze (ilości bramek); - wszystkie uniwersalne sieci reprezentujące elementarne problemy **3SAT** (klauzule) stanowią wielowyjściową sieć wielomianową *C*, która po jej rozszerzeniu o sieć C_{ext} przekształcającą sieć *C* do sieci jednowyjściowej, rozstrzyga spełnialność formuły **3SAT** *CNF*.

Prawdziwość poszczególnych elementów powyższego schematu prowadzi do wniosku o istnieniu jednostajnej sieci wielomianowej dla problemu **3SAT**.

1.3.1 Elementarny 3SAT.

Funkcja logiczna przyporządkowuje wartościom logicznym zmiennych x wartość logiczną zmiennych y i jest opisywana formalnie za pomocą wyrażeń logicznych. Wyrażenia logiczne mogą być zapisywane w wielu różnorodnych i równoważnych sobie postaciach.

Wyrażenie logiczne definiujemy następująco:

- $0, 1, x_i$ i $\neg x_i$ są wyrażeniami logicznymi oraz jeśli φ_1 i φ_2 są wyrażeniami logicznymi, to $\varphi_1 \vee \varphi_2$, $\varphi_1 \wedge \varphi_2$ są wyrażeniami logicznymi;
- wyrażenie logiczne może mieć tylko postać opisaną wyżej, ale zamiast x_i mogą wystąpić inne zmienne logiczne.

Znamy wszystkie wyrażenia logiczne opisujące funkcje dwóch zmiennych (jest ich 16). Przy ich pomocy możemy opisać każdą funkcję n zmiennych. Jeśli funkcję i zmiennych oznaczymy przez f^i, to $f^2(f^i(x_1, x_2, \dots, x_i), x_{i+1}) = f^{i+1}$. Tak więc na przykład $f^3 = f^2(f^2(x_1, x_2), x_3)$, a ponieważ w wyrażeniu mogą wystąpić dowolne zmienne, więc ogólnie można zapisać $f^3 = f^2(f^2(x_r, x_s), x_t)$.

Jeśli φ będzie wyrażeniem logicznym opisującym funkcję n zmiennych w postaci koniunkcji m funkcji trzech zmiennych:

$$\varphi = f_1^3(x_r, x_s, x_t) \wedge f_2^3(x_r, x_s, x_t) \wedge \dots \wedge f_m^3(x_r, x_s, x_t) \quad (1)$$

gdzie: $x_r, x_s, x_t \in \{x_1, x_2, \dots, x_n\}$
oraz $r \in \{1, 2, \dots n\text{-}2\}$, $s \in (r+1, r+2, \dots, n\text{-}1\}$ i $t \in \{s+1, s+2, \dots n\}$, tj. $r < s < t$,

to opuszczając oznaczenia argumentów poszczególnych funkcji można zapisać, że φ jest funkcją m zmiennych, których rolę pełnią funkcje f_i^3, $i=1, 2, \dots, m$.

$$\varphi = f^m(f_1^3, f_2^3, \dots, f_m^3) = f^2(f^{m-1}(f_1^3, f_2^3, \dots, f_{m-1}^3), f_m^3)$$

Rozwijając ostatnią postać wyrażenia dalej uzyskamy ostatecznie:

$$\varphi = f^2(f^2(\dots f^2(f^2(f_1^3, f_2^3), f_3^3)\dots, f_{m-1}^3), f_m^3) \quad (2)$$

Uzyskana postać wyrażenia pokazuje, że można $f_1^3, f_2^3, \dots f_m^3$ traktować niezależnie jako argumenty kolejnych $(m\text{-}1)$ dwu argumentowych funkcji f

[2] (wszystkie są funkcjami iloczynu logicznego realizowanymi przez bramki AND), począwszy od najgłębiej zagnieżdżonej.

W kategoriach sieci logicznych, odpowiada to sytuacji, w której $f_1^3, f_2^3, \ldots f_m^3$ stanowią niezależne fragmenty w sieci z m wyjściami, tak że *j-te* wyjście rozstrzyga spełnialność *j-tej* funkcji $f_j^3(x_r, x_s, x_t)$.

