**Курсовая работа**

**Использование бинарных решающих диаграмм для решения логических задач. Библиотека BuDDy**

по дисциплине «Математическая логика»

Выполнил

студент гр. 3530904/80104 А. А. Квашнин

Проверила: И. В. Шошмина

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2020 г.

Оглавление

[*Введение* 3](#_Toc56548965)

[Определение проблемной области 3](#_Toc56548966)

[Цель работы 3](#_Toc56548967)

[Задача работы 3](#_Toc56548968)

[*Описание предметной области* 4](#_Toc56548969)

[Представления булевых функций с помощью BDD 4](#_Toc56548970)

[Программная реализация BDD 6](#_Toc56548971)

[Описание библиотеки BuDDy 6](#_Toc56548972)

[*Ход выполнения работы* 8](#_Toc56548973)

[*Физическая интерпретация задачи* 11](#_Toc56548974)

[*Заключение* 13](#_Toc56548975)

[*Литература* 13](#_Toc56548976)

# Введение

## Определение проблемной области

Одной из наиболее распространённых проблем в современном программировании является проблема удовлетворения ограничениям (Constraint Satisfaction Problem, CSP), которая на практике решается методами программирования в ограничениях (constraint programming) [4].

Изначально проблема удовлетворения ограничениям позиционировалась как одна из задач, решаемых при создании искусственного интеллекта, разработка которого, в свою очередь, является востребованной во многих областях науки и техники и сама превратилась в самостоятельную науку[3].

Частным примером задачи удовлетворения условий является задача выполнимости (SAT), суть которой заключается в выполнимости произвольной булевой функции. Под выполнимой функцией понимается такая бинарная функция, которая при подстановке определённых значений булевых переменных возвращает значение True.

Данная курсовая работа рассматривает задачу удовлетворения ограничениям, которая затем сводится к проблеме выполнимости. В качестве математического аппарата используются специальные представления булевых функций – бинарные решающие диаграммы (Binary Decision Diagram, BDD).

## Цель работы

Целью работы является ознакомление студента с библиотекой BuDDy, программно реализующей BDD, а также решение «задачи Эйнштейна» - классического примера проблемы удовлетворения ограничениям.

Задача Эйнштейна состоит в сопоставлении N различным объектам N различных значений M свойств.

## Задача работы

Задача – реализовать программу, решающую задачу Эйнштейна, с помощью средств библиотеки BuDDy.

# Описание предметной области

## Представления булевых функций с помощью BDD

Существует множество форм представления булевых функций, зависящих от конечного числа булевых переменных, принимающих значения логического нуля и единицы.

К основным формам относятся:

* Таблица истинности;
* Семантическое дерево;
* Дизъюнктивная нормальная форма;
* Конъюнктивная нормальная форма;
* Совершенная дизъюнктивная нормальная форма;
* Полином Жегалкина;
* Иные пропозициональные формулы общего типа.

Указанные выше стандартные формы представления бинарных функций являются неудобными при решении задачи выполнимости, если число переменных велико, а ведь именно с такими видами задач сталкиваются инженеры, например, при разработке искусственного интеллекта. Именно поэтому задачу выполнимости относят к классу NP-полных задач. Более того, задача SAT стала первой задачей, для которой была доказана её NP-полнота.

Облегчает решение подобных SAT задач при работе с бинарными функциями введение новой формы представления функций – бинарной решающей диаграммы.

BDD – ациклический орграф, в котором отсутствуют повторения в структуре, с одной корневой вершиной, двумя листьями, помеченными 0 и 1, и промежуточными вершинами. Каждая вершина помечается булевой переменной. Из вершин исходит два направленных ребра, которые имеют разное графическое представление [1].

Для определения значения бинарной функции при определённой интерпретации (наборе численных значений переменных) необходимо осуществлять спуск с корневой вершины до листьев. При прохождении очередной вершины мы выбираем ребро в зависимости от значения переменной (при 0 идём по пунктирному ребру, при 1 – по сплошному).

BDD может быть построена двумя способами [5]:

* путём изменения структуры семантического дерева;
* логическими операциями над BDD меньшего размера.

Первый вариант основан на том, что BDD по своей сути является семантическим деревом без избыточностей, и может быть реализован, если мы имеем дело с упорядоченной BDD (Ordered BDD, OBDD), где все переменные представлены в диаграмме от корня к листьям в одном порядке.

