Лекция 3.
Теоремы Шеннона

Теорема Шеннона №1

- **Всякая машина Тьюринга *A* может быть преобразована в эквивалентную машину *B* не более чем с двумя внутренними состояниями.**

Доказательством теоремы будет схема построения такой машины.

Причем строить будем, по сути, универсальную машину Тьюринга, использующую **одну ленту** и имеющую лишь **два внутренних состояния**, которая сможет моделировать работу любой машины Тьюринга.

Доказательство теоремы Шеннона № 1

- Пусть машина *A* содержит:
  - *m* символов внешнего алфавита $a_{1,\ldots,j,\ldots,m}$,
  - *n* внутренних состояний $S_{1,\ldots,i,\ldots,n}$,

- Тогда машина *B* будет содержать:
  - два внутренних состояния α и β,
  - *m* обычных символов внешнего алфавита b_i , являющихся аналогами a_i,
  - Не более чем $4mn$ особенных символов b, за счет которых производится расширение внутренней памяти.

Общая идея построения

- Для произвольной машины Тьюринга A с алфавитом из m букв (символов, записываемых на ленте, включая пустой символ) и с n внутренними состояниями мы построим машину B с двумя внутренними состояниями и алфавитом не более чем из $4mn+m$ символов.

- Машина B будет работать, по существу, так же, как и машина A: во всех клетках ленты, кроме воспринимаемой считывающей головкой и одной смежной с ней, на ленте машины B записано то же, что и на ленте машины A в соответствующие такты работы двух машин.

Общая идея построения

- Машина *B* моделирует поведение машины *A*, но **хранит информацию о внутреннем состоянии машины *A*** с помощью символов, записанных в клетке под считывающей головкой, и в клетке, которую считывающая головка машины *A* собирается посетить.

- **Основная задача** — своевременно освежать эту информацию и держать ее под считывающей головкой. Если последняя передвигается, то информацию о состоянии надо перенести в новый квадрат, используя всего два внутренних состояния машины *B.*

Формальная схема построения

- Пусть символы алфавита машины A $a_1, a_2, \ldots, a_m$ и ее состояния $S_1, S_2, \ldots, S_n$.

- В машине B поставим в соответствие алфавиту машины A $\boldsymbol{m}$ элементарных символов $b_1, b_2, \ldots, b_m$.

- Затем определим **4*mn*** новых символов, соответствующих парам из состояния и символа машины А, снабженных двумя новыми двузначными индексами. Такие новые символы будем называть ***особыми***.

Формальная схема построения

- **Особый** знак машины *B* имеет формат b_{ijxy}, где:
- i - номер ленточного символа, $i = 1, 2, \ldots, m$
- j - номер внутреннего состояния машины *A*, $j = 1, 2, \ldots, n$
- x - назначение (роль) клетки: если клетка передает информацию во время «качания», то x = ”+”, а если получает – то x = ”-”. Сами клетки назовем соответственно: передатчик / приёмник.
- y - положение другой особой клетки (машина *B* не может запомнить откуда она ушла): в зависимости от того, вправо или влево от воспринимаемой клетки должна передвинуться считывающая головка при качании, y = R или L.

Два состояния машины *B* назовем α и β.

Формальная схема построения

- При **первом шаге** качания они переносят в ближайшую подлежащую посещению клетку информацию о том, вправо (α) или влево (β) от новой клетки лежит старая клетка. Эта информация нужна в новой клетке, чтобы управляющий элемент передвинул считывающую головку назад в нужном направлении. **После первого шага** информация об этом сохраняется в новой клетке с помощью записанного там символа (последний индекс y).

- Состояния α и β используются, чтобы сообщить из старой клетки в новую о факте окончания качания. После первого шага качания состояние β переносится в новую клетку вплоть до конца качания, когда переносится α. **Это означает конец операции, и новая клетка начинает затем действовать как передатчик и управляет следующим шагом вычисления.**

Формальная схема построения

- Чтобы заставить машину *B* работать аналогично машине *A*, мы заполняем начальную ленту машины *B* соответственно начальной ленте машины *A* (с заменой a_i на b_i), за исключением клетки, занимаемой считывающей головкой в начальный момент.

- Если S_j - начальное состояние машины *A* и a_i начальный символ в этом квадрате, то в соответствующем квадрате ленты машины *B* записываем $\boxed{b_{i,j,-,R/L}}$ и приводим машину *B* в состояние α.

