А.В.Никитин

Механистическое понимание логики.

Оглавление.

Этот разговор мы начали в [11]. Теперь попробуем подойти к логике более детально именно с позиций механистического понимания её возникновения, механизмов реализации, принципов нахождения пути решений, логических действий, и т.д.

Логика[1] понимается нами только в отношении человека. Но это не совсем так, или совсем не так, не знаю. Логика, как система обоснования связи объектов, явлений и действий должна существовать для всех самоуправляемых систем. Для всех или для каждой? Может быть, у каждой системы должна быть своя логика?

Тогда сразу появляется и вопрос соответствия логик. Все ли они будут оценивать логичность действия того или иного самостоятельного интеллекта по одним и тем же критериям логичности и адекватности?

Что мы вкладываем в понятие логичности?

Кажется, пришло время взглянуть на логику несколько под другим углом. Может быть, попробовать рассмотреть её не как человеческий универсальный механизм нахождения ответов на трудные задачи, а как универсальный механический принцип, присущий всем живым существам. Начиная с клетки.

Часть 1. История и реальность.

Чем отличается наше механистическое понимание логики от того, как реально развивалась автономная логическая система той же клетки?

Если сравнивать, то оказывается, что это совершенно разные пути развития механистической логики.

---

[1] **Ло́гика** (др.-греч. λογική «наука о рассуждении», «искусство рассуждения» от λόγος — «речь», «рассуждение») — наука о формах, методах и законах интеллектуальной познавательной деятельности, формализуемых с помощью логического языка. Поскольку это знание получено разумом, логика также определяется как наука о ***правильном*** мышлении. Поскольку мышление оформляется в языке в виде рассуждения, частным случаем которого являются доказательство и опровержение, логика иногда определяется как наука о способах рассуждения или наука о способах доказательств и опровержений. Логика как наука изучает способы достижения истины в процессе познания опосредованным путём, не из чувственного опыта, а из знаний, полученных ранее, поэтому её также можно определить как науку о способах получения ***выводного знания***. http://ru.wikipedia.org/?oldid=40207508

Мы развиваем математическое обоснование каждого шага в движении по пути алгоритма решения логической задачи, добиваясь логичности действий на основе математики.

Клетка идет другим путем. На основе случайных изменений. И, тем не менее, её путь развития логики также вполне обоснован и предопределен. Правда, это становится понятным только сейчас, на новом уровне понимания реальной сложности клеточной системы управления. Мы добрались до понимания истинной роли ДНК и РНК в составе клетки, разобрались во многих клеточных механизмах и процессах. Но непонятного еще так много, что работы тут - на годы.

Сравним…

Мы строили, строили…

Прежде, чем начинать разбираться в сложностях понимания логики автономных систем, надо бы разобраться в том, что сделано в направлении механистического понимания логики истории развития нашей существующей техники. В нашей истории.

Разобраться в том, что сделано в области машинных вычислений, программирования, в том направлении развития искусственного интеллекта, которое сегодня определяет направление развития автоматических систем высокого уровня интеллекта.

Вот, с этого мы и начнем. С того, что сделано…

История.

Не я, и не сегодня, поставил вопрос о механистическом[2] понимании логики. Это было сделано давно.

Но, если рассматривать этот вопрос в привязке к логической машине, на электронной основе, то начинать надо с начала прошлого века.

Интересно, что сегодня существует только одно, и весьма узкое понятие для логической машины. Это – машина вывода[3]. Вот, оказывается, как. Нет уже отдельной электронной логической машины в современной науке и компьютерной технике. Есть только программа…

А машина была, и не одна. У нас в России первыми, наверное, были «интеллектуальные машины» С.Н.Корсакова[4]. Очень интересна «мыслительная машина» А.Н.Щукарева [25]. Она была уже далеко не первой, и тем более, не последней.

С логической машины начинался Искусственный Интеллект[5]. И кибернетика[6]:

---

[2] **Механици́зм** — существовавшие в прошлом метод познания и миропонимание, рассматривающие мир как механизм. В более широком смысле механицизм есть метод сведе́ния сложных явлений к их физическим причинам; противопоставлялся витализму. Благодаря успехам физики в XVI-XVIII векам возникло желание перенести физическое миропонимание на другие науки. В качестве единственного метода подлинной науки рассматривалась математика, понимаемая (ввиду её тогдашнего уровня) в основном механистически. http://ru.wikipedia.org/?oldid=38627662

[3] **Машина вывода** — программа, которая выполняет логический вывод из предварительно построенной базы фактов и правил в соответствии с законами формальной логики. http://ru.wikipedia.org/?oldid=39673690

[4] **Семён Никола́евич Корса́ков** (14 (25) января 1787 — 1 (13) декабря 1853) — русский дворянин, изобретатель механических устройств, «интеллектуальных машин», для информационного поиска и классификации, пионер применения перфорированных карт в информатике[1]. С. Н. Корсаков является пионером русской кибернетики. Основное стремление С. Н. Корсакова — усиление возможностей разума посредством разработки научных методов и специальных устройств. В первой половине XIX века он изобрел и сконструировал ряд действующих механических устройств, функционирующих на основе перфорированных таблиц и предназначенных для задач информационного поиска и классификации http://ru.wikipedia.org/?oldid=40288113

[5] **Иску́сственный интелле́кт** (**ИИ**, англ. *Artificial intelligence, AI*) — наука и технология создания интеллектуальных машин, особенно интеллектуальных компьютерных программ. ИИ связан со сходной задачей использования компьютеров для понимания человеческого интеллекта, но не обязательно ограничивается биологически правдоподобными методами[1]. http://ru.wikipedia.org/?oldid=40614265

[6] **Киберне́тика** (от др.-греч. κυβερνητική — «искусство управления»[1]) — наука об общих закономерностях процессов управления и передачи информации в различных системах, будь то машины, живые организмы или общество. http://ru.wikipedia.org/?oldid=40379334

Термин «кибернетика» в современном понимании как наука об общих закономерностях процессов управления и передачи информации в машинах, живых организмах и обществе впервые был предложен Норбертом Винером в 1948 году[2]. Она включает изучение обратной связи, чёрных ящиков и производных концептов, таких как управление и коммуникация в живых организмах, машинах и организациях, включая самоорганизации. Она фокусирует внимание на том, как что-либо (цифровое, механическое или биологическое) обрабатывает информацию, реагирует на неё и изменяется или может быть изменено, для того чтобы лучше выполнять первые две задачи [3]. Стаффорд Бир назвал её наукой эффективной организации, а Гордон Паск расширил определение, включив потоки информации «из любых источников», начиная со звёзд и заканчивая мозгом.

Пример кибернетического мышления. С одной стороны, компания рассматривается в качестве системы в окружающей среде. С другой стороны, кибернетическое управление может быть представлено как система.

Более философское определение кибернетики, предложенное в 1956 году Л. Куффиньялем (*англ.*), одним из пионеров кибернетики, описывает кибернетику как «искусство обеспечения эффективности действия» [4]. Новое определение было предложено Льюисом Кауфманом (*англ.*): «Кибернетика — исследование систем и процессов, которые взаимодействуют сами с собой и воспроизводят себя».

Вот так. Не больше и не меньше…

Такое понимания кибернетики, как науки, было связано с резким расширением сферы её применения в середине 20 века. Тогда казалось, что кибернетика может всё…

В Википедии читаем:

Кибернетика является междисциплинарной наукой. Она возникла на стыке математики, логики, семиотики, физиологии, биологии, социологии. Ей присущ анализ и выявление общих принципов и подходов в процессе научного познания. Наиболее весомыми теориями, объединяемыми кибернетикой, можно назвать следующие:

Теория передачи сигналов
Теория управления
Теория автоматов
Теория принятия решений
Синергетика
Теория алгоритмов
Распознавание образов
Теория оптимального управления

Кроме средств анализа, в кибернетике используются мощные инструменты для синтеза решений, предоставляемые аппаратами математического анализа, линейной алгебры, геометрии выпуклых множеств, теории вероятностей и математической статистики, а также более прикладными областями математики, такими как математическое программирование, эконометрика, информатика и прочие производные дисциплины.

Сделано, действительно много. Но … постепенно приоритеты менялись, реальным оказалось только компьютерное направление, точнее, программирование. И реальный вес кибернетического подхода стал снижаться. Сегодня уже говорят о кризисе в кибернетике.

И, тем не менее, пока, кибернетика еще объединяет специалистов по логике, ИИ, машинным вычислениям, задавая свое направление движения.

Механистическим пониманием логики активно занимались все ведущие специалисты по кибернетике, вычислительной технике и ИИ. У нас это Д.А.Поспелов[7], Н.М.Амосов [26], В.М.Глушков [27], М.М.Ботвинник[8], Н.П.Брусенцов[9], Г.Н.Поваров[10] и многие другие. Пусть и в разной степени, но занимались все.

---

[7] **Дми́трий Алекса́ндрович Поспе́лов** (родился 19 декабря 1932 года) — советский специалист в области новых методов управления сложными системами, создания ЭВМ новой архитектуры и проблем искусственного интеллекта. Профессор, доктор технических наук. http://ru.wikipedia.org/?oldid=40269970

[8] **Михаил Моисеевич Ботвинник** (4 (17) августа 1911, Куоккала, Выборгская губерния, Великое княжество Финляндское — 5 мая 1995, Москва) — 6-й в истории шахмат и 1-й советский чемпион мира (1948—1957, 1958—1960, 1961—1963). Основатель и бессменный руководитель «Школы Ботвинника», где совершенствовались самые талантливые юные шахматисты Советского Союза. В течение многих лет являлся научным сотрудником Института электротехники. В последние десятилетия жизни в собственной лаборатории работал над проблемой компьютерного моделирования человеческого мышления. http://ru.wikipedia.org/?oldid=39801129

[9] **Никола́й Петро́вич Брусенцо́в** (р. 7 февраля 1925, г. Каменское (ныне Днепродзержинск), Днепропетровская область) — главный конструктор троичной ЭВМ «Сетунь», заслуженный научный сотрудник МГУ. Кандидат технических наук. http://ru.wikipedia.org/?oldid=37256430

Причина этого понятна и вполне очевидна. Логика, как наука, созданная еще в Древней Греции, не всегда отвечает на вопросы, которые ставятся перед ней машинными технологиями, применяемыми сегодня. Классическая логика все так же ориентирована на логический аппарат человека и никак не находит нужных точек соприкосновения с машинными логиками.

В России первое движение в этом направлении[11] сделала воображаемая логика Н.А.Васильева[12]:

> В нашем мире, утверждал ученый, допустимы только "положительные" ощущения, что дает нам возможность различать противоположные качества (говоря, что предмет не белого цвета, человек фактически делает заключение, что предмет красного, зеленого или какого-либо другого цвета) и иметь два качественно различные типы суждений - утвердительные и отрицательные. Если же вообразить мир, в котором возможны не только "положительные", но и "отрицательные" ощущения, то такой мир уже потребует уже иной логики и введения дополнительных качественных суждений. Подобно тому, как евклидова геометрия имеет эмпирическое обоснование через, казалось бы, чувственно "очевидный" пятый постулат, так и логика получает свое эмпирическое обоснование посредством закона (не)противоречия. Если отбросить этот закон, то наряду с утвердительными и отрицательными суждениями становится возможным ввести еще один, отличный от упомянутых, вид суждения, который Васильев назвал индифферентным[4]. Для логики, которая оперировала бы тремя видами суждений, нужен уже не закон исключенного третьего, а закон исключенного четвертого. По мере "усложнения" устройства "воображаемых" миров, усложняется и логика, которая может быть не только двух "измерений" (как аристотелева), но, вообще говоря, любого количества измерений. Однако не все логические законы представляют собой эмпирические обобщения ("материальный" аспект логики). В любой логике имеются законы, делающие возможным само рассуждение ("формальный" аспект логики). Разграничение "формального" и "материального" аспектов в логике предполагает разграничение двух формулировок закона (не)противоречия. Одно дело, когда закон (не)противоречия запрещает одновременное существование двух несовместимых признаков предмета, а другое - когда он гласит, что одно и то же суждение не может одновременно быть истинным и ложным. Первое можно отбросить, как это и делается в воображаемой логике, а второе сохраняет силу для любой мыслимой логической системы (его Васильев предложил назвать законом абсолютного разграничения истины и лжи, или законом несамопротиворечия). Минимум логических законов, необходимых для логического рассуждения, слагает металогику - науку о структурах, общих для всех мыслимых логик. [30]

По этой причине еще в 19 веке началось упорное движение в сторону формализации понятий логики, и в основном, в сторону математики. На основе этой формализации и появились математические логики, применяемые сегодня в компьютерной технике, программировании, кибернетике и робототехнике.

Следствием такого подхода стали: двоичная логика, троичная логика [28, 29], многозначные логики Я.Лукасевича[13], логика Л.Кэролла[14] [31].