В этом случае используются операции редукции, основанные на следующих правилах:

*1. Если несколько вершин орграфа помечено одним именем, и они имеют изоморфные подструктуры, то граф можно редуцировать: оставить только одну из них.*

*2. Если обе выходные дуги промежуточной вершины v орграфа ведут в одну вершину, то эта вершина v может быть выброшена.*

При завершении алгоритма мы получаем редуцированную BDD (Reduced Ordered BDD, ROBDD), и именно этот вид бинарных решающих диаграмм является канонической формой представления булевых функций.

Булевы переменные также могут быть представлены в виде решающих диаграмм, и всё множество логических операций над ними также переносится в пространство BDD. Именно этот подход построения BDD реализован в рассматриваемой библиотеке BuDDy. Пример применения произвольной бинарной логической операции к двум BDD представлен на рисунке 1:

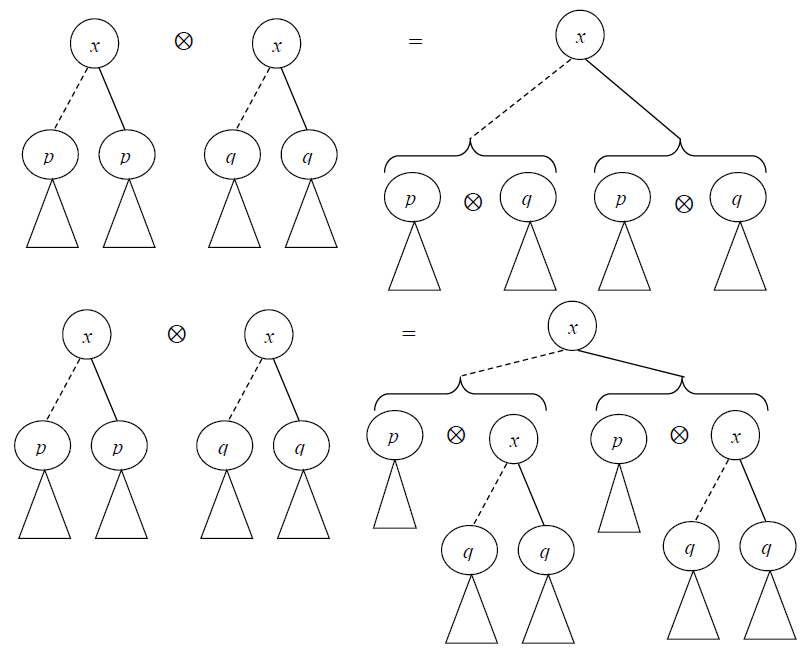


Рисунок . Логическая операция над BDD

## Программная реализация BDD

В памяти бинарная решающая диаграмма хранится в виде таблицы, описывающей все листья и вершины графа.

В строках такой таблицы хранятся номер узла, номер соответствующей ему булевой переменной и номера вершин, в которые входят рёбра, выходящие из данной вершины.

За листьями-константами закреплены номера 0 и 1.

## Описание библиотеки BuDDy

Центральным классом библиотеки является класс *bdd*, для которого перегружены следующие операторы:

* & (&=) – конъюнкция двух BDD (и соответственно с присваиванием);
* | (|=) – дизъюнкция;
* ^ (^=) – сложение по модулю 2;
* ! – отрицание;
* >> (>>=) – правосторонняя импликация;
* << (<<=) – левосторонняя импликация;
* – (– =) – следует понимать как разность характеризуемых множеств:
* a – b = a & !b, где a и b булевы функции;
* > – булева функция «больше»;
* < – булева функция «меньше»;
* == (!=) – функции сравнения двух BDD. Возвращают значения “истина” или “ложь” в виде целых 0 или 1;
* оператор потокового вывода << [1].

Оператор вывода может использоваться совместно с манипуляторами:

- *bddsat –* вывод интерпретаций, на которых булева функция истинна;

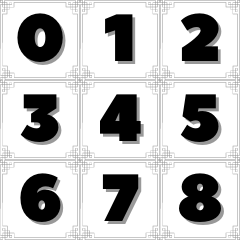
- *bddtable –* вывод значимых строк таблицы;

- *bddall –* вывод всех строк таблицы.

