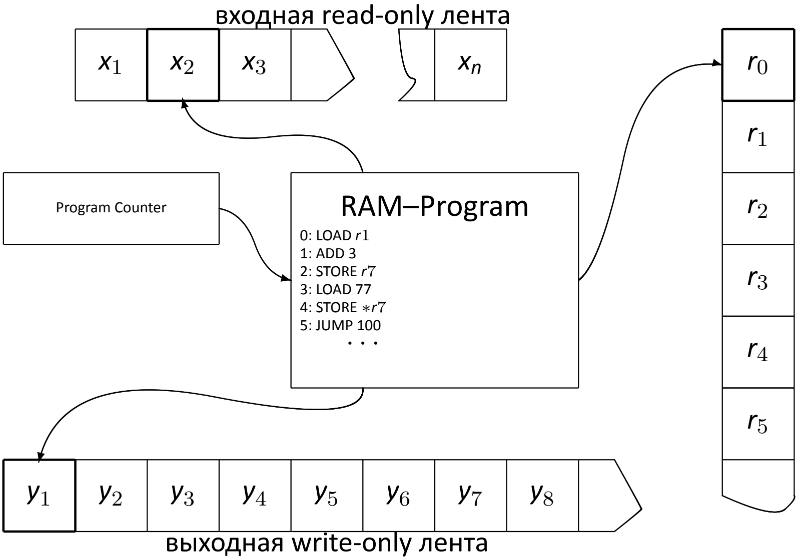
Формально об алгоритмах. Вычислительные модели

Н. Н. Кузюрин С. А. Фомин

3 декабря 2010 г.

RAM — random access machine



RAM: набор команд

LOAD OP	$r_0 \leftarrow OP$	Загрузить операнд в сумматор
STORE OP	$r_{OP} \leftarrow r_0$	Сохранить сумматор в регистре
ADD OP	$r_0 \leftarrow r_0 + OP$	Прибавить операнд к сумматору
SUB OP	$r_0 \leftarrow r_0 - OP$	Вычесть операнд из сумматора
READ OP	$r_{OP} \leftarrow input$	Загрузить ячейку из входной ленты в r_{OP} и перейти к следующей.
WRITE OP	$OP \rightarrow output$	Записать *OP* в текущую ячейку выходной ленты и сдвиг к следующей.
JUMP OP	$PC \leftarrow OP$	Установить счетчик команд в *OP*.
JGTZ OP	$PC \leftarrow OP : r_0 > 0$	Установить счетчик команд в *OP*, если $r_0 > 0$
JZERO OP	$PC \leftarrow OP : r_0 = 0$	Установить счетчик команд в *OP*, если $r_0 = 0$
HALT		Остановить работу.

RAM: моделирование FOR через GOTO

...
for *i* **in range(** $1..78$ **):**
тело цикла
...

$\Longrightarrow$

...	...	
0777	LOAD	1
0778	STORE	r_{17}
0779	начало цикла	
...	...	
1035	LOAD	1
1036	ADD	r17
1037	STORE	r17
1038	конец цикла	
1039	LOAD	r_{17}
1040	SUB	78
1041	JGTZ	0779
...	...	

RAM: меры сложности алгоритмов

- RAM потенциально мощнее любых существующих компьютеров
- но физически нереализуемы
  - оперируют бесконечной памятью
  - каждая ячейка памяти не ограничена по размеру

Однородная мера сложности

unit-cost RAM

Сложность = число выполненных команд

Более реалистична:

Логарифмическая мера сложности

Сложность команды = сумме логарифмов значений операндов

Сложность = сумма сложностей команд

Машина Тьюринга

Определение

***Машина Тьюринга** — это набор $T = \langle k, \Sigma, \Gamma, \alpha, \beta, \gamma \rangle$, где*

- $k \geq 1$ *— число лент;*
- Σ *— алфавит лент,* $\star \in \Sigma$ *— символ-пробел;*
- Γ *— конечное множество состояний,* $S, Q \in \Gamma$ *— выделенные состояния: запуск машины и завершение работы;*
- α, β, γ *— произвольные отображения:*

$$\alpha : \Gamma \times \Sigma^k \to \Gamma,$$
$$\beta : \Gamma \times \Sigma^k \to \Sigma^k,$$
$$\gamma : \Gamma \times \Sigma^k \to \{-1, 0, 1\}^k.$$