Fakt 3.1 Elementarny **3SAT**

Jeśli przyjmiemy, że funkcja φ opisana przez wyrażenie *(1)* ma postać normalną iloczynu *CNF*, odpowiadającą definicji problemu **3SAT**, to funkcje f_i^3 $i=1, 2, \ldots, m$ reprezentowane są przez klauzule trzech różnych zmiennych, wtedy każda z ośmiu możliwych postaci klauzul trzech zmiennych może być traktowana jako elementarny **3SAT**.

Tabela 1.1 Zestawienie elementarnych alternatyw i koniunkcji funkcji 3 zmiennych.

Wartościowanie zmiennych x_3 x_2 x_1			Elementarne alternatywy (klauzule)	Elementarne koniunkcje (implikanty)
0	*0*	*0*	$x_3 \vee x_2 \vee x_1$	$\bar{x}_3 \wedge \bar{x}_2 \wedge \bar{x}_1$
0	*0*	*1*	$x_3 \vee x_2 \vee \bar{x}_1$	$\bar{x}_3 \wedge \bar{x}_2 \wedge x_1$
0	*1*	*0*	$x_3 \vee \bar{x}_2 \vee x_1$	$\bar{x}_3 \wedge x_2 \wedge \bar{x}_1$
0	*1*	*1*	$x_3 \vee \bar{x}_2 \vee \bar{x}_1$	$\bar{x}_3 \wedge x_2 \wedge x_1$
1	*0*	*0*	$\bar{x}_3 \vee x_2 \vee x_1$	$x_3 \wedge \bar{x}_2 \wedge \bar{x}_1$
1	*0*	*1*	$\bar{x}_3 \vee x_2 \vee \bar{x}_1$	$x_3 \wedge \bar{x}_2 \wedge x_1$
1	*1*	*0*	$\bar{x}_3 \vee \bar{x}_2 \vee x_1$	$x_3 \wedge x_2 \wedge \bar{x}_1$
1	*1*	*1*	$\bar{x}_3 \vee \bar{x}_2 \vee \bar{x}_1$	$x_3 \wedge x_2 \wedge x_1$

W rachunku zdań, klauzula definiowana jest jako alternatywa literałów określających wartościowanie zmiennych logicznych, zaś koniunkcja literałów określana jest mianem implikantu. Ekwiwalentnymi określeniami tych pojęć w kategoriach funkcji logicznych są elementarna alternatywa i elementarna koniunkcja odpowiednio. Zestawienie możliwych do zdefiniowania ośmiu klauzul (elementarnych alternatyw) i ośmiu implikantów (elementarnych koniunkcji) funkcji trzech zmiennych zawarto w tabeli 1.1.

1.3.2 Uniwersalna sieć logiczna elementarnego 3SAT

Dla ośmiu różnych klauzul trzech zmiennych można zdefiniować 255 funkcji (bez funkcji stałej ***TRUE***) wyrażonych koniunkcją kombinacji

klauzul po jednej, po dwie itd. aż do ośmiu z ośmiu i każdej przyporządkować numer. Każda tak zdefiniowana funkcja będzie zapisana w kanonicznej normalnej postaci koniunkcyjnej (*CNF*).

