

**Rozwiązywanie równań i układów równań
nad skończonymi strukturami algebraicznymi
z wykorzystaniem SAT-solvera.**

Patryk Popek

Praca Licencjacka

Uniwersytet Marie Curie Skłodowskiej w Lublinie
Instytut Informatyki
2023

Spis treści

1	Wstęp	2
2	Opis Teoretyczny Problemu	3
2.1	Pojęcia Matematyczne	3
2.2	Pojęcia Logiczne	4
2.3	SAT-Solver	5
3	Aplikacja i algorytmy w niej zastosowane	6
3.1	Wczytanie algebry do programu	6
3.1.1	Pliki .ua	6
3.1.2	Algorytm w programie	7

1 Wstęp

Matematyka jest codzienną częścią naszego życia. Jesteśmy związani z nią nawet o tym nie wiedząc. Mijamy ją w sklepach, w różnego rodzaju statystykach widzianych w internecie, czy niektórzy pracują z nią zawodowo.

Matematyka jest królową nauk, bez której nie mogłaby powstać logika oraz fizyka, co za tym idzie. nie mogłaby też powstać informatyka. Cała informatyka, nie licząc kilku praw fizycznych spowodowanych skończonością naszego świata, jest tak naprawdę matematyką zapisaną w systemie binarnym. Idąc dalej tym tokiem, logika i informatyka są pochodną matematyki.

Celem tej pracy jest napisanie aplikacji, która rzecz czysto matematyczną, czyli równanie matematyczne zapisane w postaci funkcji będzie potrafiła rozwiązać za pomocą narzędzi logicznych takich w środowisku informatycznym

2 Opis Teoretyczny Problemu

Poniżej przedstawione są pojęcia związane z pracą.

2.1 Pojęcia Matematyczne

1. **Struktura algebraiczna:** Jest to trójka (A, B, C) , gdzie A jest dziedziną algebry, B jest dozwolonymi operacjami na zbiorze oraz C jest opisanymi relacjami elementów dziedziny względem siebie nawzajem.
2. **Grupa:** Jest to zbiór G posiadający działanie \circ , które posiada następujące własności:
 - a) Wewnętrzność: Dla dowolnych $a, b \in G$ działanie $a \circ b \in G$.
 - b) Łączność: Dla dowolnych $a, b, c \in G$ zachodzi $(a \circ b) \circ c = a \circ (b \circ c)$.
 - c) Element neutralny: Istnieje $e \in G$ takie, że dla każdego $a \in G$ $a \circ e = a$.
 - d) Odwracalność: Dla każdego $a \in G$ Istnieje $x \in G$ takie, że $a \circ x = e$.
 - e) Jeżeli dla dowolnego $a, b \in G$ zachodzi $a \circ b = b \circ a$ wtedy grupa jest grupą abelową.

Przykładową grupą abelową zgodną z powyższą definicją jest grupa $G(2, +)$ czyli XOR.

3. **Funkcja:** Jest to operacja przekształcająca zbiór X w zbiór Y . Innymi słowami, każdemu elementowi $x \in X$ przypisuje dokładnie jeden element $y \in Y$. Zbiór X nazywamy dziedziną funkcji, a zbiór Y przeciwdziedziną. Najpopularniejszą formą przedstawienia funkcji jest podanie jej wzoru na przykład $f_x = x + 2$
4. **Arność:** Jest to liczba argumentów przyjmowana przez funkcję. Jeżeli arność wynosi 0 wtedy funkcja jest funkcją stałą
5. **Liczność(eng. Cardinality):** Jest to liczba elementów znajdujących się w zbiorze. Na potrzeby tej pracy określa dziedzinę wczytanej do aplikacji struktury .
6. **Punkt przecięcia funkcji:** Mamy dwie funkcje F_1 oraz F_2 . Punkt przecięcia oznacza miejsce, w którym $F_1 = F_2$. Posiadając wzory powyższych funkcji możemy ułożyć równanie, za pomocą którego umiemy wyliczyć wartość powyższego punktu.

7. **Równanie:** Jest to wyrażenie postaci $t_1 = t_2$, gdzie t_1, t_2 są dowolnymi wyrażeniami algebraicznymi zgodnymi z używaną algebrą. Równania dzielimy na 3 typy:
 - a) Sprzeczne: Równanie nie posiada rozwiązania, czyli za zmienną równania nie da się podstawić żadnej wartości z dziedziny algebry.
 - b) Oznaczone: Równanie posiada rozwiązania, czyli za zmienną równania da się podstawić co najmniej jedną wartość z dziedziny algebry.
 - c) Tożsamy: Równanie jest zawsze rozwiązywalne bez względu na to jaką wartość z dziedziny algebry podstawimy za zmienną równania.
8. **Notacja Polska:** Sposób zapisu wyrażeń, w którym operator/nazwa funkcji znajdują się przed argumentami. Na przykład $+xy - ab$ jest równoważne zapisowi $(x + y)/(a + b)$

2.2 Pojęcia Logiczne

Poniższe definicje odnoszą się do logiki zerowego rzędu.

