# Детерминированная машина Тьюринга (ДМТ)

## Определение алгоритма

Основные атрибуты алгоритма:

* Алгоритм задается последовательностью инструкций
* Алгоритм выполняется детерминировано, т.е. для одинаковых данных выполняются одинаковые действия
* Должен существовать вычислитель, способный выполнить указанные в алгоритме инструкции
* Вычислитель должен иметь средства для хранения и отображения информации

Таким образом алгоритм – это набор инструкций для формальной модели вычислителя.

Результативность алгоритма желательное, но не обязательное условие.

Алгоритм не обязан останавливаться – он может работать бесконечно.

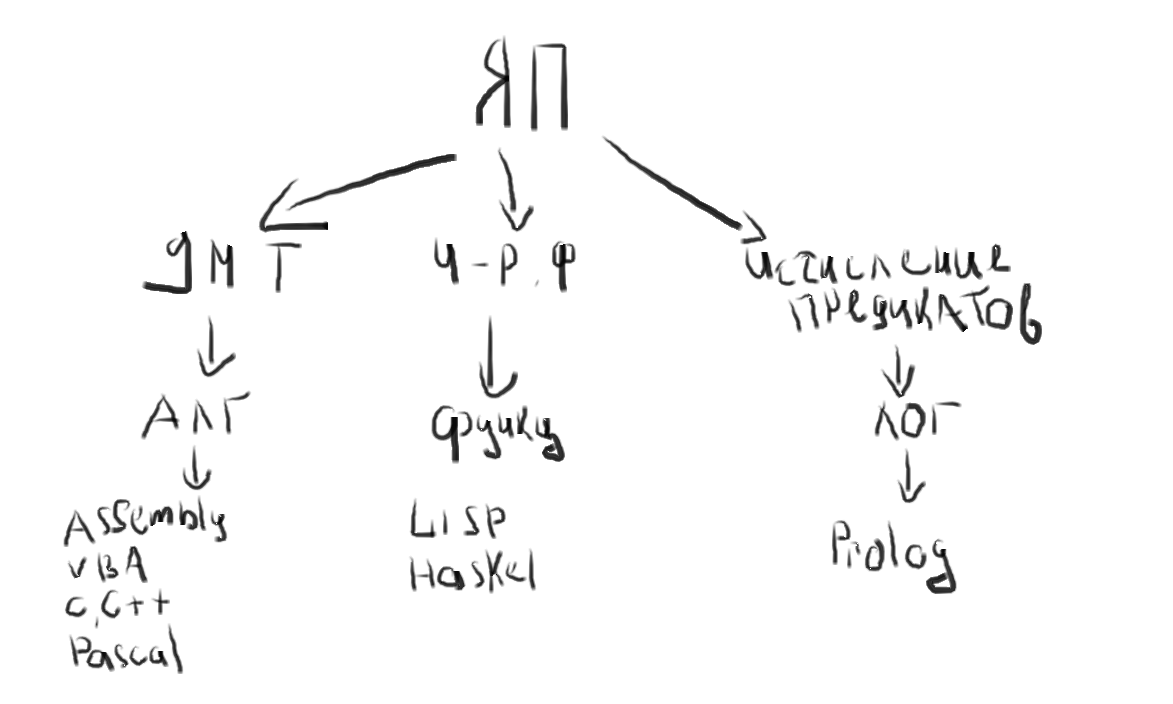
Математическая модель “ДМТ” является вычислителем, способным выполнить все указанные в алгоритме инструкции.

Существует 3 основные математические модели алгоритма:

* Детерминированная машина
* Частично рекурсивная функция
* Исчисление предикатов

Соответственно этим моделям языки программирования делятся на 3 группы:

* Алгоритмические (процедурные)
* Функциональные
* Логические



Программа на логическом языке представляет собой базу знаний, состоящую из истинных фактов и правил вывода этих фактов.

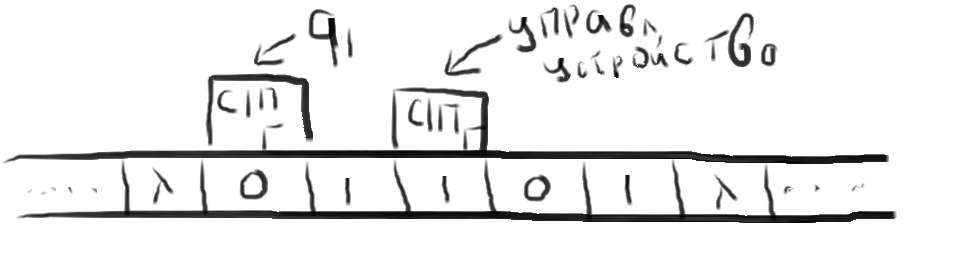
Чтобы запустить прогу нужно сформировать к базе знаний запрос или сформировать цель.