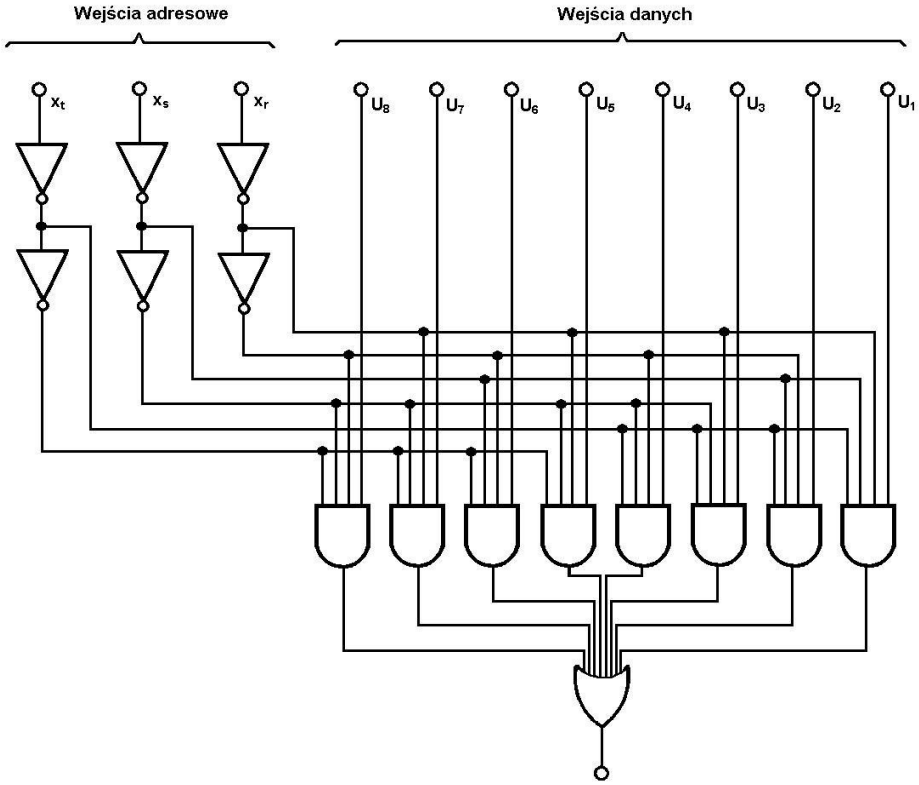
Wykorzystując dualny sposób definiowania funkcji przy pomocy implikantów otrzymamy 255 funkcji (bez funkcji stałej ***FALSE***) wyrażonych alternatywą kombinacji implikantów po jednym, po dwa itd. aż do ośmiu z ośmiu i każdej przyporządkować numer, to każda tak zdefiniowana funkcja będzie zapisana w kanonicznej normalnej postaci dysjunkcyjnej (*DNF*).

Jeśli konstruowanie sieci logicznej reprezentującej funkcje trzech zmiennych (nawet wtedy, gdy ograniczymy się do rozpatrywania 8 funkcji opisywanych przy pomocy pojedynczych klauzul) realizowane będzie wprost w oparciu o wyrażenia klauzul, to mimo, że nie będą skomplikowane, będą różnić się rozmiarem. Na przykład, dla funkcji $f^3(x_r, x_s, x_t) = (x_r \vee x_s \vee x_t)$ wystarczą 2 bramki OR, ale dla funkcji $f^3(x_r, x_s, x_t) = (\neg x_r \vee \neg x_s \vee \neg x_t)$, sieć oprócz 2 bramek OR będzie zawierała jeszcze 3 bramki NOT.

Zróżnicowany rozmiar sieci elementarnego **3SAT** *CNF* nie jest sprzyjającą okolicznością dla zadania konstruowania sieci logicznej dla całego **3SAT** *CNF*. Jednak kosztem zwiększenia rozmiaru sieci logicznej, możliwe jest operowanie uniwersalną siecią logiczną, która może reprezentować dowolną funkcję logiczną trzech zmiennych.

Taką uniwersalna siecią dla przypadku elementarnego **3SAT** *CNF*, jest sieć logiczna reprezentująca funkcję stałą ***TRUE***, a dla przypadku elementarnego **3SAT** *DNF*, sieć reprezentująca funkcję stałą ***FALSE***. W obu przypadkach, podstawą konstruowania sieci jest układ multipleksera 8/1. Rodzaj bramek i sposób ich połączenia w sieci logicznej określa schemat układu prezentowany na rysunku 1.2.

Sygnały wejściowe zmiennych x_r, x_s, x_t , będą podawane na wejścia adresowe układu. Sygnały stałych (wektory $U=[u_1, u_2, \ldots, u_8]$, binarnie interpretowane jako numer funkcji) będą podawane na wejścia danych multipleksera. Sygnał wyjściowy z bramki OR określa tablica stanów (tabela 1.2), z której jednoznacznie wynika, że sygnał na wyjściu układu odpowiada sygnałowi podawanemu na wejście danych określone przez aktualny stan wejść adresowych.



