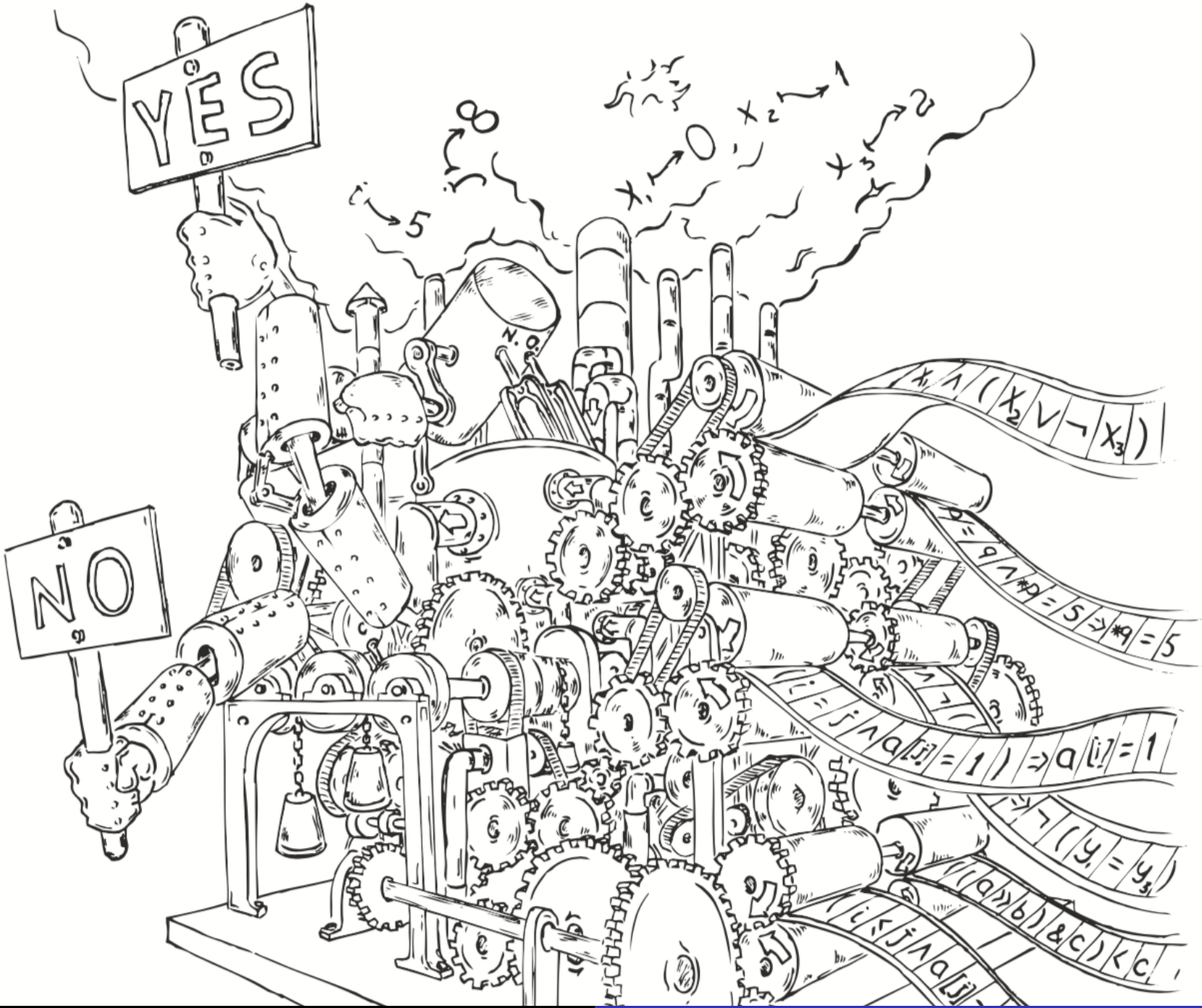

YES
NO

Задачи разрешимости логических формул и приложения
Лекция 1. Введение

Роман Холин

Московский государственный университет

Москва, 2021

О чём курс

- Что такое решатели
- Что такое SAT и SMT решатели и чем они отличаются
- Как ими пользоваться
- Как они устроены
- Где они используются

Где применяется?