Важным шагом в направлении механистического понимания логики стала Машина Тьюринга[15], тем более, если рассматривать и её многочисленные модификации. А.Тьюринг[16] занимался и основами ИИ:

---

[10] **Геллий Никола́евич По́варов** (2 февраля 1928 — 16 ноября 2004) — советский математик, философ и историк науки, профессор кафедры кибернетики Национального исследовательского ядерного университета "МИФИ", действительный член Международной академии информатизации, внесший значительный вклад в развитие отечественной кибернетики, философию науки. http://ru.wikipedia.org/?oldid=38163493

[11] См. Логика в России

[12] **Николай Александрович Васильев** (29 июня 1880, Казань — 31 декабря 1940) — философ, этик, психолог, историк, поэт и переводчик; учёный, предвосхитивший развитие системы воображаемой (неаристотелевой) и основных разделов современной неклассической логики. http://ru.wikipedia.org/?oldid=39883935

[13] **Ян Лукасе́вич** (польск. *Jan Łukasiewicz*; 21 декабря 1878, Львов — 13 ноября 1956, Дублин) — польский логик, член Польской Академии Наук (1937), один из главных представителей львовско-варшавской школы. С 1945 — профессор Королевской ирландской академии в Дублине. Работал в области логических проблем индукции и причинности и логических оснований теории вероятностей. Построил первую систему многозначной логики, а с её помощью — систему модальной логики. Разработал оригинальный язык для формализации логических выражений (т. н. Польская запись, послужившая основой для более известной обратной польской записи). По философским воззрениям — позитивист. http://ru.wikipedia.org/?oldid=40289468

[14] **Льюис Кэ́рролл** (англ. *Lewis Carroll*, настоящее имя **Чарльз Лю́твидж До́джсон** (традиционная передача; английское произношение — Додсон), *Charles Lutwidge Dodgson*; 1832—1898) — английский писатель, математик, логик, философ, диакон и фотограф. http://ru.wikipedia.org/?oldid=40454062

Работы Алана по сооружению первых ЭВМ и развитию методов программирования имели неоценимую важность, дав основу большинству исследований в области искусственного интеллекта. Он полагал, что компьютеры, в конце концов, смогут мыслить как человек, и предложил простую проверку, известную как тест Тьюринга, оценивающую способность машины мыслить: побеседуйте с ЭВМ, и пусть она убедит вас, что она — человек.

Тест Тьюринга[17] до сих пор не прошла ни одна машина, к сожалению…

Уникальность человеческой логики.

Но, вместе с движением к механистическому пониманию логики, к машинным математическим логикам, оценивалась и человеческая логика. Здесь, как мы видим, присутствует другая крайность. Та самая, с которой мы начинали рассказ о логике в [11].

Человеческая логика уникальна. И потому, поразительным оказывается даже сам факт математического доказательства её уникальности. Особенно, в отношении к искусственному интеллекту. К тому, что в любом случае, является пока только частью человеческой логики.

Не может математика в полной мере отразить всю сложность логического аппарата человеческого мозга. Философия всегда указывала на уникальность человеческой логики, принимая её за эталон во всех своих доказательствах и построениях.

И тем не менее…

Давид Гильберт[18] думал иначе. Он считал, что математика в состоянии доказать всё[19], ну почти всё…, и даже доказательство Геделем[20] своей теоремы о неполноте[21] не очень отразилось на оптимизме Гильберта.

---

[15] **Маши́на Тью́ринга (МТ)** — абстрактный исполнитель (абстрактная вычислительная машина). Была предложена Аланом Тьюрингом в 1936 году для формализации понятия алгоритма.

Машина Тьюринга является расширением конечного автомата и, согласно тезису Чёрча — Тьюринга, способна имитировать все другие исполнители (с помощью задания правил перехода), каким-либо образом реализующие процесс пошагового вычисления, в котором каждый шаг вычисления достаточно элементарен. http://ru.wikipedia.org/?oldid=38271396

[16] **А́лан Мэ́тисон Тью́ринг** (англ. *Alan Mathison Turing*; 23 июня 1912 — 7 июня 1954) — английский математик, логик, криптограф, оказавший существенное влияние на развитие информатики. Кавалер Ордена Британской империи (1945). Предложенная им в 1936 году абстрактная вычислительная «Машина Тьюринга» позволила формализовать понятие алгоритма и до сих пор используется во множестве теоретических и практических исследований. http://ru.wikipedia.org/?oldid=40174055

[17] **Тест Тьюринга** — эмпирический тест, идея которого была предложена Аланом Тьюрингом в статье «Вычислительные машины и разум» (англ. *Computing Machinery and Intelligence*), опубликованной в 1950 году в философском журнале *«Mind»*. Тьюринг задался целью определить, может ли машина мыслить. http://ru.wikipedia.org/?oldid=40263859

[18] **Дави́д Ги́льберт** (нем. *David Hilbert*; 23 января 1862 — 14 февраля 1943) — немецкий математик-универсал, внёс значительный вклад в развитие многих областей математики. В 1910—1920-е годы (после смерти Анри Пуанкаре) был признанным мировым лидером математиков. http://ru.wikipedia.org/?oldid=40418102

[19] **Те́зис Чёрча — Тью́ринга** — фундаментальное эвристическое утверждение, существенное для многих областей науки, в том числе, для математической логики, теории доказательств, информатики, кибернетики, дающее интуитивное понятие о вычислимости. Это утверждение было высказано Алонзо Чёрчем и Аланом Тьюрингом в середине 1930-х годов. Позднее были сформулированы другие практические варианты утверждения:

- физический тезис Чёрча — Тьюринга: *любая функция, которая может быть вычислена физическим устройством, может быть вычислена машиной Тьюринга*;
- Сильный тезис Чёрча — Тьюринга (*тезис Чёрча — Тьюринга — Дойча*): *любой конечный физический процесс, не использующий аппарат, связанный с непрерывностью и бесконечностью, может быть вычислен физическим устройством*. http://ru.wikipedia.org/?oldid=40678036

[20] **Курт Фри́дрих Гёдель** (нем. *Kurt Friedrich Gödel*; 1906—1978) — австрийский логик, математик и философ математики, наиболее известный сформулированной и доказанной им теоремой о неполноте. http://ru.wikipedia.org/?oldid=39716167

[21] **Теоре́ма Гёделя о неполноте́** и **втора́я теоре́ма Гёделя**[~ 1] — две теоремы математической логики о принципиальных ограничениях формальной арифметики и, как следствие, всякой формальной системы, в которой можно определить основные арифметические понятия: натуральные числа, 0, 1, сложение и умножение.

Первая теорема утверждает, что если формальная арифметика непротиворечива, то в ней существует невыводимая и неопровержимая формула. Вторая теорема утверждает, что если формальная арифметика непротиворечива, то в ней невыводима некоторая формула, содержательно утверждающая непротиворечивость этой теории. Эти теоремы были доказаны Куртом Гёделем в 1930 году (опубликованы в 1931) и имеют непосредственное отношение ко второй проблеме из знаменитого списка Гильберта. http://ru.wikipedia.org/?oldid=40275423

Но, сегодня этот оптимизм уже не так бесспорен. Если говорить о проблеме, возникшей при доказательстве теоремы о неполноте Геделя [35], то:

«…полагают, что из теоремы К. Геделя о неполноте формальных систем вытекает принципиальное различие между искусственным ("машинным") интеллектом и человеческим умом, а именно, полагают, что теорема Геделя указывает на некоторое принципиальное преимущество человеческого ума перед "умом" машинным - т.е. человек обладает способностью решать проблемы, принципиально неразрешимые для любых искусственных "интеллектуальных" систем (так называемые "алгоритмически неразрешимые" проблемы), причем ограниченность "искусственного ума" проистекает из его "формального" характера».

Когда-то книга Х.Дрейфуса [39] о той же проблеме наделала много шума в стане сторонников ИИ. Сейчас, конечно, времена другие, но, мысли и выводы, сделанные тогда, не потеряли своей актуальности. Примерно в то же время выходит работа Джона Р. Лукаса [40]. О ней много споров и сейчас. Не менее критичен и взгляд Роджера Пенроуза[22]. Примерно к этому выводу приводит и теорема Тарского[23] о невыразимости истины[24].

И все же, наука пока идет по пути Д.Гильберта в понимании человеческого интеллекта. Сегодня развивается направление эволюционных алгоритмов[25] [32, 34]. Новое дыхание получила нечеткая логика[26] Л. Заде[27]. О нечеткой логике мы уже говорили [41]. Математическая логика не оставляет попыток достижения не только Интеллекта человека, но и Разума. Конечно, на основе компьютерного программирования.

Хотя, надо признаться, сложность достижения этой цели сейчас оценивается уже более реально. Даже при сегодняшнем бурном росте компьютерных мощностей и технологий.

Мы, наконец-то поняли, что все предыдущие попытки понимания человеческого интеллекта с позиций машинной оценки не привели к результату. Мало того, они не позволили реально оценить все сложности возможного решения. Мы пока нисколько не приблизились к пониманию, что же такое – логика, как она появилась, и как влияет на

---

[22] Сэр **Ро́джер Пенро́уз** (англ. *Roger Penrose*; род. 8 августа 1931, Колчестер, Англия) — английский учёный, активно работающий в различных областях математики, общей теории относительности и квантовой теории; автор теории твисторов. В 1989 году выпущена его книга «Новый ум короля», в которой автор излагает свои мысли про теорию так называемого сильного искусственного интеллекта, обосновывая несостоятельность воплощения в жизнь такой формы искусственного интеллекта. http://ru.wikipedia.org/?oldid=40446660

[23] **Альфред Тарский** (польск. *Alfred Tarski*; 14 января 1901, Варшава — 26 октября 1983, Беркли, Калифорния) — выдающийся польско-американский математик, логик, основатель формальной теории истинности. Член-корреспондент Британской академии (1966). http://ru.wikipedia.org/?oldid=40453007

[24] **Теорема Тарского о невыразимости истины** — теорема, доказанная Альфредом Тарским в 1936 году, важный ограничивающий результат в математической логике, основаниях математики и формальной семантике. Теорема гласит, что множество истинных формул арифметики первого порядка (т.е. множество их номеров при любой фиксированной гёделевской нумерации) не является арифметическим множеством. Другими словами, понятие арифметической истины не может быть выражено средствами самой арифметики. Теорема Тарского применима к любой достаточно сильной формальной системе. http://ru.wikipedia.org/?oldid=39779973

[25] **Эволюционные алгоритмы** — направление в искусственном интеллекте (раздел эволюционного моделирования), которое использует и моделирует биологическую эволюцию. Различают различные алгоритмы: генетические алгоритмы, эволюционное программирование, эволюционные стратегии, системы классификаторов, генетическое программирование... Все они моделируют базовые положения в теории биологической эволюции — процессы отбора, мутации и воспроизводства. Поведение агентов определяется окружающей средой. Множество агентов принято называть популяцией. Такая популяция эволюционируют в соответствии с правилами отбора в соответствии с целевой функцией, задаваемой окружающей средой. Таким образом, каждому агенту (индивидууму) популяции назначается значение его пригодности в окружающей среде. Размножаются только наиболее пригодные виды. Рекомбинация и мутация позволяют изменяться агентам и приспособляться к среде. Такие алгоритмы относятся к адаптивным поисковым механизмам. http://ru.wikipedia.org/?oldid=39667532

[26] **Нечёткая логика** и теория нечётких множеств — раздел математики, являющийся обобщением классической логики и теории множеств. Понятие нечёткой логики было впервые введено профессором Лютфи Заде в 1965 году. В его статье понятие множества было расширено допущением, что функция принадлежности элемента к множеству может принимать любые значения в интервале [0...1], а не только 0 или 1. Такие множества были названы нечёткими. Также автором были предложены различные логические операции над нечёткими множествами и предложено понятие лингвистической переменной, в качестве значений которой выступают нечёткие множества. http://ru.wikipedia.org/?oldid=40566238

[27] **Лютфи Аскер Заде** (в научных работах обычно *Лютфи Заде* или *Лютфи А. Заде*, (азерб. *Lütfi Zadə — Лютфи Заде* (*Лютфали Аскерзаде*), англ. *Lutfi Asker Zadeh — Лютфи А. Заде*; род. 4 февраля 1921, Новханы, Азербайджанская ССР) — американский математик, основатель теории нечётких множеств и нечёткой логики, профессор Калифорнийского университета (Беркли). http://ru.wikipedia.org/?oldid=40548893

процесс мышления человека. Вот, очень интересная цитата, из работы В.Г.Редько [32], как раз по рассматриваемому нами вопросу:

В то время как математическая логика дает ответы на вопросы: "Каковы правила человеческой логики?" и "Как использовать правила логики?", рассматриваемая здесь пока чисто умозрительно теория происхождения логики могла бы дать ответы на более глубокие вопросы: "Почему правила человеческой логики таковы, каковы они есть?" и "Почему правила логики могут корректно использоваться?"

Резюмируя раздел, подчеркнем, что анализ "интеллектуальных изобретений" биологической эволюции и построение теории происхождения логики представляет собой очень интересную и практически нетронутую область для теоретических исследований. Более того, построение теории происхождения логики чрезвычайно актуально в связи с возможностью определенного обоснования всего научного познания.

Появилось понимание необходимости исследования этой проблемы. Здесь можно только согласиться…

Математика логики.

Когда мы говорим о математической логике, то, как мне кажется, далеко не все представляют себе в полной мере, о чем идет речь.

Ранее, в оценке сделанного учеными–кибернетиками, мы уже затронули теорию автоматов[28].

Теперь продолжим:

Практически теория автоматов применяется при разработке лексеров и парсеров для формальных языков (в том числе языков программирования), а также при построении компиляторов и разработке самих языков программирования.

Другое важнейшее применение теории автоматов — математически строгое нахождение разрешимости и сложности задач.

Там же приведен пример реализации типового автомата:

Терминология:

Символ — любой атомарный блок данных, который может производить эффект на машину. Чаще всего символ — это буква обычного языка, но может быть, к примеру, графическим элементом диаграммы.

Слово — строка символов, создаваемая через конкатенацию (соединение).

Алфавит — конечный набор различных символов (множество символов)

Язык — множество слов, формируемых символами данного алфавита. Может быть конечным или бесконечным.

Автомат:

Автомат[29] — последовательность (кортеж) из пяти элементов (Q,Σ,δ,S_0,F), где:

Q — множество состояний автомата

Σ — алфавит языка, который понимает автомат

δ — функция перехода, такая что $\delta : Q \times \Sigma \to Q$

S_0 — начальное состояние

F — множество состояний, называемых «принимающие состояния».