Rys. 1.2 Schemat układu multipleksera 8/1

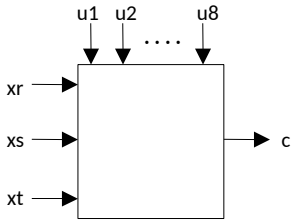
Tabela 1.2 Tablica stanów multipleksera 8/1

wejścia adresowe	x_r	*0*	*1*	*0*	*1*	*0*	*1*	*0*	*1*
	x_s	*0*	*0*	*1*	*1*	*0*	*0*	*1*	*1*
	x_t	*0*	*0*	*0*	*0*	*1*	*1*	*1*	*1*
wyjście		u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8

Pokazany na rysunku 1.2 schemat multipleksera 8/1, pozwala określić rozmiar odpowiadającej jemu sieci logicznej. Każdą występującą w schemacie czterowejściową bramkę AND można zastąpić trzema dwuwejściowymi bramkami AND. Z kolei ośmiowejściową bramkę OR można zastąpić siedmioma bramkami OR. Ostatecznie rozmiar sieci logicznej odwzorowującej układ multipleksera będzie mieć rozmiar łącznie 48 bramek, w tym 3 bramki wejściowe zmiennych, 8 bramek wejściowych stałych, 24 dwuwejściowych bramek AND, 7 dwuwejściowych bramek OR i 6 bramek NOT.

W oparciu o przeprowadzoną w tym punkcie dyskusję sieci logicznej elementarnego **3SAT**, możemy sformułować stwierdzenie.

Fakt 3.2: Każda funkcja logiczna f trzech różnych zmiennych x_r, x_s, x_t, opisana wyrażeniem w jednej z dwóch kanonicznych normalnych postaci koniunkcyjnej (*CNF*) lub dysjunkcyjnej (*DNF*), ma uniwersalną sieć logiczną o stałym rozmiarze k bramek. Przyjmuje ona postać wielobiegunnika, przedstawioną na rysunku 3, co odpowiada schematowi ideowemu z rys. 1.1 c.



Rys. 1.3 Schemat ideowy uniwersalnej sieci logicznej funkcji trzech zmiennych.

1.3.3 Konstrukcja sieci 3SAT

W przypadku rozpatrywania wszystkich możliwych 255 funkcji zdefiniowanych przez wyrażenia postaci *CNF*, musielibyśmy operować 255 różnymi wektorami stałych $U=[u_1, u_2, \dots, u_8]$.

Ponieważ przyjęliśmy, że jako elementarne **3SAT** będziemy traktować funkcje opisywane przez wyrażenia mające postać pojedynczych klauzul, to możemy ograniczyć się do rozpatrywania ośmiu funkcji definiowanych przez osiem różnych klauzul i wtedy przy konstruowaniu sieci operować będziemy ośmioma wektorami stałych U, które jednoznacznie są określane przez postaci klauzul.

$$
U = \begin{cases}
[0,1,1,1,1,1,1,1] & \text{jeśli} \quad x_3 \vee x_2 \vee x_1 \\
[1,0,1,1,1,1,1,1] & \text{jeśli} \quad x_3 \vee x_2 \vee \overline{x}_1 \\
[1,1,0,1,1,1,1,1] & \text{jeśli} \quad x_3 \vee \overline{x}_2 \vee x_1 \\
[1,1,1,0,1,1,1,1] & \text{jeśli} \quad x_3 \vee \overline{x}_2 \vee \overline{x}_1 \\
[1,1,1,1,0,1,1,1] & \text{jeśli} \quad \overline{x}_3 \vee x_2 \vee x_1 \\
[1,1,1,1,1,0,1,1] & \text{jeśli} \quad \overline{x}_3 \vee x_2 \vee \overline{x}_1 \\
[1,1,1,1,1,1,0,1] & \text{jeśli} \quad \overline{x}_3 \vee \overline{x}_2 \vee x_1 \\
[1,1,1,1,1,1,1,0] & \text{jeśli} \quad \overline{x}_3 \vee \overline{x}_2 \vee \overline{x}_1
\end{cases}
$$

W poszczególnych wektorach występują stałe *false* na kolejnych jego pozycjach. Dla określenia numeru pozycji, na których występują one w wektorze U, wystarczy by negatywne literały w zapisie klauzuli interpretować jako cyfry „1" liczby binarnej. Na przykład $(\overline{x}_3 \vee \overline{x}_2 \vee x_1) \equiv 110_2 = 6_{10}$.

