Алгоритмы для NP-трудных задач

Лекция 7: Алгоритмы расщепления

А. Куликов

Computer Science клуб при ПОМИ
http://logic.pdmi.ras.ru/∼infclub/

План лекции

1. Простой алгоритм для задачи выполнимости
2. Автоматические доказательства верхних оценок на время работы ал

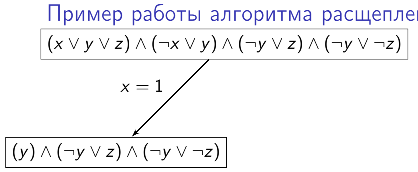
Метод расщепления

- Один из самых мощных и хорошо изученных методов для доказательства верхних оценок для NP-трудных задач.
- Рекордные теоретические верхние оценки для многих задач получены именно с помощью этого метода.
- В то же время для некоторых задач не дает ничего лучше полного перебора.
- Анализ алгоритмов, основанных на методе расщепления, как правило достаточно сложен (хотя и однообразен).
- Многие практические алгоритмы также основаны на методе расщепления.
- Мы будем рассматривать метод расщепления в применении к задачам выполнимости и максимальной выполнимости, хотя известно много таких алгоритмов и для других задач.

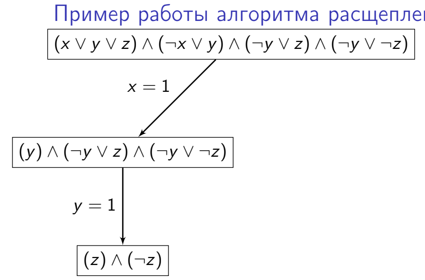
Пример работы алгоритма расщепления

$\boxed{(x \vee y \vee z) \wedge (\neg x \vee y) \wedge (\neg y \vee z) \wedge (\neg y \vee \neg z)}$