## ДМТ и тезис Тьюринга

### ДМТ как белый ящик или структурная схема

ДМТ – это автомат, который имеет бесконечную ленту, а также управляющее устройство и считывающую – пишущую головку.

Лента разделена на ячейки в каждой из ячеек записан либо пустой символ (λ), либо символ некоторого алфавита А. В каждый момент времени на ленте может быть конечное число не пустых символов.



Управляющее устройство находится в одном из конечного множества состояний. Т.е. количество состояний, как и количество символов конечно. Среди них выделяют начальное – q1 и конечное – qz. Считывающая головка в 1 момент времени обозревает 1 ячейку. Перед началом работы управляющее устройство ДМТ находится в начальном состоянии q1, а сама считывающая головка ДМТ находится в ячейке с 1 непустым символом. По достижению qz ДМТ останавливается.

Один этап работы ДМТ можно описать следующим свойством:

1. считать символ аj с текущей ячейки
2. затереть текущий символ аj и в зависимости от текущего состояния и прочитанного символа записать туда другой символ aj’
3. В зависимости от текущего состояния и прочитанного символа либо остаться на месте, либо сдвинуться на 1 шаг в соседнюю клетку
4. В зависимости от текущего состояния qi и прочитанного символа aj перейти в новое состояние qi’

Машиной Тьюринга называется совокупность <А, λ, Q, q1, δ>, где

А – Алфавит

λ – Пустые значения

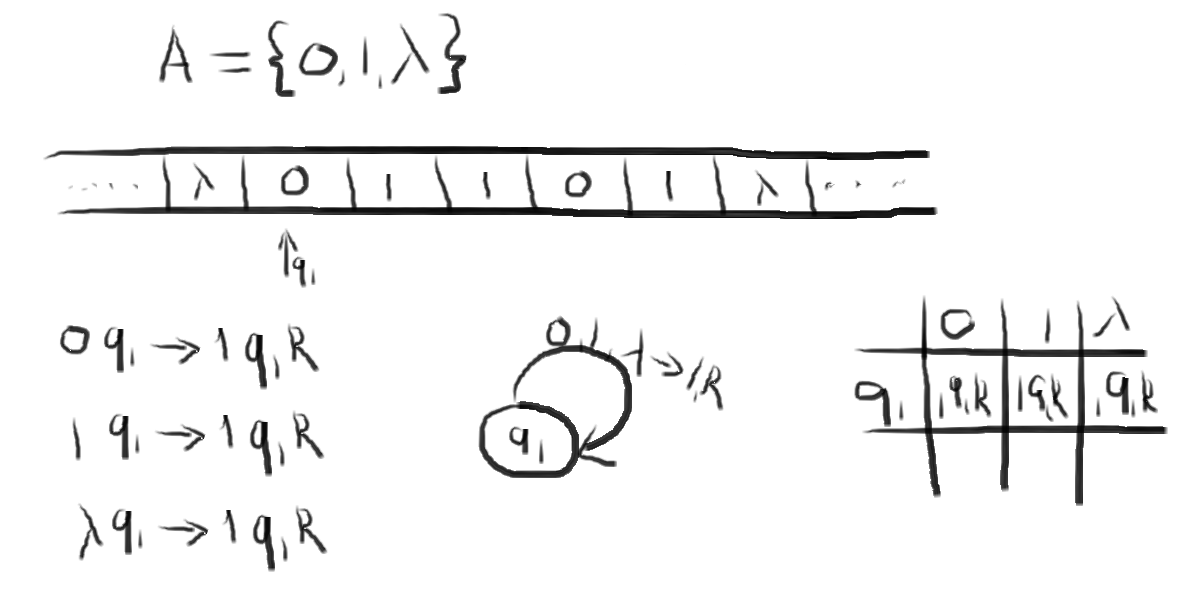
Q – Множество состояний

q1 – Первое состояние

δ – Множество команд

Пример: бесконечно двигаясь вправо заполнить всю ленту единицами.

Существует 3 способа представления алгоритма Тьюринга



Конфигурация (К) машины Тьюринга – К <α1, qi, α2>

α1 – слово, записанное слева от головки

α2 – слово, записанное справа от головки с учетом текущего символа

qi – текущее состоянии

Слева α1 от и справа α2 находятся пустые символы.

К каждой не заключительной конфигурации применима ровно 1 команда. Текущая конфигурация однозначно определяет все дальнейшее поведение машины Тьюринга.

K1 –> Ki –> Ki+1 –> …–> Kz

Начальной и заключительной конфигурациями считаются q1α и qzβ, где слева от головки МТ находится пустой символ. То есть ДМТ обозревает первый не пустой символ.

