Теоретические основы информатики / Лекция 1

Тема: **«ВВЕДЕНИЕ: Предмет Информатики и ее место в системе наук»**

Что делает наше общество «информационным обществом»?

За свою историю человечество пережило пять информационных революций:

1) Изобретение речи
2) Изобретение письменности
3) Изобретение книгопечатания
4) Изобретение телеграфа и телефона
5) Изобретение компьютеров

Каждый раз новые *информационные технологии* на несколько порядков увеличивали объем хранимой информации, скорость ее распространения и доступность.

В конце-концов, в XXI веке скорость распространения и объем хранимой информации стали практически неограниченными. Более того, существенная доля человеческого труда сегодня затрачивается не на производство материальных благ, а на обработку информации! Именно потому, что производство и потребление информации составляет существенную долю в жизни нашего общества, оно и называется **«информационным».**

Сегодня человек может совершить кругосветное путешествие, посетить другие планеты, увидеть далекие звезды и получить ответы ны любые интересующие его вопросы не выходя из дома. Мы можем гулять по вымышленным мирам населенным эльфами или киборгами-мутантами. Там мы становимся кем-то другим, не тем, кто мы есть в повседневной жизни. Но у заманчивых благ информационного общества есть и обратная сторона.

Устрашающим последствием информатизации является невиданная доселе прозрачность личности. Данные о каждом человеке собираются кредтиными организациями, сотовыми операторами сязи, налоговыми органами и, конечно, спецслужбами. Таким образом, для постороннего вмешательства в личную жизнь человека открываются самые широкие перспективы. Такие, какие и не снились тоталитарным режимам прошлого!

Обилие и доступность информации бумерангом ударило по современному человеку. На нас ежедневно обрушивается «информационный шквал»: телевидение, реклама, городской шум, Интернет и пр. Более того, пути, которыми мы сегодня получаем информацию отличаются от тех, для которых наши органы чувств сформировала эволюция. Быстрая смена кадров нв экране ТВ, приводит к

замедлению развития мозга у маленьких детей, у детей по-старше телевизор нарушает способность избирательно направлять свое внимание. Динамичные компьютерные игры развивают реакцию, но ведут к тому, что дети теряют способность к внутренней речи, начинают скучать на уроках. И даже взрослым игроманам грозит психоз связянный с потерей способности отличать вымысел от реальности.

Во зло мы используем информацию, или во благо завивит от нас. самих А свойста информации и методы ее полезного использования можно изучать с точки зрения науки.

Определение

Проблемы хранения, передачи и преобразования информации изучает наука **Информатика.**

Краткая история развития Информатики.

- **1948 Норберт Винер** «Кибернетика или управление и связь в животном и машине" / Заложены основы теории управления..

- **1948 Клод Шеннон** «Математическая теория связи»
- **1950-е В СССР** Кибернетику объявили «Лженаукой», вместе с генетикой.
- **1960-е** Методологические и общенаучные положения кибернетики в соединении с обобщением практического опыта применения компьютеров привели к созданию более развитой и конкретной системы понятий и методов, составивших основу *информатики*. А кибернетика осталась важным направлением современной методолгии и философии науки, одновременно конкретизируя свои положения и идеи в прикладных науках.
- **Конец 1960-х – начало 1970-х – "Informatique"** французский термин – **"Information+Automatique" -** Coputer Science в англоязычных странах.
- **1978 –** Современное понимание круга задач, которые решает Информатика.
- **1983 –** Создано отделение Информатики, вычислительной техники и автоматизации при АН СССР
- **Сегодня** в мировой практике за Информатикой закрепился англоязычный термин

> Computer Science is no more about computers than astronomy is about telescopes.
>
> Edsger Wybe Dijkstra

В обыденном сознании Информатика связана с компьютерами и пользовательскими приложениями, такими как текстовые процессоры или электронные таблицы. Но самое интересное скрыто, как обычно, внутри. В тех базовых принципах, которые используются при создании, как пользовательских приложений, так и самих компьютеров. Эти базовые знания и составляют **теоретические основы информатики**. Именно **теоретическая информатика** определяет будущее развитие как информатики вцелом, так и ее приложений.

Информатику удобно подразделить на

- ***Теоретическую*** *(раздел прикладной математики)*
  Теоретическая информатика *разрабатывает* математический аппарат исследования процессов хранения, обработки и передачи информации.
- ***Прикладную*** *(разделы прикладной математики, физики, психологии, биологии и пр.)*
  Прикладная Информатика в отличие от *Теоретической,* не ставит своей целью развитие математического аппарата Информатики. Прикладная информатика *использует* математический аппарат Информатики, а также достижения и методы других наук для изучения процессов хранения, обработки и передачи информации.
- ***Практическую*** (Создание прикладных программ)
  Практическая информатика относится к инженерным специальностям. Специалисты в области практической информатики занимаются разработкой *программного обеспечения*, используя достижения Теоретической и Прикладной информатики.
- ***Техническую*** *(раздел техники и технологии, инженерные задачи)*
  Техническая информатика – это тоже инженерная сфера человеческой деятельности. Но в отличие от *Практической Информатики,* к *Технической Информатике* относят вопросы конкретной реализации материальной базы (hardware) процессов хранения, обработки и передачи информации.

Заметим!

Теоретическая Информатка является **фундаментальной естественной** наукой.

Напомним, что *естественными* называются научные дисциплины, которые изучают объективные свойства мира, не зависящие от человеческого сознания.

Фундаменьальными называют такие научные дисциплины, понятия которых носят общенаучный характер, используются в других науках и практической деятельности.

Заметим!

Информатика в целом имеет **гуманитарные** (относятся к наукам об обществе) и **технические** подразделы. Значит, **Информатика** является комплексной междисциплинарной областью знания.

В этом курсе мы с вами познакомимся с Теоретической Информатикой – той базой на основе которой и создаются сегодняшние *информационные технологии*. Без этой базы, Информатика превращается в *магию* – мы используем информационные технологиии, как волшебные артефакты, без понимания того, как они функционируют.

В рамках **Теоретической Информатики**

- Проблемы хранения и передачи информации изучают **Теория информации** и **Теория кодирования**.
- Проблемы преобразования информации изучают **Теория алгоритмов , Теория вычислимости** и **Теория сложности, Теория игр**, **Теория формальных языков и грамматик, Теория автоматов** и другие области Теоретической Информатики**.**

Кроме тех разделов математики, которые относят к Теоретической Информатике, перечислим и некоторые другие разделы математики, которые активно используются в исследованиях теоретических вопросов информатики:

- Математическая Логика
- Теория множеств
- Общая Алгебра
- Линейная алгебра
- Математический анализ
- Теория Вероятностей и математическая статистика
- Комбинаторика
- Теория графов
- Топология

Каковы задачи Теоретической Информатики?

- Построение фундамента Информатики как науки.
- Обеспечение развития прикладной Информатики, практической Информатики и Технической Информатики

Каков предмет Теоретической Информатики?

- Математические методы изучения процессов хранения, передачи и преобразования информации.

Примеры:

1) Как измерять память компьютера определяется в **Теории Информации**

2) Процессоры современных компьютеров построены операторов, свойства которых были изученны в рамках **Математической логики**
3) Языки программирования разработанны на основе резудьтатов **Мат. Логики**, **Теории алгоритмов** и **формальной теории языков и грамматик**
4) Есть задачи для которых решение невозможно вычислить. **Теория вычислимости** говорит какие проблемы не решаемы.
5) Есть задачи, для которых существует алгоритм их решающий, но работать любой такой алгоритм будет дольше, чем просуществует наше Солнце. **Теория сложности** оберегает нас от бесполезной траты времени.

Заключение

Предметом теоретической информатики является математический аппарат исследования процессов хранения, передачи и обработки информации. Оказывается, эти процессы подвержены определенным общим закономерностям, которые делают Теоретическую Информатику, как и Информатику в целом наукой *фундаментальной.* Достижения и методы информатики применяются сегодня в самых различных науках:

- математике (Теория компьютеризированных доказательств, Системы компьютерной математики: Mathematica, MathCad MatLab, Maple и пр.),
- физике (Компьютерные системы управления экспериментом, компьютерные системы математического моделирования),
- психологии (Компьютерные тесты, компьютерные методы статистического анализа результатов эксперимента, достижения в области искусственного интеллекта),
- филологии (формальные грамматики Хомского, компьютерные методы анализа больших корпусов текста),
- экономике (теория игр, системы электронных торгов, компьютеризированные системы экономического моделирования, системы искусственного интеллекта)
- химии (системы виртуальной реальности, системы математического моделирования химических реакций)
- биологии (Анализ ДНК, алгоритмы анализа свертывания белков, алгоритмические подходы к анализу биохимических реакций)

И список можно продолжить! В то же время Информатика является *комплексной междисциплинарной* наукой. Информатика вобрала в себя методы, достижения и, даже проблематику других отраслей научного знания:

- математики (практически все области нашли применение в современной информатике)
- физики (кремниевые полупроводники, квантовые компьютеры, да и современные компьютеры не были бы возможны без соответствующего прогресса в квантовой механике, оптике, электронике и других областях физики)

- биологии (вычислительные устройства, использующие белковые молекулы в качестве «процессоров», вычислительные устройства, содержащие живые клетки, достижения нейрофизиологии)
- психологии (врпосы человеко-машинного интерфейса)

Информатика является относительно молодой наукой. Однако, именно ей принадлежит ведущая роль в развитии современного информационного общества. Сама же информатика строится на математической основе, которую принято называть «Теоретическая информатика». Именно Теоретической Информатике и посвящен этот курс.

Теоретические основы информатики / Лекция 2

Тема: **«Понятие информации»**

Понятие информации

Центральным понятием в Информатике, наверное и не удивительно, является понятие *информации*. Это понятие происходит от латинского informatio – «разъяснение, изложение, осведомленность». Давайте попробуем дать формальное определение этому понятитю!

Очевидно, содержание книги, телепередачи или картины можно отнести к тому, что мы в обыденном понимании называем информацией.

С другой стороны, искусствовед может почерпнуть из одной и той же картины больше информации, чем неспециалист. Так, эксперты могут определить эпоху, географическое место происхождение картины, и, даже, авторство!

Получается, мы не можем утверждать отсутствие в каком-то объекте информации на основании только того, что конкрентно нам в данный момент она недоступна. А не значит ли это, что некая информация содержтся не только в рукотворных объектах, но и скажем в метеорите, упавшем где-то в Антарктиде? Ведь нередко ученым удается определить, откуда он прилетел, и как давно упал на Землю!

Таким образом, *информация* – это некоторая неотъемлимая характеристика *материи* во Вселенной! Каждый камень хранит свой возраст, историю тех рек, на дне которых этот он некогда покоился. Так же, как кольца на стволах деревьев несут информацию обо всех климатических изменениях, что произошли за то время пока дерево было живым. И так же, как свет далеких звезд несет информацию о прошлом нашей Вселенной. Окружающий нас Мир хранит свою собственную историю, бекрайнее море информации окружаеат нас.

Так что, неудивительно, что *теоретическая информатика* является наукой фундаментальной.

В математической теории информации *понятие информации* является элементарным, *неопределяемым*. Также, как понятие точки в геометрии, понятие множества в теории множеств, или понятия времени и материи в физике. Когда мы строим строгую математическую теорию, мы должны *ввести* некоторые понятия, от которых мы должны отталкиваться при определении следующих понятий. Но эти-то первичные понятия определить не из чего! Так и получается, что теории строятся из

- Первичных неопределяемых понятий,
- Аксиом,
- Определений,
- Теорем.

В нашем случае *информация* и является как раз таким первичным неопределяемым понятием.

Правда, теория информации и не ставит целью дать формальное *определение* информации. Так же, как физика не стремиться дать определение *материи.* Так же, как физики *изучают* свойства материи и вводят количественные меры, мы будем изучать свойства информации и научимся измерять ее количественно.

Для того, чтобы можно было применить математические методы для изучения информации, пришлось отвлечься от смысла, содержания информации. В самом деле, как можно строго математически определить смысл? Или хотябы указать количественные свойства смысла?

Начнем строгое построение теории информации с нескольких дополнительных определений. Интуитивно ясно, что в материальном мире информация проявляется в материальных объектах.

Определение (носитель)

Материальный объект, который служит для представления или передачи информации будем называть *материальным носителем* информации.