Таблица работы машины B

α	b_i	$\rightarrow$	$b_{i,1,-,R}$	R	α	$i=1,2,\ldots,m$	
β	b_i	$\rightarrow$	$b_{i,1,-,L}$	L	α	$i=1,2,\ldots,m$	
β	$b_{i,j,-,y}$	$\rightarrow$	$b_{i,j+1,-,y}$	y	α	$i=1,2,\ldots,m$	$j=1,2,\ldots,n-1;$ $y=R,L$
$\frac{\alpha}{\beta}$	$b_{i,j,+,y}$	$\rightarrow$	$b_{i,j-1,+,y}$	y	β	$i=1,2,\ldots,m$	$j=2,\ldots,n;$ $y=R,L$
$\frac{\alpha}{\beta}$	$b_{i,1,+,y}$	$\rightarrow$	b_i	y	α	$i=1,2,\ldots,m$	$y=R,L$

Формальная схема построения

- Предположим, что машина *A* имеет команду:

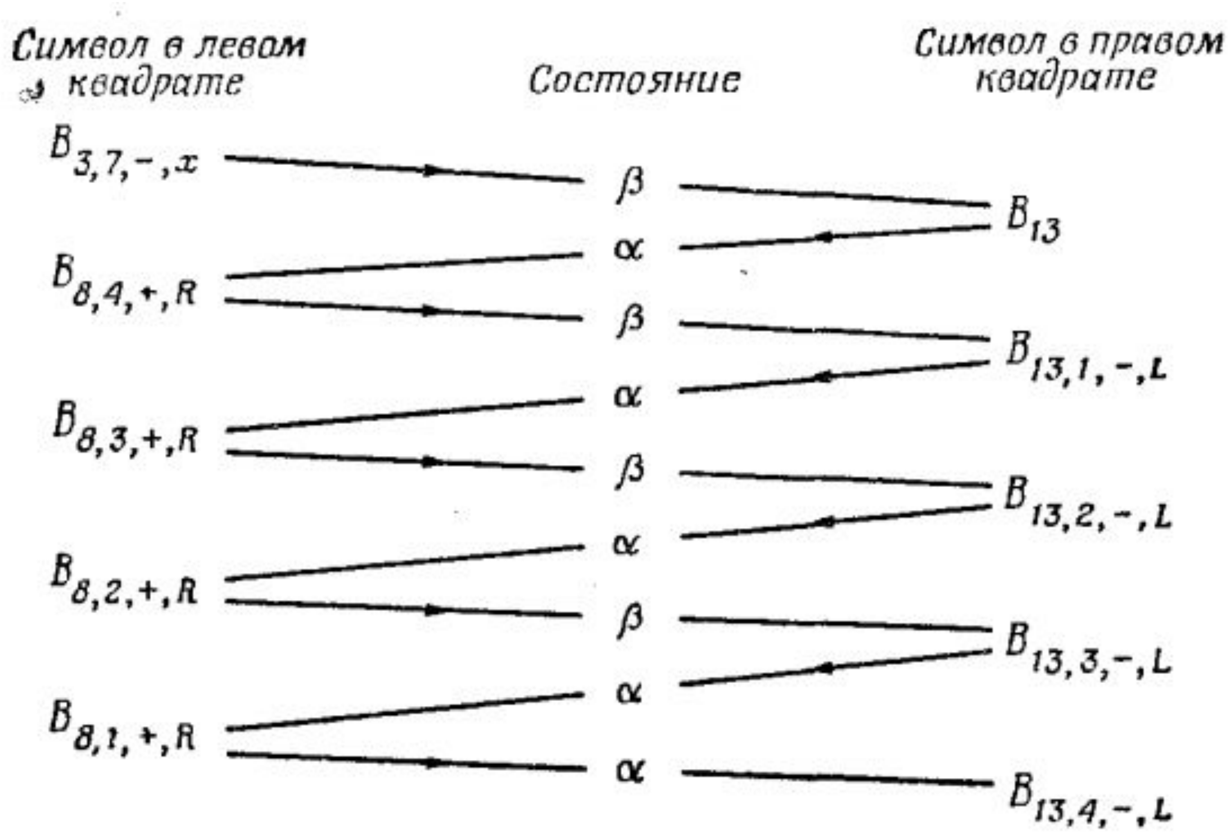
$$S_j\, a_i \to a_k\, \frac{R}{L}\, S_l$$

