Алгоритм DPLL. Разобраться, запрограммировать и сравнить с другими методами экспериментально.

DPLL

Уточнения: *чистая переменная* - переменная, встречающаяся в наборе дизъюнкт только с отрицанием или без него
Пустой дизъюнкт – дизъюнкт имеющий значение FALSE

Алгоритм DPLL работает по принципу угадывания переменных, он пытается найти такой набор значений переменных, для которого формула будет выполнима. В процессе угадывание происходит ряд упрощений, за счет чего этот алгоритм работает довольно быстро.
Есть несколько правил для упрощения:
1) Если есть дизъюнкт с одной переменной, то этой переменной во всей формуле присваивается значение TRUE или FALSE в зависимости от отрицания у переменной находящейся в этом дизъюнкте.
2) Если есть чистая переменная, то такой переменной присваивается значение в зависимости от отрицания, чтобы значение стало равно истине.
Если эти правила применить нельзя, снова угадывают значение переменной.
Алгоритм выполняется до тех пор, пока мы либо не получим пустой дизъюнкт, либо не найдем набор при котором формула выполнима.

Псевдокод алгоритмов

Псевдокод:

```
Процедура DPLL //на вход подается значение для заполнения выбранной переменной
    Начало
        Выбираем переменную, которой будем присваивать значение
        Если такая переменная есть
            Заполняем ее полученным на входе в процедуру значением
        Проверяем, не нашли ли мы решение(выполняется ли набор дизъюнкт)
            Если да, возвращаем значение TRUE
        Проверяем есть ли пустой дизъюнкт в наборе дизъюнкт
            Если да, возвращаем значение FALSE
        Повторять
            Удаляем дизъюнкты со значением TRUE
            Если есть дизъюнкт с одной переменной
                Заполняем эту переменную значением в зависимости от отрицания
            Если был найден дизъюнкт с одной переменной
                Удаляем дизъюнкты со значением TRUE
            Если существует «чистая» переменная
                Заполняем эту переменную значением в зависимости от отрицания
        Пока происходят упрощения
        Если (DPLL(TRUE))
            Возвращаем значение TRUE
        Если (DPLL(FALSE))
            Возвращаем значение TRUE
    Конец
```

Заканчивает работу при двух ситуациях:
1) когда найдет значения всех переменных, при которых формула выполнима
2) когда найдет пустой дизъюнкт.
Если после окончания процедуры возвращено значение TRUE, то формула выполнима, если FALSE формула не выполнима.

Метод резолюции

Для сравнения с алгоритмом DPLL я выбрал алгоритм, основанный на методе резолюции.

В основе разработанного алгоритма лежит взятие всех возможных резольвент в поданном наборе дизъюнкт, но с некоторыми модификациями:

1) В список не добавляются тождественно истинные дизъюнкты
2) Не добавляются дизъюнкты, являющиеся продолжением уже существующих, например:
(X1 V X2) и (X1 V X2 V not X4) второй дизъюнкт является продолжением первого.

Псевдокод:

```
Процедура Resolution
  Начало
    Упрощаем КНФ
      //Удаляем дизъюнкты со значением TRUE, используем правило поглощения
    Пока мы не прошли по всему набору дизъюнкт включая резольвенты начиная со второго ( i )
      Пока мы не прошли по всему набору дизъюнкт, начиная с первого по i
        Если можем взять резольвенту
        //не является продолжением какого-либо дизъюнкта, не имеет значения TRUE
          Берем резольвенту
          Если резольвента – пустой дизъюнкт
            Возвращаем TRUE
          Пока в наборе есть дизъюнкты с одной переменной
            Берем резольвенты с множеством дизъюнкт содержащих эту переменную
          Если получили пустой дизъюнкт
            Возвращаем TRUE
        КЦ
      КЦ
    КЦ
    Возвращаем FALSE
  Конец
```

Заканчивается:
1) Когда найдет пустой дизъюнкт
2) При взятии всех возможных резольвент.
Если алгоритм возвращает значение TRUE, то найден пустой дизъюнкт и, следовательно, формула не выполнима. Если функция возвращает значение FALSE, то функция выполнима.

Пример для анализа

$(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4) \wedge (\neg x_2 \vee \neg x_3 \vee \neg x_4) \wedge$
$(x_3 \vee x_4 \vee x_5) \wedge (\neg x_3 \vee \neg x_4 \vee \neg x_5) \wedge (x_4 \vee x_5 \vee x_6) \wedge (\neg x_4 \vee \neg x_5 \vee \neg x_6) \wedge$
$(x_5 \vee x_6 \vee x_7) \wedge (\neg x_5 \vee \neg x_6 \vee \neg x_7) \wedge (x_6 \vee x_7 \vee x_8) \wedge (\neg x_6 \vee \neg x_7 \vee \neg x_8) \wedge$
$(x_7 \vee x_8 \vee x_9) \wedge (\neg x_7 \vee \neg x_8 \vee \neg x_9) \wedge (x_1 \vee x_3 \vee x_5) \wedge (\neg x_1 \vee \neg x_3 \vee \neg x_5) \wedge$
$(x_2 \vee x_4 \vee x_6) \wedge (\neg x_2 \vee \neg x_4 \vee \neg x_6) \wedge (x_3 \vee x_5 \vee x_7) \wedge (\neg x_3 \vee \neg x_5 \vee \neg x_7) \wedge$
$(x_4 \vee x_6 \vee x_8) \wedge (\neg x_4 \vee \neg x_6 \vee \neg x_8) \wedge (x_5 \vee x_7 \vee x_9) \wedge (\neg x_5 \vee \neg x_7 \vee \neg x_9) \wedge$
$(x_1 \vee x_4 \vee x_7) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_7) \wedge (x_2 \vee x_5 \vee x_8) \wedge (\neg x_2 \vee \neg x_5 \vee \neg x_8) \wedge$
$(x_3 \vee x_6 \vee x_9) \wedge (\neg x_3 \vee \neg x_6 \vee \neg x_9) \wedge (x_1 \vee x_5 \vee x_9) \wedge (\neg x_1 \vee \neg x_5 \vee \neg x_9)$