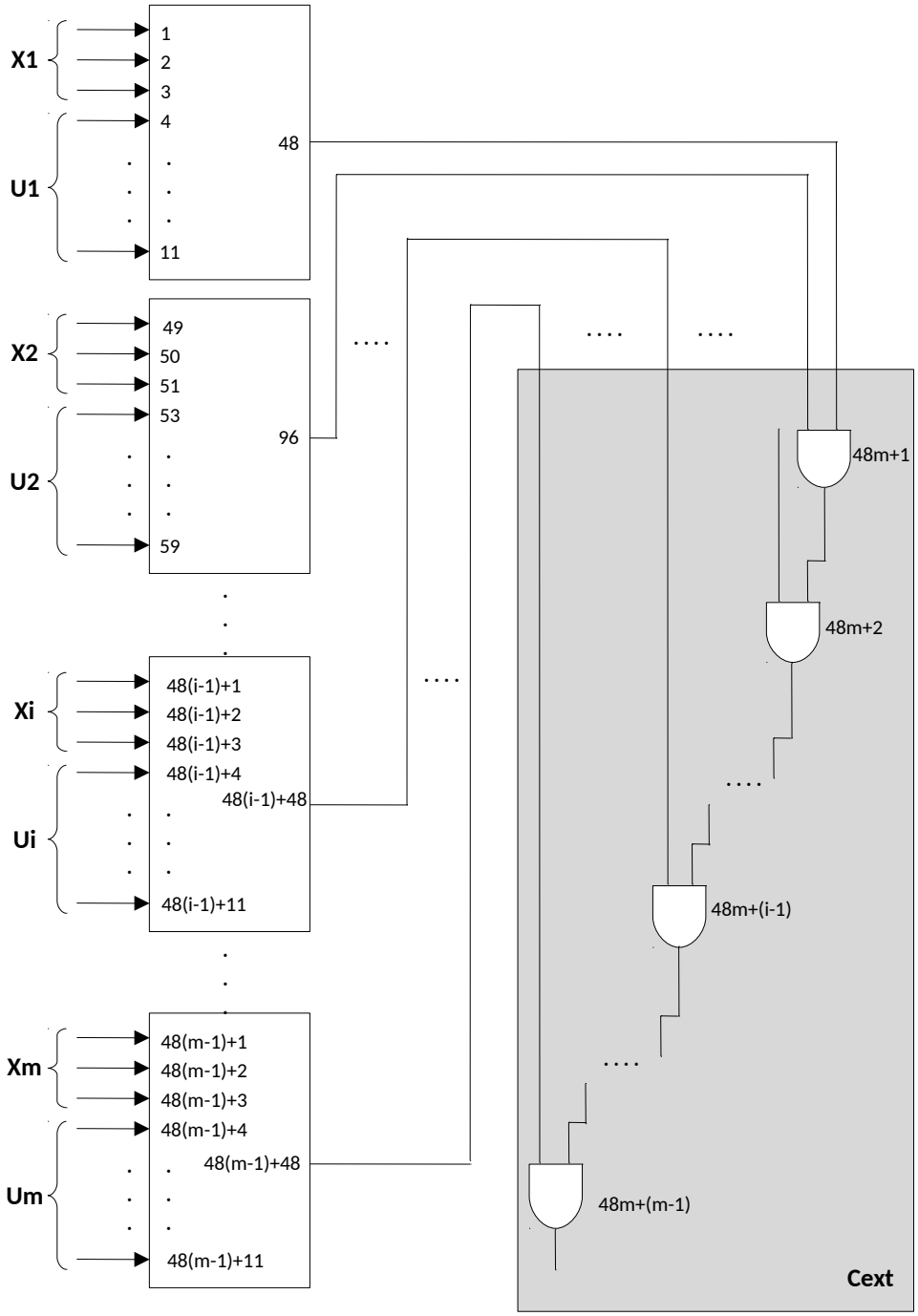
Uwaga 3.1: Należy wyraźnie zaznaczyć, że na określenie postaci wektorów U ma wprost wpływ jedynie układ negatywnych literałów w zapisie klauzuli i nie ma wpływu to z jaką kombinacją trzech zmiennych mamy do czynienia. Mogą to być dowolne kombinacje trzech zmiennych dopuszczane przez formułę *(1)* definiującą **3SAT** *CNF*.