- Тогда машина *B* будет иметь команду:

$$\alpha \quad b_{i,j,-,\frac{R}{L}} \to b_{k,l,+,\frac{R}{L}} \quad \frac{R}{L} \quad \frac{\beta}{\alpha}$$

Пусть м.А

$S_7\, a_3 \to a_8\, R\, S_4$



Формальная схема построения

- Инструкция машины *A* заменяется приведенной выше инструкцией для машины *B*. Машина *B* работает вплоть до момента, когда вместо особого символа в одной из двух особых клеток окажется записанным элементарный символ, соответствующий символу из внешнего алфавита машины *A*.
- Т.о. будет произведен набор действий, эквивалентный первой команде (инструкции) машины *A*. Далее аналогично эмулируется вторая команда и т.д. вплоть до остановки машины *A* и эквивалентной её машины *B*.
- Итого показано, как машина *A* преобразуется в эквивалентную ей машину *B* с двумя внутренними состояниями. **Q.E.D**.

Теорема Шеннона №2

Всякая машина Тьюринга *А* может быть преобразована в эквивалентную машину *С* не более чем с двумя знаками внешнего алфавита.

Доказательством будет схема построения.

Покажем, что можно построить машину *С*, работающую подобно любой заданной машине Тьюринга *А* и использующую только **два символа внешнего алфавита**, например символы 0 и 1.

Доказательство

- Пусть машина *A* содержит:
  - *n* внутренних состояний S_j,
  - *m* символов внешнего алфавита a_i,

- Тогда машина C будет содержать:
  - 2 символа внешнего алфавита: 0 и 1
  - *n* внутренних состояний T_j, являющихся аналогами S_j,
  - Некоторое количество специальных внутренних состояний (оценим его в конце доказательства).

Общая идея построения

- Пусть *l* - наименьшее целое число, для которого $m \le 2^l$.

- Тогда символам машины *A* можно сопоставить двоичные последовательности длины *l* таким образом, что различным символам будут соответствовать различные же последовательности.

- При этом пустому символу машины *A* мы ставим в соответствие последовательность из *l* нулей. Машина *C* будет работать с двоичными последовательностями.

Общая идея построения

- Элементарная операция машины *A* будет соответствовать в машине *C* переходу считывающей головки на (l – 1) клеток вправо (с сохранением считанной информации во внутреннем состоянии головки), затем обратному переходу на (l – 1) клеток влево, записи новых символов по пути и, наконец, движению вправо или влево на l клеток, в соответствии с движением считывающей головки машины *A*.

- В течение этого процесса состояние машины *A*, конечно, сохраняется и в машине *C*. Замена старого состояния новым происходит в конце операции считывания.

Формальная схема построения

- **Начальная лента** машины *С* представляет собой начальную ленту машины *А,* где каждый символ замещен соответствующей ему двоичной последовательностью. Если работа машины *А* начинается с какого-то определенного символа, то работа машины *С* начнется с самого левого двоичного символа соответствующей группы. Если машина *А* начинает работу в состоянии S_j, то машина *С* начнет работу в состоянии T_j.

- Состояниям S_1, S_2, ..., S_n машины *А* мы ставим в соответствие **состояния** $T_1, T_2, \ldots, T_n$ машины *С* (последние будут встречаться, когда машина *С* начинает операцию, считывая первый символ в двоичной последовательности длины *l*).

Формальная схема построения

- Для каждого из этих T_j определим два состояния T_{j0} и T_{j1}.
- Если машина C находится в состоянии T_j и читает символ 0, то она движется вправо и переходит в состояние T_{jo}. Если она читает 1, то движется вправо и переходит в состояние T_{j1}.
- Таким образом, с помощью этих двух состояний машина запоминает, каким был первый символ двоичной последовательности.

Формальная схема построения

- Для каждого из этих T_{j0} и T_{j1} определим опять по два состояния: T_{j00}, T_{j01} и T_{j10}, T_{j11}. Если, например, машина находится в состоянии T_{j0} и читает символ 0, то она переходит в состояние T_{j00} и т. д.

- Таким образом, с помощью этих состояний запоминается начальное состояние и два первых символа, прочитанных в процессе работы машины.

