JavaScript на службе у теоретической информатики

Виталий Брагилевский

9 ноября 2019

HolyJS 2019

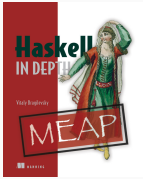
Eto ya

JetBrains
(Санкт-Петербург)

Санкт-Петербургский
государственный
университет

Факультет математики
и компьютерных наук

Glasgow Haskell Compiler
Steering Committee



Haskell in Depth

Мы займёмся теорией вычислений!

- что такое вычисление?
- всё ли можно вычислить в принципе?
- что можно вычислить за «разумное» время и в «разумном» объёме памяти?

Обязательно ли JS-разработчику знать теорию вычислений?

Обязательно ли JS-разработчику знать теорию вычислений?

НЕТ

Инструментарий теории вычислений

- решение задач
- модели вычислений
- сведения
- классы задач

Инструментарий JavaScript-разработчика

Код на github.com (Node.js 13.0.1):
bravit/talks-msk-holyjs-2019-code

Содержание доклада

Задачи и языки

Модели вычислений и тезис Чёрча—Тьюринга

Неразрешимость и теорема Райса

Классы сложности и NP-полнота

Задачи и языки

Пример задачи: проверка чисел на простоту

Постановка задачи

Дано целое число n. Проверить, является ли оно простым.

Пример задачи: проверка чисел на простоту

Постановка задачи

Дано целое число n. Проверить, является ли оно простым.

Пример решения

```
function isPrime(n) {
    // ...
}
```

Пример задачи: проверка чисел на простоту

Постановка задачи

Дано целое число n. Проверить, является ли оно простым.

Пример решения

```
function isPrime(n) {
   // ...
}

isPrime(3); // true
isPrime(5); // true
isPrime(6); // false
```

Ограничение №1: решаем только задачи разрешения

- Задача разрешения (*decision problem*) — это задача с ответом да/нет

Ограничение №1: решаем только задачи разрешения

- Задача разрешения (*decision problem*) — это задача с ответом да/нет

```
const Answer = Object.freeze({
    YES: Symbol("YES"),
    NO:  Symbol("NO")
});
```

```
class DecisionProblem  {
    constructor(name) { this.name = name }
    solve(instance) { return undefined }
    answer(instance) {
        return this.solve(instance)
                ? Answer.YES
                : Answer.NO
    }
}
```

Пример: проверка чисел на простоту

`js-for-tcs-01`

Работать с числами сложновато…

Работать с числами сложновато…

- Символы

Работать с числами сложновато…

- Символы
- Алфавит

Работать с числами сложновато…

- Символы
- Алфавит
- Слово

Работать с числами сложновато…

- Символы
- Алфавит
- Слово
- Язык

Работать с числами сложновато...

- Символы
- Алфавит
- Слово
- Язык

Примеры

- $A_1 = \{|\}, L_1 = \{||, |||, |||||, |||||||, \dots\}$

Работать с числами сложновато...

- Символы
- Алфавит
- Слово
- Язык

Примеры

- $A_1 = \{|\}$, $L_1 = \{||, |||, |||||, |||||||, \dots\}$
- $A_2 = \{0, 1\}$, $L_2 = \{0, 10, 100, 110, 1000, 1010, \dots\}$

Работать с числами сложновато...

- Символы
- Алфавит
- Слово
- Язык

Примеры

- $A_1 = \{|\}$, $L_1 = \{||, |||, |||||, |||||||, \dots\}$
- $A_2 = \{0, 1\}$, $L_2 = \{0, 10, 100, 110, 1000, 1010, \dots\}$
- $A_3 = \{c - \text{ ASCII-символ}\}$, $L_3 = \{\texttt{\0A}, \texttt{\0D}, \texttt{\0D\0A}\}$

Ограничение №2: решаем задачи распознавания

Даны язык и слово.

Принадлежит ли слово языку?

Ограничение №2: решаем задачи распознавания

Даны язык и слово.

Принадлежит ли слово языку?