Примеры:

Для **хранения** информации могут использоваться различные материальные носители: бумага, магнитный диск, камень, дерево. А какие примеры знаете вы?

Для **передачи** информации могут использоваться следующие носители: бумага, электромагнитное поле, лазерный диск. Какие примеры могли бы привести вы?

Любой носитель хранит или передает информацию посредством изменения какой-либо характеристики носителя во времени или пространстве.

Определение (сигнал, параметр сигнала)

Изменение (однократное) некоторой физической характеристики носителя, которое используется для передачи (или хранения) информации назовем *сигналом*. А значение этой характеристики, отнесенное к некорой шкале измерений назовем *параметром сигнала.*

Примеры

1. Изменение частоты колебания возуха – это сигналы передающие человеческую речь. Пераметром такого сигнала является частота воздушных колебаний, выраженная в Герцах.
2. Изменение буквы, которую видит в данный момент ваш глаз при чтении книги– это сигналы передающие информацию, содержащуюся в книге. Параметром сигнала в данном случае являются буквы соотнесенные с некоторым алфовитом. Но книгу можно использовать не только для передачи, но и для хранения информации. Если вы движете указательный палец вдоль строк текста, но не читаете текст, то с изменением координат вашего пальца меняется и та буква, на которую он указывает. В этом случае сигналом является изменение буквы при изменении координаты выделенной области книги. Параметр сигнала остается тот же.
3. Изменение намагниченности при движении по поверхности магнитного диска – это сигналы, записанные на жестком диске компьютера. Параметр сигнала – намагниченность диска, выраженная в некоторых физических единицах.
4. Изменение отражающей способности поверхности лазерного диска вдоль звуковых дорожек – это сигналы записанного на диске звука. Параметром такого сигнала является отражающая способность диска, выраженная в некоторых физических единицах.

Параметры сигнала могут быть

- *дискретные* (параметр сигнала принимает конечное множество значений)
- *и непрерывные* (параметр сигнала может принимать произвольные значения из заданного промежутка)

Определение (сообщение)

Функция f(t) изменения параметра сигнала по времени (или пространственным координатам) называется *сообщением.*

В соответствии с видом функции, сообщения могут быть *дискретными* или *непрерывными.*

Естественно, функция f(t)

- может быть непрерывной на заданном промежутке, тогда говорят, что сообщение *непрерывно по изменению параметра*, Например, непрерывно по времени.

- или принимать определенные значения только для некоторого конечного множества значений параметра, тогда говорят, что сообщение *дискретно* по изменению значения параметра. Например, дискретно по времени.

Какие сообщения возникают в примерах приведенных выше?

Мера количества информации

Как мы отметили, нас не интересует смысл сообщений. А значит, мы можем рассматривать только множество отличных друг от друга событий и, соответсвенно, сообщений о них, без содержания.

Предположим, нас интересует на каком этаже пятиэтажного здания находится деканат? А кого-то другого интересует на каком курсе у него будут читать теоретические основы информатики?

В обоих случаях существует неопределенность интересующего нас события: для него существует пять возможных исходов. Если отвлечься от смысла сообщений, отвечающих на каждый из двух вопросов, то оба ответа просто-напросто выбирают одну из пяти альтернатив. Если договориться, что все пять альтернатив равновероятны, то оба сообщения несут одинаковое количество информации, так как разрешают одинаковое количество *неопределенности*.

Сравним еще два вопроса:

- Какое число выпадет при очередном бросании игральной кости?
- Какой стороной упадет подброшенная монета?

В обоих случаях все исходы равновероятны. Но вот если в первом случае возможно целых 6 равновероятных исходов, то во втором – всего два. Так что, неопределенность, которую снимают ответы на первый вопрос больше, чем во втором случае! Так как до получения ответа число потенциально возможных ответов было больше в первом случае. Это можно сформулировать и иначе: чем меньше вероятность какого-либо события, тем большую неопределенность снимает сообщение о его появлении, и, следовательно, тем большую информацию это сообщение содержит.
Мы можем определить количество информации в сообщении как разность неопределенностей до и после события, информацию об исходе которого это сообщение содержит:

$$I=H1-H2$$

Причем, после события, всякая неопределенность относительно исхода события устраняется. То есть,

$$H2=0.$$

Очевидно, что неопределенность каким-то образом связана с числом возможных исходов. Введем численную характеристику неопределенности и назовем ее *энтропией:*

$$H=f(N)$$

Предыдущий пример наталкивает на мысль взять за меру *неопределенности* число равновероятных исходов.

Например, сообщение связяно с событием S, у которого m равновероятных исходов. Таким событием, например, может быть случайный выбор шаров при розыгрыше лотереи. Тогда, событие, соответствующее выигрышной комбинации n шаров будет иметь

$$N=m^n$$

(m в степени n) возможных тсходов.

Казалось бы, искомая мера f(N)=N количества информации найдена!

Но у меры количества информации, определенной таким образом обнаруживаются, по крайней мере, две проблемы:

1. При m=1 всякая неопределенность отсутствует, так как исход события S получается известен зарание. Но значение N отлично от нуля!
2. Пусть S1 и S2 – два независимых события. Причем, число возможных исходов события S1 равно N1, а события S2 – N2. Тогда число сообщений об исходах событий S1 и S2 равно N=N1*N2. Хотя количество информации, получаемое от двух независимых испытаний должно складываться!

Так как

$$I=H1-H2=H1,$$

количество информации будет обладать свойством аддитивности тогда и только тогда, когда этим свойством будет обладать мера неопределнности - энтропия. Таким образом, энтропия системы с m^n возможными исходами будет в n раз больше энтропии системы с m возможными исходами.

$$f(N)=f(m^n)=nf(m)$$

прологарифмируем обе части равенства

m^n=N,

получим:

log N = n log m, следовательно
n = log N / log m

подставим в выражение f(N) = nf(m) и получим:

f(N) = (f(m) / log m) log N

Для фиксированного m выражение f(m) / log m – константа. Обозначим к = f(m) / log m.

f(N) = к log N

Если мы положим f(N) = log N, это будет решением уравнения, к которому мы пришли полагаясь на свойство аддитивности энтропии. Для этого решения к = log m / log m = 1.

Определение (Кол-во информаци по Р.Хартли)

Пусть N – общее число возможных сообщений, тогда *количество информации,* преходящееся на одно сообщение I(N)=log N.

Формула Хартли не создает те две проблемы, что не позволоили нам определить f(N) тривиально (f(N) = N):

1. При m=1, log m^n = log 1 = 0
2. Пусть число взаимных исходов двух независимых событий равно N=N1*N2. Тогда I(N) = log N = log N1*N2 = log N1 + log N2 = I(N1) +I(N2).

Если все N возможных сообщений равновероятны, то вероятность каждого из них равна 1/N. Тогда формулу количества информации можно переписать:

I(N) = log N = log 1/P = -log P

Заметим:

Мы будем пользоваться альтернативным обозначением количества информации, использующем вероятность, вместо количества равновероятных исходов:

I(P)=I(N), где P – это вероятность одного из N возможных сообщений.

Договоримся использовать в формуле количиства информации логарифм по основанию два. Тогда, единице будет равно количество информации содержащееся в сообщение об исходе подбрасывания симметричной монеты:

$$I(1/2) = - \log_2 1/2 = - (-1) = 1$$

Это один *бит* (BInary uniT).

Предположим, что в результате случайного выбора букв русского языка (которых 33 плюс пробел) из мешка было получено сообщение:

В начале было Слово…

Иоанн, 1:1

Подумайте, насколько невероятно такое событие! И тем более интересно, какое количество информации в нем содержится? Абстрагируемся от смысла сообщения.

Наше составное сообщение содержит информацию о 19 независимых событий, каждое из которых соответствует одному из 34 возможных елементарных сообщений (сигналов, соответствующим буквам русского алфавита). Мы уже знаем, как посчитать количество информации в нашем сообщении:

$$I(P) = - \log_2 P = - \log_2 1/34^{19} = \log_2 34^{19} = 19 \log_2 34$$
$$\sim 96.66 \text{ бит.}$$

Однако в реальности, частота появления различных букв в текстах на русском языке неодинакова! Так что, мы не сможем измерить количество информации в произвольном русскоязычном тексте. Формула Хартли определена только для событий с равновероятным множеством исходов… Что же делать? Вспомним, чему равно количество иноформации содержащееся в сообщении, вероятность которого равна P :

$$I(P) = - \log_2 P$$

Однако, формула эта имеет смысл только в том случае, когда все сообщения равновероятны! Выход был найден Клодом Шенноном (1916-2001). Перепишем формулу количества информации еще раз:

$I(P)= - \sum^{N}_{I=1} P_i \log P_i$, все верно, так как $P_i=1/N$ для любого i – все события равновероятны.

Предположим у нас есть источник S сообщений i, причем, вероятность сообщений теперь не одинакова, и для каждого сообщения своя: P_i.

Клод Шеннон определил величину, которую он назвал *Энтропией*, или мерой неопределенности, источника S.

Определение (Формула Шеннона).

Пусть S – источник сообщений 1 … N. Пусть P_i – вероятность сообщения i. Тогда *энтропия* источника S определяется как *среднее* количества информации, содержащейся в одном сообщении:

$$H(P_1 \ldots P_N) = - \sum_{I=1}^{N} P_i \log P_i$$

Обратите внимание, что в случае равновероятных событий эта формула совпадает с формулой количества информации! Действительно, количество информации, содержащееся в сообщении M – это изменение неопределенности до и после получения этого сообщения:

I(M) = H(до сообщения M) – H(после сообщения M)

В тех случаях, когда каждое сообщение полностью снимает неопределенность связянную с источником информации, H(после сообщения M) = 0.

Давайте, еще раз посчитаем информацию содержащуюся в приведенном выше сообщении, но теперь уже с точки зрения произвольного сообщения на русском языке. Для этого нам понадобится частотность букв русского языка.

i	Символ	P(i)	i	Символ	P(i)	i	Символ	P(i)
1	_	0.175	12	Л	0.035	23	Б	0.014
2	О	0.090	13	К	0.028	24	Г	0.012
3	Е	0.072	14	М	0.026	25	Ч	0.012
4	Ё	0.072	15	Д	0.025	26	Й	0.010
5	А	0.062	16	П	0.023	27	Х	0.009
6	И	0.062	17	У	0.021	28	Ж	0.007
7	Т	0.053	18	Я	0.018	29	Ю	0.006
8	Н	0.053	19	Ы	0.016	30	Ш	0.006
9	С	0.045	20	З	0.016	31	Ц	0.004
10	Р	0.040	2	Ь	0.014	32	Щ	0.003

			1					
11	В	0.038	2 2	Ъ	0.014	33	Э	0.003
						34	Ф	0.002

Если мы подсчитаем количество информации, которое содержится в одной букве русского языка, выданной источником, генерирующим произвольные русскоязычные тексты, то получим:

$H(P_1 \ldots P_{34}) \sim 4{,}72$ бит.

Вспомним свойство аддитивности количества информации:

Свойство (аддитивности количества информации)

Пусть I(M) – количество информации, содержащееся в сообщении M, составленном из m независимых сообщений K_i. Пусть каждое сообщение K_i содержит одинаковое количество информации I(K). Тогда

$I(M)=I(K_1K_2 \ldots K_m)=mI(K)$

По свойству аддитивности количества информации легко подсчитать, что количество нформации, которое содержится в изречении от Иоанна, если оно получено от источника, генерирующего русскоязычные тексты, равно

$I(John) \sim 19 \times 4{,}72 = 90{,}44$ бит.

Кроме того, что Клод Шеннон предложил наиболее удачную меру количества информации, он еще и доказал теорему о максимальном значении функции Энтропии, или меры неопределенности.

Теорема (Шеннона о максимальном значении Энтропии)

Пусть источник S генерирует сообщения $x_1 \ldots x_n$ с вероятностями $P_1 \ldots P_n$, тогда

$H(P_1 \ldots P_n) \leq \log_2 n$

и $H(P_1 \ldots P_n) = \log_2 n$ когда $P_1 = \ldots = P_n = 1/n$

Таким образом, неопределенность источника сообщений в первом примере с высказыванием от Иоанна была максимально возможной, и количество информации, которое было передано самим сообщением было также максимально возможным для сообщения написанного кирилицей.

Как вы видите, одно и то же сообщение может передавать различное количество информации в зависимости от неопределенности, связанной с источником эти сообщения порождающим. Но эта неопределенность не может превышать границу, указанную в теореме Шеннона о *максимальном значении энтропии*.