Машина Тьюринга: Таблица

```
MT = {
        ’k’: 1,
        ’start’: ’S’,
        ’stop’: ’Q’,
        ’program’: {
        # (Состояние, символы на лентах) ->
        # -> (новое состояние, (действия на лентах))
         (’S’,  (’1’)): (’s2’, ((’*’,’R’))),
         (’s2’, (’1’)): (’s2’, ((’1’,’R’))),
         (’s2’, (’*’)): (’s3’, ((’*’,’R’))),
         (’s3’, (’*’)): (’s4’, ((’1’,’L’))),
         (’s3’, (’1’)): (’s3’, ((’1’,’R’))),
         (’s4’, (’1’)): (’s4’, ((’1’,’L’))),
         (’s4’, (’*’)): (’s5’, ((’*’,’L’))),
         (’s5’, (’1’)): (’s5’, ((’1’,’L’))),
         (’s5’, (’*’)): (’S’, ((’1’,’R’))),
         (’S’, (’*’)): (’Q’,  ((’*’,’’)))
    }}
```

Машина Тьюринга: Таблица

```
MT = {
        ’k’: 1,
        ’start’: ’S’,
        ’stop’: ’Q’,
        ’program’: {
        # (Состояние, символы на лентах) ->
        # -> (новое состояние, (действия на лентах))
         (’S’,  (’1’)): (’s2’, ((’*’,’R’))),
         (’s2’, (’1’)): (’s2’, ((’1’,’R’))),
         (’s2’, (’*’)): (’s3’, ((’*’,’R’))),
         (’s3’, (’*’)): (’s4’, ((’1’,’L’))),
         (’s3’, (’1’)): (’s3’, ((’1’,’R’))),
         (’s4’, (’1’)): (’s4’, ((’1’,’L’))),
         (’s4’, (’*’)): (’s5’, ((’*’,’L’))),
         (’s5’, (’1’)): (’s5’, ((’1’,’L’))),
         (’s5’, (’*’)): (’S’, ((’1’,’R’))),
         (’S’, (’*’)): (’Q’,  ((’*’,’’)))
    }}
```

<S>	*	⇒	[Q]	*		
<S>	1	⇒	s2	*	R	
s2	*	⇒	s3	*	R	
s2	1	⇒	s2	1	R	
s3	*	⇒	s4	1	L	
s3	1	⇒	s3	1	R	
s4	*	⇒	s5	*	L	
s4	1	⇒	s4	1	L	
s5	*	⇒	<S>	1	R	
s5	1	⇒	s5	1	L	

Симулятор Машины Тьюринга

```
def execute_turing_machine(mt, tape_in):
    T = mt[”program”]          # программа/таблица переходов
    tape = [”*”] + tape_in     # дописываем пробел слева от входа
    state = mt[”start”]        # начальное состояние
    position = 1               # положение головки
    step = 0                   # счетчик тактов
    while state != mt[”stop”] and step < 1000:
        step += 1
        if position >= len(tape):  # потенциальная бесконечность
            tape.append(”*”)       # ленты
        symbol = tape[position]
        state, (symbol_to_write, move)  = T[(state, (symbol))][:2]
        tape[position] = symbol_to_write
        if move == ”L”:
            position -= 1
        if move == ”R”:
            position += 1
```