- Продолжим такое построение состояний вплоть до $(l - 1)$ шагов. Получившееся в итоге разнообразие состояний можно обозначить через $T_{j, x1, x2, \ldots, xs}$ где $j = 1, 2, \ldots, n$; $x_j = 0, 1$; $s = 1, \ldots, (l - 1)$.

Формальная схема построения

- Если машина находится в одном из этих состояний (s < l – 1) и читает 0 или 1, то она движется вправо, и 0 или 1 делается дальнейшим индексом состояния. В тот момент, когда s становится равен (l – 1), машина читает последний символ последовательности длины l.
- **Теперь правила операций зависят от конкретных правил машины *A*.**

- Допустим, текущей инструкцией машины *A* была команда:

$$\boxed{S_j a_i \to a_k \frac{R}{L} S_p}$$

Формальная схема построения

- Машина *C* уже готова к выполнению соответствующей инструкции, а значит в дальнейших состояниях должна быть закодирована информация о трех параметрах:
- о **новом символе** a_k, который следует записать на место старого символа a_i,
- о **направлении дальнейшего движения** машины: R или L,
- о номере **нового состояния** S_p.

Формальная схема построения

- Новый символ a_k может быть закодирован двузначным кодом в виде последовательности $y_1, y_2, \ldots, y_{s-1}, y_s$, где y_i=0, 1.
- Определим два новых множества состояний, которые несколько похожи на введенное выше множество состояний *T*, но соответствуют не считыванию, а **записи**: $R_{p, y1, y2, \ldots, ys}$ и $L_{p, y1, y2, \ldots, ys}$
  - название состояния (R или L) будет ***индикатором движения машины А***,
  - первое число в индексе (р) – будет показывать ***номер нового состояния S_p машины А***,
  - индексы $y_1, y_2, \ldots, y_{s-1}, y_s$ – ***значения кода нового символа a_k***.

Формальная схема построения

- Пусть последовательность $x_{1,}$ x_2, …, x_{s-1}, x_s соответствует некоторому символу машины *A.*

- Допустим машина *A* читает этот символ в состоянии S_j, тогда она записывает символ, соответствующий двоичной последовательности $y_{1,}$ y_2, …, y_{s-1}, y_s , переходит в состояние S_p и движется, например, вправо.

- В этом случае, машина *C*, будучи в состоянии $T_{i, x1, x2, \ldots, xl-1}$ и читая символ x_l , переходит в состояние $R_{i, x1, x2, \ldots, xl-1}$, записывает символ y_l и движется влево.

Формальная схема построения

- При $s = 1$ эта запись заканчивается на символе y_1. Остается только передвинуть считывающую голову на l клеток вправо или влево, в зависимости от того, находится ли машина в состоянии R или в состоянии L.

- Это делается с помощью множеств состояний $U_{i,s}$ и $V_{i,s}$ ($i = 1, 2, \ldots, n$; $s = 1, 2, \ldots, l - 1$). В состоянии R_{ix1} машина записывает x_1, движется вправо и переходит в состояние U_{i1}.

Формальная схема построения

- В каждом из состояний *U* машина продолжает движение вправо, не записывая ничего и переходя в состояние *U* со следующим по величине индексом, пока не будет достигнуто последнее состояние *U.*
- $U_{i,s}$ вызывает движение вправо и состояние $U_{i,s+1}$ $(s < l - 1)$. Наконец состояние $U_{i,l-1}$ приводит после движения вправо к состоянию T_i, завершая тем самым цикл. Аналогично, $L_{i,x}$ приводит к движению влево и состоянию $V_{i,1}$. $V_{i,s}$ дает движение влево и $V_{i,s+l}$ $(s < l - 1)$, наконец, $V_{i,l-1}$ дает движение влево и T_i.
- Таким образом показано, как машина *A* преобразуется в эквивалентную ей машину *C* с двумя символами внешнего алфавита, **Q.E.D**.

Оценка числа состояний машины C

Оценим количество состояний машины C *сверху:*

- состояний типа T: $n\ (1+2+4+\ldots+2^{l-1}) = n(2^{l}-1)$
- состояний типа R: $n\ (2^{l-1} +\ldots+4+2)= n(2^{l}-2)$
- состояний типа L: $n\ (2^{l}-2)$
- состояний типа U: $n\ (l-1)$
- состояний типа V: $n\ (l-1)$

=> всего требуется не более $3n2^{l} +2nl -7n$ состояний.