Язык = Задача

Как бы представить язык в JavaScript…

Пример: генератор для языка натуральных чисел

```
function* natWords() {
    let w="";
    while(true) {
        w += "|";
        yield w;
    }
}
```

Пример: генератор для языка простых чисел

```
function* primeWords() {
    let i = 1;
    while(true) {
        ++i;
        if (!isPrime(i)) {
            continue
        }
        yield "|".repeat(i)
    }
}
```

Представление и распознавание языков в JS

```
class Lang extends DecisionProblem {
    constructor(gen, name) {
        super(name);
        this.gen = gen;
    }
    solve(instance) {
      // ...
    }
}
```

```
solve(instance) {
    for(let word of this.gen()) {
        if (word === instance)
            return true;
        // ГРЯЗНЫЙ ХАК:
        if (instance.length < word.length)
            return false
    }
    return false
}
```

Пример: распознавание языков как генераторов

`js-for-tcs-02`

Не слишком ли это сильные ограничения?

Не слишком ли это сильные ограничения?

Задача: вычислить четвёртое простое число.

Ограничение: можно решать только задачу распознавания.

Не слишком ли это сильные ограничения?

Задача: вычислить четвёртое простое число.

Ограничение: можно решать только задачу распознавания.

Алфавит: $\{|, \#\}$.

Не слишком ли это сильные ограничения?

Задача: вычислить четвёртое простое число.

Ограничение: можно решать только задачу распознавания.

Алфавит: $\{|, \#\}$.

Язык: $\{|\#||$

Не слишком ли это сильные ограничения?

Задача: вычислить четвёртое простое число.

Ограничение: можно решать только задачу распознавания.

Алфавит: $\{|, \#\}$.

Язык: $\{|\#||, ||\#|||$

Не слишком ли это сильные ограничения?

Задача: вычислить четвёртое простое число.

Ограничение: можно решать только задачу распознавания.

Алфавит: $\{|, \#\}$.

Язык: $\{|\#||, ||\#|||, |||\#|||||$

Не слишком ли это сильные ограничения?

Задача: вычислить четвёртое простое число.

Ограничение: можно решать только задачу распознавания.

Алфавит: $\{|, \#\}$.

Язык: $\{|\#||, ||\#|||, |||\#|||||, ||||\#|||||||$

Не слишком ли это сильные ограничения?

Задача: вычислить четвёртое простое число.

Ограничение: можно решать только задачу распознавания.

Алфавит: $\{|, \#\}$.

Язык: $\{|\#||, ||\#|||, |||\#|||||, ||||\#|||||||, |||||\#||||||||||$

Не слишком ли это сильные ограничения?

Задача: вычислить четвёртое простое число.

Ограничение: можно решать только задачу распознавания.

Алфавит: $\{|, \#\}$.

Язык: $\{|\#||, ||\#|||, |||\#|||||, ||||\#|||||||, |||||\#||||||||||, \dots\}$

Решение: строим слово $||||\#|$

Решение: строим слово `||||#|` и решаем для него задачу распознавания, добавляя в конец по `|`, пока не получим «ДА»:

```
||||#| NO
||||#|| NO
||||#||| NO
||||#|||| NO
||||#||||| NO
||||#|||||| NO
||||#||||||| YES
```

Решение: строим слово `||||#|` и решаем для него задачу распознавания, добавляя в конец по `|`, пока не получим «ДА»:

```
||||#| NO
||||#|| NO
||||#||| NO
||||#|||| NO
||||#||||| NO
||||#|||||| NO
||||#||||||| YES
```

Ответ: `|||||||`.

Пример: четвёртое простое число

`js-for-tcs-03`

С задачами всё ясно.
Вопрос: как вычислять?

