Б-3. Выполнимость

14 декабря 2024 г. 18:54

Формальные доказательства

Формальные доказательства

- Формальные доказательства — большой раздел математической логики
- ⭑ Формальное доказательство — это конечная последовательность синтаксически корректных формул, составленная по правилам, определяемым системой доказательств
- Система доказательств состоит из
  - правил вывода (получения новых формул из имеющихся в доказательстве)
  - аксиом (формул, которые можно включать в доказательство без ограничений)
- Цель формального доказательства — из набора формул-условий получить требуемую формулу-заключение

  Пример:
  - формулы — слова над алфавитом $\{S, (,)\}$ {S, левая скобка, правая скобка}
  - аксиома — слово S
  - правила вывода: любой символ S в формуле можно заменить на SS, (S) или $()$
  - вывод формулы $(()())$ из пустого набора условий:

    S - аксиома
    (S)
    (SS)
    $(()S)$
    $(()())$

  - ⭑ формула над $\{(,)\}$ выводится из пустого набора условий
  - ⇔ она является правильной расстановкой скобок
- ⭑ Такие системы доказательств называются формальными грамматиками

Доказательство теорем

Теорема состоит из набора формул, одна из которых стоит особняком (гипотеза)

- Как выглядит теорема?
  - даны условия $F_1, \ldots, F_k$ и гипотеза G
  - доказать, что из условий следует гипотеза
  - ⭑ т.е. что формула $(F_1 \wedge F_2 \wedge \cdots \wedge F_k) \to G$ — тавтология
  - ⭑ если формула $X \to Y$ — тавтология, то формула Y называется следствием X
- Простейший случай: $F_1, \ldots, F_k, G$ — булевы формулы
  - ⭑ могут быть более сложные формулы (с предикатами, кванторами и т. д.)
  - ⭑ даже для булевых формул проверка «в лоб» очень трудоемка: таблицу значений формулы из m литералов и n переменных можно вычислить за время $\Theta(m \cdot 2^n)$
- Доказательство от противного:
  - доказать, что $\overline{(F_1 \wedge F_2 \wedge \cdots \wedge F_k) \to G}$ — противоречие
  - эквивалентная формула: $F_1 \wedge F_2 \wedge \cdots \wedge F_k \wedge \bar{G}$
  - ⭑ если каждую из формул $F_1, \ldots, F_k, \bar{G}$ заменить на эквивалентную КНФ, общая формула станет КНФ
- Задача: дана КНФ, является ли она противоречием?
- ⭑ Наблюдение: Y — следствие $\overset{x \le y}{X}$ ⇔ X эквивалентна $X \wedge Y$
- ⭑ Следствие: 0 — следствие X ⇔ X — противоречие Вывели тождественно ложную формулу
- Стратегия доказательства: получить 0 как следствие исходной формулы
- Метод резолюций — система доказательств, реализующая эту стратегию

Y - следствие X, т.е. $X \le Y$ $\forall x \in X, y \in Y$.
$\Rightarrow X \sim X \wedge Y$, т.к. $\wedge$ - это минимум

*Противоречие - тождественно ложное выражение

Метод резолюций

Лемма о следствии

Лемма

Для любых булевых формул X, Y, Z формула $Y \vee Z$ — следствие формулы $(X \vee Y) \wedge (\bar{X} \vee Z)$.