Заключение

Информациия - это первичное понятие теории информации. Его мы не определяем. Мы отказались рассматривать *смысл* в математической теории информации. Но мы научились измерять *количество информации*. От самого простого случая равновероятных сообщений – формулы Хартли мы прошли путь до формулы Шеннона.

Клод Шеннон ввел удачное определение *энтропии* – меры неопределнности. Через энтропию мы можем определить количество информации, содержащееся в сообщении, как разность между неопределенностью до и после получения сообщения.

Свойство аддитивности количества информации облегчает вычисление по формуле Шеннона количества информации, содержащееся в сообщении. В случае независимых сигналов составляющих сообщение, достаточно измерить количество информации для сигнала, и помножить эту величину на длину сообщения.

Мы завершили лекцию оценкой максимально возможной неопределенности – теоремой Шеннона о максимальном значении энтропии.

Теоретические основы информатики / Лекция 3

Тема: **«Объем информации. информационные процессы. Понятия теории кодирования»**

Повторение

На прошлом занятии мы научились измерять *количество информации*. Для этого мы использовали формулу энтропии Шеннона.

Как мы измеряли количество информации, содержащееся в сообщении?

Мы использовали *формулу Шеннона.*

Мы также изучили формулу Хартли, которая применима, когда все сигналы на выходе источника сообщения равновероятны. Формула Шеннона является обобщением формулы Хартли.

На прошлом занятии мы также познакомились с Теоремой Шеннона «О максимальном значении энтропии». Эта теорема утверждает, что максимальная

неопределенность связяна с ситуацией, в которой все исходы равновероятны. Что соответствует нашему интуитивному пониманию неопределенности.

Объем данных

С одной стороны *сообщение* разрешает некоторую неопределенность. С другой – мы можем говорить о тех ресурсах, которые были затрачены на *представление* информации.

Скажиите, можно ли измерять информацию в килограммах? А в Ваттах?

Познакомимся с еще одной мерой информации, которая получила наибольшее распространение в быту.

Кроме того, какую неопределенность разрешает то или иное сообщение, можно говорить и о том сколько страниц, слов, букв понадобилось для записи этого сообщения.

Определение (Объем данных).
Объем данных в сообщении измеряется количеством символов (разрядов) в этом сообщении.

Какую другую меру объема информации вы могли бы предложить?

Измерим объем сообщения:

> Все книги мира хранят не больше информации, чем транслируется телевидением в одном большом американском городе в течение одного года. Не все биты имеют одинаковую ценность.
>
> Carl Sagan (1934–1996)

Самый простой способ – пересчитать все буквы сообщения. В данном случае их 172. Однако, это же сообщение можно было записать, используя китайские иероглифы. В таком случае, отдельным словам соответствовали бы свои символы, а в приведенной цитате 25 слов. Что же получается, объем информации уменьшился?

Надо договориться о *разрядности*, или символах, в которых мы записываем наши сообщения.

В современных компьютерах используются двоичные разряды, которые, так же как и единица *количества* информации называются *битами*. Кроме того, используется и укрупненная единица – *байт*, равный 8 битам. Единицы можно укрупнять и дальше, по степеням двойки. Для современных ЭВМ приняты следующие укрупненные единицы измерения *объема* данных:

- 1 Килобайт = 1024 бит = 2^{10} бит
- 1 Мегабайт = 1024 Килобайт = 2^{20} бит
- 1 Гигабайт = 1024 Мегабайт = 2^{30} бит
- 1 Терабайт = 1024 Гигабайт = 2^{40} бит
- 1 Петабайт = 1024 Терабайт = 2^{50} бит
- 1 Екзабайт = 1024 Петабайт = 2^{60} бит

Так как двоичные разряды используются в компьютерах для адресации памяти, именно степени 2 обычно определяют максимальный объем памяти ЭВМ.

Есть и десятичная единица объема инфрмации – *дит*.

Пример:

Посчитайте, какой объем информации использован для записи сдедующих сообщений в *битах* и *дитах*:

10011
1234598
2561651

Виды информационных процессов.

Информацию приходится измерять дабы наилучшим образом организовать известные информационные процессы. Мы уже познакомились с тремя информационными процессами, когда давали определение иинформатики – информатика изучает процессы хранения, обработки и передачи информации.

Иногда бывает удобно выделить еще два информационных процесса, которые обычно не рассматривают в рамках теоретической информатики: *получение* и *использование* информации.

1. **Получение информации.** Строго говоря, *получение информации* об объекте состоит в *измерении* количественных или сопоставлении образцам качественных характеристик этого объекта.

   Поясните, пожалуйста, это определение на примере человеческого глаза!

   Измерения обычно сопряжены с погрешностями. Именно погрешности являюься основной характеристикой качества измерения.

Абсолютная погрешность – это отклонение результата измерения от истинного значения измеряемой величины.

Относительная погрешность – это разность между истинным и наблюдаемым значением измеряемой величины, разделенная на истинное значение и помноженное на 100%. Часто удрбно рассматривать модуль *относительной погрешности.*

Приборная погрешность **- это максимально допустимая погрешность, которую гарантирует прибор.**

Методические погрешности – погрешности, вызванные теми етодами, которые использовались для обработки даных после измерения.

2. **Обработка информации.** Процессы обработки информации формализуются математическим понятием *алгоритма.*

   Определение (алгоритм) Под *алгоритмом* мы пока будем понимать некоторое описание последовательности действий.

   Позже мы дадим более строгое определение алгоритма. Само же понятие алгоритма существует на протяжении несколких тысяч лет – со времен Эвклида.

3. **Передача информации.**

   Для того, чтобы изучать характеристики передачи информации безотносительно к природе средства передачи, Клодом Шенноном была предложена общая схема передачи информации (**Схема 1**).

   В момент передачи сообщения по каналу связи сообщение может искажаться в результате внешних воздействий, или внутренних процессов. Например, чем дальше от вас собеседник, тем труднее ему разобрать ваше сообщение в хоре окружающих голосов. В этом примере голоса окружающих – это внешнее воздействие, а затухание колебагий воздуха с расстоянием – внутренний процесс характерный для этого канала связи. Такие нежелательные искажающие сообщение воздействия называются *помехами,* или *шумами*.

   Приведите, пожалуйста примеры каналов связи и шумов, искжающих сообщения по этим каналам передаваемые!

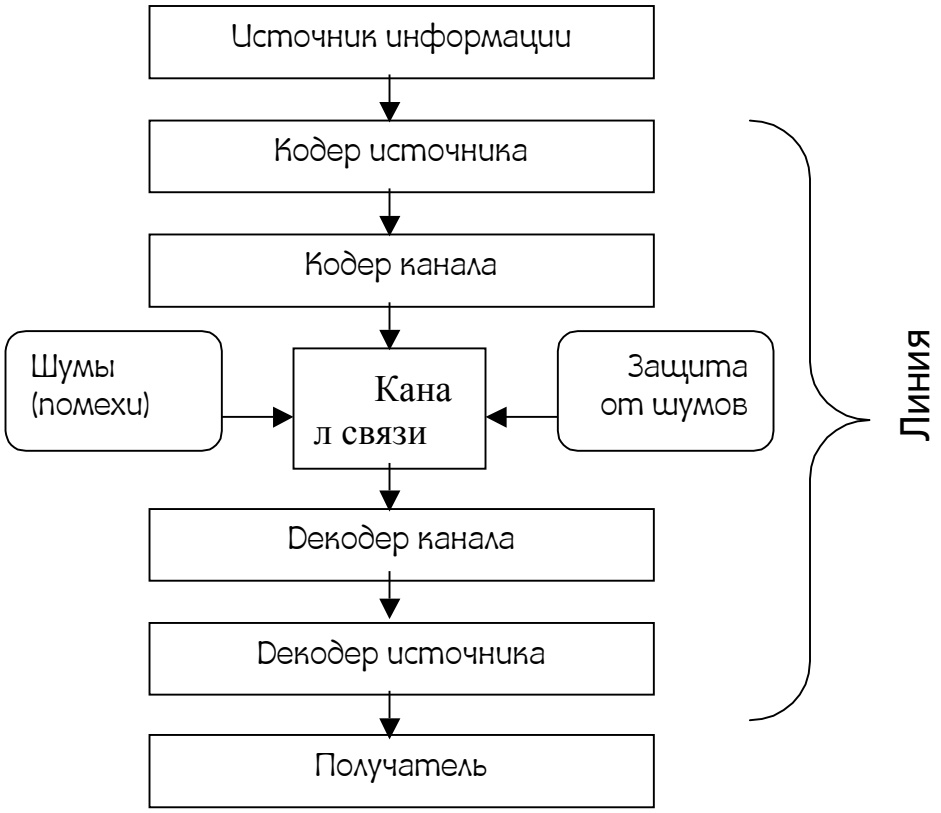


Схема 1

Отметим, что кодер источника и источник инфориации могут быть совмещены. Также возможно, что *получатель* и *источник* используют одинаковое представление информации, тогда не нужен и *декодер источника*.

a. Роль **кодера источника** – перевод исходной информации из произвольной формы в форму сообщения, состоящего из сигналов, понятных кодеру канала.
b. Роль **кодера канала** заключается в переводе информации из формы пригодной для чтения или записи в канал в форму, пригодную для передачи информации по каналу.

Приведите, пожалуйта примеры каналов связи и назовите, что и какие роли (в соответствии с нашей схемой) в ваших примерах выполняет!

Приведите, пожалуйста , примеры, где присутствует и кодер, и декодер источника.

Приведите пример, когда отсутствует кодер канала!

Характеристики канала связи

Основными характеристиками каналов связи являются:

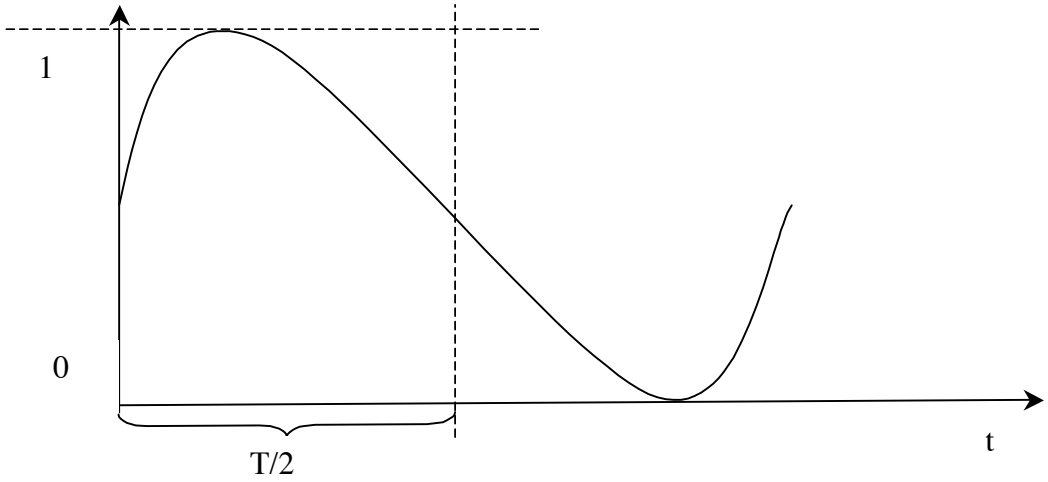
- *Скорость передачи* информации J по каналу связи – это отношение количества информации I, переданной за время t: J=I/t.

Ограничимся рассмотрением только тех каналов связи, где испоьзуются колебания для передачи информации:

- *Ширина полосы пропускания канала* – это интервал частот, используемый для передачи информации данным каналом связи.

Определение (длительность элементарного импульса)

Если параметр сигнала меняется синусоидально, то *длина элементарного импульса* определяется как τ_0=T/2, где T – это период колебаний. *Импульсом* называется та неперерывная область функции параметра сигнала от времени, где значение парамеира сигнала ближе к единице. Та область функции, которая ближе к нуль называется *паузой*.



Изменяя характеристики колебания параметра сигнала так, чтобы в заданный момент времени параметр сигнала находился в области *импульса* или *паузы,* можно передавать двоичные сообщения.

Определение (Пропускная способность) Если с передачей одного импульса связано количество информации I_{imp}, а передается она за время τ_0, то *пропускной способностью канала* называется величина $C=I_{imp}/\tau_0$.