- Напомним, что l - наименьшее целое число, для которого $m \leq 2^{l}$, значит $2^{l}<2m$, и т.о. верхняя граница числа состояний меньше, чем $6mn + n(2l - 7)$, что в свою очередь заведомо меньше чем $8mn$.

Оценка числа состояний машины С

- ИТОГО машина С будет содержать:
  - 2 символа внешнего алфавита: 0 и 1
  - Не более чем **8mn** внутренних состояний (здесь суммарно учтены как аналоги исходных состояний машины А, так и все вводимые дополнительно специальные состояния)

Нормальные алгоритмы

Допустим, что анализ производится укрупнено, не по одному символу, а сразу по несколько. Кроме того, лента является «растяжимой» - т.е. вместо одного символа можно вписывать произвольное их количество и наоборот, без процедуры сдвигания части слова.

Нормальный алгоритм Маркова – *математическое построение, предназначенное для уточнения понятия алгоритм, которое задается алфавитом и нормальной схемой подстановок, выполняемых по заранее определенным правилам.*

Нормальные алгоритмы

- <u>Формат команды</u> (строки) следующий:

$\{a_i\}$ □ $\{b_j\}$ [·],

где

$\{a_i\}$ - последовательность символов, которая ищется в слове,

□ - знак перехода к операции записи,

$\{b_j\}$ - последовательность символов, которая записывается вместо найденной последовательности,

[·] - знак принудительного окончания алгоритма (необязательный параметр).

Λ – служебный знак, обозначающий пустой символ, присутствует везде: изначально на ленте (если она пустая), справа и слева от каждого символа (если на ленте записано слово).

Нормальные алгоритмы

- Программа (алгоритм) представляет собой совокупность строк указанного вида.
- По своей сути основная операция при работе алгоритмов Маркова – это **переработка слов в некотором алфавите.** Эта переработка заключается в производстве некоторого количества замен определенных последовательностей символов. Эти замены совершаются в **СТРОГО определенном порядке,** а именно: **после каждой замены алгоритм читается с самого начала, а слово анализируется с самого первого (левого) символа.**
- Окончание работы алгоритма происходит в тот момент, когда выполняется строка, содержащая знак принудительной остановки, либо тогда, когда более ни одна строка не может быть выполнена (в слове нет ни одной из искомых последовательностей символов).

Нормальные алгоритмы

- В отличие от машин Тьюринга, алгоритмы Маркова выполняются без какого – либо устройства, осуществляющего движения и имеющего внутреннюю память. В данном случае мы можем оперировать только ленточными знаками. Сама лента в этом случае не разделяется на строгие ячейки, а имеет гибкую основу, что позволяет ей растягиваться и сжиматься исходя из того, увеличивается ли в слове число символов или уменьшается.

- Нормальные алгоритмы можно рассматривать как обобщение машины Тьюринга. В свою очередь работу машин Тьюринга можно рассматривать как переработку начального слова некоторого нормального алгоритма. Этот алгоритм получается сразу же из таблицы машины.

- Пусть существует следующая машина Тьюринга, которая печатает на чистой ленте последовательность 001001001001….

Aλ □ 0RB ; Bλ □ 0RC ; Cλ □ 1RA

- Работа алгоритма происходит следующим образом (через запятую указана пара: символ-состояние)

A, 0B, 00C, 001A

- Построим алгоритм Маркова, для чего к внешнему алфавиту {0,1} добавляем внутренний алфавит {A,B,C…}

A □ 0B

B □ 0C

C □ 1A

Λ □ A

- Изначально лента пустая (содержит только символ Λ). Последняя строчка выполнится первой, что позволит записать на ленте символ А. Имея на ленте символ А, в процессе работы алгоритма, мы последовательно получим:

Λ , A , 0B , 00C , 001A , 0010B …и т.д.,

что, по сути, означает бесконечно написание последовательности 001001001…

Тезис Маркова

Таким образом, всегда на основе машины Тьюринга довольно легко можно получить работающий алгоритм Маркова.

Любой нормальный алгоритм можно в свою очередь преобразовать в машину Тьюринга, но это более сложно. Сложности связаны с тем, что у Маркова укрупненный алгоритм, т.к. сразу читается и может быть записано несколько символов.

***Тезис Маркова**: любой вычислительный процесс можно преобразовать в нормальный алгоритм.*