Доказательство:
- пусть $F_{|\vec{b}}$ обозначает результат подстановки набора значений $\vec{b}$ в формулу F
- пусть $\vec{b}$ — произвольный набор, такой что $((X \vee Y) \wedge (\bar{X} \vee Z))_{|\vec{b}} = 1$

⇒ $(X \vee Y)_{|\vec{b}} = 1$, $(\bar{X} \vee Z)_{|\vec{b}} = 1$
- если $X_{|\vec{b}} = 1$, то $\bar{X}_{|\vec{b}} = 0 \Rightarrow Z_{|\vec{b}} = 1$
- если $X_{|\vec{b}} = 0$, то $Y_{|\vec{b}} = 1$

⇒ $(Y \vee Z)_{|\vec{b}} = 1$

⇒ $((X \vee Y) \wedge (\bar{X} \vee Z)) \to (Y \vee Z)$ — тавтология □

Метод резолюций

Клоз можем считать множеством литералов, т.к. $\vee$ - коммутативно, ассоциативная операция

Метод резолюций:
- формулы, которыми оперирует метод — это клозы (элементарные дизъюнкции)
- клоз рассматривается как множество литералов
  - порядок литералов не важен, повторяющиеся литералы стираются
- единственное правило вывода — правило резолюций:
  - если есть клозы вида $x \vee C$ и $\bar{x} \vee D$ (x — переменная), дописать клоз $C \vee D$
  - ⭑ клоз, содержащий пару литералов $\{y, \bar{y}\}$, не дописывается
  - если C и D — пустые множества литералов, дописывается пустой клоз □
- аксиом нет
- условия — все клозы КНФ, поданной на вход метода
- цель — получить пустой клоз

C, D - могут быть пустым мн-вом литералов

Метод резолюций

Клоз можем считать множеством литералов, т.к. ∨ - коммутативно, ассоциативная операция

Метод резолюций:

- формулы, которыми оперирует метод — это клозы (элементарные дизъюнкции)
- клоз рассматривается как множество литералов
  - порядок литералов не важен, повторяющиеся литералы стираются
- единственное правило вывода — правило резолюций:
  - если есть клозы вида $x \vee C$ и $\bar{x} \vee D$ (x — переменная), дописать клоз $C \vee D$
  - ★ клоз, содержащий пару литералов $\{y, \bar{y}\}$, не дописывается
  - если C и D — пустые множества литералов, дописывается пустой клоз $\square$
- аксиом нет
- условия — все клозы КНФ, поданной на вход метода
- цель — получить пустой клоз

C, D - могут быть пустым мн-вом литералов

Полнота метода резолюций

Теорема о полноте

Теорема о полноте метода резолюций

КНФ $F = C_1 \wedge \cdots \wedge C_k$ является противоречием $\Leftrightarrow$ существует доказательство методом резолюций с условиями $C_1, \ldots, C_k$ и заключением $\square$.

Доказательство достаточности:

- рассмотрим доказательство методом резолюций с заключением $\square$
- каждая формула является либо условием, либо получено по правилу резолюций из каких-то предыдущих формул
  - а значит, является следствием конъюнкции этих формул согласно лемме
- отношение «быть следствием» транзитивно
- любая формула вида $C_{i_1} \wedge \cdots \wedge C_{i_j}$ является следствием F
- $\Rightarrow$ любая формула в доказательстве является следствием F
- ★ пустой клоз является следствием формулы $x \wedge \bar{x}$, а значит, задает константу 0
- $\Rightarrow$ 0 — следствие $F \Rightarrow F$ — противоречие $\square$

т.к. = транзитивно

Комментарий:

- ★ мы доказали корректность метода: если существует доказательство, содержащее пустой клоз, то заданная КНФ действительно является противоречием
- ★ обратная импликация доказывает полноту метода: если КНФ — противоречие, то это можно доказать методом резолюций

