**Imię i nazwisko:** Jakub Pranica **Grupa:** Informatyka -13i **Data:** 24-04-2017 **Przedmiot:** Metody Programowania

**Wykładowca:** dr inż. Zbigniew Kokosiński **Semestr:** 2

**Temat:** Wykorzystując algorytm GSAT oraz WalkSAT napisać program rozwiązujący problem spełnialności dla wprowadzonej formuły logicznej.

**Problem spełnialności** – zagadnienie rachunku zdań, określające czy dla danej formuły logicznej istnieje takie przyporządkowanie zmiennych zdaniowych, żeby formuła była prawdziwa.

**Koniunkcyjna postać normalna** (ang. conjunctive normal form, **CNF**) danej formuły logicznej to równoważna jej formuła zapisana w postaci koniunkcji klauzul (zbioru formuł logicznych).

**Opis programu**: Program generuje formułę logiczną o podanych właściwościach lub prosi użytkownika o wpisanie jej   
z klawiatury w postaci CNF. Następnie prosi o wskazanie metody, którą ma użyć do znalezienia uporządkowania zmiennych boolowskich spełniającego formułę (dostępne metody: 1.Prosty WalkSAT 2.Ulepszony WalkSAT 3.GSAT).

**GSAT** oraz **WalkSAT** są algorytmami wyszukiwania lokalnego do rozwiązywania problemu spełnialności formuł logicznych. Oba algorytmy pracują na koniunkcyjnej postaci normalnej wprowadzonej formuły logicznej, której przypisują losowe wartości jej zmiennych. Jeżeli uzyskane przyporządkowanie zmiennych daje wartość "1" dla wszystkich implicentów formuły, algorytm kończy działanie. W przeciwnym wypadku jedna ze zmiennych zostaje zanegowana, co powtarzane jest do momentu uzyskania odpowiedniego przyporządkowania. WalkSAT i GSAT różnią się od siebie   
w metodzie wyboru zmiennej do zanegowania. WalkSAT najpierw wybiera losowy implicent o wartości "0", a następnie tę wartość w nim, której negacja skutkuje "zepsuciem" najmniejszej ilości gotowych już implicentów, z pewnym prawdopodobieństwem wybrania jednej ze zmiennych w sposób losowy (poprzez "zepsucie" rozumiemy zamianę wartości implicentu z "1" na "0"). Wybranie losowe (co jakiś czas), daje szansę algorytmowi na naprawienie obecnego niewłaściwego przyporządkowania i wyjście z "błędnej pętli". Uproszczona wersja WalkSAT zawsze wybiera zmienną   
w implicencie w sposób losowy. GSAT wybiera natomiast tę zmienną, której negacja minimalizuje liczbę implicentów   
o wartości "0".

Jeżeli program uzyska rozwiązanie w żądanym czasie, wypisuje je na ekran oraz zapisuje do pliku. W przypadku przekroczenia maksymalnego czasu przeznaczonego na wyszukiwanie algorytmy mogą wystartować ponownie   
z nowym losowym przyporządkowaniem.

**Napotkane problemy: 1.**Zapis i reprezentacja formuł logicznych w postaci CNF w języku programowania C++ **2.**Wygenerowanie odpowiedniej formuły logicznej oraz przypisanie do niej zmiennych zgodnie z podanymi przez użytkownika właściwościami. **3.**Wylosowanie lub wybór zmiennej, której negacja będzie optymalna w danym kroku**. 4.**Realizacja algorytmu WalkSAT oraz GSAT i wypisanie na ekran lub do pliku uzyskanego przyporządkowania wartości zmiennych spełniającego formułę. **5.**Wprowadzenie możliwości ograniczenia czasu wyszukiwania.

**Przykładowy wynik programu:** Program rozwiązuje problem spełnialności dla wprowadzonej formuły logicznej.

**(Przykład 1)** Wybierz algorytm:

1.**WalkSAT** - wybiera losowy implicent o wartości "0", a następnie neguje losowa w nim zmienną.

2.**GSAT** - wybiera te zmienna, której negacja minimalizuje liczbę implicentów o wartości "0".