1. **Zmienna zdaniowa:** jest to najmniejszy możliwy element w logice. Przyporządkowuje się jej wartość prawda(1) albo fałsz(0).
2. **Negacja:** Jest to zaprzeczenie zmiennej zdaniowej $x = 1 \rightarrow \neg x = 0$.
3. **Literal:** Jest to pojedyncze wystąpienie zmiennej $x \rightarrow \neg x$.
4. **Spełnialność funkcji:** Funkcja logiczna jest spełnialna gdy istnieje wartościowanie, które zwraca wartość logiczną funkcji 1.
5. **Prawdziwość funkcji:** Funkcja logiczna jest spełnialna przy każdym wartościowaniu, czyli zawsze zwraca wartość logiczną funkcji 1.
6. **Koniunkcja:** Funkcja logiczna postaci $x \wedge y$, zwraca 1 tylko i wyłącznie wtedy gdy zarówno $x, y = 1$.
7. **Alternatywa:** Funkcja logiczna postaci $x \vee y$, zwraca 0 tylko i wyłącznie wtedy gdy zarówno $x, y = 0$.
8. **Implikacja:** Funkcja logiczna postaci $x \rightarrow y$, zwraca 0 tylko i wyłącznie wtedy gdy $x = 1, y = 0$.
9. **Koniunkcyjna postać normalna (CNF):** Funkcja logiczna, która jest koniunkcją alternatyw.

10. **Dysjunkcyjna postać normalna (DNF):** Funkcja logiczna, która jest alternatywą koniunkcji.
11. **Klauzula Horna:** Jest to alternatywa posiadająca co najwyżej jeden literał niezanegowany.

2.3 SAT-Solver

3 Aplikacja i algorytmy w niej zastosowane

Aplikacja, którą napisałem na potrzeby tej pracy podzielona została na kilka kroków. Etapy te zostały ułożone liniowo i każdy kolejny można wykonać dopiero po ukończeniu wszystkich wcześniejszych. Wszystkie etapy są reprezentowane w interfejsie aplikacji za pomocą przycisków.

3.1 Wczytanie algebry do programu

3.1.1 Pliki .ua

Algebry, które są wykorzystywane w mojej aplikacji są reprezentowane poprzez pliki z rozszerzeniem "ua". Skrót ten pochodzi od wyrażenia "universal algebra". Są to pliki w formacie XML.

```
<?xml version='1.0' encoding='utf-8'?>
<algebra>
  <basicAlgebra>
    <algName>P4</algName>
    <cardinality>4</cardinality>
    <operations>
      <op>
        <opSymbol>
          <opName>d</opName>
          <arity>2</arity>
        </opSymbol>
        <opTable>
          <intArray>
            <row r="[0]">0,1,2,3</row>
            <row r="[1]">1,0,3,2</row>
            <row r="[2]">2,3,0,1</row>
            <row r="[3]">3,2,1,0</row>
          </intArray>
        </opTable>
      </op>
      <op>
        <opSymbol>
          <opName>m</opName>
          <arity>2</arity>
        </opSymbol>
        <opTable>
          <intArray>
            <row r="[0]">0,0,0,0</row>
            <row r="[1]">0,1,2,3</row>
            <row r="[2]">0,2,0,2</row>
            <row r="[3]">0,3,2,1</row>
          </intArray>
        </opTable>
      </op>
    </operations>
  </basicAlgebra>
</algebra>
```

Rysunek 1: Przykładowy plik reprezentujący algebrę uniwersalną

Powyższe zdjęcie przedstawia wygląd pliku opisującego algebrę. Sekcjami interesującymi nas najbardziej w tym pliku są *cardinality* oraz *operations*. *Cardinality* informuje o dziedzinie całej struktury. Jest ona niezmienna i ograniczona do zakresu $x \in \{0, 1, \dots, \textit{cardinality} - 1\} \wedge x \in \mathbb{N}$. *Operations* opisuje wszystkie dostępne operacje na strukturze. Sekcja ta jest podzielona na podsekcje *op* poświęcone poszczególnym działaniom. *OpName* informuje o nazwie działania, jest to identyfikator, po którym będzie można znaleźć działanie w programie. *Arity* określa arność funkcji reprezentowanej przez to działanie. Najważniejszą częścią opisu działania jest *opTable*. Jest to struktura przedstawiająca wyniki działania opisywanej funkcji na wszystkich możliwych wariacjach z powtórzeniami o długości *arity* dziedziny algebry. Tabele te są główną osią, dzięki której powstaje finalna formuła CNF używana w *Sat-Solverze*. Algebra pokazana stronek wyżej posiada dwa działania: Działanie *d*, które jest dwuargumentowym dodawaniem modulo 4 oraz działanie *m*, które jest dwuargumentowym mnożeniem modulo 4

3.1.2 Algorytm w programie