Очевидно, пропускная способность характеризует среднее количество информации, передаваемое по каналу за единицу времени. Измеряют пропускную способность в бит/с.

Заметтье, что скорость передачи информации ограничена сверху пропускной способностью канала: J□ I_{imp}/□.

4. **Хранение информации.** Этот тип информационых процессов можно рассматривать как особый вид передачи информации, где и отправитель и получатель – одно лицо, а время передачи данных велико.

   Однако, принципы хранения информации имеют свою специфику. Особую важность для теоретической информатики имеют, так называемые, *структуры данных*: массивы, стеки, очереди, деревья и др.

5. **Использование информации.** Все что происходит, когда информация, представленная некоторым образом, оказывает влияние на практическую деятельность, относят к процессам *использования информации.*

Теория кодирования

В общей схеме передачи информации, вы не могли не обратить внимание на то, что информация сначала кодируется источником, затем каналом, затем также дважды декодируется.

Двоичное предсавление информации в компьютерах, звуков дуквами алфавита – все этопримеры кодирования информации.

- Разработка принципов *наиболее экономного* представления информации;
- Теория передачи информации по *каналам связи*;
- Разработка приемов повышения *надежности* передачи информации.

  - все это задачи теории кдирования.

Задача кодирования.

Задача кодирования – это задача перевода дискретного сообщения из одного алфавита в другой. Причем такое преобразование не должно приводить к потере информации.

Определение (Код) Правило, описывающее ссоответствие знаков или их сочетаний первичного (исходного) алфавита знакам или их сочетаниям вторичного алфавита называется *кодом.*

Набор знаков *вторичного алфавита,* используемый для представления знаков или их сочетаний первичного алфавита мы также будем называть *кодом.*

Определение (Кодирование) *Кодирование* – это перевод информации, представленной символами первичного алфавита в последовательность кодов.

Определение (Декодирование) Операция обратная кодированию.

Определение (Обратимость) Операции кодирования и декодирования называются *обратимыми,* если их последовательное применение не приводит к потери информации.

Примеры:

- Примером обратимого кодирования является телеграф
- Примером необратимого кодирования является перевод с одного естественного языка на другой.

Матеатически услвие обратимости кодирования можно представить следующим образом:

- Пусть первичный алфавит A состоит из N знаков со средней информацией на знак I^A.
- Пусть вторичный алфавит B состоит из M знаков со средней информацией на знак I^B.
- Пусть I_{st}^A –этоколичество информации в исходном сообщении состоящем из n знаков алфавита A.
- Пусть I_{end}^B – это этоколичество информации в том же сообщении записанном с помощью m знаков алфавита B.

Тогда мы можем записать условие обратимости кодирования:

$$I_{st}^A \geq I_{tnd}^B$$

То есть операция обратимого кодирования не уменьшает количество информации, содержащееся в сообщении.

Будем считать, что символы первичного алфавита появляются на выходе источника независимо друг от друга. Тогда, полученное соотношение можно переписать:

$$nI^A \geq mI^B$$

Отношение $m/n \geq I^A/I^B$ характеризует среднее число знаков вторичного алфавита, которое приходится использовать для кодирования одного знака первичного алфавита. Обратите внимание, частота каждого символаа в сообщении в любом сообщении бесконечной длинны будет одинаковой, и будет стремиться к вероятности появления этого символа на выходе источника. Теперь мы можем строго определить среднее число знаков вторичного алфавита, которое приходится использовать для кодирования одного знака первичного алфавита независимо от сообщения!

Определение (средняя длина кода) Для некоторого источника, первичного алфавита A, состоящего из N символов, вторичного алфавита B и кода Φ определяется как взвешанная сумма кодов для всех символов первичного алфавита, взвешанных соответствующими вероятностями появления этих символов в сообщении.

$$K(\Phi,A,B) = \sum^{N}_{I=1} n_i p_i,$$

Где n_i – это длина кодового слова для i-го символа первичного алфавита.

Если предположить постоянство поведения источника сообщений во времени (*эргодичность*), то предел отношения числа знаков вторичного алфавита к числу знаков первичного алфавита, кодирующих одно и то же сообщение, длина которого стремится к бесконечности, будет, очевидно, стремится к *средней длине кода.*

$$\lim_{n\to\infty} m/n = \lim_{n\to\infty} \sum^{N}_{I=1} n_i l_i(n)/n = \sum^{N}_{I=1} n_i \lim_{n\to\infty} l_i(n)/n =$$
$$= \sum^{N}_{I=1} n_i p_i = K(\Phi,A,B).$$

Поясним, что через $l_i(n)$ мы обозначили количество i-х символов первичного алфавита в произвольном сообщении длина n которого стремится к бесконечности. Длину кодового слова, кодирующего i-ый символ первичного алфавита, мы обозначили через n_i.

Так как

$$nI^A \geq mI^B,$$

минимальное возможное значение длины кода будет

$$K_{min}(A,B) = I^A/I^B.$$

Заключение

На этом занятии мы познакомились с понятием **объема** информации. В отличие от *количества информации, объем* не является характеристикой самого сообщения. *Объем* информации характеризует ресурсы, затраченные на представление сообщения.

Затем, мы познакомились с основными видами информационных процессов: **получение, обрабтка, передача и хранение** информации. Мы рассмотрели принципы, положенные в основу научного изучения этих процессов. В дальнейшем мы будем рассматривать более подробно вопросы *обработки, передачи и хранения* информации. Процесс хранения информации можно рассматривать и с точки зрения передачи информации самому себе по очень медленному каналу связи.

С проблемами хранения информации неразрывно связаны типы представления данных. Основными типами являются: массивы, стеки, очереди, деревья.

Мы также приступили к изучению *теории кодтрования*. Дали определения *кода, кодирования, декодирования, обратимости кодирования, средней длины кода длины кода.*

Теоретические основы информатики / Лекция 4

Тема: **«Теория кодирования. Виды кодирования. Оптимальные коды»**

Теория кодирования

На прошлой лекции мы ввели определение средней длины кода. Вспомнем это определение.

Определение (средняя длина кода) Для некоторого источника, первичного алфавита A, состоящего из N символов, вторичного алфавита B и кода определяется как взвешанная сумма кодов для всех символов первичного алфавита, взвешанных соответствующими вероятностями появления этих символов в сообщении.

$$K(\,,A,B) = \sum_{I=1}^{N} n_i p_i,$$

Где n_i – это длина кодового слова для i-го символа первичного алфавита.

Мы отметили, что Так как

$$nI^A \geq mI^B,$$

минимальное возможное значение длины кода будет

$$K_{min}(A,B) = I^A / I^B.$$

В качестве меры превышения длины кода мы будем использовать *относительную избыточность кода.*

Определение (Относительная избыточность кода) *Относительной избыточностью кода* мы будем называть величину

$Q(\Box,A,B) = (K(\Box,A,B) - K_{min}(A,B))/ K_{min}(A,B) = K(\Box,A,B) / K_{min}(A,B) – 1 = K(\Box,A,B)$ $I^B/ I^A – 1.$

Для фиксированного первичного алфавита A и фиксированного вторичного алфавита B существует множество различных способов построения кода, ставящего символам из алфавита A символы или комбинации символов алфавита B. Для разных вариантов относительная избыточность тоже может быть различной. Кроме того ресурсы (машинная память, время работы), необходимые для кодирования/декодирования разных кодов могут быть различны. Однако, мы не будем рассматривать алгоритмическую оптимальность в рамках Теории Кодирования.

Определение (Оптимальный код) Для некотрого первичного алфавита A и вторичного алфавита B *асимптотически оптимальным кодом* будем называть такой способ кодирования, при котором, если длина сообщений стремится к бесконечности, избыточность кодирования стремится к нулю.

Насколько хороший код можно пстроить? На этот вопрос дает ответ *первая теорема Шеннона:*

Теорема (Основная теорема кодирования при отсутствии шумов)
При отсутствии шумов всегда возможен такой вариант кодирования сообщения, при котором относительная избыточность кода будет сколь угодно близка к нулю.

Первая теорема Шеннона дает нам уверенность в возможности оптиального кодирования. Но не дает рецепта построения такого кода. Именно поэтому теория кодирования не становится тривиальной даже в случае отсутствия шумов.

Кроме уменьшения избыточности кодирования для заданных первичного и вторичного алфавита, уменьшить длину кода можно и правильным выбором

Как видно из определения, существует всего два способа уменьшить минимально возможное значение средней длины кода $K_{min}(A,B) = I^A / I^B$: уменьшить числитель и увеличить знаменатель. Как это сделать? Вспомним, что сообщения, записанные

символами первичного алфавита генерируются нкеоторым источником S, который характеризуется вероятностями появления отдельных символов алфавита A на выходе источника. Учитывая особенности источника можно достичь нашей цели:

- Уменьшить числитель. Этого можно достичь, если учесть различие чатот символов первичного алфавита, корреляции нескольких знаков.
- Увеличить знаменатель. Этого можно достичь, если использовать такой способ кодирования, при котором появление знаков вторичного алфавита было бы равновероятным, то есь $I^B=\log_2|B|$.

Давайте учитывать различные вероятности появления отличных символов первичного алфавита на выходе источника. Но мы будем считать, что корреляций между символами, генерируемыми источником нет. То есть, источник не запоминает, какие символы он уже выдавал, и генерирует новые символы независимо от прошлого. Такие источники называют *источниками без памяти.* Тогда, минимальное значение средней длины кода можно записать следующим образом:

$$K_{min}(A,B) = I^A / \log_2|B|.$$

На практики почти повсеместно в цифровой технике используется двоичное кодирование, то есть |B|=2, а сам вторичный алфавит B состоит из нуля и единицы B={0,1}. Такое кодирование проще всего реализовать. Например, информацию можно хранить как последовательность намагниченных или ненамагниченных участков жесткого диска. Нетрудно видеть, что в этом случае

$$K_{min}(A,2) = I^A$$

Для двоичного котдирования первая теорема Шеннона может быть переформулирована.

Теорема (Вариант теоремы Шеннона). При отсутствии помех средняя длина двоичного кода может быть сколь угодно близкой к средней информации, приходящейся на один символ первичного алфавита.

Формула относителльной избыточности кодирования в случае двоичного кода принимает вид:

$$Q(\square,A,2) = K(\square,A,2) / I^A - 1.$$

Классификация способов кодирования

При кодировании можно использовать следующие особенности вторичного алфавита:

- Элементарные сигналы (0, 1) могут иметь одинаковые длительности, или разные длительности.
- Длина кода может быть одинаковой для всех знаков первичного алфавита (равномерный код), или быть различной (неравномерный код).
- Можно кодировать каждый знак первичного алфавита (алфавитное кодирование), или кодировать блоки символов первичного алфавита (блочное кодирование).

Комбинации перечисленных способов определяют основу конкретного мпособа кодирования. Мы познакомимся с несколькими такими методами.

Неравномерный код с разделителем

Будем кодировать каждую букву первичного алфавита отдельно. Условимся, что разделителем отдельных кодов будет последлвательность двух нулей “00” (признак конца знака). А разделителем слов будет последовательность трех нулей “000” (признак конца слова, пробел). Я предлагаю вам следующие правила построения кодов:

- Код признака конца буквы можно включить в код буквы, так как код признака конца буквы не используется сам по себе, отдельно от кода буквы. То есть все коды букв будут заканчиваться на «00».
- Коды букв не должны содержать двух (или более) нулей подряд нигде кроме как в конце кода. Иначе такой код будет воспринят как два кода двух различных знаков первичного алфавита.
- Код буквы, кроме Иначе ноль в нвчале кода буквы и два нуля в конце кода буквы слева образуют код пробела помимо нашего желания!
- Разделителю слов «000» всегда предшествует признак конца знака. То есть ы конце каждого слова реализуется последовательность «00000». Значит коды букв могут оканчиваться не только на «00», но и на «000», и на «0000». При этом не будеи возникать трудностей с правильным декодированием сообщения.

В соответствии с перечисленными правилами можно построить таблицу кодов для кириллицы.