Слово:

Автомат читает конечную строку символов $a_1, a_2, \ldots, a_n$, где $a_i \in \Sigma$, и называется *словом*.Набор всех слов записывается как Σ*.

Принимаемое слово:

---

[28]**Теория автоматов** — раздел дискретной математики, изучающий абстрактные автоматы — вычислительные машины, представленные в виде математических моделей — и задачи, которые они могут решать. Теория автоматов наиболее тесно связана с теорией алгоритмов: автомат преобразует дискретную информацию по шагам в дискретные моменты времени и формирует результат по шагам заданного алгоритма. http://ru.wikipedia.org/?oldid=40047094

[29]**Абстра́ктный автома́т** (в теории алгоритмов) — математическая абстракция, модель дискретного устройства, имеющего один вход, один выход и в каждый момент времени находящегося в одном состоянии из множества возможных. На вход этому устройству поступают символы одного алфавита, на выходе оно выдаёт символы (в общем случае) другого алфавита. http://ru.wikipedia.org/?oldid=36801037

Слово w ∈ Σ* принимается автоматом, если $q_n \in F$.

Говорят, что язык L *читается (принимается)* автоматом M, если он состоит из слов w на базе алфавита Σ таких, что если эти слова вводятся в M, по окончанию обработки он приходит в одно из принимающих состояний F:

$$L = \{w \in \Sigma^* | \hat{\delta}(S_0, w) \in F\}$$

Обычно автомат переходит из состояния в состояние с помощью функции перехода δ, читая при этом один символ из ввода. Есть также автоматы, которые могут перейти в новое состояния без чтения символа. Функция перехода без чтения символа называется *ϵ-переход* (эпсилон-переход).

Далее, можно самостоятельно сравнить теорию моделей[30], с формальным языком математической логики, да и с самой математической логикой, например, первого порядка[31].

Небольшая цитата:

В математической логике и информатике **формальный язык** — это множество конечных слов (строк, цепочек) над конечным алфавитом. Понятие языка чаще всего используется в теории автоматов, теории вычислимости и теории алгоритмов. Научная теория, которая имеет дело с этим объектом, называется теорией формальных языков.

В теории моделей **язык** соответствует не языку в информатике, а скорее алфавиту. Язык состоит из множеств символов, функций и отношений вместе с их арностью, а также множество переменных. Каждое из этих множеств может быть бесконечным. Из языка вместе с универсальными логическими символами составляются логические высказывания.

Комментарии излишни. Мы говорим об одной математике логики, пусть и в разных вариациях. Теперь ознакомимся с теорией алгоритмов[32]. Читаем:

Развитие теории алгоритмов начинается с доказательства К. Гёделем теорем о неполноте формальных систем, включающих арифметику, первая из которых была доказана в 1931 г. Возникшее в связи с этими теоремами предположение о невозможности алгоритмического разрешения многих математических проблем (в частности, проблемы выводимости в исчислении предикатов) вызвало необходимость стандартизации понятия алгоритма. Первые стандартизованные варианты этого понятия были разработаны в 30-х годах XX века в работах А. Тьюринга, А. Чёрча и Э. Поста. Предложенные ими машина Тьюринга, машина Поста и лямбда-исчисление Чёрча оказались эквивалентными друг другу. Основываясь на работах Гёделя, С. Клини ввел понятие рекурсивной функции, также оказавшееся эквивалентным вышеперечисленным.

Одним из наиболее удачных стандартизованных вариантов алгоритма является введённое А. А. Марковым понятие нормального алгоритма. Оно было разработано десятью годами позже работ Тьюринга, Поста, Чёрча и Клини в связи с доказательством алгоритмической неразрешимости ряда алгебраических проблем.

Следует отметить также немалый вклад в теорию алгоритмов, сделанный Д. Кнутом, А. Ахо и Дж. Ульманом. Одной из лучших работ на эту тему является книга «Алгоритмы: построение и анализ» Томаса Х. Кормена, Чарльза И. Лейзерсона, Рональда Л. Ривеста, Клиффорда Штайна.

Да, за 20-й век было сделано немало в области математики логики. Кибернетика многого добилась. Мы начали разговор о математизации логики в [11, 18,]. Сейчас мы только конкретизируем круг затрагиваемых вопросов.

Например, теория информации[33]. Здесь пересекаются пути развития теории вероятностей[34], математической статистики и криптографии[35]. Неуловимо перетекает

---

[30] **Теория моделей** — раздел математической логики, который занимается изучением связи между формальными языками и их интерпретациями, или моделями. Название **теория моделей** было впервые предложено Тарским в 1954 году. Основное развитие теория моделей получила в работах Тарского, Мальцева и Робинсона. http://ru.wikipedia.org/?oldid=34917877

[31] **Логика первого порядка** (*исчисление предикатов*) — формальное исчисление, допускающее высказывания относительно переменных, фиксированных функций и предикатов. Расширяет логику высказываний. В свою очередь является частным случаем логики высшего порядка. http://ru.wikipedia.org/?oldid=37747937

[32] **Тео́рия алгори́тмов** — наука, изучающая общие свойства и закономерности алгоритмов и разнообразные формальные модели их представления. К задачам теории алгоритмов относятся формальное доказательство алгоритмической неразрешимости задач, асимптотический анализ сложности алгоритмов, классификация алгоритмов в соответствии с классами сложности, разработка критериев сравнительной оценки качества алгоритмов и т. п. http://ru.wikipedia.org/?oldid=34209219

[33] **Теория информации** (математическая теория связи) — раздел прикладной математики, аксиоматически определяющий понятие информации[1], её свойства и устанавливающий предельные соотношения для систем передачи

алгоритм[36] из теории управления[37] в теорию информации, а теперь и в криптографию. Это понятие связывает логику и программирование в единый процесс управления вычислениями. В теорию управления составной частью входит и теория катастроф[38].

Информатика, как относительно новая наука, также внесла свою лепту – алгоритмическую теорию информации[39].

Теория алгоритмов плавно перетекает в понятие комбинаторная логика[40]. Читаем:

Комбинаторная логика и лямбда-исчисление — это такие формальные системы, в которых центральной разрабатываемой сущностью является представление об объекте. В первой из них — комбинаторной логике, — механизм связывания переменных в явном виде отсутствует, а во второй он имеется. Наличие явного механизма связывания предполагает и наличие связанных переменных, но тогда есть и свободные переменные, а также механизмы замещения формальных параметров — связанных переменных, — на фактические параметры, то есть *подстановка*.

Изначальным назначением комбинаторной логики был именно анализ *процесса подстановки*. В качестве ее сущностей планировалось использовать объекты в виде *комбинаций констант*. Лямбда-исчислению отводилась роль средства уточнения представлений об *алгоритме* и *вычислимости*. Как следствие, комбинаторная логика дает в руки инструмент для анализа процесса подстановки. Через короткий промежуток времени оказалось, что обе эти системы можно рассматривать как языки программирования (см. также комбинаторное программирование).

В обеих системах исчисляются объекты, они являются исчислениями или языками *высших порядков*, то есть имеются средства описания отображений или операторов, которые определяются на отображениях или операторах, а в качестве результата вырабатывают также отображения или операторы. Самое существенное,

---

данных. Как и любая математическая теория, оперирует с математическими моделями, а не с реальными физическими объектами (источниками и каналами связи). Использует, главным образом, математический аппарат теории вероятностей и математической статистики.

Основные разделы теории информации — кодирование источника (сжимающее кодирование) и канальное (помехоустойчивое) кодирование. Теория информации тесно связана с криптографией и другими смежными дисциплинами. http://ru.wikipedia.org/?oldid=40252588

[34] **Тео́рия вероя́тностей** — раздел математики, изучающий закономерности случайных явлений: случайные события, случайные величины, их свойства и операции над ними. http://ru.wikipedia.org/?oldid=40482365

[35] **Криптогра́фия** (от др.-греч. κρυπτός — скрытый и γράφω — пишу) — наука о методах обеспечения конфиденциальности (невозможности прочтения информации посторонним) и аутентичности (целостности и подлинности авторства, а также невозможности отказа от авторства) информации. http://ru.wikipedia.org/?oldid=40367811

[36] **Алгори́тм**, от имени учёного аль-Хорезми (перс. خوارزمی [al-Khwārazmī]) — точный набор инструкций, описывающих порядок действий исполнителя для достижения результата решения задачи за конечное время. В старой трактовке вместо слова «порядок» использовалось слово «последовательность», но по мере развития параллельности в работе компьютеров слово «последовательность» стали заменять более общим словом «порядок». Это связано с тем, что работа каких-то инструкций алгоритма может быть зависима от других инструкций или результатов их работы. Таким образом, некоторые инструкции должны выполняться строго после завершения работы инструкций, от которых они зависят. Независимые инструкции или инструкции, ставшие независимыми из-за завершения работы инструкций, от которых они зависят, могут выполняться в произвольном порядке, параллельно или одновременно, если это позволяют используемые процессор и операционная система. http://ru.wikipedia.org/?oldid=40050623

[37] **Тео́рия управле́ния** — наука о принципах и методах управления различными системами, процессами и объектами. Основами теории управления являются кибернетика и теория информации.

Суть теории управления: на основе системного анализа составляется математическая модель объекта управления (ОУ), после чего синтезируется алгоритм управления (АУ) для получения желаемых характеристик протекания процесса или целей управления. Родоначальником непосредственно «**математической теории управления**» можно считать Александра Михайловича Ляпунова — автора классической теории устойчивости движения (1892)[1] http://ru.wikipedia.org/?oldid=40150820

[38] **Теория катастроф** — раздел математики, включающий в себя теорию бифуркаций дифференциальных уравнений (динамических систем) и теорию особенностей гладких отображений. http://ru.wikipedia.org/?oldid=39704595

[39] **Алгоритмическая теория информации** — это область информатики, которая пытается уловить суть сложности, используя инструменты из теоретической информатики. Главная идея — это определить сложность (или описательную сложность, колмогоровскую сложность, сложность Колмогорова-Хайтина) строки как длину кратчайшей программы, которая выводит заданную строку. Строки, которые могут выводиться короткими программами, рассматриваются как не очень сложные. Эта нотация удивительно глубока и может быть использована для постановки и доказательства невозможности некоторых результатов таким же образом, как это делает теорема Гёделя о неполноте и проблема зависания Тьюринга. Эта область была разработана Андреем Колмогоровым, Рэем Соломонoффым (*англ.*) и Грегори Хайтиным в конце 1960-х годов. Существуют несколько вариантов колмогоровской сложности или алгоритмической информации. Наиболее широко используемая базируется на саморазграничивающих программах и в основном следует Леониду Левину (1974). http://ru.wikipedia.org/?oldid=40125273

[40] **Комбина́торная ло́гика** — раздел дискретной математики, который тесно связан с λ-исчислением, т. к. описывает вычислительные процессы. http://ru.wikipedia.org/?oldid=27289029

что именно *отображение* считается объектом. В этом их принципиальное отличие от всего многообразия других систем, в которых первичной сущностью обычно считают представление о *множестве* и его элементах.

К настоящему времени оба эти языка не только стали основой для всей массы исследований в области компьютерных наук и компьютинга, но и широко используются в теории программирования. Развитие вычислительной мощности компьютеров привело к автоматизации значительной части теоретического — логического и математического, — знания, а комбинаторная логика вместе с лямбда-исчислением признаются основой для рассуждений в терминах объектов.

Когда понимаешь эту исходную неразрывность связи математической логики с математикой, то понимаешь, что логика в том понимании, как она сегодня воспринимается, это только часть математики. А совсем не логика…

Не логика формализовалась по требованиям математики, а математика создала себе новую логику. Математическую, с самого начала.

И построено это всё на одном фундаменте – вычислительной машине. Или компьютере, кому как привычнее. Только это было и есть главное в развитии математической логики.

Остальные варианты, как альтернатива этому главному, даже не рассматривались.

Условия для развития автоматической логики.

Логика любой самостоятельной системы, развиваясь в условиях отсутствия каких-либо внешних указателей на метод формирования системы логики, будет следовать примерно одним и тем же путем.

Этот путь предопределен.

Жесткое условие автоматичности проведения логических действий диктует необходимость создания любой счетной системы. Хотя бы единичной. Но, для роста и развития логики этого явно недостаточно. Нужна более развернутая счетная система.

Почему?

Необходимость в счетных операциях возникает из-за требования унификации действий в работе с разными объектами управления. Вот тут и нужна первая форма унификации – счетная единица системы. Просто, как фиксация наличия или отсутствия чего либо. Но это и есть факт введения счетности и единицы счета.

Вторым важнейшим фактором развития счетной системы в логической системе управления надо признать необходимость определения понятий *один - много*. Это требует расширение пределов счета в логической системе за единичный предел. Не важно, сколько, важно – больше единицы. Причем, в целочисленном исчислении.

Третьим фактором развития, вынуждающим расширить счетную систему, является необходимость определения *качественных* показателей *сходства* и *различия* сравниваемых любых логических объектов. Чтобы, хотя бы отличать один объект от другого, или относить их к одному множеству.

Но, даже на этом, первичном уровне определения, логическая система управления вынуждена провести *многозначное совмещение функций* для одних и тех же счетных единиц системы. Счетные эквиваленты количественной оценки становятся еще и эквивалентами качественной оценки при сравнительных действиях. И это происходит автоматически. Просто потому, что другой возможности у системы управления нет.

Появление в логической системе управления качественных показателей автоматически переводит любую, уже созданную *счетную систему количественного учета* в *систему весовых единиц*. Теперь *счетные единицы* системы начинают исполнять функции *весовых эквивалентов* при качественной оценке в сравнении логических объектов. Это объективная необходимость в развитии любой логической системы управления.