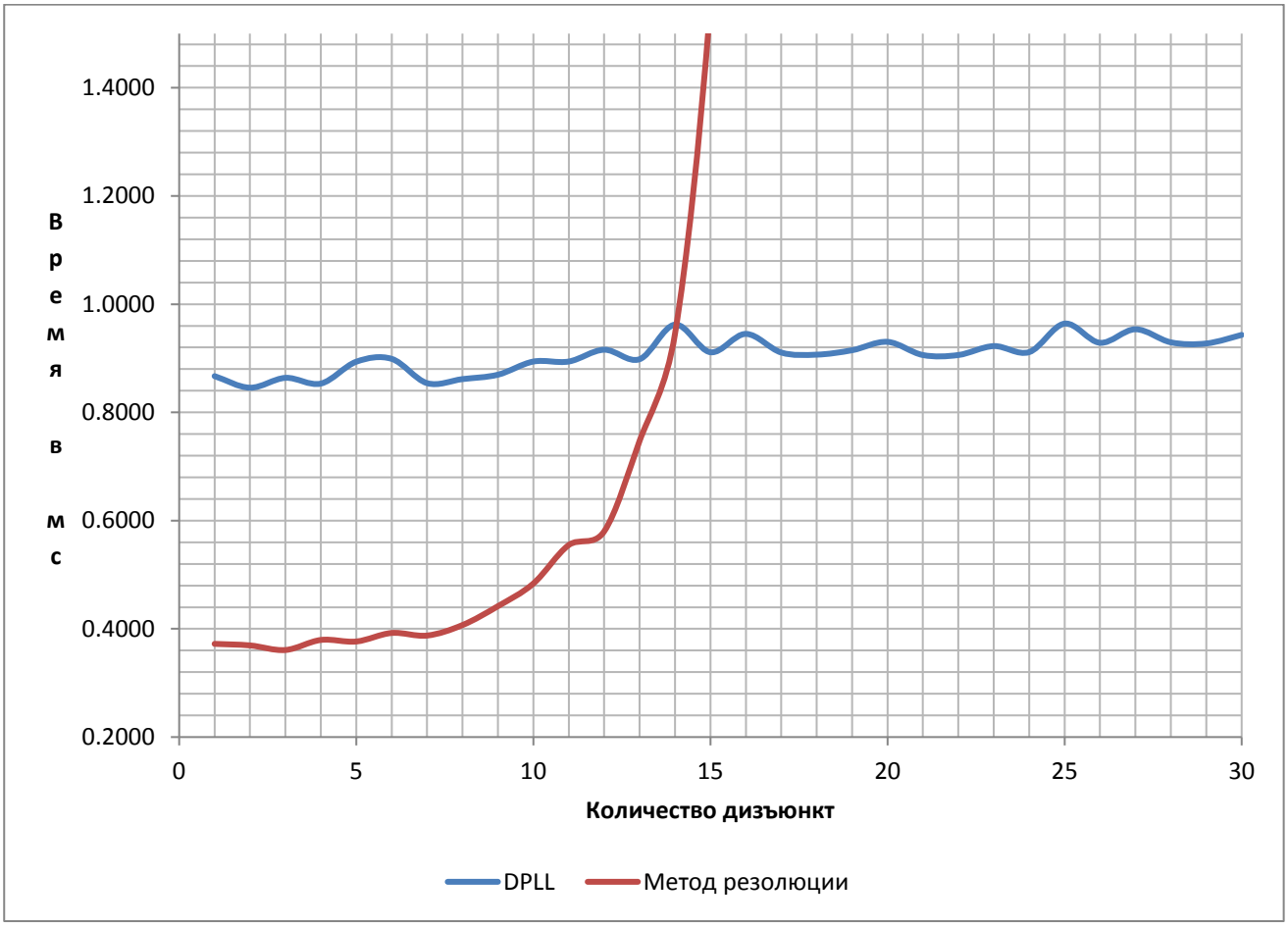
Будем выполнять сравнение экспериментального времени решения задачи SAT(задача выполнимости) алгоритма DPLL и метода резолюции, для различного количества дизъюнкт.

Для каждого случая выполним по три измерения и сосчитаем среднее время выполнения.