Симулятор Машины Тьюринга

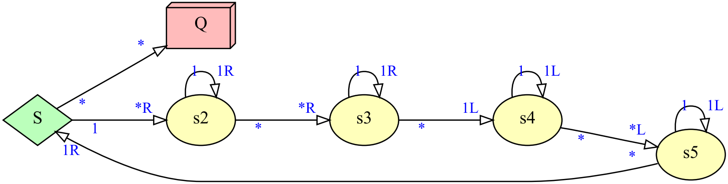
```python
def execute_turing_machine(mt, tape_in):
    T = mt["program"]          # программа/таблица переходов
    tape = ["*"] + tape_in     # дописываем пробел слева от входа
    state = mt["start"]        # начальное состояние
    position = 1               # положение головки
    step = 0                   # счетчик тактов
    while state != mt["stop"] and step < 1000:
        step += 1
        if position >= len(tape):  # потенциальная бесконечность
            tape.append("*")       # ленты
        symbol = tape[position]
        state, (symbol_to_write, move)  = T[(state, (symbol))][:2]
        tape[position] = symbol_to_write
        if move == "L":
            position -= 1
        if move == "R":
            position += 1
```

```python
MT = {
        'k': 1,
        'start': 'S',
        'stop': 'Q',
        'program': {
        # (Состояние, символы на лентах) ->
        # -> (новое состояние, (действия на лентах))
         ('S',  ('1')): ('s2', (('*','R'))),
         ('s2', ('1')): ('s2', (('1','R'))),
         ('s2', ('*')): ('s3', (('*','R'))),
         ('s3', ('*')): ('s4', (('1','L'))),
         ('s3', ('1')): ('s3', (('1','R'))),
         ('s4', ('1')): ('s4', (('1','L'))),
         ('s4', ('*')): ('s5', (('*','L'))),
         ('s5', ('1')): ('s5', (('1','L'))),
         ('s5', ('*')): ('S', (('1','R'))),
         ('S', ('*')): ('Q',  (('*',''))) 
    }}
```

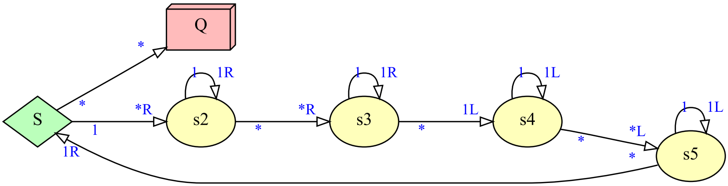
Машина Тьюринга: Удвоение строки

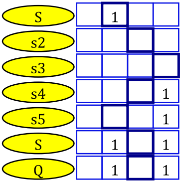
<S>	*	⇒	[Q]	*		
<S>	1	⇒	s2	*	R	
s2	*	⇒	s3	*	R	
s2	1	⇒	s2	1	R	
s3	*	⇒	s4	1	L	
s3	1	⇒	s3	1	R	
s4	*	⇒	s5	*	L	
s4	1	⇒	s4	1	L	
s5	*	⇒	<S>	1	R	
s5	1	⇒	s5	1	L	

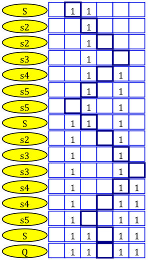
```
MT = {
        ’k’: 1,
        ’start’: ’S’,
        ’stop’: ’Q’,
        ’program’: {
        # (Состояние, символы на лентах) ->
        # -> (новое состояние, (действия на лентах))
         (’S’,  (’1’)): (’s2’, ((’*’,’R’))),
         (’s2’, (’1’)): (’s2’, ((’1’,’R’))),
         (’s2’, (’*’)): (’s3’, ((’*’,’R’))),
         (’s3’, (’*’)): (’s4’, ((’1’,’L’))),
         (’s3’, (’1’)): (’s3’, ((’1’,’R’))),
         (’s4’, (’1’)): (’s4’, ((’1’,’L’))),
         (’s4’, (’*’)): (’s5’, ((’*’,’L’))),
         (’s5’, (’1’)): (’s5’, ((’1’,’L’))),
         (’s5’, (’*’)): (’S’, ((’1’,’R’))),
         (’S’, (’*’)): (’Q’,  ((’*’,’’)))
    }}
```

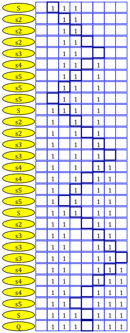


Машина Тьюринга: Удвоение строки

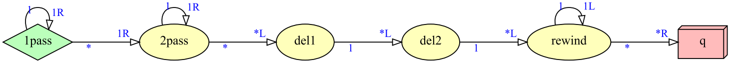






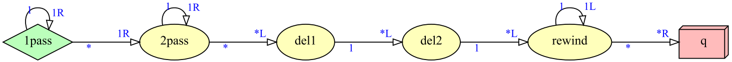


Машина Тьюринга: Унарное сложение



<1pass>	*	⇒	2pass	1	R	меняем разделитель
<1pass>	1	⇒	<1pass>	1	R	проходим первое число
2pass	*	⇒	del1	*	L	конец второго числа
2pass	1	⇒	2pass	1	R	проходим второе число
del1	1	⇒	del2	*	L	удаляем 1-ю лишнюю 1
del2	1	⇒	rewind	*	L	удаляем 2-ю лишнюю 1
rewind	*	⇒	[q]	*	R	конец
rewind	1	⇒	rewind	1	L	перематываем к началу