Uwaga 3.2: Jeśli na wszystkie wejścia stałych będą podane sygnały *true* (u_i=*1),* to rozpatrywany schemat multipleksera realizuje funkcję ***TRUE***. Z kolei, jeśli na wszystkie wejścia stałych będą podane sygnały *false* (u_i=*0)*, to multiplekser realizuje funkcję ***FALSE***.

Wobec spostrzeżeń poczynionych w *uwagach 3.1 i 3.2* przyjmiemy, że sieć realizująca funkcję ***TRUE*** dla trzech zmiennych, będzie podstawą konstruowania sieci logicznej całego **3SAT** *CNF*. Ponadto, zapisując wyrażenie *(2)* w postaci operatorowej, stosując operator prefiksowy AND, uzyskamy postać jednoznacznie określającą sposób konstruowania sieci rozszerzającej C_{ext}.

$$\mathbf{3SAT}\ CNF = AND(AND(\ \ldots\ AND(AND(f_1^3, f_2^3), f_3^3)\ldots\ , f_{m-1}^3), f_m^3)$$

W efekcie, konstrukcja sieci **3SAT** *CNF* sprowadzi się do *m-krotnego* powielenia sieci elementarnego **3SAT** reprezentowanego przez sieć realizującą funkcję ***TRUE*** i ustalenia stałej *false* na stosownej pozycji wektora U_i *(i=1, 2, … , m)*.



Rys. 1.4 Schemat ideowy sieci logicznej *3SAT CNF*

Zakładając identyczny sposób numerowania bramek wejść adresowych i wejść danych w każdym reprezentującym elementarny **3SAT** segmencie sieci oraz wykorzystując oznaczenie wielobiegunnika (rys. 1.3), otrzymaną sieć można przedstawić w postaci schematu ideowego (rys. 1.4).

Rozmiar tak skonstruowanej sieci względem *m* ilości klauzul w zapisie **3SAT** *CNF* jest liniowy i wynosi *(49·m-1)* bramek.

Dla określenia rozmiaru sieci względem *n* ilości zmiennych, trzeba uwzględnić przypadek **3SAT** *CNF*, w którym mogą występie klauzule wszystkich kombinacji po *3* z *n* zmiennych. Takich kombinacji jest *(n-2)·(n-1)·n/6*, a więc ich ilość wyraża się wielomianem trzeciego stopnia.

Uwzględniając fakt, że dla każdej kombinacji po *3* z *n* zmiennych istnieje

8 różnych klauzul, więc w najgorszym przypadku ilość klauzul w formule **3SAT** *CNF* będzie wynosić co najwyżej *4/3·n· (n-1)·(n-2)*.

Uwaga 3.3: W istocie najgorszy przypadek można ograniczyć do sytuacji, w której w formule **3SAT** *CNF* występuje maksymalnie *7* klauzul tej samej kombinacji zmiennych. Możemy tak przyjąć, bo wystarczy by tylko dla jednej kombinacji trzech zmiennych w zapisie *CNF* wystąpiło *8* klauzul, to cała formuła nie jest spełnialna (*CNF* ośmiu klauzul definiuje funkcję stałą ***FALSE***).

Ostatecznie rozmiar sieci logicznej **3SAT** *CNF* skonstruowanej w oparciu o układ multipleksera 8/1 nie jest większy niż *49·4/3·n·(n-1) (n-2)* bramek.