3.**Uruchom i porównaj oba - WalkSAT oraz GSAT.**

**3 (wybrano porównanie)** Czy zapisać wyniki do pliku? 1 - tak 2 - nie **1**

Skąd wziąć dane? 1-wygeneruj losowe 2-pozwol mi podąć **1**

Podaj ilość implicentów: **10** Podaj ilość dostępnych przy generacji zmiennych: **6**

Podaj maksymalna ilość zmiennych w jednym implicencie: **3**

Podaj maksymalny czas wyszukiwania (w sekundach): **60**

Objaśnienie symboli: **~** oznacza negacje, **|** oznacza operator logiczny OR, **&** oznacza operator logiczny AND.

**Wygenerowana funkcja (formuła):**  f(A,B,C,D,E,F)=(~F) & (~A|E) & (~C|~B) & (D|C) & (~A|D|E) & (E|A) & (~A|~C|~B) & (~C|~B) & (A|~B) & (C|E) & (~D|~E|~C) & (~F|~E) Aby kontynuować, naciśnij dowolny klawisz . . .

**Dla lepszego porównania, każdy z algorytmów rozpoczynam ze zmiennymi ustawionymi na 0:**

A=0; B=0; C=0; D=0; E=0; F=0;

Program sam decyduje, której wersji algorytmu WalkSAT użyć, gdyż dla niedużych przykładów wersja prosta jest szybsza od "ulepszonej".

Wynik dla **WalkSAT**: (znaleziony w 19 krokach)

A=1; B=0; C=0; D=1; E=1; F=0;

Wynik dla **GSAT**: (znaleziony w 21 krokach)

A=0; B=0; C=0; D=1; E=1; F=0

**(Przykład 2)** Ilość implicentów: **15** Ilość dostępnych zmiennych: **6** Maksymalna ilość zmiennych w jednym implicencie**: 4**

**Wygenerowana funkcja (formuła):** f(A,B,C,D,E,F)=(~D) & (E|~C) & (E|A) & (C|~E|A) & (~B|~F) & (F|~B|~D) & (F|B|~C|~D) & (~B|A|~F) & (E|F|~D|B) & (~A|~D|~B|E) & (A|~D) & (~F|C|~A) & (D|C) & (F|~D|A) & (~E|~A)

Wynik dla **WalkSAT**: (znaleziony w 33 krokach)

A=0; B=0; C=1; D=0; E=1; F=1;

Wynik dla **GSAT**: (znaleziony w 40 krokach)

A=0; B=1; C=1; D=0; E=1; F=0;

**(Przykład 3)** Wybierz algorytm: **2 (wybrano GSAT)** Czy zapisać wyniki do pliku? 1 - tak 2 - nie **1**

Skąd wziąć dane? 1-wygeneruj losowe 2-pozwol mi podąć **2**

Podaj ilość implicentów: **9** Podaj maksymalny czas wyszukiwania (w sekundach): **60**

Podaj 1 implicent: **~A|~E**

Podaj 2 implicent: **~C|~F**

Podaj 3 implicent: **B|E**

Podaj 4 implicent: **D**

Podaj 5 implicent: **~C|E**

Podaj 6 implicent: **F|~E**

Podaj 7 implicent: **~C|D**

Podaj 8 implicent: **~B|G**

Podaj 9 implicent: **~E|C**

**Wczytana funkcja (formuła):** f(A,B,C,D,E,F,G)=(~A|~E) & (~C|~F) & (B|E) & (D) & (~C|E) & (F|~E) & (~C|D) & (~B|G) & (~E|C) Aby kontynuować, naciśnij dowolny klawisz . . .