Модели вычислений и тезис Чёрча—Тьюринга

Алонсо Чёрч (1903–1995)

λ-исчисление (1930-е)

Переменные: x, y, z…

Применение: MN

Абстракция: $\lambda x.M$

λ-исчисление (1930-е)

Переменные: x, y, z…

Применение: MN

Абстракция: $\lambda x.M$

Функция

λ-исчисление (1930-е)

Переменные: x, y, z…

Применение: MN

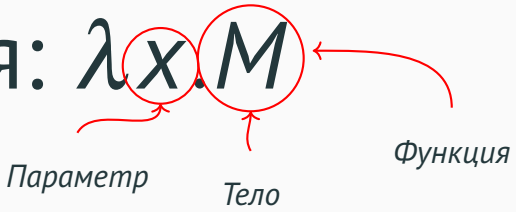
Абстракция: $\lambda x.M$

λ-исчисление (1930-е)

Переменные: x, y, z…

Применение: MN

Абстракция: $\lambda x.M$

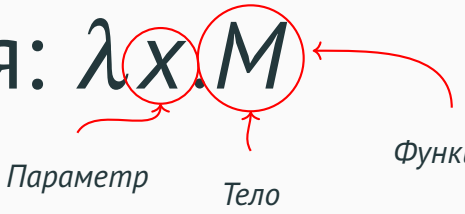


λ-исчисление (1930-е)

Переменные: x, y, z…

Применение: MN ← *Вызов функции*

Абстракция: $\lambda x.M$



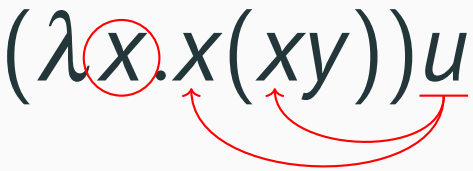
Пример λ-терма и его вычисление

$$(\lambda x.x(xy))u$$

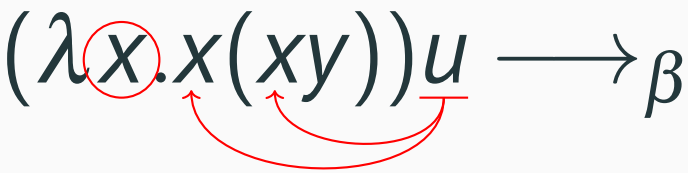
Пример λ-терма и его вычисление

$$(\lambda x.x(xy))u$$

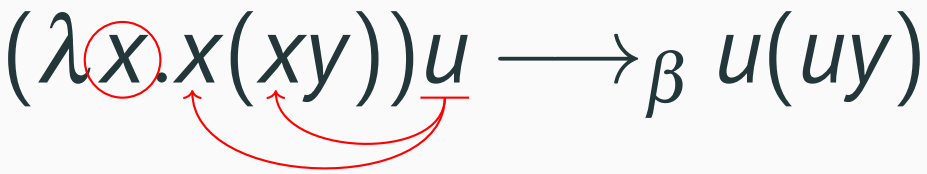
Пример λ-терма и его вычисление



Пример λ-терма и его вычисление



Пример λ-терма и его вычисление



Пример λ-терма и его вычисление

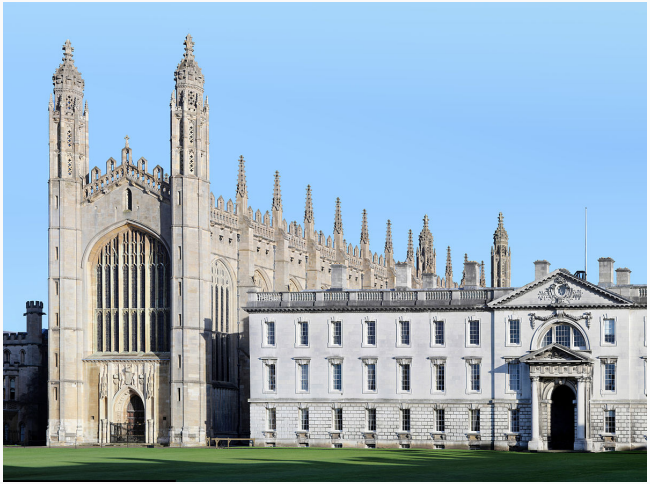
$$(\lambda x.x(xy))u \longrightarrow_\beta u(uy)$$

```
(x=>x(x(y)))(u)
```

Дальше там много интересного!