- Формальная верификация
- Поиск ошибок
- Безопасность
- Биоинформатика
- Планирование расписаний
- Автоматическое доказательство теорем
- Генерация эксплоитов

Edmund Clarke: "a key technology of the 21st century"
Donald Knuth: "evidently a killer app, because it is key to the solution of so mane other problems"

Определения

Пусть дано множество переменных V, скобки и логические связки $\vee$, $\wedge$, $\rightarrow$, $\neg$. Определим булеву формулу по индукции:

- все переменные из V - формулы.
- если A - формула, то $\neg(A)$ - формула
- если A, B - формулы, то $(A) \vee (B)$, $(A) \wedge (B)$, $(A) \rightarrow (B)$ - также формулы.

Определения

- Если $v \in V$, то v и $\neg v$ называют литералами.
- Оценка формулы - присвоение каждой переменной значения "истина"или "ложь"и последующее вычисление значения формулы. Значение формулы вычисляется по индукции.

Определения

- Булева формула выполнима - если существует такая оценка, что значение формулы при ней "истина".
- Формула противоречива, если не существует такой оценки.
- Формула является тавтологией, если при любой оценки она "истина".

Определения

- Конъюктивная нормальная форма - форма записи булевой формулы, при которой эта формула имеет вид конъюкции дизъюнкций литералов.
- Дизъюнктивная нормальная форма - наоборот.

SAT задача

- На вход подается булева формула, содержащую только нумерованные переменные, $\vee$, $\wedge$, $\neg$. Является ли формула выполнимой?

SAT задача

- Если дизъюнкты в формуле имеет длину не более 2, то сложность задачи распознавания принадлежит классу P.
- Иначе, это NP-полная задача (теорема Кука-Левина). Более того, исторически, это первая задача, чья NP-полнота была доказана.
- Что для нас значит, что задача NP-полна? Это значит, что если мы каким-то образом научимся решать такую задачу "быстро то мы сможем "быстро"решать класс NP задач.

Примеры

- Вы глава протокола на ужине для иностранных послов в некотором королевстве. Принц хочет, чтобы либо был посол из Перу, либо не было посла Катара. Королева хочет видить послов Катара или Румынии. Король не хочет видеть послов из Румынии или Перу. Кого же позвать на ужин?

Примеры

- Вы глава протокола на ужине для иностранных послов в некотором королевстве. Принц хочет, чтобы либо был посол из Перу, либо не было посла Катара. Королева хочет видить послов Катара или Румынии. Король не хочет видеть послов из Румынии или Перу. Кого же позвать на ужин?
- $(\neg p \vee q) \wedge (q \vee r) \wedge (\neg r \vee \neg q)$

Примеры

- Пусть множество натуральных чисел разбили на конечное количество непересекающихся множеств. Есть ли хотя бы одно множество, которое содержит тройку чисел, которая является Пифагоровой? Например, пусть натуральные числа разбили на два множества: множество четных и множество нечетных чисел. Тогда очевидно, что множество нечетных чисел не содержит Пифагоровой тройки.

Примеры

- Давайте упростим задачу: давайте будем делить множество натуральных чисел только на две части. Тогда ответ на задачу положительный. Достаточно представить подмножество натуральных чисел, которое нельзя разбить на 2 части так, чтобы не в одном из них не было пифагоровой тройки. С помощью SAT решателя можно выяснить, что наименьшее такое N, что множество $\{1, \ldots, N\}$ нельзя разбить на 2 множества, равно 7825.

Примеры

- А что, если мы разбиваем на три множества? Можно показать, что множество $\{1, \ldots, 10^7\}$ можно разбить.