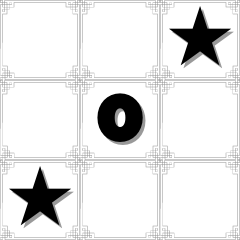
Также определены константы *bddtrue* и *bddfalse*, реализовано обращение к определённой переменной посредством функции *bdd\_ithvar* и *bdd\_nithvar*, а также подсчёт числа интерпретаций, на которых значение булевой функции – истина (это и есть реализация решения задачи SAT) с помощью функции *bdd\_satcount* [2].[[1]](#footnote-1)

**Постановка задачи**

Имеется *N = 9* объектов, расположенных следующим образом:



Соседские отношения, согласно варианту №6, задаются следующей схемой:



1. Задано M = 4 свойств, принимающих N различных значений. Задать n1 ограничений типа 1, n2 ограничений типа 2 по своему варианту. Для ограничений типа 3 и типа 4 использовать ‘’соседские’’ отношения, заданные вариантом. Придумать, как ограничения, подобные типу 3, 4, выражаются в соседских отношениях моего варианта, задать n3 ограничений подобных отношениям типа 3 и n4 ограничений подобных отношениям типа 4 в соответствии со своим заданием.
2. Необходимо самостоятельно описать дополнительный тип ограничения n7: “сумма свойств объектов-соседей не должна быть больше K”, где K – некоторое число от 0 до N\*M. K выбирается студентом.
3. Найти все возможные решения;
4. Придумать физическую интерпретацию задачи;
5. Если задача имеет не одно решение, следует добавить и/или изменить некоторые ограничения так, чтобы задача имела только одно единственное решение.
6. Если задача не имеет решений, следует удалить и/или изменить некоторые ограничения так, чтобы задача имела только одно единственное решение. Ограничение типа n7 удалять нельзя.

Для варианта №6 имеем:

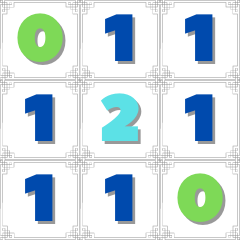
* 4 ограничения типа n1
* 5 ограничений типа n2
* 4 ограничений типа n3
* 2 ограничений типа n4

Требования к реализации (если вы сдаете на компьютере преподавателя):

* Для выполнения задания необходимо написать программу на языке C++ с использованием библиотеки BuDDy.
* Компилировать под MS Visual Studio.
* Реализация должна быть масштабируемой, т. е. изменение параметров N, M, K не должны приводить к изменению основной части программы.

# Ход выполнения работы

1. Соседские ограничения

Для всех элементов можно установить количество соседей, таким образом получаем:

* Только у элемента 4 имеется два соседа.
* Элементы 0 и 8 не имеют соседей.
* Элементы 1, 2, 3, 5, 6, 7 имеют одного соседа.

1. Описание ограничения

**1 типа:**

1. Свойство 0 у объекта 0 равно 0.
2. Свойство 0 у объекта 1 равно 1.
3. Свойство 3 у объекта 2 равно 1.
4. Свойство 2 у объекта 6 равно 7.

**2 типа:**

1. Если у объекта свойство 2 имеет значение 6, то у него свойство 1 имеет значение 5 и обратно.
2. Если у объекта свойство 0 имеет значение 3, то у него свойство 2 имеет значение 6 и обратно.
3. Если у объекта свойство 1 имеет значение 8, то у него свойство 2 имеет значение 4 и обратно.
4. Если у объекта свойство 2 имеет значение 7, то у него свойство 0 имеет значение 7 и обратно.
5. Если у объекта свойство 2 имеет значение 8, то у него свойство 0 имеет значение 2 и обратно.

**3 типа:**

1. Если у объекта свойство 0 имеет значение 5, то сверху от него стоит объект, у которого свойство 2 имеет значение 1, и обратная ситуация.
2. Если у объекта свойство 2 имеет значение 7, то сверху от него стоит объект, у которого свойство 3 имеет значение 2, и обратная ситуация.
3. Если у объекта свойство 1 имеет значение 5, то снизу от него стоит объект, у которого свойство 1 имеет значение 0, и обратная ситуация.
4. Если у объекта свойство 2 имеет значение 4, то снизу от него стоит объект, у которого свойство 0 имеет значение 8, и обратная ситуация.

**4 типа:**

1. Объект, у которого свойство 1 имеет значение 5, стоит рядом с объектом, у которого свойство 3 имеет значение 7.
2. Объект, у которого свойство 1 имеет значение 2, стоит рядом с объектом, у которого свойство 0 имеет значение 5.

**7 типа:**

1. Сумма свойств объектов-соседей не должна быть больше K=28.