http://bit.ly/lucas-lambda

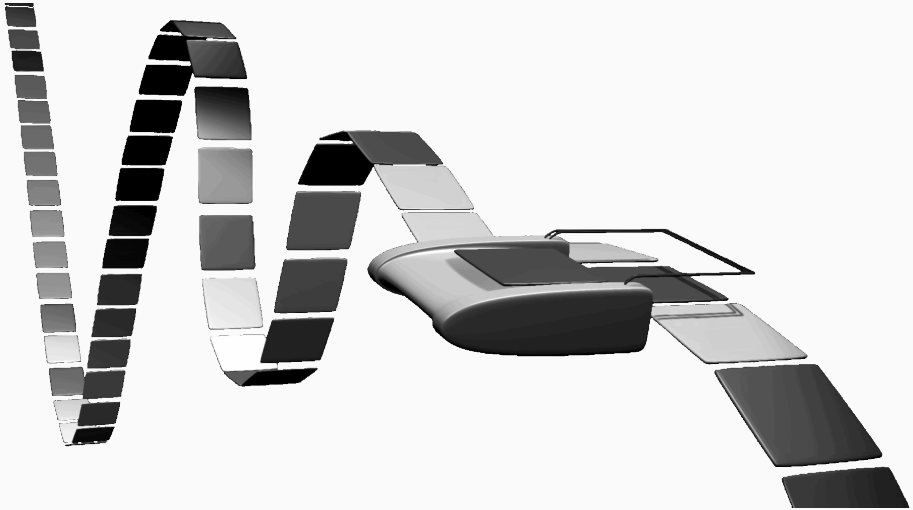
А в это время в кембриджском университете...



Осенью 1935 г.
Алан Тьюринг
слушает курс
по матлогике
и делает домашку

Алан Тьюринг (1912–1954)

Машина Тьюринга



Машина Тьюринга: интерфейс ленты

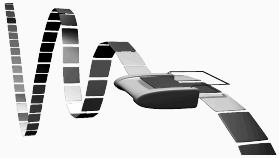
```
class Tape {
    constructor(inputWord="", blank='_') { }
    read() { }
    write(symbol) { }
    moveLeft() { }
    moveRight() { }
}
```

Машина Тьюринга: состояния

```
class State {
    constructor(numberOfStates, finalState) { }
    set(n) { }
    get() { }
    isFinal() { }
}
```

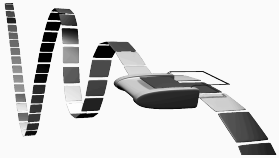
Вычисление на машине Тьюринга: один шаг

- Читаем символ с ленты



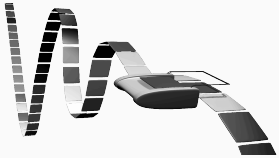
Вычисление на машине Тьюринга: один шаг

- Читаем символ с ленты
- Смотрим текущее состояние



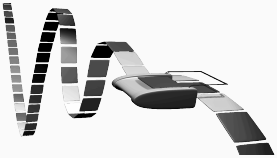
Вычисление на машине Тьюринга: один шаг

- Читаем символ с ленты
- Смотрим текущее состояние
- Делаем переход:
  - пишем символ
  - сдвигаем считывающее устройство
  - переходим в новое состояние



Работа машины Тьюринга в целом

- Начальное состояние
- Входное слово
- Машина завершает работу, если нет переходов
- Финальные (допускающие) состояния: если МТ остановилась в одном из них, то ответ «ДА»



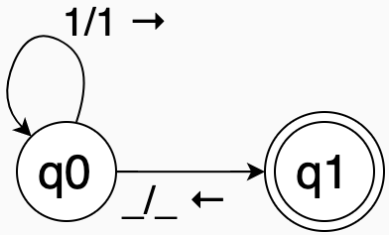
Интерфейс машины Тьюринга

```
class TuringMachine {
    constructor(state, program) { }
    run(inputWord) {
        this.reset(inputWord);
        while (!this.stopped) {
            this.step()
        }
        return this.state.isFinal()
    }
}
```