Машина Тьюринга: Унарное сложение



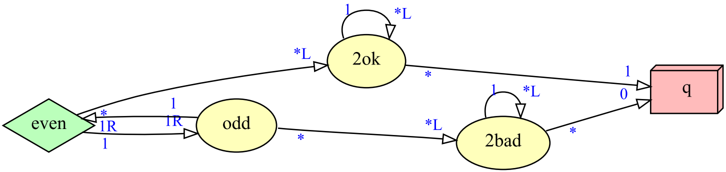
1pass		$\boxed{1}$		1	
1pass		1	$\boxed{\ }$	1	
2pass		1	1	$\boxed{1}$	
2pass		1	1	1	$\boxed{\ }$
del1		1	1	$\boxed{1}$	
del2		1	$\boxed{1}$		
rewind		$\boxed{1}$			
rewind	$\boxed{\ }$	1			
q		$\boxed{1}$			

1pass		$\boxed{1}$	1	1		1	
1pass		1	$\boxed{1}$	1		1	
1pass		1	1	$\boxed{1}$		1	
1pass		1	1	1	$\boxed{\ }$	1	
2pass		1	1	1	1	$\boxed{1}$	
2pass		1	1	1	1	1	$\boxed{\ }$
del1		1	1	1	1	$\boxed{1}$	
del2		1	1	1	$\boxed{1}$		
rewind		1	1	$\boxed{1}$			
rewind		1	$\boxed{1}$	1			
rewind		$\boxed{1}$	1	1			
rewind	$\boxed{\ }$	1	1	1			
q		$\boxed{1}$	1	1			

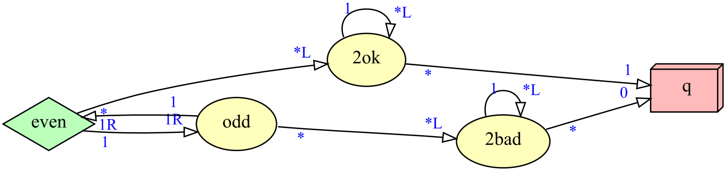
1pass		$\boxed{1}$	1		1	1	1	
1pass		1	$\boxed{1}$		1	1	1	
1pass		1	1	$\boxed{\ }$	1	1	1	
2pass		1	1	1	$\boxed{1}$	1	1	
2pass		1	1	1	1	$\boxed{1}$	1	
2pass		1	1	1	1	1	$\boxed{1}$	
2pass		1	1	1	1	1	1	$\boxed{\ }$
del1		1	1	1	1	1	$\boxed{1}$	
del2		1	1	1	1	$\boxed{1}$		
rewind		1	1	1	$\boxed{1}$			
rewind		1	1	$\boxed{1}$	1			
rewind		1	$\boxed{1}$	1	1			
rewind		$\boxed{1}$	1	1	1			
rewind	$\boxed{\ }$	1	1	1	1			
q		$\boxed{1}$	1	1	1			

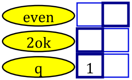
Машина Тьюринга: Распознавание четных строк

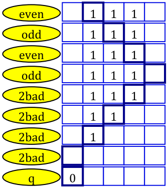
2bad	*	⇒	[q]	0		
2bad	1	⇒	2bad	*	L	
2ok	*	⇒	[q]	1		
2ok	1	⇒	2ok	*	L	
<even>	*	⇒	2ok	*	L	
<even>	1	⇒	odd	1	R	
odd	*	⇒	2bad	*	L	
odd	1	⇒	<even>	1	R	

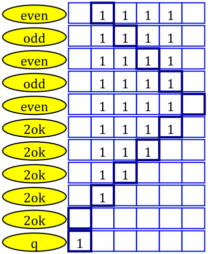


Машина Тьюринга: Распознавание четных строк

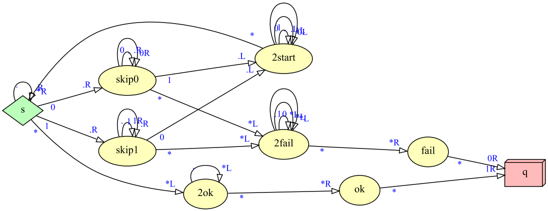


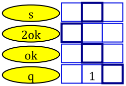


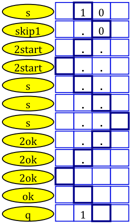


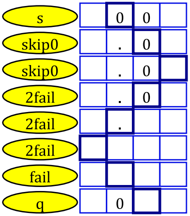


Машина Тьюринга: «Одинаковое количество 0 и 1?»