Для однозначного решения пришлось ввести дополнительные ограничения:

**1 типа:**

1. Свойство 2 у объекта 4 равно 5.
2. Свойство 1 у объекта 8 равно 6.
3. Свойство 3 у объекта 5 равно 3.
4. Свойство 0 у объекта 5 равно 8.
5. Свойство 1 у объекта 0 равно 4.
6. Свойство 0 у объекта 3 равно 6.
7. Свойство 1 у объекта 6 равно 2.
8. Свойство 2 у объекта 2 равно 0.
9. Свойство 2 у объекта 5 равно 2.
10. Свойство 3 у объекта 6 равно 5.
11. Свойство 1 у объекта 1 равно 1.
12. Свойство 3 у объекта 7 равно 4.

**2 типа:**

1. Если у объекта свойство 2 имеет значение 1, то у него свойство 3 имеет значение 6 и обратно.
2. Если у объекта свойство 0 имеет значение 6, то у него свойство 3 имеет значение 8 и обратно.
3. Если у объекта свойство 0 имеет значение 4, то у него свойство 1 имеет значение 7 и обратно.

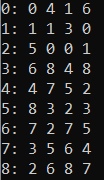
**3 типа:**

1. Если у объекта свойство 0 имеет значение 2, то сверху от него стоит объект, у которого свойство 1 имеет значение 6, и обратная ситуация.

**4 типа:**

1. Объект, у которого свойство 3 имеет значение 7, стоит рядом с объектом, у которого свойство 1 имеет значение 1.
2. Объект, у которого свойство 2 имеет значение 8, стоит рядом с объектом, у которого свойство 3 имеет значение 3.
3. Объект, у которого свойство 3 имеет значение 8, стоит рядом с объектом, у которого свойство 0 имеет значение 5.
4. Объект, у которого свойство 2 имеет значение 4, стоит рядом с объектом, у которого свойство 2 имеет значение 2.

Результат работы программы



Получено однозначное решение задачи.

# Физическая интерпретация задачи

9 человек садятся на стулья с номерами от 0 до 8. У случайных людей узнают некоторые факты, такие как любимый цвет, еда, музыка, жанр кино. Необходимо узнать все предпочтения каждого человека.

Зададим 4 вида свойств.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Свойство 0 (любимый цвет) | Свойство 1 (любимая еда) | Свойство 2 (любимая музыка) | Свойство 3 (любимый жанр кино) |
| 0 | Красный | Суши | Классическая | Боевик |
| 1 | Оранжевый | Пицца | Рок | Приключения |
| 2 | Жёлтый | Бургер | Металл | Триллер |
| 3 | Зелёный | Каша | Электронная | Драма |
| 4 | Голубой | Вафли | Рэп | Комедия |
| 5 | Синий | Печенье | Джаз | Фантастика |
| 6 | Фиолетовый | Фрукты | Поп | Фэнтэзи |
| 7 | Белый | Шаурма | Народная | Ужасы |
| 8 | Чёрный | Карри | Хип-Хоп | Детектив |

Нам известны следующие факты:

1. Любимый **цвет** у человека, сидящего на месте с номером 0 – **красный**.
2. Любимый **цвет** у человека, сидящего на месте с номером 1 – **оранжевый**.
3. Любимый **жанр** кино у человека, сидящего на месте с номером 2 – **приключения**.
4. Любимая **музыка** у человека, сидящего на месте с номером 6 – **народная**.
5. Любимая **музыка** у человека, сидящего на месте с номером 4 – **джаз**.
6. Любимая **еда** у человека, сидящего на месте с номером 8 – **фрукты**.
7. Любимый **жанр** кино у человека, сидящего на месте с номером 5 – **драма**.
8. Любимый **цвет** у человека, сидящего на месте с номером 5 – **чёрный**.
9. Любимая **еда** у человека, сидящего на месте с номером 0 – **вафли**.
10. Любимый **цвет** у человека, сидящего на месте с номером 3 – **фиолетовый**.
11. Любимая **еда** у человека, сидящего на месте с номером 6 – **бургер**.
12. Любимая **музыка** у человека, сидящего на месте с номером 2 – **классическая**.
13. Любимая **музыка** у человека, сидящего на месте с номером 5 – **металл**.
14. Любимый **жанр** кино у человека, сидящего на месте с номером 6 – **фантастика**.
15. Любимая **еда** у человека, сидящего на месте с номером 1 – **пицца**.
16. Любимый **жанр** кино у человека, сидящего на месте с номером 7 – **комедия**.
17. Если человек любит **поп музыку**, то **печенье** его любимая **еда**, и обратная ситуация.
18. Если человек любит **зелёный цвет**, то **поп** его любимая **музыка**, и обратная ситуация.
19. Если человек любит **карри**, то **рэп** его любимая **музыка**, и обратная ситуация.
20. Если человек любит **народную музыку**, то **белый** его любимый **цвет**, и обратная ситуация.
21. Если человек любит **хип-хоп музыку**, то **жёлтый** его любимый **цвет**, и обратная ситуация.
22. Если человек любит **классическую музыку**, то **фэнтэзи** его любимый **жанр**, и обратная ситуация.
23. Если человек любит **фиолетовый цвет**, то **детектив** его любимый **жанр**, и обратная ситуация.
24. Если человек любит **голубой цвет**, то **шаурма** его любимая **еда**, и обратная ситуация.
25. Если человек любит **синий цвет,** то его сосед сверху слушает **рок музыку**, и обратная ситуация.
26. Если человек любит **народную музыку,** то его сосед сверху любит **триллеры**, и обратная ситуация.
27. Если человек любит **печенье,** то его сосед снизу любит **суши**, и обратная ситуация.
28. Если человек любит **рэп музыку,** то его сосед снизу любит **чёрный цвет**, и обратная ситуация.
29. Если человек любит **жёлтый цвет,** то его сосед сверху любит **фрукты**, и обратная ситуация.
30. Если человек любит **печенье**, то он сидит рядом с человеком, который любит **жанр ужасов.**
31. Если человек любит **бургер**, то он сидит рядом с человеком, который любит **синий цвет.**
32. Если человек любит **ужасы**, то он сидит рядом с человеком, который любит **пиццу.**
33. Если человек любит **хип-хоп**, то он сидит рядом с человеком, который любит **драму.**
34. Если человек любит **детективы**, то он сидит рядом с человеком, который любит **синий цвет.**
35. Если человек любит **рэп**, то он сидит рядом с человеком, который любит **металл.**

По известным фактам были найдены предпочтения каждого человека:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № Стула | Свойство 0 (любимый цвет) | Свойство 1 (любимая еда) | Свойство 2 (любимая музыка) | Свойство 3 (любимый жанр кино) |
| 0 | Красный | Вафли | Рок | Фэнтези |
| 1 | Оранжевый | Пицца | Электронная | Боевик |
| 2 | Синий | Суши | Классическая | Приключения |
| 3 | Фиолетовый | Карри | Рэп | Детектив |
| 4 | Голубой | Шаурма | Джаз | Триллер |
| 5 | Чёрный | Каша | Металл | Драма |
| 6 | Белый | Бургер | Народная | Фантастика |
| 7 | Зелёный | Печенье | Поп | Комедия |
| 8 | Жёлтый | Фрукты | Хип-Хоп | Ужасы |

## Заключение

Я научился работать с библиотекой BuDDy, придумал ограничения различного рода для создания однозначного решения и составил физическую интерпретацию задачи. Написал программу на языке C++ в среде разработки Visual Studio. Код проекта доступен в системе контроля версий **GIT**: **https://github.com/1kvin/mathematicalLogicCourseProject**

### Литература

1. Ю.Г. Карпов. Теория автоматов. Семнадцать практических занятий. – СПб.: Питер, 2003. – 208с.: ил.  
  
2. Ю.Г. Карпов. Задачи по курсу «Математическая логика и теория алгоритмов». – М.: Издательство СПбГПУ, 2014. – 105с.  
  
3. А.Б. Беляев, И.В. Шошмина. Использование бинарных решающих диаграмм для решения логических задач. Библиотека BuDDy. - 2014. - 24 с.  
  
4. Ю.Г. Карпов. Model Checking. Верификация параллельных и распределенных программных систем. – СПб.: БХВ-Петербург, 2010. - 560 с.  
  
5. Бинарная диаграмма решений: [Электронный ресурс]:URL: [https://ru.wikipedia.org/wiki/Бинарная\_диаграмма\_реше..](https://vk.com/away.php?utf=1&to=https%3A%2F%2Fru.wikipedia.org%2Fwiki%2F%D0%91%D0%B8%D0%BD%D0%B0%D1%80%D0%BD%D0%B0%D1%8F_%D0%B4%D0%B8%D0%B0%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0_%D1%80%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D0%B9)

1. Беляев А. Б., Шошмина И. В. Использование бинарных решающих диаграмм для решения логических задач, с. 6 [↑](#footnote-ref-1)