Язык машины Тьюринга

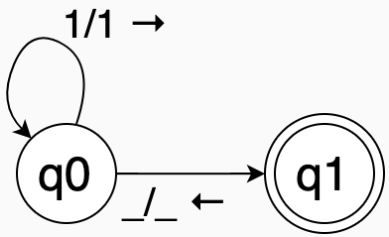
```
class TMLang extends DecisionProblem {
    constructor(tm, name) {
        super(name);
        this.tm = tm
    }
    solve(instance) {
        return this.tm.run(instance)
    }
}
```

Пример: распознавание языка из единичек




```
('1', 0) => ('1', 0, ->)
('_', 0) => ('_' ,1, <-)
```

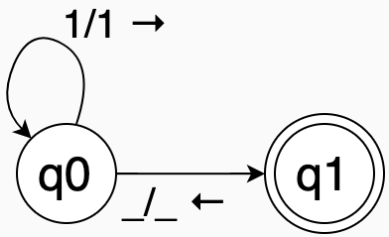
Пример: распознавание языка из единичек




```
('1', 0) => ('1', 0, ->)
('_', 0) => ('_' ,1, <-)
```

```
111
^
111
 ^
111
  ^
111_
   ^
111_
  ^
YES
```

Пример: распознавание языка из единичек




```
('1', 0) => ('1', 0, ->)
('_', 0) => ('_' ,1, <-)
```

```
111
^
111
 ^
111
  ^
111_
   ^
111_
  ^
YES
```

```
101
^
101
 ^
NO
```

Пример: распознавание языка из единичек

`js-for-tcs-04`

Бывают и другие модели вычислений!

Программа «Hello World» на языке Brainfuck (1993)

```
++++++++[>++++[>++>+++>+++>+<<<<-]>+>+>->>+[<]<-]>>.>
---.+++++++..+++.>>.<-.<.+++.------.--------.>>+.>++.
```

Бывают и другие модели вычислений!

Программа «Hello World» на языке Brainfuck (1993)

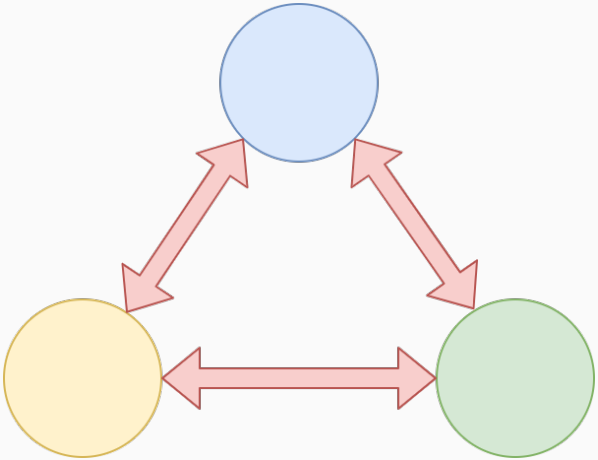
```
++++++++[>++++[>++>+++>+++>+<<<<-]>+>+>->>+[<]<-]>>.>
---.+++++++..+++.>>.<-.<.+++.------.--------.>>+.>++.
```

- Модель $\mathscr{P}''$ Коррадо Бёма (1964)

Эквивалентность моделей вычислений

- λ-исчисление (1928)
- Теория рекурсивных функций Курта Гёделя (начало 1930-х)
- Машина Тьюринга (1936)
- Машина Поста (1936)
- Нормальные алгорифмы А. А. Маркова-мл. (1954 г.)
- Модель $\mathscr{P}''$ Коррадо Бёма (1964)
- Квантовые вычисления (1990-е)
- …

Доказательство эквивалентности моделей



Тезис Чёрча—Тьюринга

Всё, что в принципе можно вычислить, можно вычислить с помощью одной из этих моделей.

Остаётся два вопроса:

- всё ли можно вычислить?
- как разделить вычисления по требуемым ресурсам?

Неразрешимость и теорема Райса

Домашнее задание

@babel/how-to

Написать плагин к Babel, который определяет, **завершается ли заданная программа на любых входных данных**, и если нет, то выдаёт соответствующее предупреждение.