код	Символ	P(i)	код	Символ	P(i)	код	Символ	P(i)
000	_	0.175	110100	Л	0.035	1111000	Б	0.014
100	О	0.090	111000	К	0.028	1111100	Г	0.012
1000	Е	0.072	111100	М	0.026	10101000	Ч	0.012
1100	Ё	0.072	1010000	Д	0.025	10101100	Й	0.010
10000	А	0.062	1010100	П	0.023	10110000	Х	0.009

10100	И	0.062	1011000	У	0.021	10110100	Ж	0.007
11000	Т	0.053	1011100	Я	0.018	10111000	Ю	0.006
11100	Н	0.053	1101000	Ы	0.016	10111100	Ш	0.006
101000	С	0.045	1101100	З	0.016	11010000	Ц	0.004
1101100	Р	0.040	1110000	Ь	0.014	11010100	Щ	0.003
110000	В	0.038	1110100	Ъ	0.014	11011100	Э	0.003
						11111100	Ф	0.002

У нас есть формула, по которой мы можем найти среднюю длину кода для данного способа кодирования:

$$K(Code1,Cyr,2)= \sum_{I=1}^{34} P_i k_i = 5,5$$

где k_i - это длина i-го кодового слова.

Таким образом избыточность кодирования в данном случае равна:

$$Q(Code1,Cyr,2) = K(Code1,Cyr,2) / I^A - 1 = 5,5 / 4,72 - 1 \sim 0,17$$

Это значит, что при данном способе кодирования мы будем вынуждены передавать на 17% больше информации, чем содержится в исходном сообщении. В итоге загружая канал связи больше, чем требуется!

Существует ли способ кодирования при котором отпала бы необходимость в разделителе знаков?

Префиксные коды

Оказывается, такое возможно!

Условие Фано. Неравномерный код может быть однозначно декодирован, если никакой из кодов не совпадает с началом (префиксом) какого-либо другого, более длинного кода.

Например: Если имеется код «110» то в качестве кодов нельзя использовать последовательности «1», «11», но можно использовать «0», «10» и пр.

Рассмотрим пример префиксного кода:

а	л	м	р	у	ы
10	010	00	11	0110	0111

Попробуйте декодировать сообщене: 00100010000111010101110000110

Начинать следует слева направо, последовательно вычеркивая обнаруженные коды, и записывая соответствующие им знаки первичного алфавита.

Префиксный код Шеннона-Фано

В 1948-1949 гг. Клод Шеннон и Роберт Фано независимо предложили префиксный код, названный в последствие в их честь.

Рассмотрим этот префиксный код на примере. Пусть имеется первичный алфавит: a_1, a_2, … a_6 с вероятностями появления этих символов в сообщении соответственно 0,3; 0,2; 0,2; 0,15; 0,1; 0,05. Расположим эти знаки в таблице в порядке убывания их вероятностей.

Знак	**Вероятность (p_i)**	**Разряды кода**				**Код**
		1	**2**	**3**	**4**	
a_1	0,30	0	0			00
a_2	0,20	0	1			01
a_3	0,20	1	0			10
a_4	0,15	1	1	0		110
a_5	0,10	1	1	1	0	1110
a_6	0,05	1	1	1	1	1111

Кодирование осуществляется следующим образом. Все знаки делятся на две группы, так чтобы суммы вероятностей в кажждой группе были приблизительно равны. В нашем примере в первую группу попадают знаки a_1, a_2, все остальные знаки попадают в другую группу. Установим в ноль первый знак кодов для всех символов из первой группы, и установим равным единицы первый знак кодов всех символов из второй группы. Продолжим деление каждой группы по той же схеме до тех пор, пока не получим группы, состоящие из одного элемента. Эта процедура изображена в таблице.

Полученный код удовлетворяет условию Фано, следовательно он является *префиксным.*

Средняя длина этого кода равна

$$K(\text{Шеннона-Фано, A,2}) = \sum_{I=1}^{N} n_i p_i = 0{,}3*2+0{,}2*2+0.2*2+0{,}15*3 +0{,}1*4+0.05*4 = 2{,}45$$

Среднее количество информации на один символ первичного алфавита равно

$$I^A = -\sum_{I=1}^{N} P_i \log P_i = 2{,}39 \text{ бит.}$$

Теперь по известной нам формуле найдем избыточность кода Шеннона –Фано.

$$Q(\text{Шеннона-Фано, A, 2}) = K(\text{Шеннона-Фано, A, 2}) / I^A - 1 =$$
$$= 2{,}45/2{,}39 - 1 \sim 0{,}025.$$

То есть избыточность кода Шеннона-Фано для нашего игрушечного алфавита составляет всего около 2,5 %.

Для русского алфавита этот избыточность кодирования кодом Шеннона-Фано составила бы примерно 0,0147.

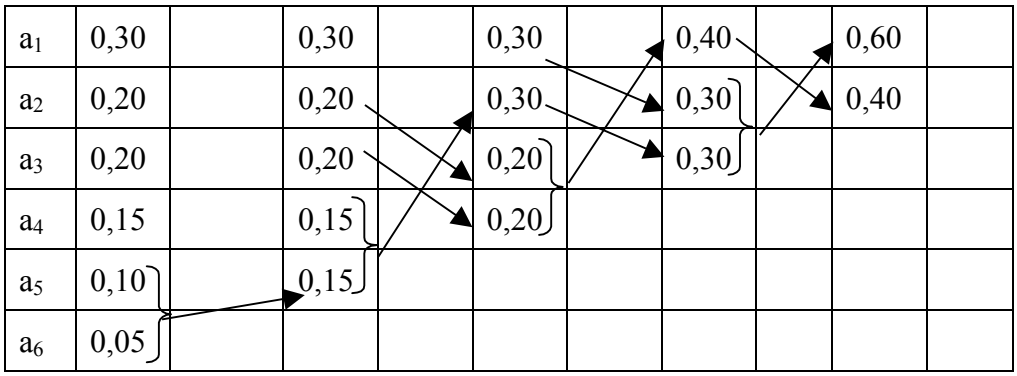
Префиксныей код Хаффмана

В 1952 году Давид Хаффман показал, что предложенный им метод кодирования является *оптимаольным префиксным кодом для дискретных источников без памяти.* Именно такие источники сообщений мы с вами договорились рассматривать.

Алгоритм кодирования методом Хаффмана состоит из двух этапов. На первом этапе исходный алфавит на каждом шаге сокращается на один символ и на следующем шаге рассматривается новый, сокращенный первичный алфавит. На следующем этапе происходит собственно кодирование.

Мы познакомимся с кодом Хаффмана на том же примере, что был рассмотрен при изучении кода Шеннона-Фано.

Знак	Вероятности				
	Исходного алфавита	**Промежуточных алфавитов**			
		A^1	**A^2**	**A^3**	**A^4**



На первом шаге алгоритма два символа исходного алфавита с наименьшими вероятностями объединяются, чтобы получить новый символ. Вероятность нового символа есть сумма вероятностей тех символов, которые в него вошли. Таким образом, получаем новый алфавит, который содержит на один символ меньше чем предыдущий. На следуещем шаге алгоритма та же процедура применяется к новому алфавиту. И так до тех пор, пока в очередном алфавите не остается только двух символов.

Теперь начинается второй этап алгоритма кодирования по Хаффману. Мы нумеруем символы всех промежуточных алфавитов, начиная с последнего. В нашем примере – с A^4.

Знак	Вероятности									
	Исходного алфавита		**Промежуточных алфавитов**							
			A^1		**A^2**		**A^3**		**A^4**	
a_1	0,30	**00**	0,30	**00**	0,30	**00**	0,40	**1**	0,60	**0**
a_2	0,20	**10**	0,20	**10**	0,30	**01**	0,30	**00**	0,40	**1**
a_3	0,20	**11**	0,20	**11**	0,20	**10**	0,30	**01**		
a_4	0,15	**010**	0,15	**010**	0,20	**11**				
a_5	0,10	**0110**	0,15	**011**						
a_6	0,05	**0111**								

В A^4 всего два символа. Они получают соответственно гомера 0 и 1. В алфавите A^3 уже три символа. Причем, один из знаков алфавита A^4, назовем этот знак «предок», был получен объединением двух символов алфавита A^3, назовем первый из этих символов «дочкой», а второй «сыном». Номера этих двух знаков формируются следющим образом. К номеру «предка» приписываются справа 0, чтобы получтиь номер дочки, и 1 – чтобы получить номер «сына». Следующая итерация алгоритма

по той же схеме нумерует знаки алфавита A^2. Процесс останавливается при достижении первичного алфавита A – коды для знаков первичного алфавита получены.

Посчитаем среднюю длину кодового слова для кода Хаффмана и нашего первичного алфавита A.

$$K(\text{Хаффман, A, 2}) = \sum_{I=1}^{N} n_i p_i = 0{,}3*2+0{,}2*2+0.2*2+0{,}15*3 +0{,}1*4+0.05*4 = 2{,}45$$

Эта величина оказывается равной K(Шеннона-Фано, A, 2)! А значит и избыточность кодирования алфавита A кодом Хаффмана будет равной избыточности кодирования алфавита A кодом Шеннона-Фано.

$$Q(\text{Хаффман, A, 2}) = K(\text{Хаффман, A, 2}) / I^A - 1 = K(\text{Шеннона-Фано, A, 2}) / I^A - 1 = Q(\text{Шеннона-Фано, A, 2}) \sim 0{,}025.$$

Однако, в тех случаях когда вероятности символов первичного алфавита сильно разнятся, ситуация меняется. Код Хаффмана обладает существенно меньшей избыточностью.

Например, для русского языка избыточность кодирования кодом Хаффмана оказывается равной примерно 0,0090. Это заметно меньше избыточности кодирования методом Шеннона-Фано!

Другие коды.

В заключение приведем несколько хорошо известных вам кодов, и опишем их в соответствии с нашей классификацией кодов.

- **Азбука морзе** – пример алфавитного кодирования с неравномерной длительностью элементарных сигналов.
- **Код Бодо –** пример равномерного алфавитного двоичного кодирования. Этот код используется для передачи телеграмм. Каждое кодовое слово кода Бодо состоит из пяти символов. Всего можно закоировать 32 знака испоьзуя двоичные последовательности длины пять. Однако, в русском языке используются 34 символа, включая пробел. Именно поэтому в телеграммах “е” и “ё” обозначаются одинаково, так же, как и для твердого и мягкого знака используется один и тот же символ. Кроме того, знаки препинания в телеграммах тоже заменяются словами.
- **Байтовый код –** еще один пример равномерного алфавитного двоичного кодирования. Этот код широко используется в вычислительной технике.

Теоретические основы информатики / Лекция 5

Тема: **«Алгоритм. Машина Тьюринга. Теория автоматов.»**

Развитие понятия алгоритма.

Сегодня мы с вами подробнее познакомимсяс принципами изучения процесса *обработки информации*. В матиматике этот процесс формализуется при помощи понятия *алгоритма*. Мы уже давали неформальное определение этого понятия. Одним из первых алгоритмов известных человечеству является алгоритм Эвклида нахождения *наибольшего общего делителя (300 г. д. н. э.)*. Слово «алгоритм» намного моложе. Оно происходит от имени узбекского математика Мухаммада ибн Муса абу Абдалла аль-Хорезми аль-Маджуси аль-Кутрубулли, жившего в средние века. По книгам этого математика средневековые европейцы изучали арифметику.

Несмотря на долгую историю, долгое время понятие алгоритма не имело строгого математического представления. В 1936 английский математик Алан Тьюринг активно работал над созданием математической модели интеллектуального творчества. Кроме нервного расстройства Алана Тьюринга, его работа подарила человечеству математическую модель, получившую в последствии название «Машины Тьюринга». Был предложен *тезис* , получивший название тезиса *Черча-Тьюринга.*

> Машина Тьюринга может моделировать любой алгоритм, выполнимый на любом физически реализуемом устройстве.

Подавляющее большинство математиков сошлось на том, что этот тезис верен. Однако, доказать его истинность или ложность не представляется возможным. Мы либо верим, либо нет…

Именно тезис Черча-Тьюринга стал отправной точкой науки, которая иначе была бы невозможна, я говорю об Информатике!

Однако, очень скоро математики осознали, что одного только понятия алгоритма недостаточно для изучения обработки информации. В середине 1960-х годов Хартманис (Hartmanis) и Стернс (Stearns) придумали искусственную вычислимую функцию, значение которой не мог вычислить ни один компьютер! Оказалось, что для достаточно длинных входных параметров компьютер будет работать так долго, что он испарится прежде чем закончит вычисления!

Как же можно узнать, как много времени поребуется компьютеру на решение той или иной задачи? Нам придется принять на веру еще одно утверждение, с которым в свое время согласились почти все математики.