Если количественный учет требовал введения хотя бы *двух* разных счетных понятий, как *один – много*, то минимальная качественная оценка приводит к необходимости введения уже *трех* разных счетных понятий. Это *эталон* и *сравниваемые* объекты.

Дальнейшее увеличение количества счетных понятий системы уже определяется уровнем логической системы управления.

Одни и те же счетные эквиваленты применяются уже в *двух* разных логических *действиях*. Как количество, и как – качественная характеристика.

Мы выделили понятие *действие*[41]. И это очень важный момент в понимании самостоятельного развития логической системы управления.

Действие фиксирует любое изменение в системе.

Если бы изменение, вносимое системой в процессе управления своими объектами, всегда имело только одно направление, то оно вообще не фиксировалось бы, как отдельное понятие.

В этом нет необходимости.

Появление любого *другого характера* изменения в системе управления уже требует фиксации. Как *другое действие* по отношению к уже имеющемуся в системе.

Надо отличить *одно* действие от *другого*.

Заметим, что пока мы даже не определяем, какое логическое действие в системе было исходным. И было ли оно вообще…

Разговор не идет о каком-то определенном действии, и тем более, о математическом. Математики еще нет. Есть только её необходимость. Но, как и в каком направлении будет развиваться математика автономной системы управления пока неизвестно. Мы не указываем ни количественных, ни качественных определителей счетной системы. Мы пока знаем только то, что они нужны. И потому, их необходимо определять. Этого требует автоматическая система управления. Пусть клетки…, но уже требует. Иначе она не может существовать.

Как только появилось различие в логических действиях, автоматическая система управления должна их различить.

Это можно сделать единственным способом. Соотнести каждое действие с каким либо имеющимся эквивалентом логической системы, чтобы иметь возможность их различить, хотя бы на уровне сравнения разных эквивалентов.

Какой эквивалент можно взять для этого? Нужен глобальный эквивалент, потому что определение идет на уровне системы.

У системы есть только один вариант. Взять счетные эквиваленты в качестве определителей разных действий. Соотнести каждый счетный эквивалент с каким либо логическим действием. Сколько счетных эквивалентов есть в системе, столько же должно быть и разных логических действий, автоматически исполняемых в любой точке системы управления.

Только при этом условии можно гарантировать *автоматическое исполнение* разных действий и в нужный момент.

Но, это приводит к возникновению *третьей* функции для каждой счетной единицы системы – *эквивалента действия*. Теперь уже действие вносит свои требования в необходимое количество *разных* счетных единиц логической системы.

Количество разных базовых действий должно соответствовать количеству разных счетных единиц логической системы.

Это жесткая необходимость. Исключения из этого правила быть не может. Как бы нам этого не хотелось.

Дальнейшее расширение набора логических действий, включая и математические, идет по пути усложнения их эквивалентов. Но, это уже следующий шаг в развитии системы управления.

---

[41] **действие** — целенаправленная активность, реализуемая во внешнем или внутреннем плане; единица деятельности. В отечественной психологии представления о Д. как специфической единице человеческой деятельности введены С. Л. Рубинштейном и А. Н. Леонтьевым.… **Большая психологическая энциклопедия**

ДЕЙСТВИЕ —структурная единица деятельности, относительно завершенный отдельный акт человеческой деятельности, для которого характерны направленность на достижение определенной осознаваемой цели, произвольность и преднамеренность индивидуальной активности.

Таким образом, *на первом этапе формирования системы управления каждая счетная единица системы становится эквивалентом количественной и качественной оценки, а так же и определителем базового действия*.

Эти функции, со своей стороны, определяют количество разных единиц в счетной системе автономной логики. И очень жестко, надо отметить…

Потому, что только так можно обеспечить автоматическую работу системы логического управления. Других вариантов нет.

По этому пути развития пойдет любая автономная система логического управления.

Это, в свою очередь, создает *единственный путь развития логической системы*, как *системы получения и сравнения эквивалентов*.

Это и есть – логика. В её механистическом понимании.

Развитие машинной логики клетки.

Мы посмотрели, как могли формироваться базовые составляющие для обеспечения автоматической работы логической системы управления. Они позволяют создать исполнительное действие. Но этого явно недостаточно. Надо оценить его эффективность. Получить его *результат.*

С точки зрения автономной логической системы, это начало формирования другого набора базовых понятий – *цель* и *результат*.

Это очень важный момент развития автономной системы управления клетки. Хотя бы потому, что сам момент начала формирования этих понятий связан с развитием понимания *действия*, как основной составляющей логики.

Вполне возможно, что основным формируемым понятием в этот момент стала *цель*, которая и привела к формированию всех остальных понятий логики.

Конечно, понятие цели возникло из жизненно важных для системы управления компонентов. И потому, *контролируемых* на автоматическом уровне. Снижение запаса этих компонентов в системе управления автоматически включало сигнал аварийного состояния и вся система переводилась на *поиск* нужного компонента. Это можно трактовать, как возникновение цели. Как снижение заряда аккумуляторов автоматически включает сигнал необходимости их подзарядки в наших устройствах: ноутбуках, мобильных телефонах…

Но, тут есть важное различие. Сигнал тревоги некому принять. Нет систем, контролирующих этот вид информации, и нет действий, реагирующих на него. Их надо еще создать. Это может быть сделано только на опыте *многократного повторения* такой опасности, как снижение уровня или *отсутствие* самого необходимого. Хотя бы по факту *случайного* исправления ситуации в какой-то момент. Для этого необходимо *запомнить* это исправление ситуации. И *применить*.

Для этого и нужен *результат*. Только он позволяет проводить многократное повторение любых действий в нужном *направлении*. Для достижения *нужного* результата.

Для автономной системы управления клетки это очень сложно. Очень.

На этом этапе происходит дальнейшее увеличение функциональной нагрузки на счетную систему. Счетные и весовые характеристики счетных эквивалентов становятся действительно – многофункциональными. Просто потому, что других компонентов в автоматической системе логического управления – нет. Это и информация, это и действие, это и цель, и результат, и … логический ответ. До него мы еще доберемся.

Но, для этого нужна *память*. Хоть какая. И *каналы управления и контроля*. Нужен сам факт *реакции* на то или иное *управляющее* воздействие.

Собственно, только с этого момента *логическая система* становится *системой управления*.

Причем, сразу – *системой логического управления*. И опять, потому, что других возможностей развития для клетки не существует [11].

Этот переход от логики эквивалентов к логике управления на основе, скорее всего, *случайного сложения необходимых компонентов*, длился очень долго. Миллионы лет.

Слишком много новых компонентов должно было появиться в автономной логической системе клетки.

Должны были появиться устойчивые связи. И память. Самая простейшая, скорее всего, косвенного, а не прямого запоминания информации, т.е. по результату, одному из многих. Может быть, сразу в виде физического объекта, как результата случайного действия.

Но, так или иначе, а *система логического управления автономного объекта* клетки, появилась. Хоть и в самом простом, автоматическом варианте.

Мы этот вариант знаем, как систему *управления по отклонениям*. Вся задача управления этой системы сводится к одному: при возникновении отклонения в работе возвращать параметры управления в состояние «как было». Просто поддерживать *постоянство* режима существования на заданном уровне. Ликвидировать возникающие отклонения. И всё.

Этот режим управления не изменился и сейчас. Система логического управления клетки работает на эту же цель – поддерживать постоянство параметров управления. В любых условиях. И вся логика клетки работает на это.

Но, нет в мире ничего постоянного. Всё меняется. Так или иначе…

Кстати, фиксируя этот этап развития автономной логической системы управления, мы всё так же пока не говорим о конкретной системе счетных единиц, не говорим об уровне сложности, пусть и первого способа автоматического управления *по отклонениям*.

А надо бы сказать.

Сложность и многофункциональность автоматических систем управления по отклонениям может быть огромной. Расширение управления по горизонтали начальных ограничений не имеет. Они появляются только вместе с расширением системы управления, как естественные препятствия этому расширению. В основном – *качественного* характера.

Охватить каналами управления и обратной связи можно все контролируемые объекты, до которых может дотянуться система управления.

Объект есть, вот он, но … что контролировать и чем управлять?

Единого эквивалента, скорее всего, не найдется. Его надо создать. Пусть и случайным перебором всех возможных вариаций. Время есть, вроде бы…, но его – нет. Отклонения возникают, а времени на поиск эквивалентов нет. Надо их искать в процессе управления.

Аварийную ситуацию надо менять срочно, и тут любой *исход* может быть и правильным, и … последним, если он неверный. Условия управления оказываются очень жесткими.

Конечно, такие острые моменты в процессе управления встречаются относительно редко, чаще есть и время, и возможности для реализации функции управления по отработанным *шаблонам*. Да, конечно, шаблоны стали первыми *стандартными схемами* управления. Они возникли из случайных факторов, зафиксированных изменений, запомненных системой.

Чтобы понять всю сложность управления на этом этапе развития автономной системы логического управления той же клетки, надо сравнить её с существующими у нас автоматами. И оказывается, все автоматические системы, созданные человеком, находятся именно на этом уровне управления. Все. Даже самые сложные, компьютерные …

А для клетки это был только первый этап развития системы управления.

Делайте выводы…

Пока мы просто перечислим понятия, возникшие на этом этапе логики. Это *сходства* и *различия*, *сравнение* и *выбор*, *управление* и *контроль*, *действие* и *результат*, границы и *ограничения*.

Вместе с этими понятиями возникли отдельные и новые функции системы: *копирование*, как *запоминание* через *противоположность*. С созданием *дополнения* информации её противоположностью. Пока, хотя бы, как элемент *памяти*.

Новым техническим средством, позволяющим создавать *каналы управления* по возникающим *шаблонам* управления, стала *логическая машина*.

Чисто техническое устройство, реализующее *функцию управления* по определенному объекту. Та самая *исполнительная схема*, позволяющая проводить автоматическое *управление по отклонениям* на основе *обратной связи*.

Логическая машина стала техническим *центром функционального управления* объектами в границах автономного *логического пространства* клетки. Таких центров управления в клетке на этом этапе сразу стало *много*. И все они работали по *разным* шаблонам управления. В зависимости от объекта и функции управления.

Все основные понятия логики возникли на этом этапе. На этапе создания *автомата*, как единственно возможного варианта системы автономного логического управления.

Война машин.

Появление логической машины в системе логического управления клетки стало обязательным условием для дальнейшего наращивания логического потенциала управления.

Логическая машина обеспечивала необходимый уровень управления автоматической системы, достаточный для стабильного существования, но он не обеспечивал наличие ресурсов для этого. Необходимые компоненты надо было находить.

Проще всего найти уже готовые. Их проще применить, и меньше проблем с переработкой, ведь она уже частично проведена. Но вот находятся эти необходимые компоненты, а проще говоря, пища, нужная для выработки энергии, находятся они в запасах других, таких же, клеточных объектов, управляемых логическими машинами.

Рациональность логики сделала свое дело. Началась война машин, логических машин…

Эта вполне реальная война[42] глобального характера идет уже миллиарды лет.

Появились вирусы, клетки обзавелись средствами *нападения* и *защиты*. Клеточный мир разделился на *хищников* и *жертв*. Хотя, это деление весьма условно. Любой вирус и любая клетка может оказаться и жертвой, и хищником в разных условия и обстоятельствах.

Война шлифует оружие и развивает защиту, двигает прогресс, независимо от желания сторон. И если начинали войну простейшие логические машины, то сейчас их развитие поражает воображение. Особенно, если учесть, что единственным устойчивым инструментом развития была и остается случайность.

Потому, что автомат так и остался автоматом, хоть и с развитой логической системой. Автоматом, неспособным на целенаправленные, и самое главное, самостоятельные изменения. Это может сделать только случайность и естественный отбор.

Но, логическая машина клетки сделала необходимые выводы, закрепив *случайность*, как законный фактор развития логики и логического выбора. Может быть, по той же случайности, не знаю, но закрепила. Если критерии качественного выбора нечеткие, то главным становится… случайный выбор.

Мы говорим – любое решение подходит, это и есть случайный выбор по нечетким критериям. Или: эх, была, не была, как повезет, … и действуем по случайному выбору в зависимости от обстоятельств, а не от логических аргументов.

Война логических машин закрепила *противоположность*, как *дополнение* к информации, блокирующее её активность в составе химических молекул. Появилось понятие *симметрии*. Пусть и в зачаточном состоянии, но достаточном для начала *ориентации* в пространстве.

Появилась ДНК[43], как способ хранения информации, более защищенный, чем в составе РНК[44]. Та же двойная спираль, а надежность значительно выше. Теперь уже ДНК стала

---

[42] http://www.nanonewsnet.ru/articles/2010/gonka-vooruzhenii-dvigatel-evolyutsii

[43] **Дезоксирибонуклеи́новая кислота́ (ДНК)** — макромолекула, обеспечивающая хранение, передачу из поколения в поколение и реализацию генетической программы развития и функционирования живых организмов. Основная роль ДНК в клетках — долговременное хранение информации о структуре РНК и белков. http://ru.wikipedia.org/?oldid=40771092

главным хранилищем информации, а РНК – рабочей копией. Появились и достойные средства защиты и нападения – *белки*[45].

Оказалось, что они намного превосходят РНК по многим параметрам. Позволяют создавать запас *энергии*, так важной для нормальной работы клетки и её логической машины. Белки обладают более высокой химической активностью. Но, самое главное – белки способны создавать направленное *движение*.

Белки быстро стали лидировать в составе синтезируемых веществ клетки. Сегодня синтез белка мы рассматриваем, как основной процесс клеточной деятельности.