Количество дизъюнкт	Алгоритм DPLL в миллисекундах	Метод резолюции в миллисекундах	Среднее время выполнения **DPLL**	Среднее время выполнения **метод резолюции**
1	*0.8697*	*0.3710*	*0.8670*	*0.3723*
	0.8847	*0.3742*		
	0.8467	*0.3718*		
2	*0.8609*	*0.3773*	*0.8457*	*0.3694*
	0.8215	*0.3572*		
	0.8546	*0.3737*		
3	*0.8365*	*0.3529*	*0.8639*	*0.3609*
	0.8896	*0.3533*		
	0.8656	*0.3766*		
4	*0.8317*	*0.3771*	*0.8533*	*0.3795*
	0.8668	*0.3885*		
	0.8613	*0.3730*		
5	*0.8314*	*0.3687*	*0.8937*	*0.3766*
	0.9809	*0.3948*		
	0.8687	*0.3664*		
6	*0.9233*	*0.4074*	*0.8989*	*0.3924*
	0.9127	*0.3747*		
	0.8608	*0.3952*		
7	*0.8199*	*0.3939*	*0.8541*	*0.3876*
	0.8748	*0.3845*		
	0.8675	*0.3844*		
8	*0.8474*	*0.4019*	*0.8616*	*0.4070*
	0.8712	*0.4066*		
	0.8661	*0.4125*		
9	*0.8352*	*0.4322*	*0.8697*	0.4419
	0.8506	*0.4406*		
	0.9233	*0.4528*		

10	*0.9142*	*0.4835*	*0.8940*	*0.4837*
	0.8866	*0.4981*		
	0.8811	*0.4694*		
11	*0.8807*	*0.5594*	*0.8940*	*0.5549*
	0.8865	*0.5459*		
	0.9149	*0.5593*		
12	*0.9569*	*0.6032*	*0.9158*	*0.5801*
	0.8965	*0.5700*		
	0.8941	*0.5672*		
13	*0.9035*	*0.7140*	*0.8983*	*0.7474*
	0.8949	*0.7804*		
	0.8965	*0.7480*		
14	*0.8938*	*0.9861*	*0.9623*	*0.9452*
	0.9019	*0.9162*		
	1.0911	*0.9332*		
15	*0.9517*	*1.57*	*0.9112*	*1.5533*
	0.8701	*1.54*		
	0.9118	*1.55*		
16	*0.9355*	*2.44*	*0.9451*	*2.5933*
	0.9186	*2.89*		
	0.9812	*2.45*		
17	*0.9087*	*4.79*	*0.9106*	*4.6133*
	0.9204	*4.31*		
	0.9028	*4.74*		
18	*0.8991*	*5.71*	*0.9067*	*5.6900*
	0.9052	*5.61*		
	0.9159	*5.75*		
19	*0.9048*	*7.49*	*0.9149*	*7.3500*
	0.9158	*7.19*		
	0.9241	*7.37*		
20	*0.9114*	*8.89*	*0.9304*	*8.9300*
	0.9399	*8.98*		
	0.9398	*8.92*		
21	*0.9343*	*11.3*	*0.9060*	*11.5333*
	0.8822	*11.6*		
	0.9015	*11.7*		
22	*0.8814*	*14.7*	*0.9060*	*14.8333*
	0.9186	*14.9*		
	0.9181	*14.9*		
23	*0.88*	*21.5*	*0.9226*	*21.6333*
	0.9474	*21.7*		
	0.9403	*21.7*		
24	*0.9157*	*31.1*	*0.9114*	*30.5333*
	0.9225	*30.2*		
	0.896	*30.3*		
25	*0.9664*	*283*	*0.9641*	*269.3333*
	0.9379	*275*		
	0.9879	*250*		

26	*0.9628*	*767*	*0.9286*	*773.0000*
	0.923	*753*		
	0.9001	*799*		
27	*0.9094*	*3580*	*0.9536*	*3583.3333*
	0.9055	*3581*		
	1.046	*3589*		
28	*0.9261*	*9457*	*0.9295*	*9524.0000*
	0.9438	*9520*		
	0.9186	*9595*		
29	*0.9173*	*18091*	*0.9274*	*18088.0000*
	0.9142	*18222*		
	0.9506	*17951*		
30	*0.9277*	*30266*	*0.9432*	*30329.3333*
	0.923	*30813*		
	0.979	*29909*		
31	*Переполнение стека*	*56785*	*Переполнение стека*	*57402.0000*
		57342		
		58079		
32	*Переполнение стека*	*94637*	*Переполнение стека*	*95146.0000*
		94813		
		95988		