Доказательство необходимости

- Проведем индукцию по числу n переменных в F
- **База индукции:** $n = 1$
  - F — противоречие $\Rightarrow$ F содержит клозы x и $\bar{x}$
  - $\Rightarrow$ по правилу резолюций из x и $\bar{x}$ выводится пустой клоз
- **Шаг индукции:**
  - пусть $F = F(x_1, \ldots, x_n)$, $S = \{C_1, \ldots, C_k\}$
  - считаем, что клоз не может содержать одновременно x_i и $\bar{x}_i$
    - если такой клоз есть, он задает константу 1 и может быть удален из F
  - построим два множества клозов, S^+ и S^-:
  - $S^+ = \{C \in S \mid \text{в } C \text{ нет переменной } x_n\} \cup \{C \mid (C \vee x_n) \in S\}$
  - $S^- = \{C \in S \mid \text{в } C \text{ нет переменной } x_n\} \cup \{C \mid (C \vee \bar{x}_n) \in S\}$
  - ★ докажем, что КНФ $F^+ = \bigwedge_{C \in S^+} C$ является противоречием:
  - пусть существует набор значений $b_1, \ldots, b_{n-1}$ такой, что $F^+_{|b_1,\ldots,b_{n-1}} = 1$
  - рассмотрим значения всех клозов из множества S на наборе $b_1, \ldots, b_{n-1}, 0$:
    - если клоз C не содержит переменную x_n, то $C_{|b_1,\ldots,b_{n-1},0} = C_{|b_1,\ldots,b_{n-1}} = 1$
    - если клоз имеет вид $C \vee x_n$, то $(C \vee x_n)_{|b_1,\ldots,b_{n-1},0} = C_{|b_1,\ldots,b_{n-1}} = 1$
    - клоз вида $C \vee \bar{x}_n$ превращается в 1 за счет значения $b_n = 0$
  - $\Rightarrow$ $F_{|b_1,\ldots,b_{n-1},0} = 1$, что невозможно, так как F — противоречие
  - ★ аналогично, $F^- = \bigwedge_{C \in S^-} C$ является противоречием
    - к гипотетическому набору, выполняющему F^-, надо добавить $b_n = 1$
  - ★ по предположению индукции, из каждого из множеств S^+, S^- можно вывести пустой клоз

Шаг индукции — окончание

- Рассмотрим вывод пустого клоза из множества S^+
  - если в выводе участвовали только клозы из S, то из S выводим пустой клоз
  - пусть в выводе участвовал хотя бы один клоз $C \in S^+ \setminus S$; тогда $(C \vee x_n) \in S$
  - $\Rightarrow$ построим вывод из S, заменив в выводе из S^+ каждый клоз из $S^+ \setminus S$ на соответствующий клоз из S
  - $\Rightarrow$ во всех следствиях из таких клозов добавится литерал x_n
  - $\Rightarrow$ из S выводится клоз x_n
- ★ аналогично, из вывода пустого клоза из S^- получим вывод клоза $\bar{x}_n$ из S
  - $\Rightarrow$ из клозов x_n и $\bar{x}_n$ получим пустой клоз $\square$

Комментарий:

- ★ искать доказательства методом резолюций может компьютер
  - существуют различные стратегии оптимизации поиска вывода
- ★ на более общем варианте метода резолюций (для формул логики первого порядка) основан язык Пролог

★ Если формула F не является противоречием, то метод резолюций заканчивает работу, когда не может вывести больше ни одного нового клоза

- по построенным клозам можно восстановить набор значений, выполняющий F

Пример доказательства методом резолюций

Пример доказательства методом резолюций

Гипотеза: Васю скоро уволят

1 условие ... 2 условие ... 3 условие ... 4 условие

Вася всегда приходит на совещание, если босс его позвал. Если босс хочет видеть Васю, он зовет его на совещание. Если босс не хочет видеть Васю и не зовет его на совещание, то Васю скоро уволят. Вася не пришел на совещание. Докажите, что его скоро уволят.