Zatem sieć ma rozmiar wielomianowy. Dodatkowo jest to sieć, którą łatwo wykorzystać jako sieć jednostajną. W tym celu wystarczy na wszystkie wejścia adresowe multiplekserów X_i *(i=1 ,2, …, m)* podać stałe *true*. Pamiętamy, że na wejścia adresowe multiplekserów podawane są zmienne $x_r, x_s, x_t \in \{x_1, x_2, \dots, x_n\}$ dla $r \in \{1, 2, \dots n\text{-}2\}$, $s \in (r+1, r+2, \dots, n\text{-}1\}$ i $t \in \{s+1, s+2, \dots n\}$, tj. $r < s < t$, występujące w poszczególnych klauzulach, zaś podanie wartościowań x_i = *true* jest wymogiem definicji sieci jednostajnej. Dla wektora wartościowań zmiennych $[x_1, x_2, \dots, x_n]$ określającego inny dowolny sposób wartościowania tych zmiennych, sieć pozostaje siecią rozstrzygającą.

Pozostaje uzasadnić, że konstrukcję sieci można wykonać w pamięci *log n*. Proces powielania układu elementarnego **3SAT** (multiplekser 8/1) wymaga jednej zmiennej do pamiętania *m* ilości klauzul i jednego licznika do indeksacji kolejnych sieci elementarnych **3SAT**

oraz ich wejść. Potrzebny jest także licznik do obliczania numeru wejścia stałych, na które należy podać stałą *false*. Połączenia odpowiednich bramek wymagają tylko bezpośrednich operacji na indeksach i stąd są łatwe do wykonania w pamięci logarytmicznej.

1.4 Podsumowanie

Sformułowana we wstępie teza o istnieniu jednostajnych sieci wielomianowych dla problemu **3SAT** *CNF* prowadzi wprost do stwierdzenia równości klas problemów ***P*** i ***NP***.

Wykazanie prawdziwości stawianej tezy stanowi zasadniczą część prezentowanej pracy. Pokazano, że wykorzystując dualizm funkcji logicznych, możliwe jest traktowanie koniunkcji klauzul występujących w zapisie **3SAT** *CNF* jako koniunkcji elementarnych funkcji trzech zmiennych. Pokazano, że wykorzystując elementy teorii układów kombinacyjnych, możliwe jest zdefiniowanie uniwersalnej sieci logicznej dla dowolnej elementarnej funkcji logicznej trzech zmiennych. Taka uniwersalna sieć charakteryzuje się stałym rozmiarem wyrażonym przez *48* bramek logicznych AND, OR i NOT. W szczególności dotyczy to każdej z *8* możliwych postaci klauzul trzech zmiennych. Pokazano, że rolę takiej uniwersalnej sieci logicznej może spełniać sieć skonstruowana w oparciu o układ multipleksera 8/1.

Ostatecznie pokazano, że sieć logiczna pełnego **3SAT** *CNF*, w najgorszym przypadku charakteryzuje się rozmiarem wyrażanym przez $O(n^3)$. Ponadto pokazano, że taką sieć można skonstruować w pamięci logarytmicznej oraz, że może to być sieć jednostajna.

Wybór problemu **3SAT** dla pokazania istnienia jednostajnej sieci wielomianowej nie jest przypadkowy[1]. Przede wszystkim problem spełnialności ***SAT*** jest problemem pierwotnym dla rozważań w kategoriach *ZUPEŁPOŚCI*. Wszystkie wyniki w odniesieniu do niego, dotyczą więc zarówno do problemów ***NP*** - *zupełnych* (problem **3SAT**) jak również problemów ***P*** - *zupełnych* (problem **2SAT**) Poza tym **3SAT** został wybrany dlatego, by w ocenie i interpretacji wyniku pracy uniknąć ***NP*** – *trudności*, podobnej do tej z jaką mieliśmy przy ocenie i weryfikacji rozwiązania problemu **KNAPSACK**.