Анализ полученных значений

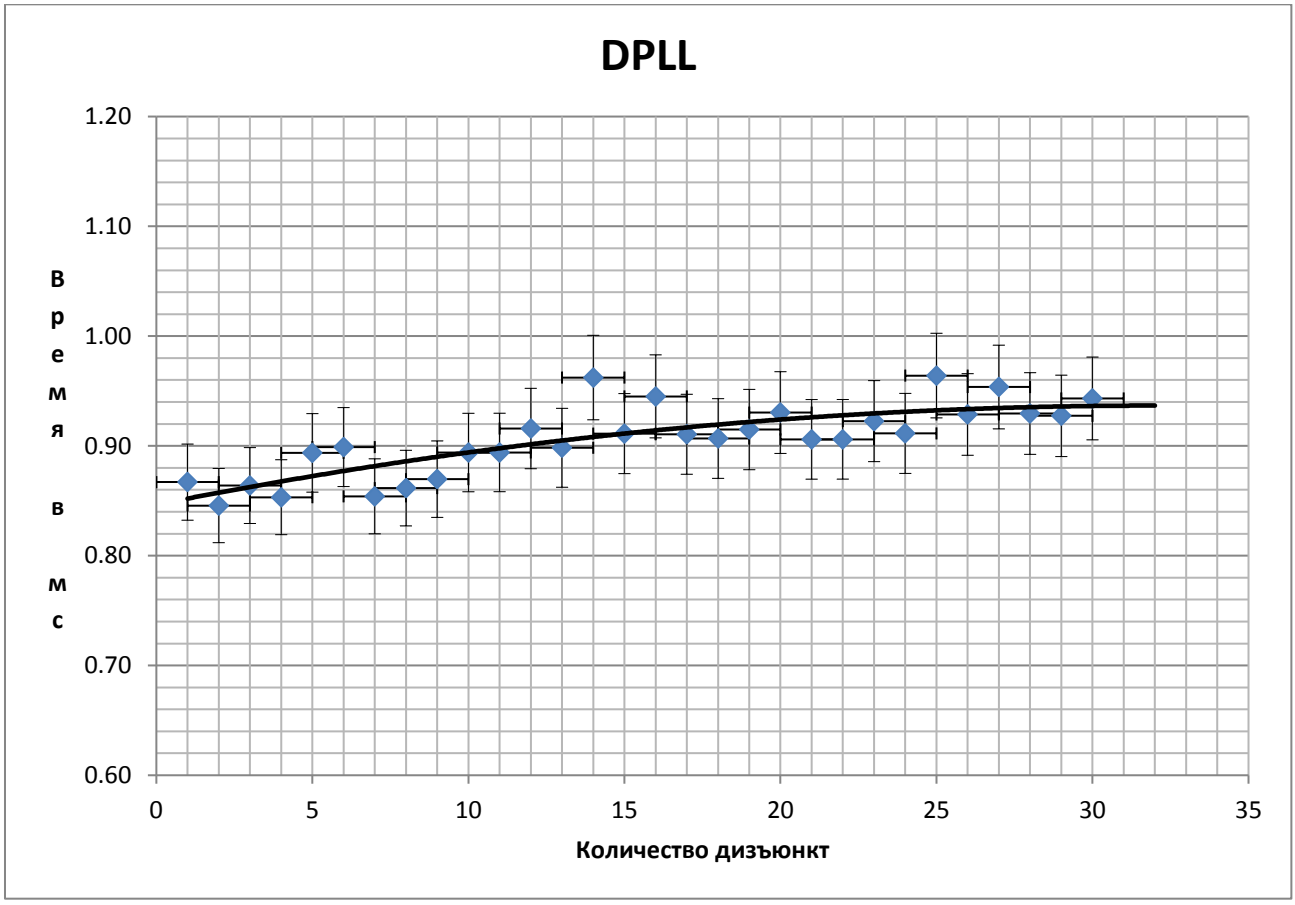
На графике заметно, что при малом количестве дизъюнкт метод резолюции работает быстрее, но, начиная с 14 дизъюнкта, метод резолюции начинает резко замедляться, что в дальнейшем ставит под вопрос целесообразность его применения. Алгоритм DPLL же, показывает стабильный результат на протяжении всего эксперимента, прирост времени очень мал.

Начиная с 31 дизъюнкта, в алгоритме DPLL возникает переполнение стека (алгоритм разработан с помощью использования рекурсии), метод резолюции продолжает работать при таких условиях, но очень медленно.

DPLL

На графике мы видим, что алгоритм DPLL хоть и работает медленнее при малом количестве дизъюнкт – более стабилен, чем метод резолюции. Скорость возрастания времени работы у алгоритма DPLL схожа с логарифмическим ростом времени работы в среднем. Сложность алгоритма, рассчитать довольно тяжело, так как в его основе лежит рекурсия. Визуализировать его работу можно в виде двоичного дерева. В таком случае, функция сложности содержит логарифм. Если судить по графику, то примерно O(log(n)).

Рост времени работы алгоритма больше связан с количеством переменных в формуле, чем с количеством дизъюнкт.



Метод Резолюции

Рост времени выполнения у метода резолюции – полиномиальный сложность его(грубо) O(n^2) (Относительно количества дизъюнкт). В худшем случае, мы возьмем все возможные резольвенты. Худший случай, когда формула имеет решение, так как алгоритм пытается опровергнуть формулу.