- Запишем теорему, которую надо доказать:
  - ⋆ $\Big((invite \to attend) \wedge (see \to invite) \wedge ((\overline{see} \wedge \overline{invite}) \to fire) \wedge (\overline{attend})\Big) \to fire$
- Отрицание теоремы:
  - ⋆ $(invite \to attend) \wedge (see \to invite) \wedge ((\overline{see} \wedge \overline{invite}) \to fire) \wedge (\overline{attend}) \wedge \overline{fire}$
- КНФ отрицания теоремы и множество клозов:
  - ⋆ $(\overline{invite} \vee attend) \wedge (\overline{see} \vee invite) \wedge (see \vee invite \vee fire) \wedge (\overline{attend}) \wedge (\overline{fire})$
  - $S = \{\overline{invite} \vee attend,\ \overline{see} \vee invite,\ see \vee invite \vee fire,\ \overline{attend},\ \overline{fire}\}$

Доказательство:

1.	$see \vee invite \vee fire$	условие
2.	$\overline{fire}$	условие
3.	$see \vee invite$	по правилу резолюций из 1,2
4.	$\overline{see} \vee invite$	условие
5.	$invite$	по правилу резолюций из 3,4; $invite \vee invite = invite$
6.	$\overline{invite} \vee attend$	условие
7.	$attend$	по правилу резолюций из 5,6
8.	$\overline{attend}$	условие
9.	□	по правилу резолюций из 7,8 жалко Васю...

Выполнимость булевой формулы (SAT)

Задача SAT

- ⋆ Метод резолюций — один из способов решения задачи SAT (комбинаторная задача)
  - от satisfiability (выполнимость)
- SAT: дана КНФ $F = \bigwedge_{i=1}^{\ell} C_i$, определить, выполнима ли она
  - если F выполнима, обычно нужно предъявить пример
    - т.е. булев вектор $\vec{b}$ такой, что $F_{|\vec{b}} = 1$
  - если F — противоречие, иногда нужно предъявить доказательство
- SAT — трудная задача
  - NP-полная
- SAT — самая важная NP-полная задача
  - для нее существуют эффективные с практической точки зрения решатели (SAT-solvers)
- ! Очень часто оптимальный способ решения других трудных задач состоит в том, чтобы перекодировать задачу в задачу SAT и скормить решателю
  - так решают
  - ⋆ задачи планирования
  - ⋆ задачи верификации железа и софта
  - ⋆ комбинаторные задачи вроде раскраски и гамильтонова цикла

Задача о гамильтоновом пути в форме SAT

Путь, который содержит все вершины графа (без повторений)

- HAMILTONIAN PATH: дан граф $G = (V, E)$, определить, есть ли в нем гамильтонов путь
  - при ответе «да» предъявить пример такого пути
- Опишем преобразование HAMILTONIAN PATH в SAT:
  - переменные: x_{ij}, $i, j \in V = \{1, \dots, n\}$
  - семантика: $x_{ij} = 1 \Leftrightarrow j$ — i-я вершина в гамильтоновом пути т.е. на i-ой позиции в пути находится j
- Клозы разбиваются на 5 групп:

определяют, что мы ищем именно гамильтонов цикл

  1. $x_{1j} \vee \dots \vee x_{nj}$ для всех $j = 1, \dots, n$
     - вершина j есть в гамильтоновом пути
  2. $\bar{x}_{ij} \vee \bar{x}_{kj}$ для всех $i, k, j = 1, \dots, n$, $i \neq k$ $= \overline{(x_{ij} \wedge x_{kj})}$
     - вершина j не входит в гамильтонов путь дважды
  3. $x_{i1} \vee \dots \vee x_{in}$ для всех $i = 1, \dots, n$ т.е. i-ое место чем-то занято
     - на i-ом месте в гамильтоновом пути стоит какая-то вершина
  4. $\bar{x}_{ij} \vee \bar{x}_{ik}$ для всех $i, k, j = 1, \dots, n$, $j \neq k$ $= \overline{(x_{ij} \wedge x_{ik})}$
     - на i-ом месте в гамильтоновом пути есть только одна вершина

опр. граф

  5. $\bar{x}_{ij} \vee \bar{x}_{(i+1)k}$ для всех $i = 1, \dots, n-1$, $k, j = 1, \dots, n$, $(j, k) \notin E$
     - соседние вершины в гамильтоновом пути должны быть соединены ребром
- если формула выполнима, переменные, равные 1, задают гамильтонов путь