В логической системе, точнее в ДНК, появились *триплеты*[46] нуклеотидов, кодирующие ту или иную аминокислоту. *Триплетное кодирование* стало основой синтеза белков по программе РНК.

Триплет можно рассматривать, как *машинное слово* логической системы клетки. Для процесса синтеза белка оно таковым и является. ДНК стало основной долговременной *памятью* клеточной логической машины управления. РНК стала играть роль *оперативной памяти* и *исполнительной команды*.

Возникновение субъекта Я.

Как говорят, этот процесс[47] спровоцировал вирус, прорвавшийся внутрь клетки и заставивший её работать по своим законам [11]. Очень возможно. По крайней мере, это объясняет резкий скачок в совершенствовании логики управления и переход системы на *принцип единоначалия*[48].

Видимо, логическая машина вируса оказалась функционально сильнее разрозненных логических машин клетки, разбросанных в её логическом пространстве управления. И вирус взял управление на себя. Логические машины клетки сами оказались управляемыми объектами … логической машины вируса.

Только на этом этапе стало возможным появления центрального понятия логики – *цель*. Цель - жесткое требование *центра управления*, не имеющее *альтернативы* в *результате*. Только достигать. Любыми средствами.

---

[44] **Рибонуклеи́новые кисло́ты** (**РНК**) — одна из трех основных макромолекул (две другие — ДНК и белки), которые содержатся в клетках всех живых организмов.

Так же, как ДНК, РНК состоит из длинной цепи, в которой каждое звено называется нуклеотидом. Каждый нуклеотид состоит из азотистого основания, сахара рибозы и фосфатной группы. Последовательность нуклеотидов позволяет РНК кодировать генетическую информацию. Все клеточные организмы используют РНК (мРНК) для программирования синтеза белков. http://ru.wikipedia.org/?oldid=40220454

[45] **Белки́** (**протеи́ны**, **полипепти́ды**[1]) — высокомолекулярные органические вещества, состоящие из соединённых в цепочку пептидной связью альфа-аминокислот. В живых организмах аминокислотный состав белков определяется генетическим кодом, при синтезе в большинстве случаев используется 20 стандартных аминокислот. Множество их комбинаций дают большое разнообразие свойств молекул белков. Кроме того, аминокислоты в составе белка часто подвергаются посттрансляционным модификациям, которые могут возникать и до того, как белок начинает выполнять свою функцию, и во время его «работы» в клетке. Функции белков в клетках живых организмов более разнообразны, чем функции других биополимеров — полисахаридов и ДНК. Так, белки-ферменты катализируют протекание биохимических реакций и играют важную роль в обмене веществ. Некоторые белки выполняют структурную или механическую функцию, образуя цитоскелет, поддерживающий форму клеток. Также белки играют важную роль в сигнальных системах клеток, при иммунном ответе и в клеточном цикле. http://ru.wikipedia.org/?oldid=40408933

[46] **Триплет** (лат. *triplus* — тройной) — в генетике, комбинация из трёх последовательно расположенных нуклеотидов в молекуле нуклеиновой кислоты. В информационных рибонуклеиновых кислотах (иРНК) триплеты образуют так называемые кодоны, с помощью которых в иРНК закодирована последовательность расположения аминокислот в белках[1]. http://ru.wikipedia.org/?oldid=37571395

[47] Согласно гипотезе *вирусного эукариогенеза (англ.)русск.*, окруженное мембраной ядро, как и другие эукариотические элементы, произошли вследствие инфекции прокариотической клетки вирусом. Это предположение основывается на наличии общих черт у эукариот и некоторых вирусов, а именно геноме из линейных цепей ДНК, кэпировании мРНК и тесном связывании генома с белками (гистоны эукариот принимаются аналогами вирусных ДНК-связывающих белков). По одной версии, ядро возникло при фагоцитировании (поглощении) клеткой большого ДНК-содержащего вируса.[5] По другой версии, эукариоты произошли от древних архей, инфицированных поксвирусами. Это гипотеза основана на сходстве ДНК-полимеразы современных поксвирусов и эукариот.[6][7] Также предполагается, что нерешенный вопрос о происхождении пола и полового размножения может быть связан с вирусным эукариогенезом.[8] http://ru.wikipedia.org/?oldid=39785179

[48] **ПРИНЦИП ЕДИНОНАЧАЛИЯ** - принцип построения организации, согласно которому подчиненный должен принимать полномочия только от одного начальника и быть ответственным только перед ним.

В ядре клетки возник центральный управляющий объект, диктующий свои условия и изменяющий параметры управления всей клетки в зависимости от своих потребностей. На месте *одноуровневой* системы функционального управления возникла сначала *двухуровневая*, а затем и *многоуровневая система управления*.

Это потребовало создания сначала *логического ответа* на основе единиц системы, теперь получивших статус *логических состояний*, в качестве универсального результата множества разных управляемых процессов для сравнения эффективности их управления. Затем – *образа*, функционального описания того или иного логического понятия или объекта, как *универсального средства формирования памяти по ассоциативным связям, на основе качественных характеристик понятий, целей, объектов, действий*.

Логическая машина центра управления стала обрабатывать информацию, поступающую от функциональных логических машин клетки и формировать управляющие воздействия. Это потребовало создания большой *основной памяти*, *оперативной памяти*, а так же *единого кодирования информации* во всей логической системе клетке.

В ядре[49] клетки сформировалось ядрышко[50]. Ядро стало основным хранилищем памяти и обработки информации ДНК и РНК. Ядро стало и высшим уровнем управления логической системы клетки. Это логическая машина клетки.

Объемы обработки информации резко увеличились и простые шаблоны, сформированные, чаще всего по случайному стечению обстоятельств, уже не совсем подходили для всех процессов управления. Нужны были универсальные шаблоны управления, отработанные и эффективные. Так появились основные *алгоритмы* управления – *правила* и *постулаты* системы.

Отработка эффективных алгоритмов управления требует многократного их использования в различных условиях и на разных логических объектах. В реальных условиях управления это очень длительный процесс с непредсказуемым результатом. Но, в начале, видимо так и было сделано.

Только потом процесс отработки был перенесен из реальности в *условность*. В процесс *мнимого управления*, не предполагающий реальных действий, а только их фиксацию.

Но, сразу возникла очень большая проблема. Логическая машина не имела контроля времени и работала только *в условиях реального времени*. Теперь потребовалось отделить мнимые процессы от фактического управления. Прежде всего, по времени.

Это потребовало отдельного контроля *Настоящего*, т.е. фактического момента управления, *от прошлого и будущего*, как мнимых временных величин, используемых в процессе условного управления.

Технически это потребовало создания еще одного уровня управления, и не менее, а более мощного, чем уровень фактического управления в реальном времени. Появление нового уровня отработки логической информации в условном времени, отличном от реального, потребовало и создания надежного *разделения условности и реальности*.

Так возникло *сознание*[51], аппарат контроля реального времени системы в составе логической машины управления. В начале своего развития, на момент появления, сознание

---

[49] **Ядро** (лат. *nucleus*) — это один из структурных компонентов эукариотической клетки, содержащий генетическую информацию (молекулы ДНК). В ядре происходит репликация — удвоение молекул ДНК, а также транскрипция — синтез молекул РНК на молекуле ДНК. В ядре же синтезированные молекулы РНК подвергаются ряду модификаций, после чего выходят в цитоплазму. Образование субъединиц рибосом также происходит в ядре в специальных образованиях - ядрышках. http://ru.wikipedia.org/?oldid=39785179

[50] **Ядрышко** находится внутри ядра клетки, и не имеет собственной мембранной оболочки, однако хорошо различимо под световым и электронным микроскопом. В ядрышке происходит синтез рРНК РНК полимеразой I, ее созревание, сборка рибосомных субчастиц. В ядрышке локализуются белки, принимающие участие в этих процессах. Некоторые из этих белков имеют специальную последовательность — сигнал ядрышковой локализации (NoLS, от англ. **N**ucle**o**lus **L**ocalization **S**ignal). Следует отметить, самая высокая концентрация белка в клетке наблюдается именно в ядрышке. В этих структурах было локализовано около 600 видов различных белков, причем считается, что лишь небольшая их часть действительно необходима для осуществления ядрышковых функций, а остальные попадают туда не специфически. http://ru.wikipedia.org/?oldid=39643796

[51] Термин **сознание** является трудным для определения, поскольку данное слово используется и понимается в широком спектре направлений. Сознание может включать мысли, восприятия, воображение и самосознание и пр. В разное

было чисто техническим уровнем контроля настоящего времени. Небольшой надстройкой в логической машине. Не более.

Но, как только появилось сознание, так сразу возникла реальная потребность контроля и реального, и условного времени системы. Потому, что реальное управление и условная отработка алгоритмов управления шла параллельно и одновременно.

Возникла необходимость и потребность в новом уровне управления, который бы мог контролировать работу, как реального управления, так и условного. И принимать решения во всех случаях.

Такой уровень управления возник. Мы его знаем, как – Личность. Субъект Я.

Уровень управления *Личность*[52] может оперировать всеми задачами, решаемыми логической машиной, как в реальном, так и в условном времени. Субъект Я может переключать Сознание, т.е. направление контроля, с одного времени на другое, с процесса управления на процесс условной отработки. Личность контролирует работу всей системы только на основе логических ответов и их сборных образов.

С другой стороны, теперь уже Личность контролирует формирование памяти логической системы, создает системы поиска и связи, фиксирует внешнюю информацию и требует создания её функциональной копии в логической машине.

Личность стала принимать управляющие решения, как в реальном времени, так и в условном. И требовать исполнения своих решений. И уж тем более, исполнения своей главной цели – продления своего управления на неограниченный срок.

С этого момента и возникла Жизнь, как существование Личности.

Мы не будем отождествлять субъект Я, как личность, управляющая клеткой, с Личностью человека. Это разные уровни развития.

Клетка создала *субъект Я, как техническую надстройку в логической машине, обладающую каким-то набором качеств, для решения проблем управления*.

Так мы и будем его рассматривать. Все остальные стороны развития личности, в том числе и философские, к этому уровню пока отношения не имеют.

Дальнейшее развитие логической системы управления клетки.

Конечно, скорее всего, дальнейшее развитие, произошедшее в логической системе управления клетки, носило характер только начала реальных изменений тех понятий логики, о которых мы говорим.

Тем не менее, даже в таком зачаточном состоянии, их уже можно рассматривать, как вполне четкие направления, которые полностью развились уже на другом уровне. На уровне мозга. Но, если уже на этом уровне эти новые понятия логики появились, то, тем более, о них надо начинать разговор…

Тут надо вспомнить, что война логических машин на клеточном уровне продолжается непрерывно уже миллиарды лет, идет сейчас и будет идти в будущем. Этого достаточно для понимания того, что совершенствование логической системы управления клетки все время идет в жестких условиях борьбы за выживание.

Все преобразования клеточной системы управления, как и качественное преобразование клеточных колоний в клеточные структуры, и далее, в клеточные организмы обусловлены этим постоянным фактором.

время оно может выступать как тип ментального состояния, как способ восприятия, как способ взаимоотношений с другими. Оно может быть описано как точка зрения, как Я. Многие философы рассматривают сознание как самую важную вещь в мире. С другой стороны, многие ученые склонны рассматривать это слово как слишком расплывчатое по значению для того, чтобы его использовать.

В общем смысле сознание временами также означает состояние бодрствования и ответную реакцию на окружающий мир в противоположность состояниям сна или комы. http://ru.wikipedia.org/?oldid=40459443

[52] Под «**личностью**» понимают: 1) человеческого индивида как субъекта отношений и сознательной деятельности («лицо» — в широком смысле слова) или 2) устойчивую систему социально значимых черт, характеризующих индивида как члена того или иного общества или общности. http://ru.wikipedia.org/?oldid=40717859

Мы вполне справедливо называем этот процесс *эволюцией*[53].

Об эволюции в Википедии:

Существует несколько эволюционных теорий, объясняющих механизмы, лежащие в основе эволюционных процессов. В данный момент общепринятой является синтетическая теория эволюции, являющаяся развитием теории Дарвина.[1] СТЭ позволяет объяснить связь субстрата эволюции (гены) и механизма эволюции (естественный отбор). В рамках СТЭ эволюция представляет собой процесс изменения наследственных черт в популяциях организмов в течение времени, превышающего продолжительность жизни одного поколения.[2] Мутации, рекомбинация, поток и горизонтальный перенос генов приводят к появлению изменчивости наследственных черт в популяциях. Под действием естественного отбора особи с определенным фенотипом (и определенным набором наследственных черт) будут более успешны чем другие, то есть будут иметь более высокую вероятность выжить и оставить потомство. Таким образом в популяции будет увеличиваться доля таких организмов, у которых есть наследственные черты, обладающие селективным преимуществом. Другим важным механизмом эволюции является генетический дрейф.

Здесь хотелось бы отметить, что любые эволюционные изменения возможны только при действии *инстинкта самосохранения*[54]:

Вся жизнь есть осуществление одной цели, а именно, охранения самой жизни, неустанная работа того, что называется общим инстинктом жизни.

Эти слова принадлежат физиологу И.П.Павлову[55].

Наличие и развитие эволюционного процесса на уровне клетки рассмотрено в [50].

Но уже само понятие инстинкта[56] предполагает наличие *личности* у любого представителя Живого.

И конечно, для высокоразвитой клетки.

Представителя эукариотов[57]. Например, для инфузории[58] и жгутиконосца[59]…

Наличие инстинкта самосохранения у клетки подтверждаются, например, [45, 46]. Такие изощренные способы защиты невозможны на уровне простых автоматов. Как показано в [49], эволюция начинается на клеточном уровне и имеет вполне рациональное направление.