### Тезис Тьюринга

Всякий алгоритм может быть реализован ДМТ.

### ДМТ как черный ящик

ДМТ – это модель вычислителя, способного выполнить следующий набор элементарных операций:

{+,-\*,/,&,OR,XOR,WHILE,READ,WRITE,END}

### Функции правильно вычислимые по Тьюрингу

Будем считать, что алфавит A состоит из 3 алфавитов: исходного, промежуточного и конечного.

Промежуточный алфавит образует символы, которые могут появиться в процессе работы и слова, записанные на ленте, будут иметь вид словарных векторов.

α1\* α2 \* ... \* αn

Пусть

V – Это слово или словарный вектор на исходном алфавите

W – Это слово или словарный вектор на результатном алфавите

ДМТ правильно вычисляет функцию f, если:

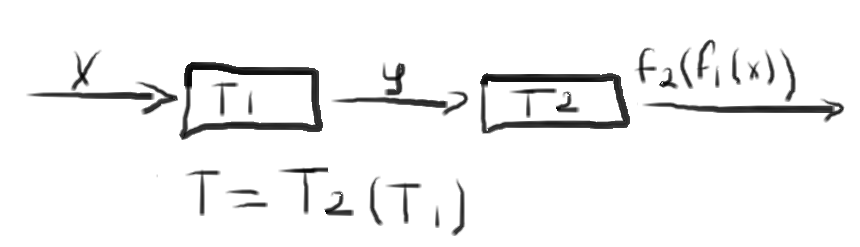
* 1. Для любых v ∈ V и w ∈ W – существует такая функция f(v) = w, то q1v –> qzw
  2. Если f(v) не определена, то ДМТ, запущенная из q1v будет работать бесконечно

Если ДМТ реализует функцию f, которая не определена в точке V, то на входном слове V ДМТ будет работать бесконечно.

### Операции над машинами Тьюринга

Композиция двух функций f1(x) и f2(y) – g(x) = f2(f1(x)). Чтобы g(x) была определена на данном x достаточно чтобы f1 была определена на x и f2 была определена на f1(x).

Теорема. Если f1(x) и f2(y) вычислимы по Тьюрингу, то их композиция так же вычислима по Тьюрингу.

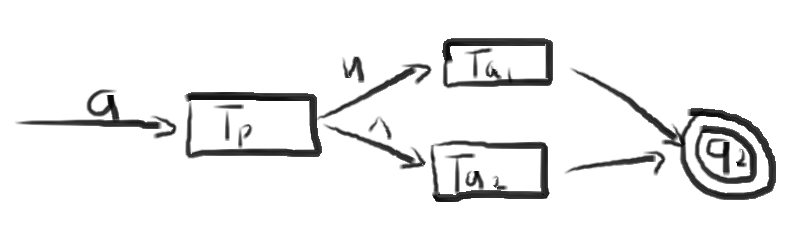


Типа кароче хошь налить супа – сначала приготовь его мудила, а потом наливай

f(a) является условием перехода к g1(a) или g2(a) условию p(a) ,если она задана следующим описанием. Если p(a) истинно, то g1(a), иначе g2(a).

Теорема 2:

Если g1(a) g2(a) и p(2) Вычислимы по Тьюрингу, то условный переход по p(a) так же вычислим по Тьюрингу.



Пример: построить машину, суммирующую любые слагаемые

Сумма, полученная на 1 слагаемом – это начальная сумма на 2 слагаемом.

Таким образом существует возможность построить ДМТ, реализующую любое элементарное действие из набора:

{+,-\*,/,&,OR,XOR,WHILE,READ,WRITE,END}

С помощью операции композиции мы можем строить какие-угодно сложные машины Тьюринга, используя в качестве блоков машины Тьюринга для элементарных действий.

Другими словами – для каждого алгоритма можно построить реализующую его ДМТ.

Можем ли мы считать, что у нас построено универсальное устройство – интерпретатор(ДМТ), способный выполнить любой алгоритм? – Нет.

### Универсальная машина Тьюринга или U–машина

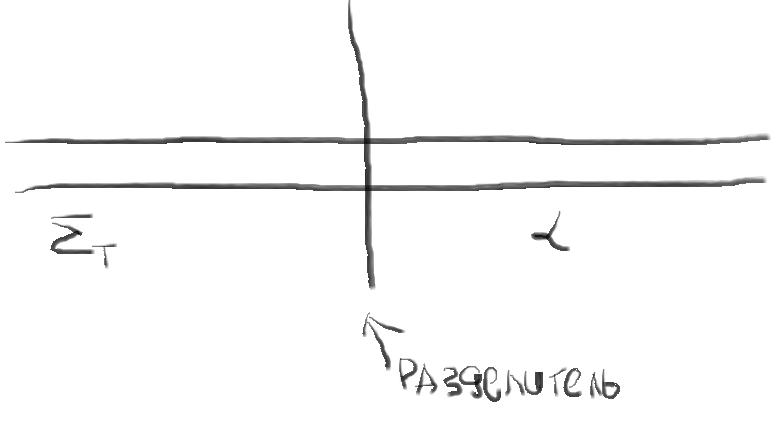
Согласно тезису Тьюринга – любой алгоритм может быть реализован ДМТ, то есть для любого алгоритма можно построить уникальную машину Тьюринга, которая будет выполнять только этот алгоритм.