Проблема останова/Halting problem

Постановка задачи

По заданному описанию компьютерной программы и входным данным определить, завершится ли её выполнение.

Проблема останова/Halting problem

Постановка задачи

По заданному описанию компьютерной программы и входным данным определить, завершится ли её выполнение.

- Это первый пример неразрешимой задачи.

Проблема останова/Halting problem

Постановка задачи

По заданному описанию компьютерной программы и входным данным определить, завершится ли её выполнение.

- Это первый пример неразрешимой задачи.
- Доказательство первым построил Тьюринг.

Проблема останова/Halting problem

Постановка задачи

По заданному описанию компьютерной программы и входным данным определить, завершится ли её выполнение.

- Это первый пример неразрешимой задачи.
- Доказательство первым построил Тьюринг.
- Доказательство неразрешимости многих других задач выполняется путём сведения к ним проблемы останова.

Сведение?

```
class EvenLengthProblem extends DecisionProblem {
    solve(instance) {
        return instance.length % 2 === 0
    }
}
class EvenNumberProblem extends DecisionProblem {
    solve(instance) {
        return instance % 2 === 0
    }
}
```

Сведение!

```
function evenLength2evenNumber(from) {
    return from.length
}
```

Сведение!

```
function evenLength2evenNumber(from) {
    return from.length
}
```

Проверка корректности сведения

```
import { strict as assert } from 'assert';
function testReduction(pr1, pr2, reduce, inst1) {
    let instance2 = reduce(inst1);
    assert.ok(pr1.answer(inst1) ===
              pr2.answer(instance2))
}
```

Пример: сведения и задача-дополнение

`js-for-tcs-05`

Теорема Райса

Всякое нетривиальное семантическое свойство компьютерных программ алгоритмически неразрешимо.

Теорема Райса

Всякое нетривиальное семантическое свойство компьютерных программ алгоритмически неразрешимо.

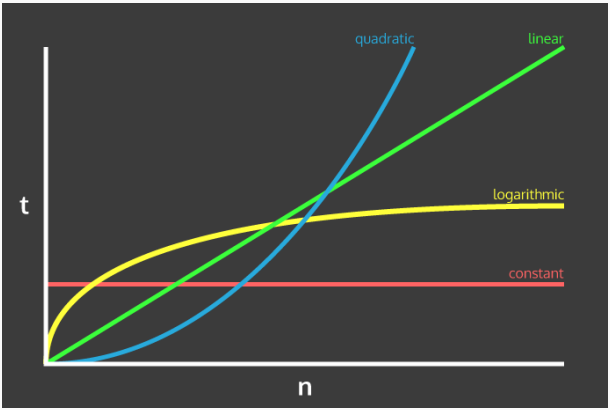
Пример нетривиального семантического свойства: «эта программа в процессе работы печатает единичку».

Классы сложности и NP-полнота

Временная сложность и «большое О»

Tim Roberts, Time Complexity/Big O Notation

https://medium.com/javascript-scene/time-complexity-big-o-notation-1a4310c3ee4b



А нам всё равно!

Класс сложности P

Класс **P** содержит все задачи, которые можно решить (все языки, которые можно распознать) за *полиномиальное* время.

А нам всё равно!

Класс сложности P

Класс **P** содержит все задачи, которые можно решить (все языки, которые можно распознать) за *полиномиальное* время.

Класс сложности EXP

Класс **EXP** содержит все задачи, которые можно решить (все языки, которые можно распознать) за *экспоненциальное* время.

Однако есть классы поинтереснее!

Класс сложности NP

Класс **NP** содержит все задачи, корректность ответа «ДА» для которых *можно проверить* за полиномиальное время (при наличии *сертификата* решения).

Пример: задача о сумме элементов подмножества

Задача. Дано множество целых чисел A и целое число T. Существует ли такое подмножество A, сумма элементов которого равна T?

Пример: задача о сумме элементов подмножества

Задача. Дано множество целых чисел A и целое число T. Существует ли такое подмножество A, сумма элементов которого равна T?