Это утверждение получило название «Сильного тезиса Черча-Тьюринга»:

> Машина Тьюринга может *эффективно* моделировать любой физически реализуемый вычислительный процесс.

Так же, как и первое утверждение сильный тезис Черча-Тьюринга является недоказуемым утверждением.

В соответствии с новым подхом к изучению проблем были определены различные классы сложности вычислительных задач. Наиболее широкую известность полусили два:

> **P** – класс проблем вычислимых машиной Тьюринга за время, которое выражается как полином от длины входных данных.
> **NP** – класс проблем, для которых машина Тьюринга может проверить правильность решения за время, которое выражается как полином от длины входных данных.

Понятно, что если мы можем быстро найти решение, мы можем так же быстро проверить решение на правтльность простым сравнением. Так что, P ⊆ NP.

В 1972 Ричард Карп догадался выделить самые трудные задачи из NP в отдельный класс NPC – класс NP-полных задач. Куда вошли те задачи, алгоритм быстрой проверки решения которых можно было использовать для быстрой проверки решения всех задач из NP.

В чем заключается ценность класса NPC? Раньше програмист мог до потери пульса пытаться найти быстрый алгоритм для решения какой-то проблемы. Теперь, если он покажет, что поставленная проблема пренадлежит классу NPC, он сможет заявить, что быстрое решение этой проблемы могло бы решить множество других сложных проблем. То есть, программист может заявить:

> Я не могу решить эту проблему, но и все эти известные люди не смогли решить ее!

А это уже не отмазка, а обоснование сложности поставленной задачи! Так родилась *Теория Сложности.* Этот раздел теоретической информатики изучает вопросы сложности вычисления задач. Причем под *сложностью* понимают большие запросы ресурсов времени и машинной памяти для решения задачи. Не следует путать вычислительную сложность с трудностью построения алгоритма, решающего поставленную задачу!

> **Пример:** Постройте алгоритм, который определяет существует ли такой набор значений аргументов логической формулы, при котором она принимает значение Истина.

Вполне понятно, что проверить подстановку значений на истинность значения формулы можно быстро. А вот можно ли так же быстро решить исходную задачу?

Если вы дадите ответ на поставленный вопрос, вы решите математическую проблему, за которую обещан 1000000 долларов: «Равны или нет классы P и NP?»

С тех пор как были определены классы P и NP, было выполнено много работы. Иерархическая структура классов вычислительной сложности насчитывает Более 300 классов сложности!

А все начиналось когда-то с машины Тьюринга. Она и по мей день является основным инструментом изучения свойств алгоритмов. Собственно говоря, под алгоритмом в математике понимают машину Тьюринга, а под маштной Тьюринга – алгоритм. Прежде чем приступить к изучению машиня Тьюринга нам понадобится познакомиться со специальной терминологией теоретической информатики.

Как описать задачу для маштны Тьюринга.

Попробуйте описать задачу незнакомому человеку, и вы поймете, как трудно описать проблему машине, а точнее машине Тьюринга. Должен быть общий язык, который понятен и нам, и машинам. Но прежде чем мы изучим этот язык, нужно договориться о том, что такое «проблема» или «задача». В математике мы должны строго определять все существенные понятия, которые используем.

Начнем с изучения «словоря» терминов теоретической информатики.

Опр. Алфавит – это любое непустое конечное множество.

Примеры: {1,2,3,4,5,6,7,8,9,0}; {a,b,c,d,e,f}

Опр. Слово (цепочка, строка) над алфавитом Σ – это конечная последовательность символов из алфавита Σ.

Примеры: 1234534; 3432; abba; fade; dgdg.

Замечание: Через Σ^* будем обозначать множество всех конечных последовательностей над алфавитом Σ. А через Σ^n – множество всех последовательностей длины n над алфавитом Σ.

Опр. Для любых двух слов x,y $\in \Sigma^*$ результат бинарной операции **конкатенации** определим как новое слово xy.

Пример: Σ={0,1}; x=010; y=1. Тогда xy=0101, а yx=1010.

Опр. Нам потребуется специальный символ □, который называется **пустой символ.**

Опр. **Язык** - это множество слов.

Опр. **Классом** называется любое (в том числе и бесконечное) множество языков.

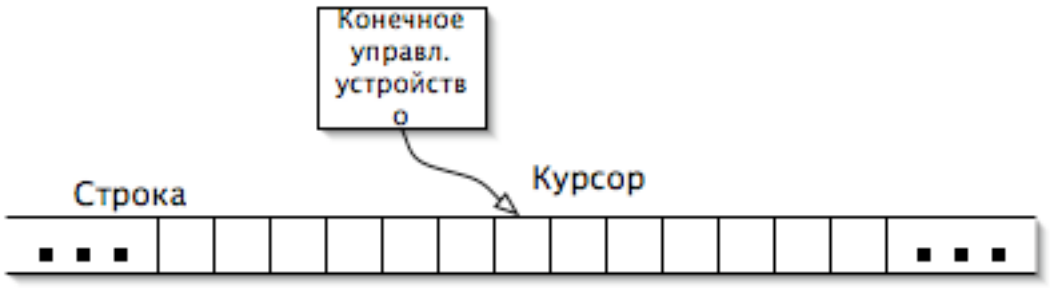
Теперь давайте попробуем определить, формально вычислительную проблему.

Опр. Пусть дана некоторая проблема в произвольной форме. Переформулируем ее так, чтобы решение проблемы для каждого аргумента записанного в некотором фиксированном алфавите □ (в том числе и для пустого аргумента) заключалось в односложном ответе «Да» или «Нет» (1 или 0). **Вычислительной проблемой** назовем *язык*, состоящий из всех слов, которым соответствует положительный ответ в решении поставленной проблемы.

Пример: Проблема: «Определить является ли данное число простым?»
Язык: PRIMES = {w| w – простое число}.

Таким образом, в Информатике под проблемой понимается язык, а под языком – проблема. Язык – это математическая формализация понятия *проблема.* Так же, как машина Тьюринга была математической формализациея понятия *алгоритм.* Вот только мы до сих пор так и не услышали, что же такое машина Тьюринга!

Машина Тьюринга.



Опр. Машина Тьюринга – это четверка M=(K, □, □, s).

1. K – конечное множество состояний, s □ K называется начальным состоянием.
2. □ - алфавит машины M. Мы полагаем, что K и □ - непересекающиеся множества. Множество □ всегда содержит специальные символы □ и □: левый ограничитель и «пустой» символ.

3. Наконец, □ - это *программа* , или *переходное отображение* K□□ в (K□ {h, 1, 0})□□□{□ ,□ ,-}. Мы полагаем h – это состояние останова, 1 – это "принимающее" состояние, а 0 - "отвергающее" состояние. Символы □ ,□ ,- задают движения курсора на один символ влево, на один символ вправо, и оставаться на месте. Мы пологаем символы движения курсора не принадлежат K□ □.

Машина Тьюринга работает в дискретном времени. На каждом шаге можно полностью описать текущее состояние машины, включающее и всю информацию записанную на ленте в данный момент. Естественно, в любой конечный момент времени строка, записанная на ленте будет конечной (несчитая ячеек, заполненных пустым символом □).

Формально конфигурация машины Тьюринга определяется следующим образом.

Опр. Конфигурация машины Тьюринга M состоит из трех элементов: (q, w, u). Здесь q □ K– это текущее состояние машины M, а w, u □ □* - строки. Причем, w – это строка слева от курсора, включая тот символ, на который в данный момент указывает курсор. А u – это строка, возможно пустая, справа от курсора. Пустую строку мы будем обозначать □.

Начальная конфигурация машины Тьюринга всегда (s, □, □).

Опр. Определим **работу** машины Тьюринга M следующим образом. Машина Тьюртнга стартует из начального состояния. На ленте записано входное слово. Дальше машина работает по шагам. Переход от одной конфигурации к другой осуществляется в соответствии с программой (переходным отображением) машины Тьюринга. Машина Тьюринга останавливается, если она достигает одного из *финальных* состояний: h, 1, 0. Если машина Тьюринга M останавливается в состоянии h , мы пишем M(x)=y, где x – это входное слово, а y – это слово, записанное на ленте справа от символа □ вплоть до первого пустого символа □ в момент останова машины. Если машина останавливается в состоянии 1 или 0, мы пишем M(x)=1 или M(x)=0 соответственно. Если машина Тьюринга никогда не остановится, мы пишем M(x)=↑.

В процессе работы машина Тьюринга не должна переходить влево от символа □. То есть, если в программе машины Тьюринга есть состояния p и q такие , что □(q, □) = (p, □, D), то □=□, а D = □ .

Направо курсор может двигаться без ограничений. Мы считаем, что лента справа от входного слова заполнена пустыми символами □.

Опр. Пусть L □ (□-{□})* - некоторый язык. Пусть M – это машина Тьюринга, такая что □ x □ □*, если x□L то M(x)=1, и если x□L то M(x)=0. Тогда мы говорим, что машина M **распознает** язык L.

Опр. Язык, который распознается какой-либо машиной Тьюринга называется **рекурсивным.**

Опр. Пусть L Ã (S-{l })* - некоторый язык. Пусть M – это машина Тьюринга, такая что " x Œ S*, если x L то M(x)=1, и если x L то M(x)=↑. Тогда мы говорим, что машина M **принимает** язык L.

Опр. Язык, который принимается какой/либо машиной Тьюринга называется **рекурсивно перечислимым.**

Приведем пример языка.

Опр. Определим язык O_2 над алфавитом $\{0,1\}^*$ как множество всех слов, которые содержат равное количество нулей и единиц.

Этот язык формализует вычислительнуб проблему, которую можно сформулировать:

Для данного входа определить, является ли поданная на вход двоичная последовательность ***сбалансированной*** *(содержит одинаковое число нулей и единиц).*

Окакзывается язык O_2 *сбалансированных слов* рекурсивный!

Пример. Построим машину Тьюринга, язык O_2. Определим МТ M=<K, , ,s>.

1. Конечное множество состояний K={s,q_0,q_1,q}
2. Алфавит S={–, ,0,1,x}
3. Программа d

p K	sŒÂ	**(p, s)**
s	–	(s, –,Æ)
s	0	(q_0, x,)
s	1	(q_1, x,)
s	x	(s, x,)
q_0	0	(q_0, 0,Æ)
q_0	1	(q, x, ¨)
q_0	x	(q_0, x,)
q_1	1	(q_1, 1,Æ)
q_1	0	(q, x, ¨)
q_1	x	(q_1, x,)
q	0	(q, 0, ¨)
q	1	(q, 1, ¨)
q	x	(q, x, ¨)
q	–	(s, –,Æ)

s	□	(“да”, □,□)
q_0	□	(“нет”, □, □)
q_1	□	(“нет”, □, □)

Пример вычисления на МТ М:

1. s, □011□
2. s, □011□
3. q_0, □x11□
4. q, □xx1□
5. q, □xx1□
6. s, □xx1□
7. s, □xx1□
8. s, □xx1□
9. q_1, □xxx□
10. “нет” □xxx□

Операции над машинами Тьюринга (композиция и итерация алгоритмов)

1. Пусть машины T_1 и T_2 имеют общий алфавит □. Пусть $K_1=\{k_1, \ldots,k_n\}$– множество состояний T_1, а $K_2=\{k_1, \ldots,k_t\}$ – множество состояний T_2. *Композиция* $T= T_1*T_2$ двух машин Тьюринга – это машина Тьюринга Т с тем же рабочим алфавитом □ и набором внутренних состояний $K_1=\{k_1, \ldots,k_n, k_{n+1}, \ldots,k_{n+t}\}$, и программой, эквивалентной последовательному выполнению машин T_1 и T_2.
2. По индукции определяется n-ая *степень* машины Тьюринга Т.
3. Операция *итерации* применима к одной машине Тьюринга и определяется следующим образом. Пусть T_1 имеет несколько заключительных состояний. Выберем ее r-е заключительное состояние и отождествим уго в схеме машины с ее начальным состоянием. Получится машина Т – результат итерации T_1.

Теоретические основы информатики / Лекция 6

Тема: «Теория автоматов. Другие модели вычислений»

Машина Тьюринга – это самая важная и хорошо известная модель вычислений. Однако, существует большое разнообразие различных формализаций алгоритмов. Познакомимся с некоторыми из них.