Еще немного про SAT

- ★ SAT остается вычислительно трудной даже при ограничении, что задана константа $k \geqslant 3$ и каждый клоз содержит не более k литералов (задача k-SAT)
- ★ Общепринятая в настоящее время гипотеза экспоненциального времени утверждает существование констант $s_k > 0$ для любого $k \geqslant 3$ таких, что ни один алгоритм не может решить задачу k-SAT за время, меньшее $2^{s_k \ell}$
  - ⋆ фразу «не может решить» следует понимать так: для любого алгоритма найдется бесконечная серия «трудных» КНФ с разным числом клозов ℓ, на проверку выполнимости которых алгоритм затратит время $\Omega(2^{s_k \ell})$
- ★ Особенность SAT: трудные примеры встречаются редко
  - важную роль играет отношение числа клозов ℓ к числу переменных n
  - если клозов мало, обычно есть много выполняющих наборов и такой набор можно быстро найти
  - если клозов много, обычно формула невыполнима и противоречие находится быстро
  - на границе попадаются трудные формулы (либо выполнимые с очень малым числом выполняющих наборов, либо невыполнимые, но такие, что некоторые наборы выполняют почти все клозы)

Распространение переменной

Распространение переменной

Упрощает задачу, сокращает перебор

Распространение переменной

Распространение переменной

Упрощает задачу, сокращает перебор

! Надо организовать данные так, чтобы мы могли удалять литерал за константное время

- Пусть КНФ состоит из клозов $C_1, \dots, C_\ell$ и зависит от переменных $x_1, \dots, x_n$
  - можно считать, что КНФ F задана двумя массивами:
    - $L[1..2n]$: в $L[i]$ хранится список номеров клозов, в которые входит литерал x_i (при $i \leqslant n$) либо литерал $\bar{x}_{i-n}$ (при $i > n$)
    - $C[1..\ell]$: в $C[i]$ хранится список номеров литералов, которые входят в клоз C_i (номер $i > n$ означает литерал $\bar{x}_{i-n}$)
  - ⋆ при переводе КНФ в такую форму можно сразу отбросить клозы, содержащие два противоположных литерала одновременно
- Распространение переменной (unit propagation) — процедура упрощения КНФ
  - ⋆ Если в F есть клоз, состоящий из единственного литерала (x_i либо $\bar{x}_i$), то набор $(b_1, \dots, b_n)$ выполняет F только при условии, что b_i выполняет данный клоз
  - ⇒ значение b_i определено однозначно
    - пусть литерал равен x_i, т.е. $b_i = 1$; случай $b_i = 0$ аналогичен
  - ⇒ можно присвоить значение b_i и упростить формулу:
  - ♣ клозы, содержащие x_i, выполнены — их можно удалить
  - ♠ из клозов, содержащих $\bar{x}_i$, можно удалить этот литерал (он равен 0)
    - если получился пустой клоз, то F невыполнима
  - ⋆ можно создать очередь одноэлементных клозов
    - очередь пополняется при выполнении пункта (♠)
  - ⋆ распространение переменной выполняется в цикле, пока очередь непуста
- ⋆ КНФ $F \xrightarrow{\text{распространение переменной}}$ КНФ $UP(F)$
  - $UP(F) = 1$, если удалены все клозы
  - $UP(F) = 0$, если встретился пустой клоз
  - $UP(F)$ выполнима ⇔ F выполнима
- ★ Распространение переменной выполняется за время O(число литералов в F)

Распространение переменной (2)

Пример:

$F = (a \vee d) \wedge (c \vee d \vee \bar{a}) \wedge (\bar{b} \vee \bar{c} \vee \bar{d}) \wedge (\bar{a}) \wedge (a \vee b \vee \bar{c})$