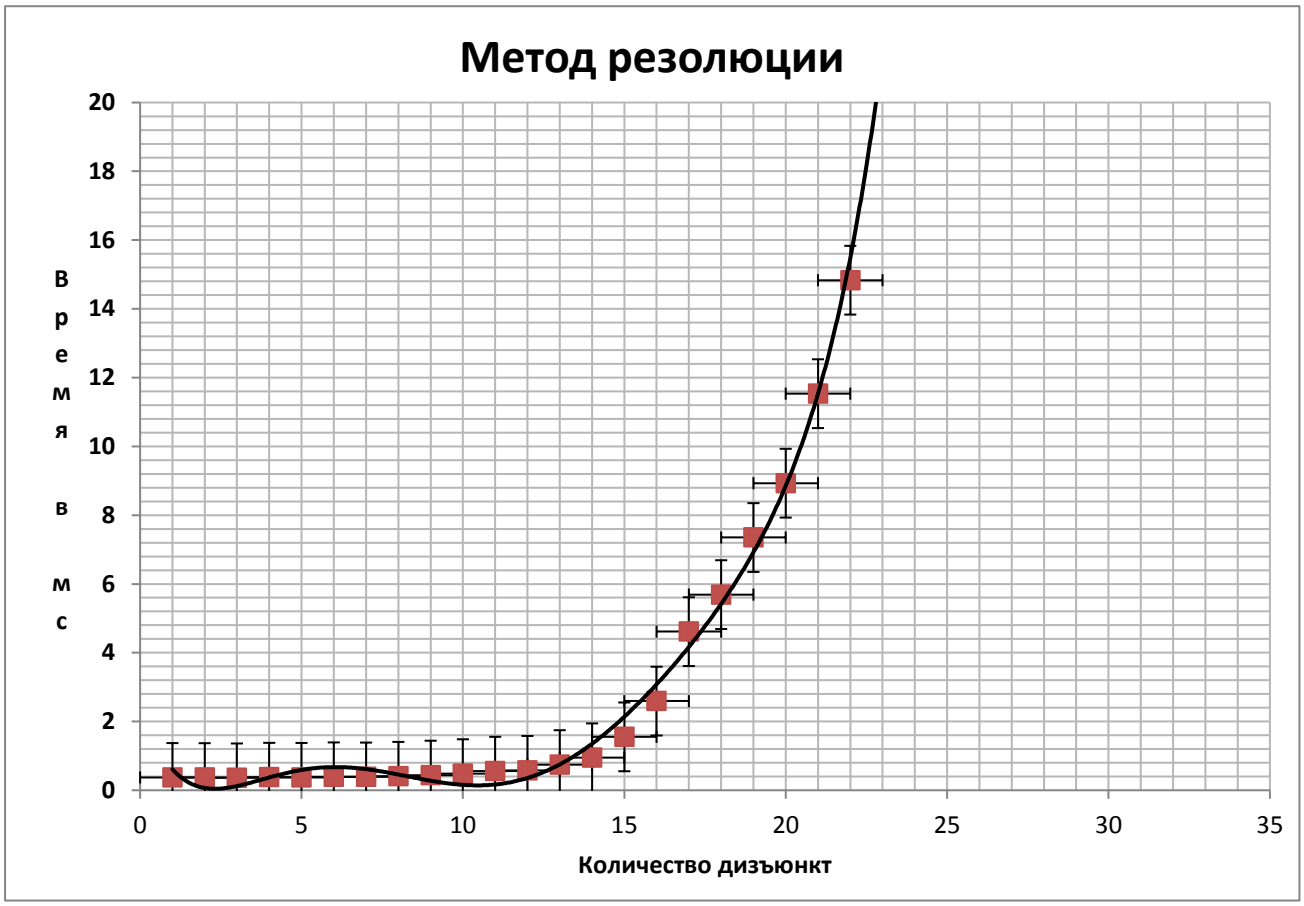
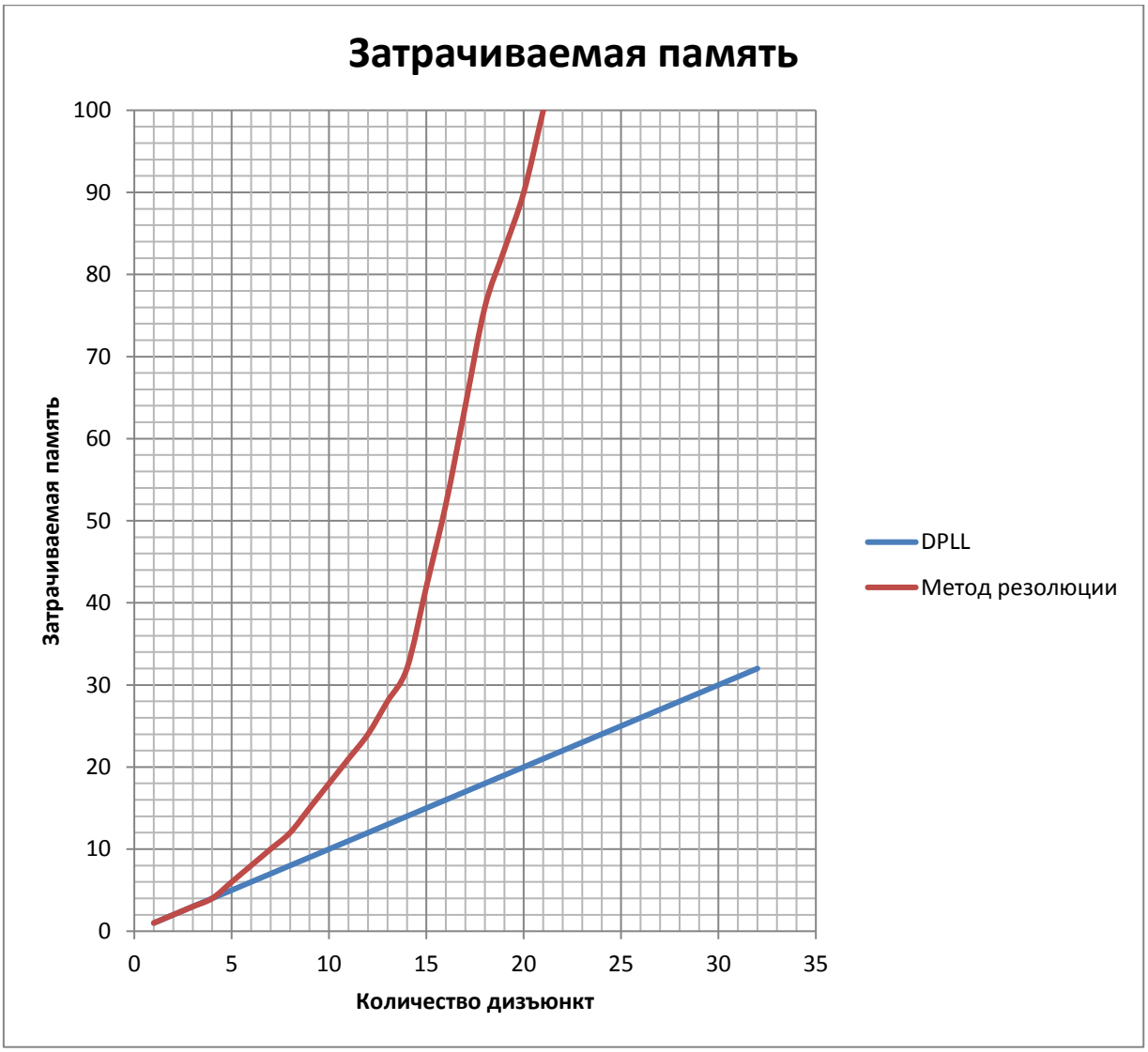