Пример

- Зачем разобрали этот пример? Потому что решение этой задачи сродни тому, как мы решаем задачи верификации и поиска ошибок. После публикации "Symbolic Model Checking without BDDs"1999 рост интереса людей, занимающиеся верификацией и поском ошибок к решателям сильно возрос.
- Поиск ошибок - это поиск контрпримера.
- Верификация - доказательство того, что контрпримера нет.
- Сейчас верификаторы и сканеры ошибок - основные "потребители"SAT и SMT решателей.

Проверка кода на эквивалентность

```
if(!a && !b) h();
else if(!a) g();
else f();

if(a) f();
else if(b) g();
else h();
```

Проверка кода на эквивалентность

Представим процедуры как будевские переменные:
if $\neg a \wedge \neg b$ then h
else if $\neg a$ then g
else f

if a then f
else if b then g
else h

Проверка кода на эквивалентность

Представим процедуры как будевские переменные:
if $\neg a \wedge \neg b$ then h
else if $\neg a$ then g
else f

if a then f
else if b then g
else h
Скомпилируем код в КНФ:
compile(if x then y else z) $\equiv (\neg x \vee y) \wedge (x \vee z)$

Проверка кода на эквивалентность

Скомпилируем код в КНФ:

compile(if x then y else z) $\equiv (\neg x \vee y) \wedge (x \vee z)$

if $\neg a \wedge \neg b$ then h else if $\neg a$ then g else f $\equiv$

$(\neg(\neg a \wedge \neg b) \vee h) \wedge ((\neg a \wedge \neg b) \vee$ (if $\neg a$ then g else f)) $\equiv$

$(a \vee b \vee h) \wedge ((\neg a \wedge \neg b) \vee ((a \vee g) \wedge (\neg a \vee f)))$

Проверка кода на эквивалентность

Скомпилируем код в КНФ:

compile(if x then y else z) $\equiv (\neg x \vee y) \wedge (x \vee z)$

if $\neg a \wedge \neg b$ then h else if $\neg a$ then g else f $\equiv$

$(\neg(\neg a \wedge \neg b) \vee h) \wedge ((\neg a \wedge \neg b) \vee$ (if $\neg a$ then g else f)) $\equiv$

$(a \vee b \vee h) \wedge ((\neg a \wedge \neg b) \vee ((a \vee g) \wedge (\neg a \vee f)))$

if a then f else if b then g else h $\equiv$

$(\neg a \vee f) \wedge (a \vee$(if b then g else h)) $\equiv$

$(\neg a \vee f) \wedge (a \vee ((\neg b \vee g) \wedge (b \vee h)))$

SMT решатель

- Что, если нам хочется задавать более сложные вопросы решателю? Например, разрешима ли такая формула: $(x = 4) \wedge ((y = 7) \vee (x = y))$
- Или такая: $(x + y = 3) \wedge (y - z = 7) \wedge (z * 2 = 4)$
- Или такая: $(lenght(s) = 3) \wedge (s[0] =' a') \wedge (s[1] =' b') \wedge (s[2] =' c')$

SMT решатель

Пропозиционной логики для этого не достаточно. Для описания таких систем используется логика первого порядки и теории.

- равенства
- равенства и неинтерпритируемых функций
- линейной арифметики
- векторы битов
- массивов