Машина Тьюринга: «Одинаковое количество 0 и 1?»

s		$\boxed{1}$	1	0	
skip1		.	$\boxed{1}$	0	
skip1		.	1	$\boxed{0}$	
2start		.	$\boxed{1}$	.	
2start		$\boxed{.}$	1	.	
2start	$\boxed{}$	.	1	.	
s		$\boxed{.}$	1	.	
s		.	$\boxed{1}$	.	
skip1		.	.	$\boxed{.}$	
skip1		.	.	.	$\boxed{}$
2fail		.	.	$\boxed{.}$	
2fail		.	$\boxed{.}$		
2fail		$\boxed{.}$			
2fail	$\boxed{}$				
fail		$\boxed{}$			
q		0	$\boxed{}$		

s		$\boxed{1}$	0	0	1	
skip1		.	$\boxed{0}$	0	1	
2start		$\boxed{.}$	.	0	1	
2start	$\boxed{}$	.	.	0	1	
s		$\boxed{.}$	.	0	1	
s		.	$\boxed{.}$	0	1	
s		.	.	$\boxed{0}$	1	
skip0		.	.	.	$\boxed{1}$	
2start		.	.	$\boxed{.}$	.	
2start		.	$\boxed{.}$	.	.	
2start		$\boxed{.}$	.	.	.	
2start	$\boxed{}$	.	.	.	.	
s		$\boxed{.}$	.	.	.	
s		.	$\boxed{.}$	.	.	
s		.	.	$\boxed{.}$	.	
s		.	.	.	$\boxed{.}$	
s		.	.	.	.	$\boxed{}$
2ok		.	.	.	$\boxed{.}$	
2ok		.	.	$\boxed{.}$		
2ok		.	$\boxed{.}$			
2ok		$\boxed{.}$				
2ok	$\boxed{}$					
ok		$\boxed{}$				
q		1	$\boxed{}$			

s		$\boxed{1}$	1	0	0	
skip1		.	$\boxed{1}$	0	0	
skip1		.	1	$\boxed{0}$	0	
2start		.	$\boxed{1}$	.	0	
2start		$\boxed{.}$	1	.	0	
2start	$\boxed{}$	.	1	.	0	
s		$\boxed{.}$	1	.	0	
s		.	$\boxed{1}$	.	0	
skip1		.	.	$\boxed{.}$	0	
skip1		.	.	.	$\boxed{0}$	
2start		.	.	$\boxed{.}$	.	
2start		.	$\boxed{.}$	.	.	
2start		$\boxed{.}$	.	.	.	
2start	$\boxed{}$	.	.	.	.	
s		$\boxed{.}$	.	.	.	
s		.	$\boxed{.}$	.	.	
s		.	.	$\boxed{.}$	.	
s		.	.	.	$\boxed{.}$	
s		.	.	.	.	$\boxed{}$
2ok		.	.	.	$\boxed{.}$	
2ok		.	.	$\boxed{.}$		
2ok		.	$\boxed{.}$			
2ok		$\boxed{.}$				
2ok	$\boxed{}$					
ok		$\boxed{}$				
q		1	$\boxed{}$			

Универсальная Машина Тьюринга

Определение

$k+1$ *ленточная МТ* Т ***универсальна**, если*

- *для любой k-ленточной МТ S (над алфавитом Σ)*
- *существует программа p $\in \Sigma_0^*$, на которой* Т *симулирует S.*

Универсальная Машина Тьюринга

Определение

$k+1$ ленточная МТ T ***универсальна**, если*

- *для любой k-ленточной МТ S (над алфавитом Σ)*
- *существует программа $p \in \Sigma_0^*$, на которой* T *симулирует S.*

Теорема

$\forall k \geq 1$ и $\forall \Sigma$, существует $k+1$ ленточная универсальная МТ.