График затрачиваемой памяти для двух методов



Затрачиваемая память измеряется в количестве дизъюнкт, с которым работает программа. Один дизъюнкт: 14 байт, не считая переменных содержащихся в нем.

Алгоритм DPLL не использует дополнительной памяти, он работает с уже имеющимся набором дизъюнкт. Метод резолюции тратит память на хранение полученных в ходе работы метода резольвент, на графике можно увидеть полиномиальный рост затрат памяти.

Вывод:

Метод резолюции, как и любой метод основанный на полном переборе, довольно медленный и неэффективный, алгоритм DPLL намного эффективнее метода резолюции в плане скорости и самого подхода к решению задачи выполнимости. Но в данной работе используется рекурсивная реализация метода DPLL – это значит, что для решения больших задач этот алгоритм не подходит, надо использовать не рекурсивную реализацию, для получения максимального эффекта.

Другой пример и протоколы работы программы к нему

(X1 V X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(not X1 V X3)^(X2 V not X3 V X4)

DPLL

Assign a variable X1 the value true
(X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(X3)^(X2 V not X3 V X4)
Simplify clauses with value true
(not X3 V not X4)^(not X2 V X4)^(X3)^(X2 V not X3 V X4)
Variable X3 one in clause
Assign a variable X3 the value true
(not X4)^(not X2 V X4)^(X2 V X4)
Variable X4 one in clause
Assign a variable X4 the value false
(not X2)^(X2)
Variable X2 one in clause
Assign a variable X2 the value false
(false)
Take another variable value:
(false)
The empty clause contained in the formula!
FAILURE

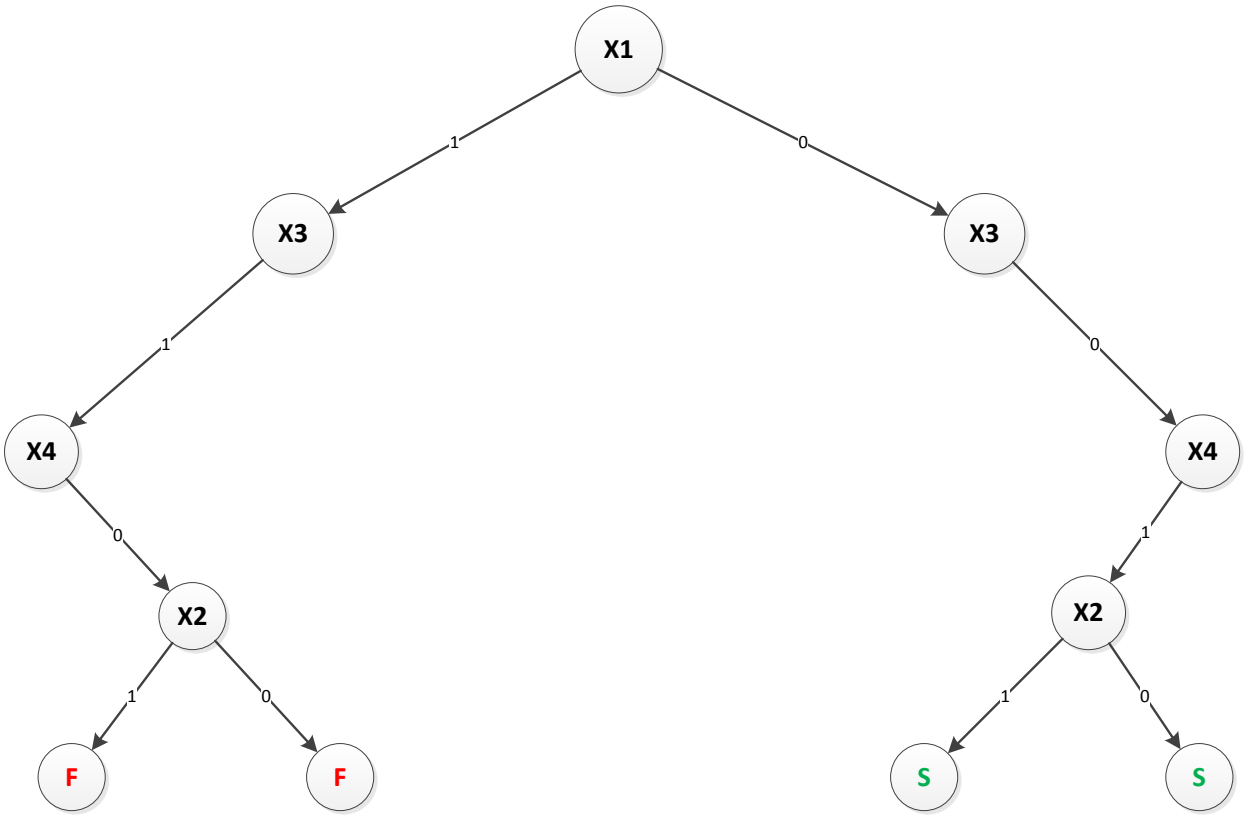
Second Computation

Assign a variable X1 the value false
(X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(X3)^(X2 V not X3 V X4)
Simplify clauses with value true
(X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(X2 V not X3 V X4)
Variable X3 in the whole formula is included with denial
Assign a variable X3 the value false
(X2 V X4)^(not X4)^(not X2 V X4)^(X2 V X4)
Simplify clauses with value true
(X2 V X4)^(not X2 V X4)
Variable X4 in the whole formula is included without denial
Assign a variable X4 the value true
(X2)^(not X2)
Simplify clauses with value true