- в очереди единственный клоз $\bar{a}$, достаем его
- ♣ удаляем клозы $\bar{a}$ и $c \vee d \vee \bar{a}$
- ♠ удаляем a из клозов $a \vee b \vee \bar{c}$ и $a \vee d$ (новый клоз d добавляем в очередь)
- текущий список клозов: $d, \bar{b} \vee \bar{c} \vee \bar{d}, b \vee \bar{c}$
- достаем клоз d из очереди
- ♣ удаляем клоз d
- ♠ удаляем $\bar{d}$ из клоза $\bar{b} \vee \bar{c} \vee \bar{d}$
- очередь пуста, получаем $UP(F) = (b \vee \bar{c}) \wedge (\bar{b} \vee \bar{c})$
- ⋆ $UP(F)$ можно выполнить, положив $c = 0$
- ⇒ набор $a = 0, c = 0, d = 1$ выполняет F (при любом b)

- Дополнение к распространению переменной: правило чистой переменной
  - ⋆ если в результате удаления клоза (♣) у литерала не осталось вхождений в формулу, то переменной этого литерала присваивается значение, превращающее этот литерал в 0
  - это позволяет выполнить все клозы, содержащие противоположный литерал, и тем самым упростить текущую КНФ
  - ⋆ когда в примере получено $UP(F) = (b \vee \bar{c}) \wedge (\bar{b} \vee \bar{c})$, литерал c не имеет вхождений, и присвоение $c = 0$ позволяет выполнить оба оставшихся клоза
- ⋆ В дальнейшем под $UP(F)$ мы понимаем формулу, полученную из F применением обоих правил

Процедура DPLL

Процедура DPLL

Рекурсивный алгоритм

- Процедура DPLL — это алгоритм оптимизированного перебора, решающий задачу SAT
  - основан на статьях Дэвиса–Патнема (1960) и Дэвиса–Логманна–Лавлэнда (1962)
- Пусть $DPLL(F)$ — булево значение, возвращаемое алгоритмом на входе F
- ⋆ Рекурсивная запись процедуры DPLL:
  - выбрать переменную x
  - вернуть $DPLL(F) = DPLL(UP(F \wedge x)) \vee DPLL(UP(F \wedge \bar{x}))$
- Комментарии:
  - ⋆ формулы F и $(F \wedge x) \vee (F \wedge \bar{x})$ эквивалентны
  - ⋆ алгоритм представляет вычисление деревом:
    - с каждым узлом связана «остаточная» формула, которую нужно выполнить;
    - некоторой переменной x остаточной формулы присваивается значение 1, формула упрощается распространением переменной и присваивается дочернему узлу
    - если формулу не удалось выполнить, вычисление возвращается в родительский узел и выполняется присвоение $x = 0$
  - ⋆ клоз x ($\bar{x}$) добавляется не к формуле, а сразу в очередь, чтобы запустить распространение переменной
- Скорость работы перебора зависит от эвристики выбора переменной x_i
  - пример эвристики: выбирается переменная с максимальным числом вхождений в клозы минимальной длины
  - цель — увеличить ресурс использования распространения переменной
- Используется много других оптимизаций для сокращения перебора
- ⋆ DPLL до сих пор лежит в основе многих SAT-решателей

Хорновская выполнимость

Хорновская выполнимость

- Существуют частные случаи задачи SAT, для которых существуют полиномиальные (и даже линейные) алгоритмы
- ★ КНФ называется **хорновской**, если каждый клоз содержит не более одной переменной без отрицания
  - Пример: $F = (\bar{a} \vee b \vee \bar{c}) \wedge (\bar{b} \vee \bar{d}) \wedge (a) \wedge (\bar{a} \vee d)$
- ★ Задача SAT с хорновской КНФ также называется хорновской (HornSAT)

Теорема

Задача HornSAT может быть решена за время $O(m)$, где m — число литералов в формуле.