Пример: $A = \{1, 3, 5\}$, $T = 8$. Ответ: «ДА».

Пример: задача о сумме элементов подмножества

Задача. Дано множество целых чисел A и целое число T. Существует ли такое подмножество A, сумма элементов которого равна T?

Пример: $A = \{1, 3, 5\}$, $T = 8$. Ответ: «ДА».

Сертификат решения: подмножество $A' = \{3, 5\}$.

Пример: задача о сумме элементов подмножества

Задача. Дано множество целых чисел A и целое число T. Существует ли такое подмножество A, сумма элементов которого равна T?

Пример: $A = \{1, 3, 5\}$, $T = 8$. Ответ: «ДА».

Сертификат решения: подмножество $A' = \{3, 5\}$.

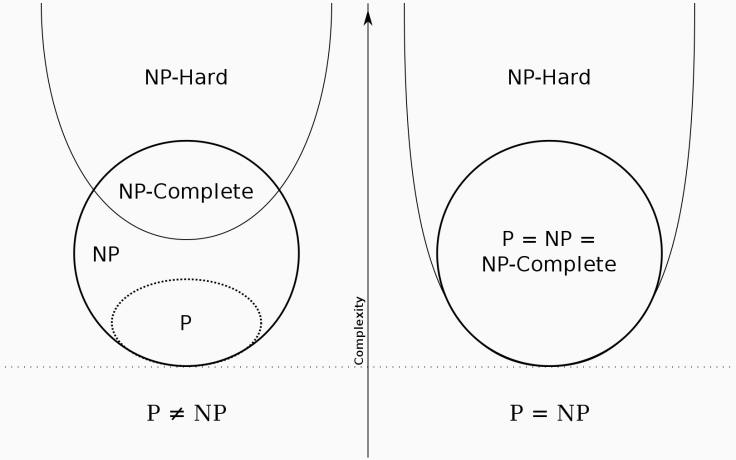
Проверка: $3 + 5 = 8 = T$ — полиномиальное время!

NP-трудные и NP-полные задачи

Задача называется *NP-трудной*, если к ней *сводится* любая задача из класса NP.

NP-трудная задача называется *NP-полной*, если она принадлежит классу NP.

Открытая проблема: P=?NP



Credit: Behnam Esfahbod

Пример NP-полной задачи: задача SAT

Булевы формулы и выполнимость

- Булева формула содержит переменные и логические связки (отрицания, конъюнкции и дизъюнкции).
- Булева формула называется выполнимой, если существует подстановка переменных, которая обращает её в истину.

Пример NP-полной задачи: задача SAT

Булевы формулы и выполнимость

- Булева формула содержит переменные и логические связки (отрицания, конъюнкции и дизъюнкции).
- Булева формула называется выполнимой, если существует подстановка переменных, которая обращает её в истину.

Задача SAT

Выполнима ли заданная булева формула?

Пример

- Формула $\phi(x, y, z) = x \vee (y \wedge z)$ — выполнима

Пример

- Формула $\phi(x, y, z) = x \vee (y \wedge z)$ — выполнима
- Подстановка: $\{x = 0, y = 1, z = 1\}$

Пример

- Формула $\phi(x, y, z) = x \vee (y \wedge z)$ — выполнима
- Подстановка: $\{x = 0, y = 1, z = 1\}$
- Проверка: $\phi(0, 1, 1) = 0 \vee (1 \wedge 1) = 0 \vee 1 = 1$

Пример: решение задачи SAT

`js-for-tcs-06`

Формула в КНФ: $(x_1 \vee \overline{x_3}) \wedge (x_2 \vee x_3 \vee \overline{x_1})$

Это всё!

Задачи и языки

Модели вычислений и тезис Чёрча—Тьюринга

Неразрешимость и теорема Райса

Классы сложности и NP-полнота

Теория вычислений — это интересно!

Vitaly Bragilevsky

vitaly.bragilevsky@jetbrains.com

_bravit (ru)

VBragilevsky (en)

bravit111

bravit_about (channel, ru)