---

[53] Биологическая **эволюция** — естественный процесс развития живой природы, сопровождающийся изменением генетического состава популяций, формированием адаптаций, видообразованием и вымиранием видов, преобразованием экосистем и биосферы в целом. http://ru.wikipedia.org/?oldid=40651143

[54] **Инстинкт самосохранения** — это врожденная форма поведения живых существ в случае возникновения опасности, действия по спасению себя от этой опасности. http://ru.wikipedia.org/?oldid=38897440

[55] **Ива́н Петро́вич Па́влов** (14 (26) сентября 1849, Рязань — 27 февраля 1936, Ленинград) — один из авторитетнейших учёных России, физиолог, психолог, создатель науки о высшей нервной деятельности и представлений о процессах регуляции пищеварения; основатель крупнейшей российской физиологической школы; лауреат Нобелевской премии в области медицины и физиологии 1904 года *«за работу по физиологии пищеварения»*. http://ru.wikipedia.org/?oldid=40773658

[56] **Инсти́нкт** — совокупность врождённых тенденций и стремлений, играющих мотивационную роль в формировании поведения. В узком смысле, совокупность сложных наследственно обусловленных актов поведения, характерных для особей данного вида при определённых условиях.[1] Инстинкты составляют основу поведения животных. У высших животных инстинкты подвергаются модификации под влиянием индивидуального опыта. http://ru.wikipedia.org/?oldid=40418424

[57] **Эукарио́ты**, или **Я́дерные** (лат. *Eukaryota* от греч. εὐ- — хорошо и κάρυον — ядро) — домен (надцарство) живых организмов, клетки которых содержат ядра. Все организмы, кроме бактерий и архей, являются ядерными (вирусы и вироиды также не являются эукариотами, но не все биологи считают их живыми организмами). http://ru.wikipedia.org/?oldid=40161853

[58] **Инфузории**, или **ресничные**[1] (лат. *Ciliophora*) — тип протистов из группы *Alveolata*. Есть подвижные и прикреплённые формы, одиночные и колониальные. Форма тела инфузорий может быть разнообразной, размеры одиночных форм от 10 мкм до 4,5 мм. Живут в морях и пресных водоёмах в составе бентоса и планктона, некоторые виды — в интерстициали, почве и во мхах. Многие инфузории — комменсалы, симбионты и паразиты других животных: кольчатых червей, моллюсков, рыб, земноводных, http://ru.wikipedia.org/?oldid=38771767

[59] **Жгутиконо́сцы** — жизненная форма протистов. Используют жгутики для локомоции и/или создания токов воды, приносящих пищу. Среди жгутиконосцев много как свободноживущих форм, так и паразитов и симбионтов животных. Среди них есть одноклеточные моноэнергидные и полиэнергидные формы, а также колониальные (например, *Eudorina*) и многоклеточные (*Volvox*) формы. В целом для жгутиконосцев характерна тенденция к мелким размерам клеток и осмотрофному питанию, хотя среди них встречаются также очень крупные фаготрофные формы. http://ru.wikipedia.org/?oldid=40188570

С другой стороны, первые многоклеточные организмы, с которых сегодня принято начинать историю Живого, выглядели совсем не столь высокоинтеллектуальными, как это представлялось ранее [47].

Таким образом, применение понятия, хотя бы технической, личности по отношению к клеткам вполне реально и объективно. Вместе с понятием субъекта Я…

Появление у клетки личности и сознания почти автоматически должно было перевести логическую систему управления высшего уровня, а теперь это уровень субъекта Я, на другой вид фиксации логических действий системы.

На этом уровне требуется унификация понятий и действий в новую систему понимания. И такая унификация была проведена.

Единым средством фиксации всех изменений, внешних и внутренних стало *событие*[60]. У понятия «событие» много толкований. Мы пока ограничимся только одним.

Событие, это зафиксированное изменение состояния.

Если логическая система фиксирует *любое* изменение состояния, это – событие. И не важно, изменение чего именно. Реальности, решения логической задачи, информации, цели или состояния. Это всё – события. И все они имеют одинаковый начальный статус в логике представления.

Правда, потом все начинает изменяться. С событиями происходит то же самое, что и с целями. Одни становятся главными и определяющими, вторые уходят в тень. До поры, до времени. Третьи становятся контрольными точками теперь уже новых логических связей, причинно- следственных. Оказывается, чтобы произошло событие А, обязательно должно произойти событие В. И наоборот, если произошло событие В, то произойдет и событие А. Или может произойти. События А и В оказались связанными. Событие В оказалось *условием* для осуществления события А.

Возник новый вид логической связи событий – связь или *логический переход по условию*. Условия, как определяющие факторы осуществления событий быстро стали самостоятельными логическими понятиями. И теперь, для любого события должны быть определены условия его совершения.

Для логической системы, *условие - это задача, требующая решения параллельно с основной, в заданный момент времени.*[11]

В русском языке понятие условия[61] оказалось весьма многогранным. В логике, в том числи и математической, под условием понимается очень широкий спектр применяемых вариаций. От логической связки «если -… то», до дополнительных параметров для решения задачи.

Пока сформулируем это понятие примерно так: *Условие – это внешний или внутренний фактор, косвенно влияющий на ход решения логической задачи.*

И иногда, самым радикальным образом, к сожалению…

С другой стороны, если *условие* это то, что должно предшествовать или произойти одновременно, то *условность, это предполагаемый вариант решения, предшествующий реальному*. Это понимание условности мы уже рассматривали чуть ранее, как отработку алгоритма *мнимого управления,* вне реального времени.

Условное решение не предполагает совершения каких-то реальных действий, это лишь *прогноз* решения. Один из возможных вариантов.

На этом уровне главным и основным логическим объектом системы стал *образ*. Если раньше он применялся, как чисто техническое описание качественных характеристик реальных объектов, то тут он стал единственной формой отображения информации. Любой составляющей логики управления: объекта, действия, результата, логического ответа…, в том числе и целей.

---

[60] **Собы́тие** — то, что имеет место, происходит, наступает в произвольной точке пространства-времени; значительное происшествие, явление или иная деятельность как факт общественной или личной жизни; подмножество исходов эксперимента. Википедия http://ru.wikipedia.org/?oldid=34986177

[61] **Условие** - какое или чего. То, что делает возможным что-н. другое, от чего зависит что-н. другое, что определяет собою что-н. другое. Толковый словарь Ушакова http://www.slovopedia.com/3/211/846133.html

Логический ответ стал основным вариантом *результата* решения. В образном отображении. На высших стадиях развития Живого это стало *эмоциями*.

Повторим, основными понятиями логики на уровне субъекта Я стали *событие*, *образ* и *логический ответ*. Этого потребовала введенная ранее *образная система адресации памяти* логической машины…

Память клетки.

Да, на этом уровне автономных систем управления еще можно определить систему памяти. Хоть и очень приблизительно.

Понятно, что адресация памяти выполняется набором последовательности в единицах системы. Понятно, что каждый новый объект хранения получает свой адрес в памяти на основе установления последовательности связей с другими объектами.

Непонятно – как?

Давайте разбираться…

Клетка имеет локальное пространство памяти в виде цепочек ДНК и РНК. ДНК упакованы в хромосомы. Видимо, это отдельные массивы памяти, имеющие отдельные центры управления чтения этих массивов. Они распаковываются по мере необходимости, копируются в рабочие копии РНК и только тогда начинается их активное использование.

Понятно, что этот вид долговременной памяти имеет только один вид чтения информации – последовательный. Но вот «считывающих головок» - рибосом[62], для этого может быть использовано сколько угодно. Так что их «параллельность» в работе надо учитывать.

Основные процессы копирования ДНК и чтения информации с РНК идут в ядре клетки. Параллельно и последовательно. Контролирует этот процесс ядро – центр управления логической системы клетки.

Все эти процессы исследовались автором достаточно внимательно и обстоятельно в [19-24]. РНК, как информационная копия ДНК, перед началом её массового использования в качестве, например, матрицы для трансляции белка, проходит первичное чтение и полное переформатирование. Происходит резка РНК на части, потом следует сшивка этих частей в определенном порядке, с присоединением начального участка инструкций (интронов[63]) [19] по работе с этой информацией, и конечного, для нормализации схода рибосом с этой информационной цепи. Эти процессы называются процессингом[64] и сплайсингом[65] РНК [19].

В результате этих процессов в центральной части иРНК остаются только экзоны[66] [19], информационные массивы для триплетной трансляции белков.

Теперь уже процесс считывания информации идет строго последовательно. От начала иРНК к её концу. В компьютерной памяти такое расположение информации в какой-то мере эквивалентно *однонаправленному списку*.

---

[62]**Рибосома** — важнейший немембранный органоид живой клетки сферической или слегка эллипсоидной формы, диаметром 100—200 ангстрем, состоящий из большой и малой субъединиц. Рибосомы служат для биосинтеза белка из аминокислот по заданной матрице на основе генетической информации, предоставляемой матричной РНК, или мРНК. Этот процесс называется трансляцией. Синтез рибосом у эукариот происходит в специальной внутриядерной структуре — ядрышке. http://ru.wikipedia.org/?oldid=40662199

[63]**Интрон** — участок ДНК, который является частью гена, но не содержит информации о последовательности аминокислот белка. http://ru.wikipedia.org/?oldid=38681536

[64] В ходе ряда последовательных стадий **процессинга** из мРНК удаляются некоторые фрагменты, ненужные в последующих стадиях, и происходит редактирование нуклеотидных последовательностей. http://ru.wikipedia.org/?oldid=40191189

[65]**Сплайсинг** (от англ. splice — сращивать или склеивать концы чего-либо) — процесс вырезания определенных нуклеотидных последовательностей из молекул РНК и соединения последовательностей, сохраняющихся в «зрелой» молекуле, в ходе процессинга РНК. Наиболее часто этот процесс встречается при созревании информационной РНК (мРНК) у эукариот, при этом путём биохимических реакций с участием РНК и белков из мРНК удаляются участки, не кодирующие белок (интроны) и соединяются друг с другом кодирующие аминокислотную последовательность участки — экзоны. Таким образом незрелая пре-мРНК превращается в зрелую мРНК, с которой считываются (транслируются) белки клетки. http://ru.wikipedia.org/?oldid=40091101

[66]**Экзон** — это последовательность ДНК, которая представлена в зрелой РНК. http://ru.wikipedia.org/?oldid=36729113

Но, если учитывать, что все процессы, как копирование РНК, предварительный процессинг и сплайсинг, считывание информации и трансляция белка в ядре клетки идут одновременно и тысячи РНК находятся в рабочей области, то надо признать и наличие оперативной памяти в ядрышке и ядре клетки. Там, скорее всего, используются системы памяти *с произвольной выборкой*. По какой-то системе кодирования адресов выборки. С этим еще предстоит разобраться…

Вместе же, эти устройства памяти клетки образуют уже знакомую нам систему центральных понятий с кольцами аксиом многовекторной системы определения. Понятия и аксиомы формируются в ядре, а выходные образы объектов формируются в нормализованных иРНК. Это делается для последующего воспроизведения этих информационных образов в физическом виде с помощью, например, трансляции белка.

Как мы видим, вполне четко реализуется программа действий, записанная в памяти ядра и РНК. Все действия соответствуют общему алгоритму решения, принятому в клетке за эталон. Глобальный эталон, между прочим…

В его основе почти стандартный набор действий, уже не раз, рассмотренный в настоящей работе.

Коренным отличием технической организации памяти клетки и мозга является физический путь установления логической связи. На уровне клетки это производится изменением ДНК и РНК, а так же усложнением ядра. На уровне мозга закрепляется новой дендритной[67] или синаптической[68] связью аксона нейрона. Как аксон находит путь к нужной мишени[69], это отдельный разговор. Пока это непонятно даже ученым.

Но, главное, связь образуется, и образуется самостоятельно. Если мы хотим использовать этот принцип формирования системы памяти, то необходимо, в первую очередь, решить эту очень непростую задачу. Я пытался решить эту задачу, и что-то даже получилось [14-17]. Хотя, признать это законченным решением невозможно. Нет пока реального решения, только первичные подходы, не более. Позже был еще один подход к решению [18], но опять, только в общих чертах…

Если говорить о компромиссном решении на основе существующих вариантов реализации устройств памяти, то можно рассматривать, как подходящие: устройства оперативной памяти компьютера и флеш-память. Но тогда необходимо дополнительное согласование целей и способов такого использования. Тоже не самая простая задача…, а, учитывая разные применяемые системы счисления, задача становится очень сложной.

Единственное, что можно констатировать на данном этапе, то, что решение теоретически есть. Вопрос в его реализации. Что это дает, не знаю.

Когда-то проблема принципа формирования памяти стала одним из факторов остановки реализации логической машины для искусственного интеллекта. Но, не единственным. Понятно, что во времена формулирования задач логической машины эти вопросы даже не рассматривались. Не было тогда таких знаний о работе клетки, о роли ДНК и РНК, об их системе оснований, но вопрос о нахождении принципа формирования памяти уже был. Ответа не было. Точного ответа нет и сейчас, но хоть стали понятны основные подходы к его пониманию.