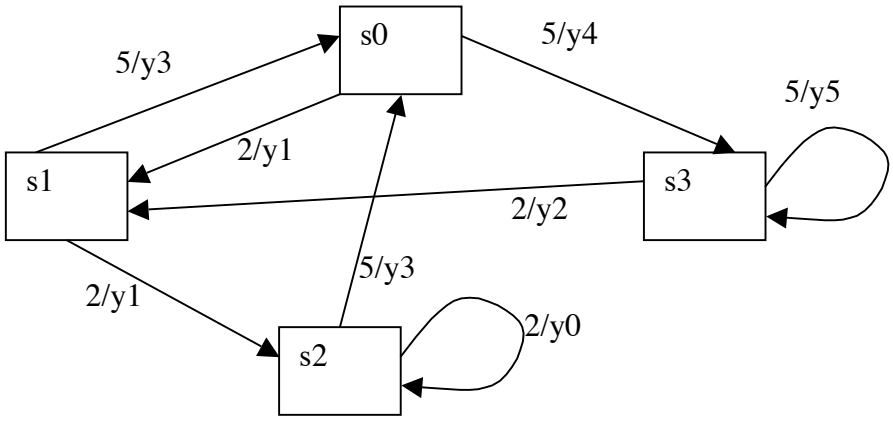
Автоматы.

Если

1. запретить курсору машины Тьюринга двигаться налево и стоять на месте (оставить только движение вправо на каждом шаге), а также
2. запретить МТ останавливаться до тех пор, пока не будет достигнут конец входного слова
3. заставить МТ останавитъся, как только конец входного слова будет достигнут

то получиться вычислительная модель под названием *клнечный автомат.*

Конечные автоматы удобно представлять в виде ориентированных графов:



Здесь:

1. Множество состояний К:
   a. s0 – начальное нейтральное состояние;
   b. s1 – немного расстроен;
   c. s2 – в гневе;
   d. s3 – счастлив.
2. Входной алфавит $\Sigma_{вх}$: {2,5} – пятерки и двойки, которые ученик получает в школе.ё
3. Выходной алфавит $\Sigma_{вых}$:
   a. y0 – брать ремень;
   b. y1 – ругать сына;
   c. y2 – успокаивать сына;
   d. y3 – надеяться;
   e. y4 – радоваться;
   f. y5 – ликовать;

Автомат «Глупый родитель» определяется набором:

ГР = (K, $\Sigma_{вх}$, $\Sigma_{вых}$, δ_K, δ_λ, s)

В отличие от МТ мы отдельно определяем *входной* и *выходной алфавит*. Кроме того, функция переходов определяется отдельно от функции выходов:

δ_K: K×$\Sigma_{вх}$ → K
δ_λ: K×$\Sigma_{вх}$ → $\Sigma_{вых}$

вместо привычной

δ: K×Σ → K×Σ×D.

Такой автомат называется *конечным автоматом типа Мили.*

Опр. Состояние s конечного автомата называется *достижимым*, если существует такое слово g, что этот автомат, начав работу в чвоем начальном состоянии, после чтения этого входного слова, окажется в состоянии s.

Опр. Два автомата A и B называются *эквивалентными,* если
1. их входные алфавиты совпадают,
2. и реализуемые ими отображения входных слов в выходные последовательности также совпадают.

Опр. Если автомат (K, $\Sigma_{вх}$, $\Sigma_{вых}$, δ_K, δ_λ, s) такой, что
δ_K: K×$\Sigma_{вх}$ → K
δ_λ: K → $\Sigma_{вых}$
Такой автомат называется автоматом типа Мура.

Утв. Автоматы типа Мура и типа Мили эквивалентны с точки зрения вычимслительной мощности.

Приведем примеры автоматов типа Мура и типа Мили :

Лекция 7

Кибернетика

Кибернетика – наука, начало которой положил Норберт Винер, в 1948 г. опубликовав книгу «Кибернетика, или управление и связь в животном и машине». Само слово «Кибернетика» происходит от греческого слова «кормчий», или «рулевой». Начала *информатики* были заложены в *кибернетике*, поэтому изучая теоретические основы информатики, мы должны познакомиться с кибернетикой.

Предмет кибернетики

Сегодня предметом кибернетики являются

> системы любой природы, способные воспринимать, хранить, перерабатывать информацию и использовать ее для управления и регулирования.

Разделы *кибернетики:*

1. *Исследование операий*. Под исследованием операций понимают применение математических методов для обоснования принятия решений. Под «операцией» в данном случае понимают *управляемое мероприятие.* А *решением* называют выбор одной альтернативы из числа возможных, лицом, ответственным за проведение этого мероприятия. Решение, наилучшим образом способствующее достижению цели, поставленной организаторами мероприятия, называется *оптимальным.* Исследование операций - классический раздел кибернетики, который сформировался как самостоятельная наука. Именно этот раздел математики принес больше всего Нобелевских премий математикам… правда в области экономики. Именно в этой области человеческой деятельности последствия неправильных решений измеряются в звонкой монете. Исследование операция включает в себя следующие разделы:
    a. *Математическое программирование* – обоснование планов, программ хозяйственной деятельности. Математическое программирование включает в себя:
        - *Линейное программирование;*
        - *Нелинейное программирование;*
        - *Динамическое программирование.*
    b. *Теория массового обслуживания* – опирается на теорию случайны процессов.
    c. *Теория игр* – позволяет обосновать решения, принимаемые в условиях неполноты информации.
2. *Системы автоматизированного и автоматического регулирования.* Этот автономный раздел кибернетики изучает проектирование систем автоматизированного оегулирования и управления технологическими производственными процессами.
3. *Распознование образов.* Этот раздел кибернетики изначально возник из технических потребностей робототехники. Роботам требуется распознавать препятствия, нужные детали при сборке машин и пр. Сегодня распознавание образов является неотъемлимой частью любой системы автоматизированного видеонаблюдения, систем сканирования текстов и др.
4. *Искусственный интеллект.* Раздел, посвященный моделированию интеллектуальной деятельности человека. Сегодня *интеллектуальные технологии* применяются очень широко. Начиная с умных игрушек типа

Sony Aibo, и заканчивая текстовыми процессорами, компьютерными системами перевода, поисковыми машинами и, конечно, автопилотом.

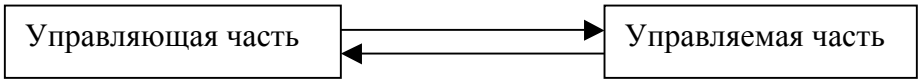
Кибернетика основана на единой методологии изучения *систем,* которая носит название *Теория стсем и системный анализ.*

Система – это предельно широкое понятие, котрое не поддается строгому определению. Приведем несколько подходов к определению систем.

- *Система* – это тень цели на среде, то есть средство достижения некоторой цели. В этом случае под целью понимают какое-то конечное состояние как самой системы, так и окружающего ее мира.
- *Система* обладает структурой, система не является простой суммой элементов. Элементы, объединенные в систему, преобретают какие-то новые качества, не свойственные каждому элементу в отдельности. Например, машина может перевозить пассажиров. Ни одна ее отдельно взятая деталь таким качеством не обладает.

Кибернетика изучает только *управляемы системы.* Характерной особенностью таких систем является их возможность изменять свое состояние под влиянием управляющих воздействий.

Схематическое изображение управляемой системы:



Стрелками обозначены воздействия, т.е. обмен информации. Управляющее учтройчтво вырабатывает сигналы управления на основании информации о состоянии управляемой части системы.

Совокупность правил, по которым информация, поступающая в управляющее устройство преобразуется в управляющие сигналы называется *алгоритмом управления.*

Таким образом, *управление* – это воздействие на объект, выбранное из множества возможных воздействий на основании имеющейся для этого информации. Целью управления является улучшение функционирования или развития управляемого объекта.

Можно выделить четыре основные типа задач управления:

1. *Регулирование* – задача поддержания параметров системы - управляемых величин – на вблизи некоторых заранее заданных *целевых* значений, не смотря на возмущения, которые влияют на эти значения. Примером регулирования является задача, которую решает термостат.

2. *Выполнение программы* – задача, которая возникает, когда *целевые* значения управляемых величин не заданы заранее, а изменяются во времени согласно некоторому заранее известному плану. Примером выполнения программы является развитие зародыша из яйцеклетки.
3. *Слежение* – это задача поддержания управляемых величин как можно ближе к значениям, характеризующим текущее состояние системы, которые в свою очередь меняются непредвиденным образом. Примером задачи слежения является задача определения объема производства в соответствии с изменяющимся спросом.
4. *Оптимизация* – это задача установления наилучшего в определенном смысле режима работы или состояния управляемого объекта. Примером может служить покупка наилучшего автомобиля при минимальных затратах, или задача минимизации потерь при транспортировак продуктов и пр.

Управляемые системы можно подразделить на:

1. *Разомкнутые* – в таких системах информация о значениях, которые принимают управляемые величины в результате управления, не используется при формировании управляющих воздействий.
2. *Замкнутые* – в этих системах при формировании управляющих воздействий используется информация о значениях управляемых величин.

Схема разомкнутой управляемой системы:

Схема замкнутой управляемой системы:

Важным понятием кибернетики является *обратная связь.*

Обратная связь, увеличивающая влияние входных воздействий (возмущений), называется *положительной.* Положительная обратная связь используется в системе управления радиолокатором, следящим за некоторым источником сигнала.

Обратная связь, уменьшающая влияние входных воздейчтвий называется *отрицательной.* Примером системы с отрицательной обратной связью является стекло, темнеющее под воиянием яркого света. Отрицательная обратная связь используется для восстановления в системе равновесия, нарушенного в результате внешнего воздействия на систему.

Функции человека и машины в системах управления

Системы управления хозяйством предприятий или территорий (городов, государств), использующие ЭВМ для переработки и хранения информации, а людей для принятия окончательных решений, получили название Автоматизированных Систем Управления (АСУ).

В АСУ

машина хранит и перерабатывает большие массивы информации. Преобразует информацию к виду, удобному для восприятия и анализа человеком;
человек принимает управленческие решения.

В АСУ компьютеры выполняют рутинную трудоемкуб нетворческую работу, освобождая человеку время для творческой деятельности. Однако, существуют ситуации, в которых физиологические особенности человека не позволяют ему принимать продуманные решения в реальном масштабе времени, а последствия неправильных решений могут быть катастрофическими. Например, ситуации связанные с защитой ядерного реактора, запуска космического аппарата, контроля ускорителя частиц и пр. В таких ситуациях требуется переход к полностью *автоматическим* системам управления, которые невзирая на сложность ситуации исключили бы необходимость участия в них человека. Создание таких систем не только повысило бы безопасность существующих технологий, но и разгрузило бы человека, предоставив ему больше свободного времени. Это было бы по силам системам, обладающим интеллектом, подобным человеческому. Но что случится с

человечеством, если такие системы будут созданы? Будут ли такие системы обладать хоть в какой-то мере правами, присущими человеку?

Задача построения полноценных систем *искусственного интеллекта* является не только трудновыполнимой, но и не однозначной с этической, социальной и даже политической точек зрения.

Лекция 8

Искусссственный интеллект

Три закона роботов:

1) Робот не может причинить вред человеку, или через бездействие позволить чеовеку причинить себе вред;
2) Робот обязан подчиняться приказам, которые ему отдает человек, но только в тех случаях, когда это не противречит Первому Закону.
3) Робот должен защищать свое существование до тех пор, пока это не противоречит первым двум Законам.

Айзек Азимов.

В середине 1960-х годов кто-то впервые поставил вопрос о том, можно ли построить машину, которая будет мыслить. И положил начало исследованиям в области компьютерного моделирования процессов человеческой интеллектуальной деятельности – *Искуственному интеллекту*. Изначально возникнув из попыток Алана Тьюринга построить «математическую модель математиеа», Информатика вернулась к исходной точке. Как определить искусственный интеллект?

Самый общий и до сих пор наиболее популярный из всех известных подходов имеет название «Тест Тьюринга»:

> В две комнаты связанные между собой только компьютерными терминалами помещают двух *человек,* а в третью – *машину*, находящуюся в общей компьютерной сети с первыми двумя комнатами. Испытуемые не знакомы и не знают в какой комнате «кто» находится. Их задача назвать комнату, в которой располагается компьютер, а задача машины обмануть людей – выдать себя за человека. Машина считается прошедшей *тест Тьюринга,* если ей удается одурачить людей.

На сегодняшний день не известно компьютерной системы, которой удалось бы пройти тест Тьюринга.