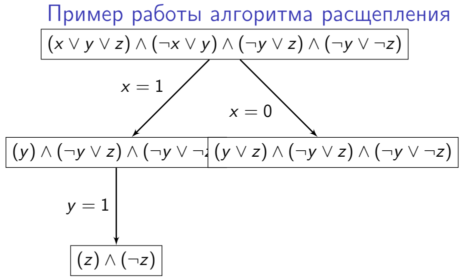
Пример работы алгоритма расщепления



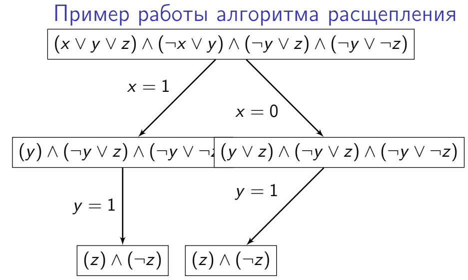
Пример работы алгоритма расщепления



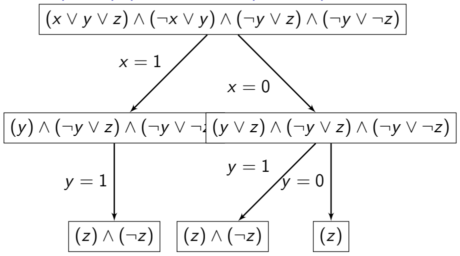
Пример работы алгоритма расщепления



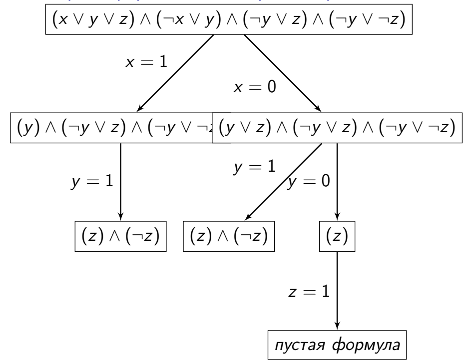
Пример работы алгоритма расщепления



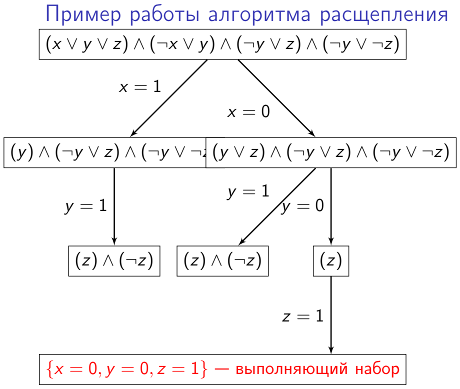
Пример работы алгоритма расщепления



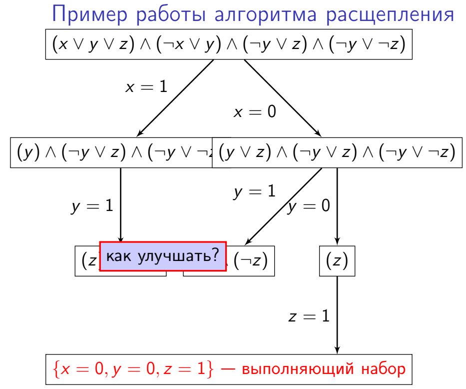
Пример работы алгоритма расщепления



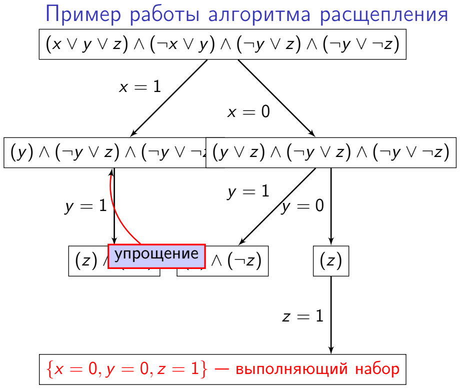
Пример работы алгоритма расщепления



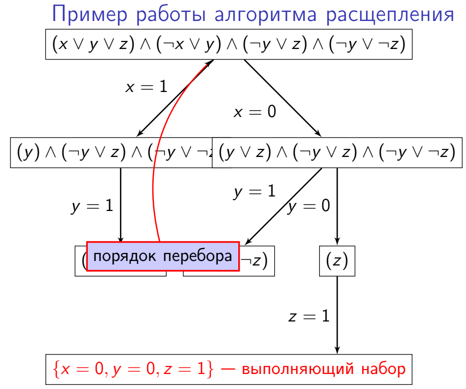
Пример работы алгоритма расщепления



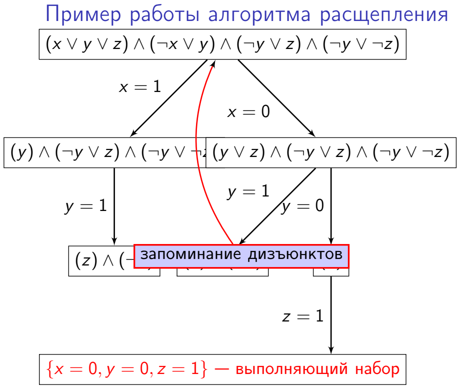
Пример работы алгоритма расщепления



Пример работы алгоритма расщепления



Пример работы алгоритма расщепления



План лекции

1. Простой алгоритм для задачи выполнимости
2. Автоматические доказательства верхних оценок на время работы а

Правила упрощения

Определение

К формуле F применимо правило упрощения (simplification/reduction rule), если по этому правилу F можно заменить за полиномиальное время на F', так что:

- сложность F' меньше сложности F,
- выполняющий набор для F можно построить из выполняющего набора для F' за полиномиальное время.

Единичный клоз

Единичный клоз

Если формула содержит единичный клоз (unit clause), то входящему в него литералу можно присвоить значение 1.

Пример

$F = (x \vee y \vee \neg z) \wedge (\neg x) \wedge (z \vee \neg y) \wedge (\neg y \vee \neg x \vee \neg z)$
$F[x = 0] = (y \vee \neg z) \wedge (z \vee \neg y)$

Чистый литерал

Чистый литерал

Если формула содержит чистый литерал (pure literal), т. е. литерал, отрицание которого не входит в формулу, то ему можно присвоить значение 1.

Пример

$F = (x \vee y \vee z) \wedge (\neg x) \wedge (x \vee \neg y) \wedge (\neg y \vee \neg x \vee z)$
$F[z = 1] = (\neg x) \wedge (x \vee \neg y)$

Резолюция

Резолюция

Если $F = F' \wedge (x \vee C) \wedge (\neg x \vee D)$ и F' не содержит ни x, ни $\neg x$, заменить F на $F' \wedge (C \vee D)$.