Универсальная Машина Тьюринга

Определение

$k+1$ ленточная МТ T ***универсальна**, если*

- *для любой k-ленточной МТ S (над алфавитом Σ)*
- *существует программа $p \in \Sigma_0^*$, на которой* T *симулирует S.*

Теорема

$\forall k \geq 1$ *и* $\forall \Sigma$, *существует $k+1$ ленточная универсальная МТ.*

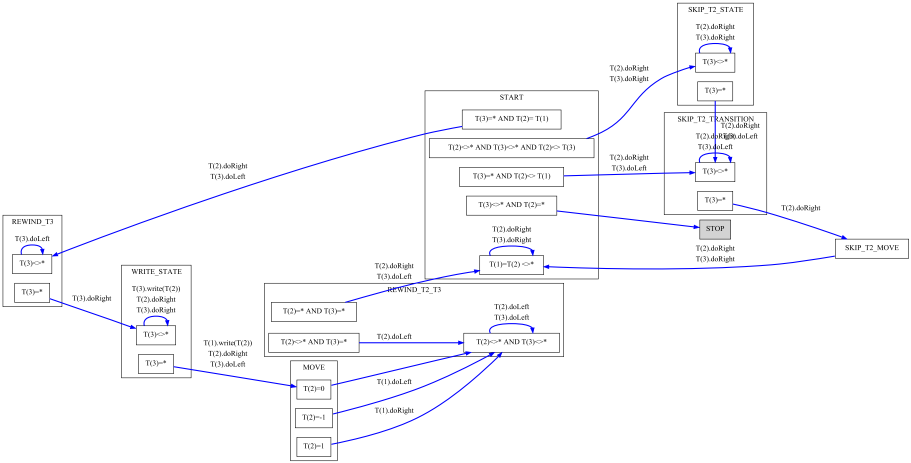
Теорема

Для любой k-ленточной МТ S *существует одноленточная МТ* T *,*

- $\forall x \in \Sigma_0^*$, T *останавливается на $x \Leftrightarrow$*
  - *на x останавливается* S
  - *на ленте* T *то, что после остановки на последней ленте* S.

Для x разрешимого S *за N шагов $\rightarrow$ время работы* T *будет $O(N^2)$.*

Универсальная Машина Тьюринга



Вычислимость и разрешимость

Определение

Функция $f : N \to N$ является ***вычислимой****, если существует такая машина Тьюринга* T*, что если на вход ей подать представленный в некоторой кодировке* ***x****, то*

1. *если функция f определена на* ***x****, и $f(\boldsymbol{x}) = \boldsymbol{y}$, то машина* T *останавливается на входе* ***x****, и на выходе у нее записано* ***y****;*
2. *если функция f не определена на* ***x****, то машина* T *зацикливается (не останавливается за любое конечное число шагов) на входе* ***x****.*

Вычислимость и разрешимость

Определение

*Функция $f : N \to N$ является **вычислимой**, если существует такая машина Тьюринга* T*, что если на вход ей подать представленный в некоторой кодировке **x**, то*

1. *если функция f определена на **x**, и $f(\boldsymbol{x}) = \boldsymbol{y}$, то машина* T *останавливается на входе **x**, и на выходе у нее записано **y**;*
2. *если функция f не определена на **x**, то машина* T *зацикливается (не останавливается за любое конечное число шагов) на входе **x**.*

Определение

*Множество S (язык L) является **разрешимым**, если существует такая машина Тьюринга* T*, что если на вход ей подать элемент $x \in S$ (слово $l \in L$), то она остановится и выведет «1».*
Иначе ($x \notin S$, $l \notin L$), T *останавливается и выводит «0».*

Неразрешимость

Упражнение

Докажите, что существуют невычислимые по Тьюрингу функции $y = f(x)$.
Использовать мощностные соображения.

Неразрешимость

Упражнение

Докажите, что существуют невычислимые по Тьюрингу функции $y = f(x)$.
Использовать мощностные соображения.

Задача

Проблема остановки (halting problem). *Для данной машины Тьюринга* M *и входа x определить, остановится ли машина Тьюринга* M, *начав работу на x?*

Неразрешимость

Упражнение

Докажите, что существуют невычислимые по Тьюрингу функции $y = f(x)$. Использовать мощностные соображения.