Другой подход к определению, характеризует интеллектуальные компьютерв через задачи, которые они способны решать:

> Любая задача, для которой не известен алгоритм решения, может быть отнесена к сфере Искусственного Интеллекта.

Четыре периода в истории ИИ:

- **1960-е – нач. 1970-х гг.** Попытки пострить мыслящие машины: моделирование процессов, свойственных человеку (свободный диалог, игры, доказательство теорем, решение задач, создание картин, сочинение стихов и музыки).
- **1970-е гг.** Исследование в области *представления знаний* и формального представления *умозаключений* (формальные преобразования строк).
- **с конца 1970 – по сегодняшний день.** Разработка узко специализированных *экспертных систем*, ориентированных на конкретные предметные области.
- **1990-е гг.** Исследования ЭВМ, построенных на принципах, иных нежели современные универсальные компьютеры.

Направления информатики, где методы искуственного интеллекта находят применение:

1) Восприятие и распознавание образов. Методы использующие «понимание» и логическое рассуждение при обработке визуальной и речевой информации.
2) Автоматическое доказательство математических теорем.
3) Игры. Самай ранняя область применения искуственного интеллекта. Игры характеризуются конечным набором возможных действий, в четко определенных условиях. Соответственно, именно в играх успехи искусственного интеллекта особенно впечетляющи. Уровень человека средних способностей был вскоре превзойден. Однако, искусственному интеллекту не удается так же легко превзойти лучших из людей. Основная сложность на пути ИИ – это отсутствие опыта. Человек испльзует весь жизненный опыт, накопленный за много лет даже при принятии решений в казалось бы ограниченных локальных ситуациях.
4) Решение задач. Например, изобретательство.
5) Понимание естественного языка. Ставится задача анализа и синтеза текстов на естественных языках. Трудности связяны с некоторыми особенностями естественных языков, которые и сделали их универсальным средством общения:

a. неполнота
b. неточность
c. нечеткость
d. грамматическая некорректность
e. избыточность
f. зависимость от контекста
g. неоднозначность

6) Вывявление и представление знаний специалистов в экспертных системах.

Если на заре ИИ ставился вопрос:

> «Может ли компьютер мыслить?»,

сегодня ставится другой вопрос:

> «Достаточно ли хорошо человек понимает как он мыслит, чтобы передать эту способность машине?».

В связи с этим, исследования в ИИ тесно связаны с исследованиями психологии и физиологии человека.

Представление знаний в системах ИИ

В отличие от базы данных, база *знаний* предполагает, что информация хранится в форме пригодной для логического вывода. Проплема поиска формы описания знаний пригодной для использования интеллектуальными алгоритмами называется проблемой *представления знаний.*

Продукционные правила

Самый простой формой представления знаний являются *системы продукций* вида:

> ЕСЛИ A1, A2, … AN, ТО B.

Когда N=0, то есть B истинно всегда, *продукция* называется *фактом.*

Пример:

ЕСЛИ

a. y является отцом x
b. z является братом y

ТО

z является дядей y.

Наряду с базой знаний, интеллектуальная система всегда обладает неким механизмом получения новых знаний из уже имеющихся. Такой механизм называется *механизмом вывода.*

Пример:

Предположим, наряду с продукцией из предыдущего примера в базе знаний содержится *факт:*

ЕСЛИ
ТО
Сергей является отцом Антона,
Семен является братом Сергея.

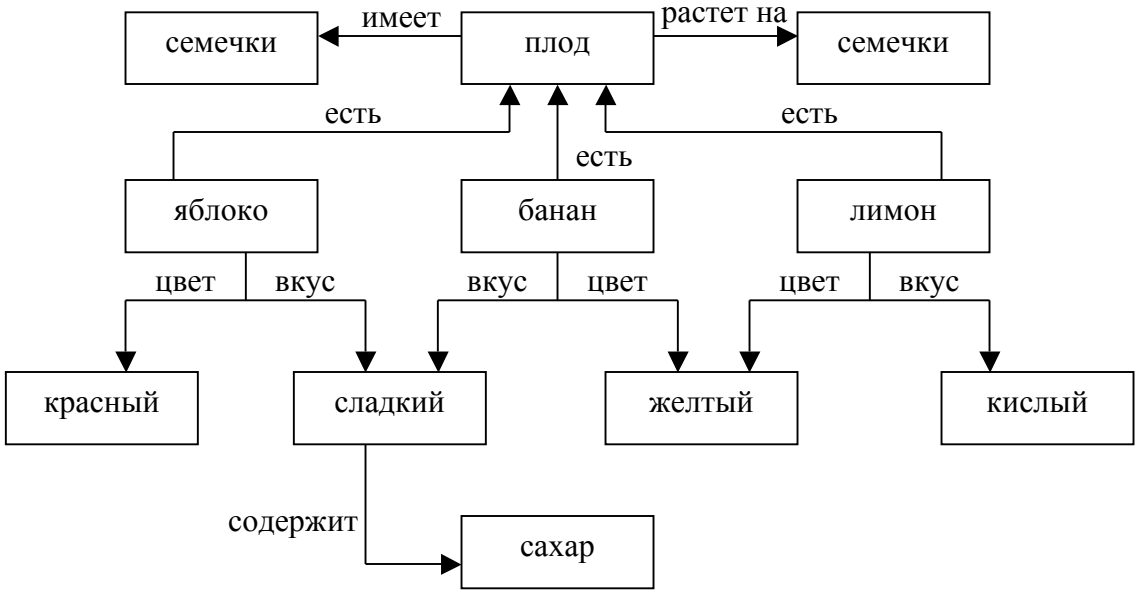
Из имеющихся продукция можно формально вывести, что Семен является дядей Антона.

Механизм, который осуществляет сопоставление предикатов (логических выражений, включающих переменные), фактов а также значений, которые могут принимать переменные в предикатах и является примером *механизма вывода.*

Семантическая сеть

Этот подход основанн на изображении понятий (сущностей) с помощью точек, а отношений между ними с помощью дуг.

Пример: Семанимческая сеть, относящаяся к понятию «фрукты»:



Фреймы

Фрейм – это совокупность *слотов.* Каждый слот имеет *имя,* и *содержание.* В качестве содержания слота может выступать любая *структура данных, процедура* или *другой фрейм.*

Пример: Фрейм человека

Класс: животное
Структурные элементы: голова, шея, руки, ноги, …
Рост: 30-220 см
Вес: 1-200 кг
Хвост: нет
Фрейм аналогии: Обезьяна

Замечание: Существуют и друеие спочобы представления знаний. В частности гибридные методы, основанные на уже перечисленных.

Характеристики машинного представления знаний

1) *Внутренняя интерпретируемость*. У каждой информационной единицы базы знаний есть уникальное в рамках системы имя.
2) *Структурность*. Элементы базы знаний связаны между собой отношениями. Особенно важную роль играют отношения типа «часть – целое», «род – вид», «элемент – класс».
3) *Семантическая метрика*. Между всеми элементами базы знаний устанавливается степень смысловой близости.
4) *Активность*. Источником активности системы является появление новых фактов или описание событий, установление новых связей между элементами базы знаний.

Моделирование рассуждений

Рассуждение – это формирование новых предложений, высказываний, суждений на основе уже имеющихся предложений, высказываний и суждений.

Моделирование рассуждений с помощью логики предикатов первого порядка.

Предикат – это конструкция вида P(x1, … xn), являющаяся утверждением относительно (выражающая связь между) объектами (или свойствами) x1, … xn. Символ P в этом случае называют *предикатным символом,* а символы x1, … xn – *термами.*

Термы могут быть только трех типов:

1) константа;
2) переменная;
3) функция от x1, … xn.

Примеры: впадает(Волга, Каспийское море);
впадает(X, Каспийское море);
впадает(X, Y).

Формулы логики предикатов определяются рекурсивно:

1) Предикат есть формула.
2) Если A, B – формулы, то A, B, A&B, A|B, A->B, ~A – тоже формулы.
3) Если A формула, x – переменная, то ∀xA и ∃xA – тоже формулы.
4) Других формул нет.

Рассмотрим особый вид формул, называемых *фразами Хорна*: B1,B2, … , Bn -> A.
Где : B1,B2, … , Bn, A – предикаты.

Частными случаями фраз Хорна является предикат A, когда все B1,B2, … , Bn являются истинными высказываниями.

Другим частным случаем фраз Хорна является B1,B2, … , Bn -> ложь.
Записывается B1,B2, … , Bn -?. Такую фразу называют *вопросом*. (B->A = ~B|A).

Рассмотрим предметную область: сдачу экзамена. Введем обозначения:

A – студент успешно сдает экзамен;
B – студент посещал занятия;
C – студент освоил учебный материал;
D – студент занимался самостоятельно;
E – студент подготовил шпаргалку.

Ограничим знания о предметной области следующими утверждениями:

Студент успешно сдает экзамен, если студент освоил учебный материал;
Студент освоил учебный материал, если студент посещал занятия и студент занимался самостоятельно;
Студент посещал занятия;
Студент занимался самостоятельно.

В наших обозначениях знания о предметной области можно записать:

C->A
B,D ->C

B
D

Существкет два типа логического вывода.

1) Рассмотрим пример *вывода из фактов:*

B, D, B, D ->C != C
C, C->A != A

2) Рассмотрим пример *ответа на вопрос.*

Немного изменим описание предметной области:

A-?
C->A
B,D ->C

Тогда вывод будет таким:

A-?, C->A != C-?
B,D ->C, C-? != B,D-?
B,D-? != ~B|~D

то еть, мы ответили почему студент не сдает экзамен успешно.

Рассмотрим еще один пример того, как система может дать ответ на ваш вопрос.

дает(логика, сила, вы) –> получает(вы, сила);
дает(логика, сила, вы);
получает(вы, сила)-?;

Вывод:

дает(логика, сила, вы) –> получает(вы, сила), дает(логика, сила, вы)!=
получает(вы, сила);
получает(вы, сила), получает(вы, сила)-? != истина;

То есть противоречие означает положительный ответ на вопрос.

Логика предикатов нашла свою реализацию в языке Пролог. Именно это направлении ИИ особенно популярно в Японии.

Распознавание образов

Так называют совокупность методов и средств автоматического восприятия и анализа окружающего мира.

Задачами теории распознавания являются:

1) Автоматическое чтение печатного или рукописного текста;
2) Восприятие речи;
3) Медицинская, психологическая и педагогическая диагностика;
4) Дистанционная идентификация объектов.

Основные задачи и методы распознавания образов

1) *Сравнение объектов*. Пример: Периодическая таблица Менделеева, классификация растительного и животного мира;

2) *Нахождение принципа классификации* объектов. Пример: распознавание самолетов, составление коллекции монет. Заранее не известен принцип классификации.

3) *Составление словаря признаков.* Пример: изготовление автомата для размена монет. Следует выбрать признаки характерные для каждого типа монет.

4) *Описание классов и объектов на языке признаков.* Каждый объект определяется как вектор, состоящий из значений набора признаков. Например, все треугольники можно определить как трехмерные вектора, состоящии из длин сторон треугольника, который они определяют. Задачу распознавания тогда можно определить как набор функционаов, определенных на векторном пространстве признаков. Каждый из этих функционалов принимает максимальное значение тольео для тех наборов признаков, которые соответствуют объектам, принадлежащих одному классу. Так например, можно определить функционал максимальный только для равнобедренных треугольников. Так же можно описать задачу распознаания буквы из N буквенного алфавита.

5) *Кластеризация в метрическом пространчтве.* Так же, как и в предыдущем случае, объекты представляются векторами. Но теперь, мы определяем расстояние между ними, и в один класс помещаем только соседние объекты.

6) *Метод словаря.* Это кластеризация на основе заданного списка представителей классов – словаря.

7) *Распознавание машиной Тьюринга.* Сторится машина Тьюринга (или автомат), которая распознает заданный *язык*.

8) *Распознавание изображений.* Изображение оцифровывается в виде растра или векторного описания. Затем задача решается одним из вышеперечисленных способов.

Заключение

Основной тенденции в ИИ сегодя является интеллектуальный интерфейс. В идеале, взаимоотношение человека и ммашины должно происходить так же, как и между двумя людьми. То есть человек ставит задачу в произвольной форме, а машина подбирает наилучшее представление для вывода решения. Это снимает не толлько психологическую нагрузку с пользователя, но и делает ненужным специальное обучение работе с ЭВМ.