1 W kolejnych pracach przygotowywanych do publikacji pokazano konstrukcję jednostajnej sieci wielomianowej dla problemu **MULT**(n)

Poprawności opisanej i dyskutowanej konstrukcji sieci (i całej rodziny sieci) nie można podważyć, chyba że: - po pierwsze zakwestionowana zostanie definicja rodziny sieci jednostajnej, w tym wymóg by każda sieć z rodziny sieci była siecią skonstruowaną w pamięci *log n* i by była rozstrzygającą dla słów wejściowych, takich że $x_i = 1$ dla każdego $i= 1, \ldots, n$, gdzie *n* jest tak jak w [3, s.285][2] długością słowa określającego sposób wartościowania zmiennych występujących w formule **3SAT** odwzorowywanej przez sieć logiczną, a nie tak jak w [3, s.43][3] długością słowa opisującego problem w kategoriach maszyny Turinga; - po drugie zakwestionowane zostaną równoważność modelu maszyny Turinga z modelem sieci logicznych i wskazany zostanie błąd w przywołanym we wstępie twierdzeniu wiążącym złożoność sieci logicznych ze złożonością czasową [1, s. 401-405],

W istocie opisana konstrukcja sieci logicznej C_n jest niczym innym jak redukcją wielomianową formuły **3SAT** *CNF* $\varphi (x_1, x_2, \ldots , x_n)$ do problemu **CIRCUIT VALUE**, realizowaną podobnie jak pokazana w [3, s. 184-186] redukcja dowolnego języka $L \in P$ do **CIRCUIT VALUE**, sprowadzająca się do powielania identycznych elementarnych sieci i następnie łączenia odpowiednich bramek wejściowych i wyjściowych.

Pomijając to, że strukturalnie sieć wielomianowa dla **3SAT** nie różni się od sieci dla **2SAT** (w sieci **2SAT**, miejsce multiplekserów 8/1 w sieci **3SAT** zajmują multipleksery 4/1), warto zwrócić uwagę na jeszcze jeden aspekt faktu stwierdzenia możliwości skonstruowania jednostajnej sieci wielomianowej dla **3SAT**.

Otóż, to że dla **3SAT** można skonstruować sieć logiczną o wielomianowej ilości bramek, otwiera drogę do „odważnych" prób skonstruowania sprzętowej implementacji algorytmu rozwiązania **2SAT** i **3SAT** oraz do „jeszcze bardziej odważnej" próby opracowania algorytmów w polilogarytmicznym czasie równoległym przy całkowitej pracy wielomianowej. Uwieńczenie tych „jeszcze bardziej odważnych" prób sukcesem rozstrzygałoby jeszcze jedną interesującą kwestię ***NC*** *(klasa Nick'a) versus* ***P***.

---

2 „ … wiemy, że sieć logiczna o *n* zmiennych wejściowych może obliczać dowolną funkcję logiczną *n* zmiennych. Równoważnie możemy myśleć, że sieć akceptuje pewne słowa długości *n* z $\{ 0, 1 \}^*$ i odrzuca pozostałe. W tej sytuacji słowa $x = x_1 \ldots x_n \in \{ 0, 1 \}^*$ traktujemy jako wartościowanie zmiennych wejściowych sieci, … „

3 „By rozwiązać taki problem używając maszyny Turinga, musimy najpierw zdecydować, w jaki sposób będziemy zapisywać (reprezentować) przykład za pomocą słowa."

Pokazanie istnienia sieci jednostajnej dla **3SAT** *CNF* jest równoważne dla stwierdzenia, że dla **3SAT** *CNF* istnieje algorytm wielomianowy. Skonstruowanie takiego algorytmu, niezależnie od jego praktycznego znaczenia, może być potraktowane jako element weryfikacji przedstawionym wyników.

BIBLIOGRAFIA

[1] Sipser Michael, Wprowadzenie do teorii obliczeń, Warszawa, WNT, 2009

[2] Traczyk Wiesław, Układy cyfrowe. Podstawy teoretyczne i metody syntezy, Warszawa, WNT, 1986

[3] Papadimitriou Christos, Złożoność obliczeniowa, Warszawa, WNT, 2007