быстрее - невозможно, т.к. за линейное читаем формулу

- Доказательство:
  - пусть F — хорновская КНФ
  - применим распространение переменной и вычислим $UP(F)$
    - как уже обсуждалось, это требует времени $O(m)$
  - если $UP(F) \in \{0, 1\}$ — мы уже получили ответ
  - иначе каждый клоз содержит хотя бы два литерала
  - ⇒ каждый клоз содержит литерал вида $\bar{x}$
  - ⇒ присвоим всем оставшимся переменным нули
  - ⇒ $UP(F)$ выполнима ⇒ F выполнима □

т.е. если не получили 0 => выполнима

- ★ Тем же способом решается SAT для **двойственных хорновских КНФ**, в которых каждый клоз содержит не более одного литерала с отрицанием

Хорновская выполнимость (2)

Пример 1:

$F = (a \vee \bar{b} \vee \bar{c}) \wedge (\bar{b} \vee \bar{c} \vee d) \wedge (\bar{d} \vee \bar{c}) \wedge (c) \wedge (\bar{d} \vee e) \wedge (\bar{a} \vee \bar{c} \vee d \vee \bar{e})$

- распространяем c: $a \vee \bar{b}, \bar{b} \vee d, \bar{d}, \bar{d} \vee e, \bar{a} \vee d \vee \bar{e}$
- распространяем $\bar{d}$: $a \vee \bar{b}, \bar{b}, \bar{a} \vee \bar{e}$
- распространяем $\bar{b}$: $\bar{a} \vee \bar{e}$
- присваиваем нули оставшимся переменным: $a = e = 0$
- ⇒ набор $a = 0, b = 0, c = 1, d = 0, e = 0$ выполняет F

Пример 2:

$F = (a \vee \bar{b} \vee \bar{c}) \wedge (b \vee \bar{c} \vee \bar{d}) \wedge (d \vee \bar{c}) \wedge (c) \wedge (\bar{d} \vee e) \wedge (\bar{a} \vee \bar{c} \vee \bar{d} \vee \bar{e})$

- распространяем c: $a \vee \bar{b}, b \vee \bar{d}, d, \bar{d} \vee e, \bar{a} \vee \bar{d} \vee \bar{e}$
- распространяем d: $a \vee \bar{b}, b, e, \bar{a} \vee \bar{e}$
- распространяем b: $a, e, \bar{a} \vee \bar{e}$
- распространяем e: $a, \bar{a}$
- распространяем a: □
- ⇒ F невыполнима

2 - выполнимость

2-выполнимость

Частный случай SAT, который можно решить за полиномиальное время

3-SAT — трудная

2-SAT — простая

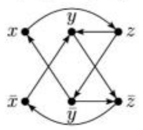
- КНФ, в которой каждый клоз состоит из двух литералов, называется **2-КНФ**
- ★ Задача SAT с 2-КНФ называется 2-выполнимость (2-SAT)
- ★ Формула $l_1 \vee l_2$, где l_1 и l_2 — литералы, эквивалентна $\bar{l_1} \to l_2$ и $\bar{l_2} \to l_1$
- Пусть дана 2-КНФ F; построим по ней орграф $G(F)$ (**граф импликаций**):
  - вершины — литералы из F
  - каждому клозу $l_1 \vee l_2$ сопоставлены ребра $(\bar{l_1}, l_2)$ и $(\bar{l_2}, l_1)$

Будет 2n вершин

- Эквивалентная формулировка 2-SAT на языке графа импликаций:
  - ★ существует ли раскраска ϕ графа импликаций в цвета $\{0, 1\}$ такая, что
    - (i) $\phi(l) \neq \phi(\bar{l})$ для любой вершины l и
    - (ii) $\phi(l_2) \geqslant \phi(l_1)$ для любого ребра (l_1, l_2)?
  - ϕ с указанными свойствами будем называть **булевой раскраской**
  - ◇ по транзитивности, если l_2 достижима из l_1, то $\phi(l_2) \geqslant \phi(l_1)$

Удалось раскрасить => выполнены все импликации => => выполнены все клозы

Пример: $F = (x \vee y) \wedge (\bar{y} \vee \bar{z}) \wedge (\bar{x} \vee z) \wedge (\bar{z} \vee y)$



граф импликаций $G(F)$:

2-выполнимость (2)

Лемма

Существует булева раскраска орграфа $G(F)$ ⇔ не существует переменной x, для которой вершины x и $\bar{x}$ взаимно достижимы в $G(F)$.