Пример

- $(x \vee \neg y \vee z) \wedge (\neg z) \wedge (z \vee u) \wedge (\neg u \vee \bar{x}) \wedge (y \vee z)$
  $\leftrightarrow (\neg y \vee z \vee \neg u) \wedge (\neg z) \wedge (z \vee u) \wedge (y \vee z)$
- $(x \vee y \vee \neg z) \wedge (\neg x) \wedge (x \vee \neg y) \wedge (\neg y \vee \neg x \vee z)$
  $\leftrightarrow (\neg x) \wedge (x \vee \neg y)$

Оценка времени работы алгоритмов расщепления

Пусть K — некоторая мера сложности формул в КНФ (к примеру, кол-во переменных или кол-во клозов).

Формально

Если алгоритм всегда расщепляет с рекуррентным неравенством вида

$$T(K) \leq T(K - t_1) + \cdots + T(K - t_j) + \mathrm{poly}(K),$$

то его время работы в худшем случае составляет

$$\mathrm{poly}(|F|)\alpha^{K(F)},$$

$\alpha = \tau(t_1, \ldots, t_j)$ — единственный положительный корень уравнения $x^{-t_1} + \cdots + x^{-t_j} = 1$.
$(t_1, \ldots, t_j)$ называется вектором расщепления, а $\alpha = \tau(t_1, \ldots, t_j)$ — числом расщепления.

Оценка времени работы алгоритмов расщепления

Неформально

Чем меньше сложности формул, на которые алгоритм расщепляет, тем быстрее алгоритм работает.

Пример

- $T(K) \leq T(K - 2) + T(K - 3) \leftrightarrow 1.325^K$
- $T(N) \leq 2 \cdot T(N - 6) + 2 \cdot T(N - 7) \leftrightarrow 1.239^N$

Алгоритм

Алгоритм

$\mathrm{DPLL\text{-}SAT}(F)$

- применять правила упрощения единичный клоз, чистый литерал и резолюция до тех пор, пока хотя бы одно из них применимо
- если F не содержит ни одного клоза, выдать "выполнима"
- если F содержит пустой клоз, выдать "невыполнима"
- если F содержит литерал, входящий в нее хотя бы два раза положительно и хотя бы два раза отрицательно, положить x равным этому литералу; в противном случае взять в качестве x любой литерал
- вернуть $(\mathrm{DPLL\text{-}SAT}(F[x = 1])$ or $\mathrm{DPLL\text{-}SAT}(F[x = 0]))$

Анализ алгоритма

Лемма

Время работы алгоритма $\mathrm{DPLL\text{-}SAT}$ не превосходит 1.415^K, где K — кол-во клозов входной формулы.

Доказательство

- достаточно показать, что в каждой ветке удаляется хотя бы два клоза: $\tau(2, 2) = \sqrt{2} = 1.414\ldots$
- это ясно, если нашелся литерал, который входит в формулу хотя бы дважды и положительно и отрицательно
- назовем (i, j)-литералом литерал, входящий в формулу ровно i раз положительно и ровно j раз отрицательно

Доказательство (продолжение)

Доказательство

- ясно, что упрощенная формула не содержит $(0,k)$- и $(k,0)$-литералов (это как раз чистые литералы), а также $(1,1)$-литералов (они удаляются резольвированием)
- значит, осталось рассмотреть случай, когда формула состоит только из $(1,2)$- и $(2,1)$-литералов

Доказательство (продолжение)

Доказательство

- рассмотрим произвольный литерал x:

$$F = (x \vee \dots) \wedge (x \vee \dots) \wedge (\neg x \vee \dots) \wedge \dots$$

- поскольку F не содержит единичных клозов, в клозе $(\neg x \vee \dots)$ есть еще хотя бы один литерал; назовем его y
- в $F[x=0]$ y будет $(2,0)$-, $(0,2)$- или $(1,1)$-литералом
- в любом из этих случаев y будет удален правилами упрощения, что повлечет за собой удаление хотя бы одного клоза
- итак, и $F[x=1]$, и $F[x=0]$ (после упрощения) содержат хотя бы на два клоза меньше, чем F

$\square$

План лекции

1. Простой алгоритм для задачи выполнимости
2. Автоматические доказательства верхних оценок на время работы ал

Анализ алгоритмов расщепления

- как правило, анализ расщепляющего алгоритма состоит из длинного списка случаев
- в каждом случае показывается, что входной пример можно разбить на несколько примеров, чьи размеры на нужную константу меньше, чем размер исходного примера
- доказательство для каждого случая представляет собой простое комбинаторное рассуждение
- такой разбор случаев можно проводить автоматически

Программа для автоматического доказательства верхних оценок для NP-трудных задач

Вход

- NP-трудная задача, сформулированная в терминах КНФ формул
- множество правил упрощения для данной задачи
- верхнюю оценку

По входным данным программа пытается найти

алгоритм расщепления для данной задачи, который использует данные правила упрощения и имеет время работы, удовлетворяющее данной оценке.