---

[67] **Дендрит** (от греч. δένδρον — «дерево») — дихотомически ветвящийся отросток нервной клетки (нейрона), воспринимающий сигналы от других нейронов, рецепторных клеток или непосредственно от внешних раздражителей. Проводит нервные импульсы к телу нейрона (соме). Дендриты могут образовывать синаптические контакты с аксонами (аксодендритические) и дендритами (дендро-дендритические). http://ru.wikipedia.org/?oldid=39429674

[68] На конце аксона находится синаптическое окончание — концевой участок терминали, контактирующий с клеткой-мишенью. Вместе с постсинаптической мембраной клетки-мишени синаптическое окончание образует синапс. Через синапсы передаётся возбуждение. http://ru.wikipedia.org/?oldid=37025286

[69] **Аксональное наведение**, **аксональный поиск пути** (англ. *Axon guidance, axon pathfinding*) — сложный процесс роста аксона к своей цели, зависящий от множества факторов. В период развития нервной системы аксоны преодолевают большие расстояния, с исключительной точностью достигая определённых частей других клеток, формируя сложную архитектуру связей. Изучение взаимодействий, влияющих на выбор пути, важно для лучшего понимания того, как строится мозг, что могло определить его эволюцию, какие факторы способствуют развитию патологий. http://ru.wikipedia.org/?oldid=34294343

Сегодня уже есть основания для возврата к теории логической машины. На новом уровне понимания. Появление логической машины уже не кажется несбыточной мечтой.

Организация работы мозга.

Главным отличием в системе организации работы логической машины на этом уровне является характер логических связей. На уровне мозга все связи вещественны. Это аксоны с синапсами и дендриты. Отростки нейронов, по которым передаются информационные импульсы.

Электрический импульс, как единственный вид передачи информации на этом уровне, перевел изменение всей системы памяти на новый уровень унификации.

Нейрон[70] остался клеткой с памятью на основе ДНК и РНК. Теперь он стал локальной логической машиной, в которой стало два уровня информации. Внешняя информация приходит и уходит в виде электрических импульсов, а внутренняя обработка информации обрабатывается ядрышком.

Уже известно, что под действием электрических импульсов нейрон может формировать *временные* ДНК, которые не входят в хромосомы, а формируются непосредственно в ядре. Эти временные ДНК, видимо кодируют внешнюю информацию. И когда необходимость в этой информации исчезает, то временные ДНК уничтожаются.

Как мы уже знаем, входная информация нейрона формируется, в основном, *дендритами*. Через эти разветвленные нейронные окончания импульс от других нейронов приходит в клетку. Эти окончания образуют связи с соседними нейронами их аксонами.

Аксон[71] – канал выходной информации, уходит от нейрона на огромные расстояния по клеточным меркам, имеет огромное количество синаптических контактов. В среднем, около 10тыс. По этим контактным переходам – *синапсам*[72] импульс передается другим нейронам логической системы. Десятки миллиардов нейронов нашего мозга создают сотни триллионов контактов для реализации логических связей.

С учетом клеточной памяти сложность мозга, как логической машины системы управления нашего организма становится весьма внушительной.

Но, при всей своей сложности, мозг, это только сложнейший коммутатор, соединяющий нейроны в логическую машину управления, надстройка над клеточной памятью. Машину, сумевшую резко ускорить работу за счет перехода на импульсную систему передачи информации в системе управления. Это мы уловили…

---

[70] **Нейрон** (от др.-греч. νεῦρον — волокно, нерв) — это структурно-функциональная единица нервной системы. Эта клетка имеет сложное строение, высокоспециализирована и по структуре содержит ядро, тело клетки и отростки. В организме человека насчитывается более ста миллиардов нейронов. http://ru.wikipedia.org/?oldid=40800375

Нейрон состоит из одного аксона, тела и нескольких дендритов, в зависимости от числа которых нервные клетки делятся на униполярные, биполярные, мультиполярные. Передача нервного импульса происходит от дендритов (или от тела клетки) к аксону, а затем сгенерированный потенциал действия от начального сегмента аксона передается назад к дендритам [1]. Если аксон в нервной ткани соединяется с телом следующей нервной клетки, такой контакт называется аксо-соматическим, с дендритами — аксо-дендритический, с другим аксоном — аксо-аксональный (редкий тип соединения, встречается в ЦНС). http://ru.wikipedia.org/?oldid=37025286

[71] **Аксон** (греч. ἄξον — ось) — нейрит, осевой цилиндр, отросток нервной клетки, по которому нервные импульсы идут от тела клетки (сомы) к иннервируемым органам и другим нервным клеткам.

Нейрон состоит из одного аксона, тела и нескольких дендритов, в зависимости от числа которых нервные клетки делятся на униполярные, биполярные, мультиполярные. Передача нервного импульса происходит от дендритов (или от тела клетки) к аксону, а затем сгенерированный потенциал действия от начального сегмента аксона передается назад к дендритам [1]. Если аксон в нервной ткани соединяется с телом следующей нервной клетки, такой контакт называется аксо-соматическим, с дендритами — аксо-дендритический, с другим аксоном — аксо-аксональный (редкий тип соединения, встречается в ЦНС). http://ru.wikipedia.org/?oldid=37025286

[72] **Си́напс**[1] (греч. σύναψις, от συνάπτειν — обнимать, обхватывать, пожимать руку) — место контакта между двумя нейронами или между нейроном и получающей сигнал эффекторной клеткой. Служит для передачи нервного импульса между двумя клетками, причём в ходе синаптической передачи амплитуда и частота сигнала могут регулироваться. Термин был введён в 1897 г. английским физиологом Чарльзом Шеррингтоном. http://ru.wikipedia.org/?oldid=40779833

С другой стороны, это и новый уровень обработки информации. Теперь уже на основе импульсной логики. Потому, что один импульс, пришедший к нейрону, не способен возбудить генерацию новых импульсов для дальнейшей передачи по логическим цепям мозга. Нужны дополнительные импульсы или иные воздействия, переводящие нейрон из состояния приемника импульсов в режим передачи. При этом характер передаваемой информации отличается от принятой. Это указывает на участие клеточной памяти в логике мозга.

И вся эта потрясающая воображение сложность … вырастает и развивается самостоятельно. Увеличивается количество нейронов, растут аксоны, как-то находящие нужные точки соединения не только с соседними, но и с весьма удаленными нейронами или другими клетками организма, вырастают дендритные окончания, … система наращивает сложность на основе собственных механизмов развития.

Еще немного о памяти.

Как только основные задачи обоснования логических связей стали эквивалентами единиц измерения системы, так сразу под них стала работать и вся система памяти. Как это происходит?

Если появилась пара сравнения, так сразу должна быть найдена и логическая связь между ними. Какая? Любая, подходящая. У нас выбор хоть и невелик, но он есть. Единиц – четыре. И в работу включаются все задачи одновременно. Строятся цепочки связей по всем решаемым задачам.

Возникает вопрос, с чем связываются логические составляющие сравниваемых объектов для их логического обоснования? С такими же связями предыдущих решений. И чем больше предыдущих решений, тем мощнее становятся связи настоящего решения.

Выходит, что все логические обоснования построены на собственной базе? Да. Видимо так, если вся база логического обоснования построена одной системой с самого начала.

Но, это практически невозможно. Потому, что даже система логики клетки строилась множеством поколений. И передавалась по наследству. В ДНК, РНК…, вместе с базовой системой логического обоснования. Да и системой памяти.

Но, вот текущую достройку системы логического обоснования и связи каждая клетка, и не только она, строит уже на собственном опыте. В этом смысле память индивидуальна.

Для логической машины это означает, что первичную систему логического обоснования и образования памяти должны создавать мы, основываясь на клеточной логике. Но память каждой логической машины все равно станет индивидуальной, с отличиями от эталона.

Часть 2. Механистическая логика клетки.

В [11] уже рассматривался подход к живой клетке, как основе нашего логического мышления. Там заложены основы логики.

Да, с одной стороны, это, хоть и очень сложный, но, биологический автомат. А с другой – форма Жизни, со всеми атрибутами этого понимания. Клетка рождается и умирает. Она обладает самостоятельностью действий. Она двигается, питается, нападает и защищается, размножается, … живет. Всё это невозможно без развитой системы логики.

К такой форме Жизни, как клетка, вполне применимо понятие «субъекта Я». Клетка так и ощущает себя. Она обладает всеми необходимыми формальными признаками обособленного и самостоятельного индивида.

При этом надо сказать об отказе от эмоциональных и психологических составляющих этой концепции. На уровне клетки об этом говорить не приходится. Можно говорить о технократическом подходе механистического понимания субъекта Я, о личности и её логике.

Вот с этого мы и начнем…

Технические характеристики основ логики клетки.

Конечно, технической основой применяемой автоматической логики автономной системы управления клетки стала логическая машина. Логические формулы и правила создавались по её возможностям. Это так.

С другой стороны, развитие логики стимулировало и развитие технической составляющей, усиление возможностей логической машины. Эти процессы взаимосвязаны.

Таким образом, говоря о технических составляющих логики клетки, мы косвенно характеризуем и технический уровень логической машины.

Мы уже говорили об этом. И не раз. Например, в [11]. Рисовали схемы управления и взаимодействие уровней. Обсуждали систему единиц, основные понятия логики. Всё это уже было. И не раз. Сейчас разговор пойдет немного о другом …

О жесткой необходимости некоторых технических требований к системе логики клетки.

Об автоматизме логики.

Система единиц логики клетки.

Это уже не раз обсуждаемый вопрос. Вроде бы всё сказано.

Нет, не всё.

Сегодня клеточная ДНК имеет четыре разных единицы. Это основания: A,T,G,C.

Мы это уже давно знаем. И всё же…

Сначала разберемся с количеством. Почему - четыре?

Нашему компьютеру вроде бы вполне достаточно двух – 1 и 0. А тут – четыре.

Исходные очевидности:

- Клетка не умеет считать. Или умеет только до одного. Дальше уже – МНОГО. Это мы, надеюсь, не оспариваем. Клетка не может отличить в ДВУХ одинаковых логических объектах, понятиях, действиях … и т.д., одно от другого. Она в состоянии различать только РАЗНЫЕ объекты. Сколько? Сколько угодно, сколько позволяют её технические возможности в поиске различий. И всё же, есть ограничение и в этом.
- *Для каждого* логического объекта или действия клетка должна иметь глобальный *эталон*. Это жесткая необходимость для системы, не имеющей основ *количественного счета*. Таким эталоном может быть только *единица системы*. Таким образом, количество разных единиц системы определяет и все количественные характеристики всех логических понятий и действий. Количественное разнообразие логики клетки четко ограничено количеством единиц системы. В данном случае, *универсальным числом* для всех логических построений стало число *четыре*.
- *Видовое разнообразие понятий* одного уровня так же ограничено этим числом. Нам придется это учитывать. Это требование системы. Меньше может быть, а вот больше – нет. Единственное возможное расширение тут – использование НИЧЕГО, как пятой единицы. И только в крайнем случае. Но для логики клетки это очень сложно. Об этом мы уже говорили. Правда, уровней может быть много…
- ДЕЙСТВИЕ, как отдельное понятие, имеет базовое разнообразие, равное количеству единиц системы. *Базовых действий* системы - четыре. Логических. Математика тут не главная. Какие действия? Это еще предстоит выяснить.

Здесь надо различать понятие МНОГО и УЧЕТНОЕ РАЗНООБРАЗИЕ. Много, это сколько угодно, а вот *учитывать одновременно* система может только ЧЕТЫРЕ РАЗНЫХ… из этого МНОГО.

Собственно, это и явилось причиной постоянной *унификации*, проводимой в логической системе клетки.

Весовые единицы системы [11]. Клетке необходимо отличать *часть* от *целого*, *один* от *много*. Это *разные* весовые единицы. Но, в любом случае, их количество не должно превышать четырех.

Мы ввели весовые единицы: *0,1 – часть целого, качество, 1 – учетная единица множества и любое целое, 1(0) – целое полученное объединением множества в единицу учета, 10 – много, все что больше одного*.

Количество весовых единиц соответствует количеству единиц системы. Понятие *ничего* – не учитывается. Оно входит в неопределяемое множество, как и понятие МНОГО.

Так как понятие *Цель* является глобальным для системы управления и может включать в себя объект, действие, качество, результат, и т.д. мы ввели универсальный определитель – *значимость*, как отдельное качество. Появление или присвоение качества *значимости* любой составляющей логики делает её целью, с самым высоким приоритетом - 1(0).

Единицы системы оказали существенное влияние на организацию физического *перемещения клетки в пространстве*, её *движения*. Как количественные или счетные величины, так и как весовые. Перемещение в пространстве сразу стало дискретным, а значит, ритмичным и тактируемым. Без счетных процессов это организовать сложно.

С другой стороны, начало движения или следующий импульс движения должен происходить без участия сложной логической машины, на основе набора составляющих: энергии, состояния двигательного механизма – сокращения или растяжения, готовности системы к этому акту, и, может быть, запускающего механизма.

Это достигается применением весовых единиц – *части (качества)* и *целого*. Счетная система клетки должна обеспечивать этот процесс на автоматическом уровне, простым счетом или сложением качеств до получения целого, как команды для движения. Видимо, так это и может быть организовано [44].

Вот теперь вроде бы становится понятным такое количество единиц системы логики клетки. *Единицы системы служат постоянными глобальными эталонами всех составляющих логики.* Других у системы нет.

И никто в неё других знаний о систематике не добавлял. В отличие от нашего компьютера. В него мы постарались вложить свои знания.

Функциональная нагрузка единиц системы, даже при таком их количестве, очень велика и многообразна.

Связь логики и математики.

Конечно, такая связь должна быть на уровне единиц системы. Иначе автоматическим операциям просто неоткуда взяться. Только установление эквивалентности единицы системы и логического действия может запустить процесс автоматических действий. Об этом мы говорили уже не раз. Равноправность объекта и действия должна быть реализована в их эквивалентности на уровне единиц системы. Точно так же единицы системы должны стать эквивалентами результата и логического ответа.

И только тогда возможны автоматические операции на всех уровнях логической системы на основе единых, в данном случае – счетных, эквивалентов.