т.е. x и $\bar{x}$ лежат в одной компоненте сильной связности

- Доказательство необходимости:
  - существование такой переменной x влечет $\phi(x) = \phi(\bar{x})$ согласно (◇), что нарушает первое условие для булевой раскраски □

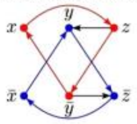
$\varphi(x) \le \varphi(\bar{x}) \wedge \varphi(\bar{x}) \le \varphi(x) \Downarrow$

- Доказательство достаточности:
  - разобьем $G(F)$ на **компоненты сильной связности**
  - отношение **достижимости** компонент — отношение порядка, дополним его до линейного порядка $\leqslant$
    - т.е. выполним топологическую сортировку компонент
  - по условию, вершины x и $\bar{x}$ лежат в разных компонентах для любой переменной x
  - ⇒ положим $\phi(x) = 1$ ($\phi(x) = 0$), если $\mathrm{comp}(x) > \mathrm{comp}(\bar{x})$ ($\mathrm{comp}(x) < \mathrm{comp}(\bar{x})$)
    - все вершины любой компоненты имеют один цвет
  - ⇒ $\phi(x) \neq \phi(\bar{x})$ для всех x, условие (i) выполнено
  - пусть существует ребро (l_1, l_2) такое, что $\phi(l_1) = 1, \phi(l_2) = 0$
  - ⇒ существует ребро $(\bar{l_2}, \bar{l_1})$, $\phi(\bar{l_2}) = 1, \phi(\bar{l_1}) = 0$
  - ⇒ $\mathrm{comp}(l_1) < \mathrm{comp}(l_2)$ и $\mathrm{comp}(\bar{l_2}) < \mathrm{comp}(\bar{l_1})$
  - из нашего определения ϕ следует $\mathrm{comp}(\bar{l_1}) < \mathrm{comp}(l_1)$ и $\mathrm{comp}(l_2) < \mathrm{comp}(\bar{l_2})$
  - ⇒ противоречие с тем, что $\leqslant$ — порядок
  - ⇒ $\phi(l_2) \geqslant \phi(l_1)$ для любого ребра (l_1, l_2), условие (ii) выполнено □

2-выполнимость (3)

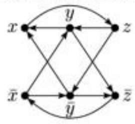
Примеры:

$F = (x \vee y) \wedge (\bar{y} \vee \bar{z}) \wedge (\bar{x} \vee z) \wedge (\bar{z} \vee y)$ выполнима:



В графе $G(F)$ две компоненты,
красные вершины красим в 0, синие — в 1

$F' = F \wedge (x \vee \bar{y})$ невыполнима:



В графе $G(F')$ единственная компонента,
ее нельзя раскрасить

Теорема

Задача 2-SAT может быть решена за время $O(\ell)$, где ℓ — число клозов в формуле.

- Доказательство:
  - построим по формуле F граф $G(F)$, в нем 2ℓ ребер
  - найдем компоненты сильной связности и отсортируем их топологически
    - например, и алгоритм Косараю, и алгоритм Тарьяна ищут компоненты за линейное от числа ребер время и выдают их в топологически отсортированном виде
  - если $\mathrm{comp}(x) = \mathrm{comp}(\bar{x})$ для какой-нибудь вершины x, возвращаем 0
  - иначе выполняем булеву раскраску $G(F)$ и возвращаем полученные значения
  - все шаги требуют времени $O(\ell)$ □