Пример входа

Допсутим, программа получила задание доказать, что существует алгоритм для выполнимости, который использует правила удаления единичных клозов и чистых литералов и имеет время работы не хуже 1.619^K, где K — количество клозов входной формулы.

Предобработка

- программа должна доказать, что для любой упрощенной формулы можно найти $(1,2)$-расщепление (то есть расщепление, которому соответствует рекуррентное неравенство $T(K) \leq T(K-1) + T(K-2) + \mathrm{poly}(K)$)
- каждый литерал упрощенной формулы входит в нее хотя бы один раз положительно и хотя бы один раз отрицательно (все остальные литералы являются чистыми и удаляются соответствующим правилом)
- более того, если есть литерал, который входит в формулу хотя бы дважды, то расщепление по нему дает требуемое неравенство:

$$\boxed{(x \vee a \vee \dots) \wedge (x \vee b \vee \dots) \wedge (\bar{x} \vee c \vee \dots) \wedge \dots}$$

$$\swarrow \qquad \searrow$$

$$\boxed{(c \vee \dots) \wedge \dots} \qquad \boxed{(a \vee \dots) \wedge (b \vee \dots) \wedge \dots}$$

Предобработка

Таким образом

Программе нужно доказать, что упрощенную формулу, состоящую только из $(1,1)$-литералов, всегда можно хорошо расщепить.

Автоматический разбор случаев

упрощенные формулы

Автоматический разбор случаев

упрощенные формулы

$(x \vee \ldots) \wedge (\bar{x} \vee \ldots) \wedge \ldots$

Автоматический разбор случаев

упрощенные формулы

$(x \vee \ldots) \wedge (\bar{x} \vee \ldots) \wedge \ldots$

$(x) \wedge (\bar{x} \vee \ldots) \wedge \ldots$

Автоматический разбор случаев

упрощенные формулы

$(x \vee \ldots) \wedge (\bar{x} \vee \ldots) \wedge \ldots$

$(x) \wedge (\bar{x} \vee \ldots) \wedge \ldots$ $y \in (x \vee \ldots)$

Автоматический разбор случаев

упрощенные формулы

$(x \vee \ldots) \wedge (\bar{x} \vee \ldots) \wedge \ldots$

есть единичный клоз $(x) \wedge (\bar{x} \vee \ldots) \wedge \ldots$ $y \in (x \vee \ldots)$

Автоматический разбор случаев

упрощенные формулы

$(x \vee \ldots) \wedge (\bar{x} \vee \ldots) \wedge \ldots$

есть единичный клоз $(x) \wedge (\bar{x} \vee \ldots) \wedge \ldots$ $y \in (x \vee \ldots)$

$(x \vee y \ldots) \wedge (\bar{x} \vee \bar{y} \ldots) \wedge \ldots$

Автоматический разбор случаев

упрощенные формулы

$(x \vee \ldots) \wedge (\bar{x} \vee \ldots) \wedge \ldots$

есть единичный клоз $(x) \wedge (\bar{x} \vee \ldots) \wedge \ldots$ $y \in (x \vee \ldots)$

$(x \vee y \ldots) \wedge (\bar{x} \vee \bar{y} \ldots) \wedge \ldots$

$(x \vee y \ldots) \wedge (\bar{x} \vee \ldots) \wedge (\bar{y} \vee \ldots) \wedge \ldots$

Автоматический разбор случаев

упрощенные формулы

$(x \vee \ldots) \wedge (\bar{x} \vee \ldots) \wedge \ldots$

есть единичный клоз $(x) \wedge (\bar{x} \vee \ldots) \wedge \ldots$ $y \in (x \vee \ldots)$

$(2, 2)$-расщепление по x $(x \vee y \ldots) \wedge (\bar{x} \vee \bar{y} \ldots) \wedge \ldots$

$(x \vee y \ldots) \wedge (\bar{x} \vee \ldots) \wedge (\bar{y} \vee \ldots) \wedge \ldots$

Автоматический разбор случаев