**Losowo przypisane początkowe wartości:** A=0; B=0; C=0; D=0; E=1; F=1; G=1;

**Neguję zmienną D:** A=0; B=0; C=0; D=1; E=1; F=1; G=1;

**Neguję zmienną G:** A=0; B=0; C=0; D=1; E=1; F=1; G=0;

**Neguję zmienną G:** A=0; B=0; C=0; D=1; E=1; F=1; G=1;

**Neguję zmienną B:** A=0; B=1; C=0; D=1; E=1; F=1; G=1;

**Neguję zmienną E:** A=0; B=1; C=0; D=1; E=0; F=1; G=1;

Koniec. **Uzyskane spełniające formułę przyporządkowanie:** A=0; B=1; C=0; D=1; E=0; F=1; G=1;

**Czas wykonania:** 0.014000 (sekundy)

**Pseudokod:**

**Wypisz na ekran:** "Program rozwiązuje problem spełnialności dla wprowadzonej formuły logicznej. \n"

**Wypisz na ekran:** "Skąd wziąć dane? 1-wygeneruj losowe 2-pozwol mi podąć \n"

**Jeżeli użytkownik wpisał 2, wygeneruj potrzebne dane - w przeciwnym wypadku poproś go o wpisanie formuły logicznej w postaci CNF.**

**Wypisz na ekran:** "Podaj maksymalny czas wyszukiwania (w sekundach):\n"

**Zapisz podany czas w zmiennej maxtime.**

**Wypisz na ekran:** "Wybierz algorytm: (1.Prosty WalkSAT 2.Ulepszony WalkSAT 3.GSAT) \n"

**Zapamiętaj wybrany algorytm.**

**Przypisz losowe wartości zmiennym boolowskim (we wprowadzonej formule logicznej).**

**Utwórz funkcję** sprawdzWszystkie()**, która zwraca true, gdy wszystkie implicenty mają wartość "1".**

**Dopóki** nie wszystkie implicenty mają wartość "1"**, wykonuj w pętli:** while( !sprawdzWszystkie() )

**Jeżeli wybrano prosty WalkSAT:**

**Wybierz losowy implicent o wartości "0", a następnie losową w nim zmienną do zanegowania.**

**Jeżeli wybrano ulepszony WalkSAT:**

**Wybierz losowy implicent o wartości "0", a następnie tę zmienna w nim, która zamieni najmniej "gotowych" implicentów na wartość "0".**

**Jeżeli wybrano GSAT:**

**Wybierz tę zmienną, której negacja minimalizuje liczbę implicentów o wartości “0”.**

**Zaneguj wybraną zmienną (zmień jej wartość na przeciwną:**

int war = zmienne[indeks];

if(war == 1) zmienne[indeks] = 0; else zmienne[indeks] = 1;

**Jeżeli maksymalny czas wyszukiwania został przekroczony** if(aktualnyCzas>maxtime**)**

**,to przerwij wyszukiwanie i rozpocznij ponownie z innym przyporządkowaniem początkowym.**

**w przeciwnym wypadku kontynuuj wykonywanie pętli warunkowej.**

//Gdy pętla zakończyła się, aktualne przyporządkowanie zmiennych jest rozwiązaniem problemu spełnialności dla wprowadzonej formuły logicznej.

**Wypisz na ekran i zapisz do pliku uzyskane przyporządkowanie.**

**Wnioski:** Na podstawie analizy wyników zwracanych przez program, można stwierdzić, iż program działa i prawidłowo posługuje się algorytmem WalkSAT oraz GSAT, w celu znalezienia optymalnego przyporządkowania zmiennych boolowskich stanowiącego rozwiązanie problemu spełnialności dla wprowadzonej przez użytkownika formuły logicznej. GSAT i WalkSAT są reprezentantami algorytmów wyszukiwania lokalnego - polegają one na niewielkich modyfikacjach rozwiązania początkowego (np. losowego), ocenie wszystkich uzyskanych w ten sposób rozwiązań pochodnych i wyborze najlepszego z nich jako nowego rozwiązania początkowego.

**Literatura:**

* *Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein*: **Wprowadzenie do algorytmów.** Warszawa: Wydawnictwa Naukowo-Techniczne, 2003.
* *Zbigniew Kokosiński* **wykład i prezentacja nr. 8** Metody programowania, Politechnika Krakowska 2017.
* **pl.wikipedia.org/wiki** **/Koniunkcyjna\_postać\_normalna** oraz **pl.wikipedia.org/wiki/Problem\_spełnialności**
* **en.wikipedia.org/wiki/WalkSAT**