Рассмотрим экземпляр задачи выполнимости, заданный условиями C1, ..., Ck по множеству булевых переменных x1, ..., xn . Экземпляр называется монотонным, если литерал в каждом условии состоит из неинвертированной переменной, то есть каждый литерал равен xi для некоторого i, а не . Монотонные экземпляры задачи выполнимости решаются очень легко: они всегда выполнимы присваиванием каждой переменной 1. Предположим, имеются три условия (x1 Ҏ x2 ), (x1 Ҏ x3 ), (x2 Ҏ x3 ). Экземпляр является монотонным, и присваивание, задающее все три переменные равными 1, действительно выполняет все условия. Но это не единственное выполняющее присваивание; также можно было присвоить x1 и x2 значение 1, а x3 значение 0. В самом деле, для любого монотонного экземпляра естественно задать вопрос о минимальном количестве переменных, которым необходимо присвоить 1 для его выполнения.

Глава 8. NP-полнота и вычислительная неразрешимость Задача монотонной выполнимости с минимумом истинных переменных формулируется так: для заданного монотонного экземпляра задачи выполнимости и числа k существует ли выполняющее присваивание для экземпляра, в котором не более k переменных задано значение 1? Докажите, что задача является NP-полной.

1.**Сначала нужно доказать, что наша задача лежит в классе NP.** Для этого опишем недетерминированную машину Тьюринга, которую ее решает. Первым делом МТ недетерминированно порождает всевозможные присваивания. Так как это происходит независимо на различных лентах, потребуется линейное от количества переменных число шагов. Теперь осталось проверить для каждого присваивания, что условия выполняются. Это также требует полиномаильного числа шагов. Если найдется хоть одно такое присваивание, то ответ в задаче положительный. В качестве сертификата используются набор значений переменных.

2. **Теперь сведём Зада́чу выполни́мости бу́левых фо́рмул (SAT или ВЫП) к нашей задаче за полиномиальное время.**

В [теории сложности вычислений](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D1%81%D0%BB%D0%BE%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D0%B8_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D0%B9) сведение задачи R1(Задача вершинного покрытия){\displaystyle R\_{1}} к {\displaystyle R\_{2}}R2 по Куку — это [полиномиальный](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_P) по времени [алгоритм](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC) (другими словами, [машина Тьюринга](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%B0_%D0%A2%D1%8C%D1%8E%D1%80%D0%B8%D0%BD%D0%B3%D0%B0) с полиномиальным временем работы), решающий задачу R1 {\displaystyle R\_{1}}при условии, что функция, находящая решение задачи R2{\displaystyle R\_{2}}, ему дана в качестве [оракула](https://ru.wikipedia.org/wiki/%D0%9E%D1%80%D0%B0%D0%BA%D1%83%D0%BB_(%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D1%81%D0%BB%D0%BE%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D0%B8_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D0%B9)), то есть обращение к ней занимает всего один шаг.

Допустим, у нас есть алгоритм, решающий нашу задачу за полиномиальное время:

Что подаётся (конъюкция дизъюнкций литералов без отрицаний)

Условия записываются следующим образом:

List<List<int>> conditions=

{{1,2,3},

{3,2,4},

{1,5}}

Алгоритм понимает это так:

(x1 ИЛИ x2 ИЛИ х3) И (х3 ИЛИ х2 ИЛИ х4) И (х1 ИЛИ х5)

Будем называть алгоритм решения нашей задачи solveOurTask

Возвращается ответ: нет или да(существует ли присваивание)

Что такое задача вершинного покрытия?

Задача о вершинном покрытии состоит в поиске вершинного покрытия наименьшего размера для заданного графа (этот размер называется числом вершинного покрытия графа).

Также вопрос можно ставить как эквивалентную задачу разрешения:

На входе: Граф G и положительное целое число k.

**Вопрос: Существует ли вершинное покрытие C для графа G размера не более k?**

Итак, у нас есть экземпляр задачи вершинного покрытия

Список вершин:0,1,2,3.4,5(например)

И список того, какие из них соединены рёбрами: (2,3), (3,4), (3,1), (3,0), (0,1)

А, теперь, внимание, ~~магия~~ сведение: список вершин заменяем на переменные (т.е получаем х1, х2, х3, х4, х5), а рёбра заменяем на элементарные дизъюнкции (х2 v x3) And (x3 v x4) And (x3 v x1) And (x3 v x0) And (x0 v x1).Почему это сведение корректно? Ну смотрите: чтобы вышеописанное условие было истинным, нужно, чтобы в каждой дизъюнкции хотя бы одна переменная была равна 1, и каждая дизъюнкция символизирует определённое ребро, причём если переменная равна 1, то это означает, что соответствующая вершина принадлежит вершинному покрытию. Проще говоря, если мы найдём допустимое присваивание, то оно 100%-но будет являться вершинным покрытием, т.к. это всё равно что взять минимум по одной вершине с каждого ребра. Таким образом, подставляем «сведённое условие задачи выполнимости в solveOurTask и, с учётом сведения, получаем ответ на вопрос.

Таким образом, мы получили полиномиальный алгоритм, решающий задачу SAT. А это означает, что задача SAT сводится к нашей задаче, т.е сложность нашей задачи не более, чем сложность задачи SAT