Другой подход заключатся в создании универсальной машины Тьюринга или U–машины, которая выполняет функции интерпретатора. Для этого у нас должны быть хранимые, модифицируемые и выполняемые программы, а также устройство, которое способно эти программы способно выполнить.

Рассмотрим проблему построение универсального устройства интерпретатора в рамках модели ДМТ. Хранимая, модифицируемая и выполняемая программа – это система команд(aiqi–> ai’qi’αk) МТ. Входные данные для программы – это входное слово α. Результат работы машины Т на слове α – выходное слово Т(α), если машина Т останавливается на слове α.

Получаем, что нужно построить такую универсальную ДМТ, что для любой МТ Т системой команд(∑Т) и входным словом α мы получим следующее U(∑Т, α) = Т(α). Для машины Т входным словом является слово (α), а для машины U входным является система команд (∑Т) и входное слово (α).

Как на ленте U машины разместить входное слово этой U машины? Разделим ленту U машины на две бесконечные полу–ленты. В левой ленте запишем систем команд машины Т. В правой полу–ленте – входное слово.



Машина U читает текущий символ входного слова на правой полу–ленте и ищет подходящую команду в левой полу–ленте. Затем возвращается на правую полу–ленту и переписывает текущий символ, согласно прочитанной команде, а дальше переходит в новое состояние. В результате работы U машины на правой ленте будет записано слово Т(α).

Проблема, возникающая при построении U–машины связана с тем, что U–машина должна иметь фиксированный алфавит. Систему команд и входное слово α машины T нельзя переписать на ленту машины U потому что их алфавиты могут не совпадать. Выход заключается в том, чтобы символы из алфавитов машины Т и U кодировать словами алфавита U (0 и 1).

### Проблема остановки машины Тьюринга. Алгоритмически неразрешимые проблемы

Рассмотрим проблему разрешимости с точки зрения машин Тьюринга: Машина Тьюринга вычисляет значение некоторой функции натуральных аргументов. Каждая машина представляет собой некую функцию. Верно ли обратное – каждая ли функция вида f:N–>N может быть реализована некоторой машиной Тьюринга. Поскольку каждая машина Тьюринга однозначно задается своей программой, а программу можно записать как последовательность команд, то есть каждую последовательность команд можно записать как строчку конечной длины, в конечном алфавите, то мы имеем счётное количество строк.

То есть множество всех машин Тьюринга или всех алгоритмов – счётно. В тоже время количество функций f:N­–>N несчётно. Следовательно, существуют функции, которые нельзя вычислить за конечное число шагов с помощью алгоритма. Такие функции называются невычислимыми.

Таким образом алгоритмически неразрешимых задач гораздо больше, чем разрешимых – таких задач несчётное количество.

Если про задачу доказано, что не существует алгоритма, который решал бы эту задачу за конечное число шагов, то такая задача считается алгоритмически неразрешимой.

Рассмотрим одну такую задачу, которая называется проблема остановки машины Тьюринга.

Результативность – желательное свойство алгоритма, ДМТ на некоторых входных словах может работать бесконечно.

Построить вместе с универсальным интерпретатором универсальный отладчик. То есть такой алгоритм В, который бы позволял определить для любого алгоритма а и входного слова α – будет ли остановка А на α или он будет работать бесконечно.

В рамках модели ДМТ эта модель формулируется следующим образом: Построить МТ B, такую что для любой МТ А и входного слова α такую, что B(∑A, α) = Истина, если А остановится на α.

Теорема: универсального отладчика не существует, то есть не существует машины Тьюринга В, решающая проблему остановки для произвольной машины Тьюринга Т.

Смысл этой теоремы в том, что не существует одного универсального отладчика, способного отладить любую программу. Для каждого алгоритма проблема отладки должна решаться отдельно.

## Недетерминированная машина Тьюринга А. Классификация задач по степени сложности

### Вычислительная сложность алгоритмов

Сложность задачи – сложность наилучшего алгоритма, известного для ее решения.

Сложность алгоритма – число элементарных шагов, необходимых для решения наихудшего из всех возможных случаев, допускающих применение алгоритма.

Число элементарных шагов выражается как функция от размерности входных данных задачи.