SUCCESS
runtime: 230.6 ms //много из-за вывода протокола(в программе можно отключить протокол)
X1 = 0;
X2 = 0;
X4 = 1;
X3 = 0;

Дерево обхода значений для этого примера DPLL

(X1 V X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(not X1 V X3)^(X2 V not X3 V X4)

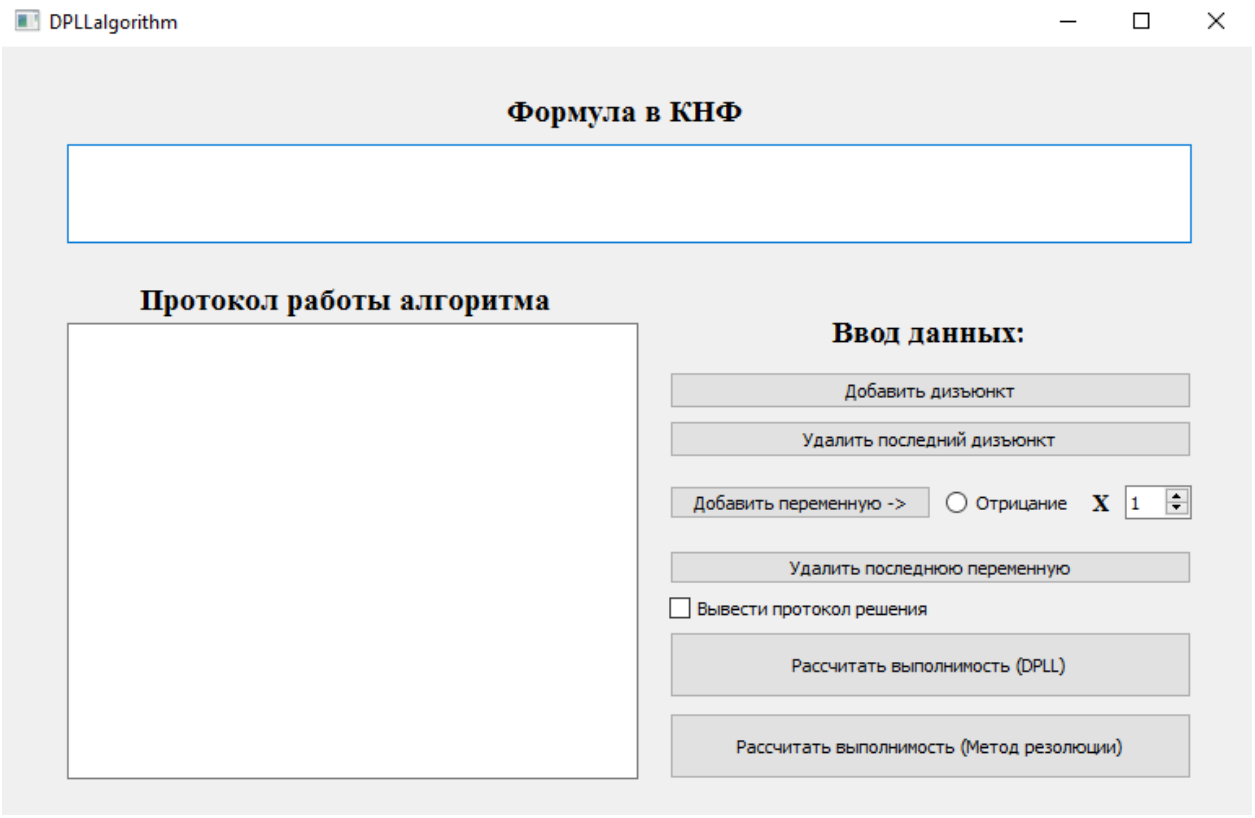


Для этого примера не важно какое значение примет переменная X2.