Задача

Проблема остановки (halting problem). *Для данной машины Тьюринга* M *и входа x определить, остановится ли машина Тьюринга* M*, начав работу на x?*

Теорема

Проблема остановки алгоритмически неразрешима.

Неразрешимость проблемы остановки

Пусть есть $\mathrm{T}(\mathrm{M}, x)$: «Да» $\Leftrightarrow$ M останавливается на x, иначе «Нет»

$\mathrm{T}^{diag}(X) \equiv \mathrm{T}(X, X)$

$\mathrm{T}^{killmepls}(X)$:

- $\mathrm{T}^{diag}(X)$=«да» $\Rightarrow \mathrm{T}^{killmepls}$ зацикливается,
- $\mathrm{T}^{diag}(X)$=«нет» $\Rightarrow \mathrm{T}^{killmepls}$ останавливается.

Остановится ли $\mathrm{T}^{killmepls}$ на $\mathrm{T}^{killmepls}$?

- «Да» $\Rightarrow \mathrm{T}^{diag}(\mathrm{T}^{killmepls}) =$ «нет»
  $\Rightarrow \mathrm{T}^{killmepls}$ не должна останавливаться на $\mathrm{T}^{killmepls}$.
- «Нет» $\Rightarrow \mathrm{T}^{diag}(\mathrm{T}^{killmepls}) =$ «да»
  $\Rightarrow \mathrm{T}^{killmepls}$ должна останавливаться на $\mathrm{T}^{killmepls}$.

Противоречие.

Упражнения

Упражнение

Докажите, что также неразрешима версия задачи «HALT» — «остановка на пустом слове», т. е. для данной МТ T определить, остановится ли она на пустом слове.

Упражнения

Упражнение

Докажите, что также неразрешима версия задачи «HALT» — «остановка на пустом слове», т. е. для данной МТ T определить, остановится ли она на пустом слове.

Упражнение

Докажите, что не существует алгоритма, который выписывает одну за другой все машины Тьюринга, которые не останавливаются, будучи запущенными на пустой ленте.

Упражнения

Упражнение

Докажите, что также неразрешима версия задачи «HALT» — «остановка на пустом слове», т. е. для данной МТ T определить, остановится ли она на пустом слове.

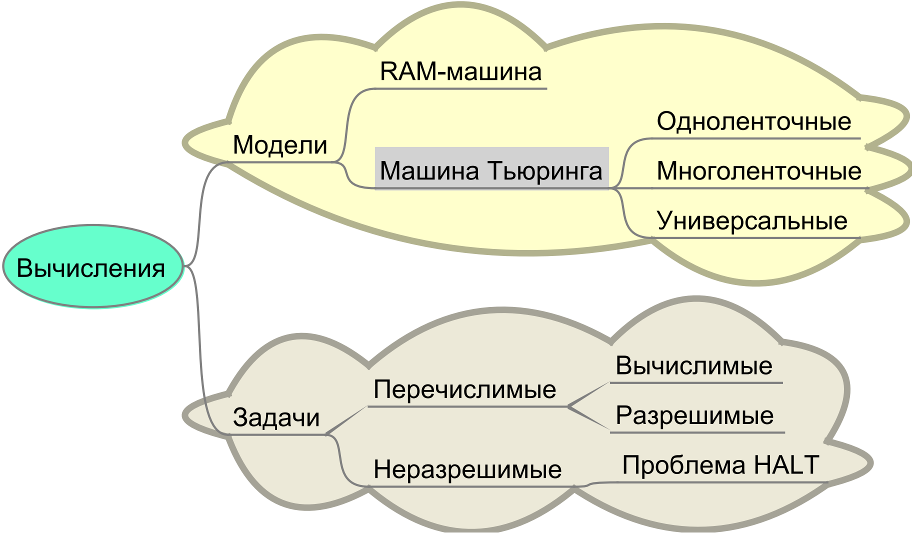
Упражнение

Докажите, что не существует алгоритма, который выписывает одну за другой все машины Тьюринга, которые не останавливаются, будучи запущенными на пустой ленте.

Упражнение

Существует ли алгоритм, который выписывает одну за другой все машины Тьюринга, которые останавливаются, будучи запущенными на пустой ленте?

«Карта памяти» лекции



Интернет поддержка курса

`http://discopal.ispras.ru/`

Вопросы?