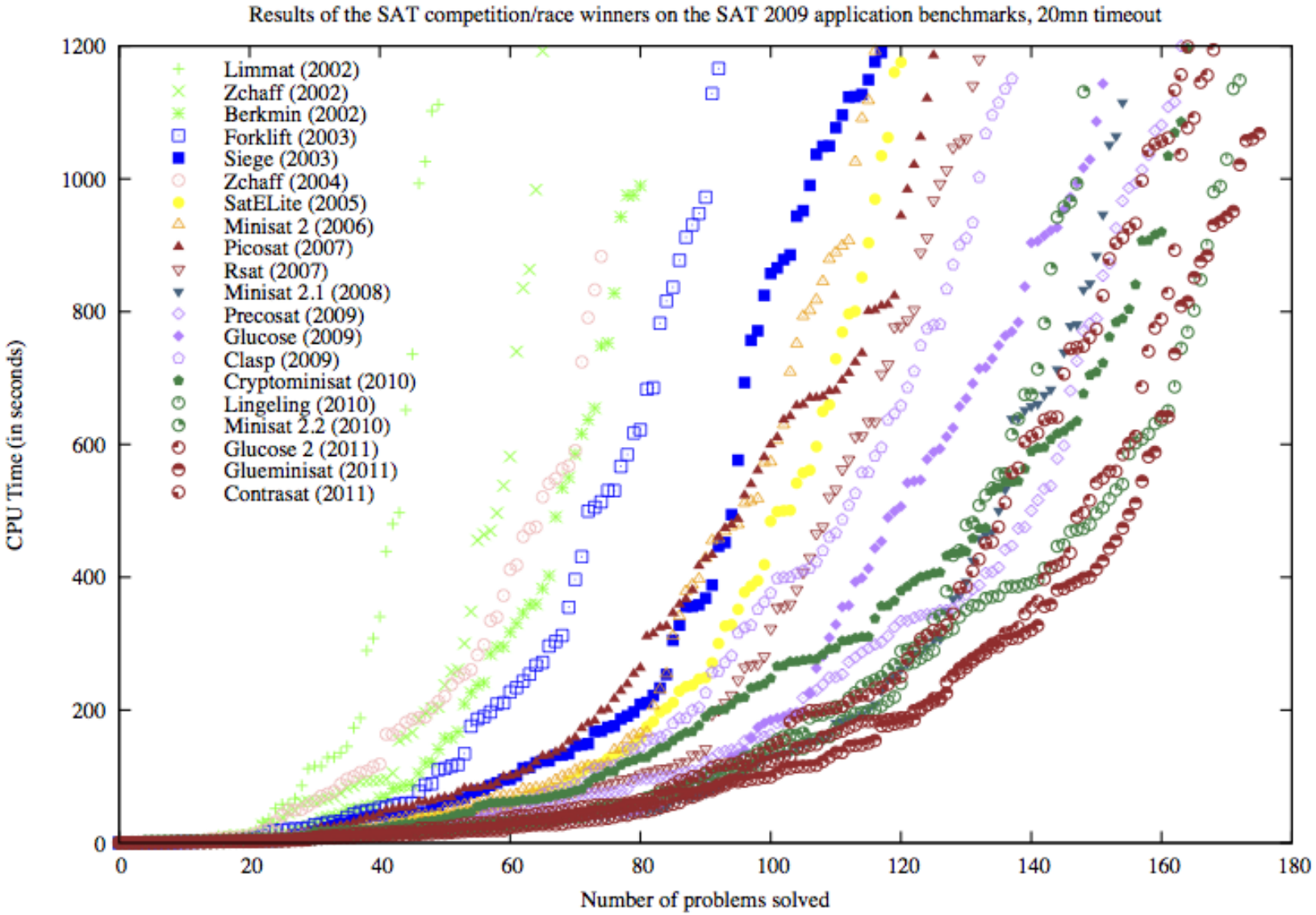
Большая часть SMT решателей использует SAT решатели.