Метод Резолюции

Take the resolution of (not X3 V not X4) and (X1 V X2 V X4) clauses:
(X1 V X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(not X1 V X3)^(X2 V not X3 V X4)^(X1 V X2 V not X3)
Take the resolution of (not X2 V X4) and (X1 V X2 V X4) clauses:
(X1 V X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(not X1 V X3)^(X2 V not X3 V X4)^(X1 V X2 V not X3)^(X1 V X4)
Take the resolution of (not X2 V X4) and (not X3 V not X4) clauses:
(X1 V X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(not X1 V X3)^(X2 V not X3 V X4)^(X1 V X2 V not X3)^(X1 V X4)^(not X2 V not X3)
Take the resolution of (not X1 V X3) and (X1 V X2 V X4) clauses:
(X1 V X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(not X1 V X3)^(X2 V not X3 V X4)^(X1 V X2 V not X3)^(X1 V X4)^(not X2 V not X3)^(X2 V X3 V X4)
Take the resolution of (not X1 V X3) and (not X3 V not X4) clauses:
(X1 V X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(not X1 V X3)^(X2 V not X3 V X4)^(X1 V X2 V not X3)^(X1 V X4)^(not X2 V not X3)^(X2 V X3 V X4)^(not X1 V not X4)
Take the resolution of (X2 V not X3 V X4) and (not X3 V not X4) clauses:
(X1 V X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(not X1 V X3)^(X2 V not X3 V X4)^(X1 V X2 V not X3)^(X1 V X4)^(not X2 V not X3)^(X2 V X3 V X4)^(not X1 V not X4)^(X2 V not X3)
Take the resolution of (X2 V not X3 V X4) and (not X2 V X4) clauses:
(X1 V X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(not X1 V X3)^(X2 V not X3 V X4)^(X1 V X2 V not X3)^(X1 V X4)^(not X2 V not X3)^(X2 V X3 V X4)^(not X1 V not X4)^(X2 V not X3)^(not X3 V X4)
Take the resolution of (X2 V not X3 V X4) and (not X1 V X3) clauses:
(X1 V X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(not X1 V X3)^(X2 V not X3 V X4)^(X1 V X2 V not X3)^(X1 V X4)^(not X2 V not X3)^(X2 V X3 V X4)^(not X1 V not X4)^(X2 V not X3)^(not X3 V X4)^(not X1 V X2 V X4)
Take the resolution of (X1 V X4) and (not X3 V not X4) clauses:
(X1 V X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(not X1 V X3)^(X2 V not X3 V X4)^(X1 V X2 V not X3)^(X1 V X4)^(not X2 V not X3)^(X2 V X3 V X4)^(not X1 V not X4)^(X2 V not X3)^(not X3 V X4)^(not X1 V X2 V X4)^(X1 V not X3)
Take the resolution of (X1 V X4) and (not X1 V X3) clauses:
(X1 V X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(not X1 V X3)^(X2 V not X3 V X4)^(X1 V X2 V not X3)^(X1 V X4)^(not X2 V not X3)^(X2 V X3 V X4)^(not X1 V not X4)^(X2 V not X3)^(not X3 V X4)^(not X1 V X2 V X4)^(X1 V not X3)^(X3 V X4)
Take the resolution of (not X2 V not X3) and (not X1 V X3) clauses:
(X1 V X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(not X1 V X3)^(X2 V not X3 V X4)^(X1 V X2 V not X3)^(X1 V X4)^(not X2 V not X3)^(X2 V X3 V X4)^(not X1 V not X4)^(X2 V not X3)^(not X3 V X4)^(not X1 V X2 V X4)^(X1 V not X3)^(X3 V X4)^(not X1 V not X2)
Take the resolution of (X2 V X3 V X4) and (X2 V not X3 V X4) clauses:
(X1 V X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(not X1 V X3)^(X2 V not X3 V X4)^(X1 V X2 V not X3)^(X1 V X4)^(not X2 V not X3)^(X2 V X3 V X4)^(not X1 V not X4)^(X2 V not X3)^(not X3 V X4)^(not X1 V X2 V X4)^(X1 V not X3)^(X3 V X4)^(not X1 V not X2)^(X2 V X4)
Take the resolution of (X2 V not X3) and (not X1 V X3) clauses:
(X1 V X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(not X1 V X3)^(X2 V not X3 V X4)^(X1 V X2 V not X3)^(X1 V X4)^(not X2 V not X3)^(X2 V X3 V X4)^(not X1 V not X4)^(X2 V not X3)^(not X3 V X4)^(not X1 V X2 V X4)^(X1 V not X3)^(X3 V X4)^(not X1 V not X2)^(X2 V X4)^(not X1 V X2)
Take the resolution of (X2 V not X3) and (not X2 V not X3) clauses:
(X1 V X2 V X4)^(not X3 V not X4)^(not X2 V X4)^(not X1 V X3)^(X2 V not X3 V X4)^(X1 V X2 V not X3)^(X1 V X4)^(not X2 V not X3)^(X2 V X3 V X4)^(not X1 V not X4)^(X2 V not X3)^(not X3 V X4)^(not X1 V X2 V X4)^(X1 V not X3)^(X3 V X4)^(not X1 V not X2)^(X2 V X4)^(not X1 V X2)^(not X3)
Take all resolutions with one-variable clause: (not X3)
(X1 V X2 V X4)^(not X2 V X4)^(not X1)^(X1 V X4)^(X2 V X4)^(not X1 V not X4)^(not X1 V X2 V X4)^(X4)^(not X1 V not X2)^(X2 V X4)^(not X1 V X2)
Take all resolutions with one-variable clause: (not X1)
(X2 V X4)^(not X2 V X4)^(X4)^(X2 V X4)^(X4)^(X2 V X4)
Take all resolutions with one-variable clause: (X4)
SUCCESS
runtime: 1554.87 ms//много из-за вывода протокола(в программе можно отключить протокол)

Программа (графический интерфейс программы)



Ввод производится помощью кнопок справа. В центральном верхнем поле отображается введенное выражение. Выражение вводится в форме КНФ. В левом окне отображается протокол работы выбранного алгоритма, отключить протокол можно убрав флажок с поля «Вывести протокол решения»(по умолчанию отключён), это значительно ускорит работу алгоритмов. Выбрать с помощью какого алгоритма нужно решить задачу выполнимости можно там же, в правом нижнем углу.

Подробнее об этих алгоритмах и самой задаче выполнимости булевых формул я расскажу во время проведения альтернативного экзамена.