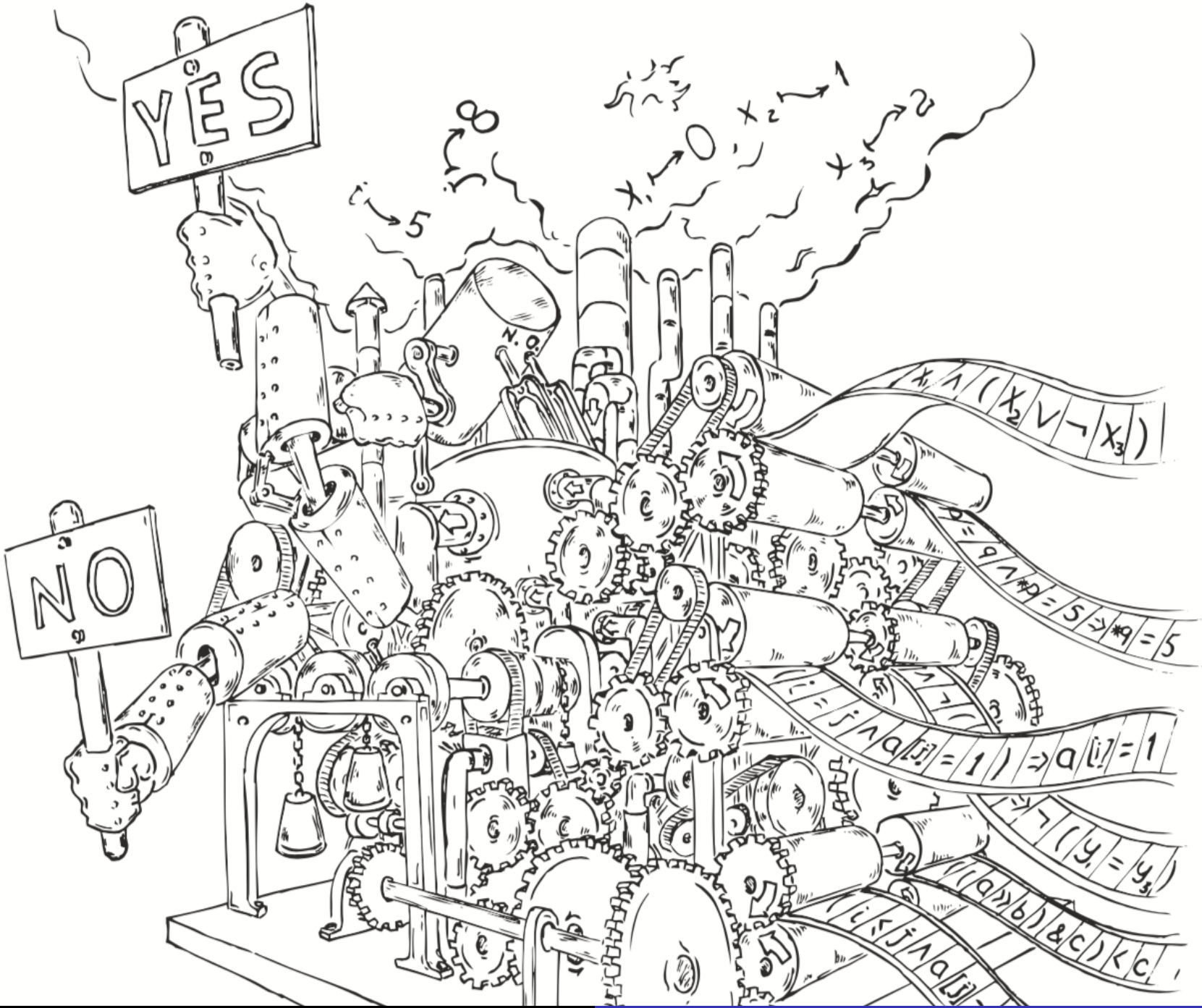
SMT-lib

- Специальный list-подобный язык для записи уравнений, подающийся на вход решателю.

Listing 3. Integer Example in SMT-LIB

```
(set-logic QF_LIA)
(declare-fun x () Int)
(declare-fun y () Int)
(declare-fun z () Int)
(assert (= (+ (* 6 x) (* 2 y) (* 12 z)) 30))
(assert (= (+ (* 3 x) (* 6 y) (* 3 z)) 12))
(check-sat)
```


Results of the SAT competition/race winners on the SAT 2009 application benchmarks, 20mn timeout
CPU Time (in seconds)
Number of problems solved
0
200
400
600
800
1000
1200
0
20
40
60
80
100
120
140
160
180
Limmat (2002)
Zchaff (2002)
Berkmin (2002)
Forklift (2003)
Siege (2003)
Zchaff (2004)
SatELite (2005)
Minisat 2 (2006)
Picosat (2007)
Rsat (2007)
Minisat 2.1 (2008)
Precosat (2009)
Glucose (2009)
Clasp (2009)
Cryptominisat (2010)
Lingeling (2010)
Minisat 2.2 (2010)
Glucose 2 (2011)
Glueminisat (2011)
Contrasat (2011)


YES
NO