То же самое мы пытались сделать и в наших вычислительных машинах. Но, не довели дело до логического завершения. Потому, что простота двоичной системы счета с бинарной записью отодвинула задачу установления эквивалентности объектов и действий в неопределенность, а точнее, просто не дала этого сделать. Выход был найден на другом уровне. На уровне записи действия последовательностью из 0 и 1. Той же бинарной записью. Теперь есть некоторая логическая неопределенность понимания числа и действия. Записи оказались логически идентичны. Потому и возникла острая необходимость жестко разделить команды и данные в компьютере.

У нас система единиц позволяет создать эквивалентность объекта и действия на самом первом уровне. На уровне счетных единиц. На уровне триплета, машинного слова

логической системы. И выделить для каждого из основных логических действий одну счетную единицу системы весовых соотношений.

Весовые единицы системы: 0,1; 1; 1(0); 10;

Например, так:

1(0) - Ассоциация в одно множество;

1 - Эквивалентность;

10 - Полярность до противоположности;

0,1- Движение по причинно-следственной связи;

Пока это только примерное установление эквивалентности. Потом, возможно, потребуется пересмотр, но пока так…

В данном случае мы установили счетный эквивалент для целой задачи логического обосновании. В общем понимании – действия. А вот объекты, с которыми надо работать, могут быть любыми. Это цели, образы, объекты, действия, результаты, логические ответы, условия, события, …. Все составляющие этой логики управления.

Напомним, что сейчас мы говорим о задачах самой логики. О задачах логического обоснования получаемых связей, между объектами и понятиями логики. Решение этих задач диктуется системой связей объектов памяти логической машины.

Собственно, это и есть система памяти. Она основана на связях.

Шаблоны и алгоритмы.

О шаблонах[73] и алгоритмах управления можно говорить в двух вариациях. В первом случае это *информационная последовательность* из единиц системы. Во втором, это *техническое устройство* в составе логической машины, реализующее действия по этому шаблону и алгоритмы.

Если говорить о клеточной логической машине, то шаблоны и алгоритмы в ней, скорее всего, существуют в обоих вариантах. Информационные последовательности закреплены в РНК и ДНК. Технические устройства реализованы в ядерных лабиринтных структурах.

Вот здесь начинаются отличия этой логики от классической и математической.

Автономная логическая система не может работать на основе нашей абстрактной математики. Она её не понимает. Всё её понимание складывается из образов реальности, как-то скопированных в химическую или иную память, и принятых в качестве эталона для оценки. Понятно, что образы, так или иначе, но отобразились в памяти в виде информации. На каком-то носителе.

Далее уже идет *сравнительная* обработка. Одни образы сравниваются с другими, отбирается повторяющаяся информация. Она воспроизводится и снова сравнивается. Принцип остался неизменным. Клетка сохраняет исходную понимаемость того, с чем она работает. Образ.

Этот же сравнительный информационный образ становится основой функционального технического устройства, формирующего процесс *копирования образа* в реальном исполнении.

С этой стороны *шаблоном можно назвать одно место сборки копии образа в специализированном техническом устройстве при любом варианте копирования.*

Например, для получения белка на рибосоме, шаблоном будет триплетная последовательность РНК. Техническое устройство – рибосома, двигается по одноцепочной РНК от начала до конца, собирая белок на основе триплетной последовательности.

---

[73] **Шаблон** (нем. Schablone, от франц. echantillon — образец) в технике, приспособление или инструмент для проверки правильности формы ряда готовых изделий; образец, по которому изготовляются однородные изделия.

Яндекс.Словари›БСЭ, 1969-1978

Шаблон в переносном значении — образец, пример, которому подражают.

http://ru.wikipedia.org/?oldid=40247013

Ранее[11] мы дали определение шаблона. Уточним его: *Шаблон, это логический признак сравнения, действия или взаимодействия, общий для всей цепи ассоциативной или причинно-следственной связи*.

Это о шаблоне памяти в системе управления.

Да, память клетки тоже шаблонная. По крайней мере, адресация памяти. Она идет по цепи ассоциативной или причинно-следственной связи общего признака или качества сравниваемых объектов. Эти связи и создают образ объекта в памяти системы. Шаблон оказывается цепочкой связей, уходящей в бесконечность своего представления конкретных образов.

Да, шаблон, это первый вид примитивов. Он так и используется. В качестве составной части сложных конструкций.

Если сборка копии образа требует нескольких последовательных действий и перемещений или нескольких частей, необходимых для соединения их в общую копию, то такой последовательный цикл получения копии образа уже можно назвать *алгоритмом*.

Мы так и понимаем *алгоритм*, как *последовательность действий* для решения той или иной задачи. Понятно, что алгоритм может состоять и из нескольких шаблонов с переходами от одного к другому.

Более сложные алгоритмы и шаблоны, связанные в единый процесс, иногда и многовариантный, уже составляют *модель*[74] решения задачи управления. Процесс создания и отработки моделей мы называем *моделированием*.

Сегодня моделирование в машинной логике, в частности, при решении широкого спектра сложных логических задач используется очень широко. В основном, это *имитационное моделирование*[75].

Если отвлечься от сложной математики, обратив внимание только на сам процесс формирования и отработки модели решения, то клетка делает то же самое.

Моделирование производится в реальном времени и выражается во множестве создаваемых физических копий образа объекта с небольшими отклонениями. Это сказывается недостаток средств моделирования в логической машине клетки в режиме «мнимого управления или решения», который восполняется фактическим исполнением моделей с отбором лучших уже при применении. Вполне логичное решение.

Сложность задач и относительная ограниченность средств логической машины клетки требует жесткого *ограничения набора базовых шаблонов и алгоритмов*. Это нормальное ограничение для сравнительной логики клетки. Но, тем не менее, это и обеспечивает

---

[74] **Моде́ль** (фр. *modèle*, от лат. *modulus* — «мера, аналог, образец») — это упрощенное представление реального устройства и/или протекающих в нем процессов, явлений.

Построение и исследование моделей, то есть *моделирование*, облегчает изучение имеющихся в реальном устройстве (процессе, …) свойств и закономерностей. Применяют для нужд познания (созерцания, анализа и синтеза). Моделирование является обязательной частью исследований и разработок, неотъемлемой частью нашей жизни, поскольку сложность любого материального объекта и окружающего его мира бесконечна вследствие неисчерпаемости материи и форм её взаимодействия внутри себя и с внешней средой. http://ru.wikipedia.org/?oldid=40866282

[75] **Имитационное моделирование** — метод, позволяющий строить модели, описывающие процессы так, как они проходили бы в действительности. Такую модель можно «проиграть» во времени как для одного испытания, так и заданного их множества. При этом результаты будут определяться случайным характером процессов. По этим данным можно получить достаточно устойчивую статистику.

Имитационное моделирование — это метод исследования, при котором изучаемая система заменяется моделью, с достаточной точностью описывающей реальную систему и с ней проводятся эксперименты с целью получения информации об этой системе. Экспериментирование с моделью называют имитацией (имитация — это постижение сути явления, не прибегая к экспериментам на реальном объекте).

Имитационное моделирование — это частный случай математического моделирования. Существует класс объектов, для которых по различным причинам не разработаны аналитические модели, либо не разработаны методы решения полученной модели. В этом случае аналитическая модель заменяется имитатором или имитационной моделью.

Имитационным моделированием иногда называют получение частных численных решений сформулированной задачи на основе аналитических решений или с помощью численных методов[1].

Имитационная модель — логико-математическое описание объекта, которое может быть использовано для экспериментирования на компьютере в целях проектирования, анализа и оценки функционирования объекта. http://ru.wikipedia.org/?oldid=40166420

приемлемую *однотипность подхода* к способам решения задач моделированием, обеспечивая при этом достаточное *разнообразие* моделей, так необходимое для *выбора*.

Логичность логики.

Как мы видим, вполне адекватная логичность действий может объясняться очень простым и ограниченным набором готовых алгоритмов, исполняемых в автоматическом режиме. Нет в этих алгоритмах ничего такого, что требовало бы отдельного философского осмысления, как большой проблемы логики Живого.

Вот эти глобальные алгоритмы логического обоснования:

- *Порядок следования;*
- *Обобщение и детализация;*
- *Сравнение и уравнивание эквивалентов.*
- *Полярность и противоположность на основе использования вещественной или мнимой оси симметрии;*

Все они взаимосвязаны, из одного появляется другое, как один из вариантов расширения понимания или альтернатива.

Все они начинались с очень простого понимания и расширились до сложнейшей системы понятий и определений.

Сегодня можно указать только первый элемент системы логического обоснования, это - *порядок следования*. С него все начиналось в логике клетки. Как и когда появились остальные алгоритмы, установить уже вряд ли удастся.

Но есть еще один *первичный элемент системы автономной логики – Цель*.

Это прямое следствие появления личности в логической системе управления клетки.

Субъект Я появился, как высший уровень принятия решения, как определитель локальности Живого, как хранитель существования логической системы во времени и пространстве.

Всё началось с ХОЧУ. С Цели. С логического обоснования этого. Для самого себя.

Мы постараемся найти те основные условия, при которых образование логики, как основы системы управления станет не только возможным, но и обязательным.

Собственно, эти условия нам уже хорошо известны. Но …

Они стали основными в понимании основ логики, когда были изменены условия её появления. Как только в основу понимания были поставлены автоматические условия её возникновения, как и принципы её применения, так сразу стали очевидными и основные условия.

Но именно они обеспечили не только возможность появления логики на уровне автоматических операций, но и неизбежность её появления.

Все глобальные алгоритмы логического обоснования были установлены системой, скорее всего, случайно. Но, однажды установленные, и многократно проверенные, они уже вошли в арсенал логики системы управления. И потому требовали уточнения и расширения своего понимания, как основа логического обоснования действий.

Личность субъекта Я развивалась в соответствии с пониманием роста сложности условий своего существования. Нет, это еще не интеллект, это пока автомат, но уже с начальными признаками интеллекта. Все направления развития логики субъекта Я очень долго формировались и расширялись … случайно. Время для этого было. Миллиард лет, и не один [11], это вполне достаточный срок для понимания закономерности в случайных событиях. Даже на уровне клетки.

Главные направления развития отражают процесс становления автоматической логики. Они все взаимосвязаны. Мне очень сложно сказать, где заканчивается одно и начинается другое, и потому любое деление будет субъективным. Потому, что это части одной логики. Логики клетки. И пересечение неизбежно. Слишком уж все тут переплетено….

Порядок следования.

Вот это понятие надо установить основным в начале рассказа о логике клетки. Просто фиксация порядка следования, независимо от его логического понимания.

Это клетка должна была установить на этапе принятия ДНК и РНК, как основы своей памяти. Само построение этих эквивалентов памяти клетки обязывает придерживаться этого правила или принципа. В них *порядок следования*[76] является естественной частью построения этих молекул памяти. И если эти молекулы стали основой памяти клетки, то и принцип их построения так же вошел в систему, как основа логики.

Принцип, вроде бы простой, но он позволяет создать всю логику, вместе с памятью, о которой мы еще будем говорить далее.

Странно, словосочетание такое встречается часто, а его понятие в словарях я не нашел. А это в данном случае и есть главное.

Что же определяет порядок следования?

Он определяет очень важные для логики понятия.

Это *последовательность*[77]. Это *начало* и *конец* последовательности. Это и, *что за чем следует*[78]. Что *раньше*, а что *позже*. В этом основа понимания *причины* и *следствия*[79].

Клетка использовала этот принцип во всех логических построениях. Начиная с принципа логического обоснования. Основы логики.

Математическая логика ввела понятие *логическое следование*[80] [42]. Это стало основой, в том числе, и для разработки логики отношений[81]:

> Отношения порядка играют большую роль в логических исчислениях, т.к. логическое следование (доказуемость формул) упорядочивает высказывания и их формульные образы по-разному — от линейного порядка, преобладающего в аксиоматических конструкциях, до тех или иных видов частичного порядка (древовидные структуры). Алгебраический подход к представлению «законов мышления» существенно использует порядковые структуры: булева алгебра является частным случаем решетки, а она есть вид частично упорядоченного множества.

Здесь, как мы видим, порядок следования не является основным понятием, скорее, наоборот. В логике отношений рассматриваются только частности этого большого понятия.

---

[76] **Нуклеотидная последовательность**, генетическая последовательность — порядок следования нуклеотидных остатков в нуклеиновых кислотах. Определяется при помощи секвенирования. http://ru.wikipedia.org/?oldid=28379909

[77] **ПОСЛЕДОВАТЕЛЬНОСТЬ**, числа или элементы, расположенные в организованном порядке. **Последовательности** могут быть конечными (имеющие ограниченное число элементов) или бесконечными, как полная **последовательность** натуральных чисел 1, 2, 3, 4 ….dic.academic.ru›Научно-технический энциклопедический словарь

[78] **Очередь (очередность, последовательность)** О и очередность имеют в обыденном сознании и в науке следующие значения: 1) О, – суть некое целостное образование, совокупность, множество, состоящее из элементов, расположенных в определенном порядке (последовательности, расположения) друг возле друга на основе какого-либо правила, алгоритма следования, программы, расписания или расположения (диспозиции) друг за другом - как в пространстве, так и во времени, обозначаемое порядковыми числительными; 2) О - это само место, где существует, наличествует объект или же такой объект временно отсутствует ("вакантное место" в О); 3) О означает (в смысле хрональности) следование, движение событий или состояний объектов любого рода в пространственно-структурном смысле, иначе говоря, - их дискретное распределение (во времени) друг после друга (в целом, - в пространстве-времени), также обозначаемые порядковыми числительными. В словарях имеются и другие значения. http://www.chronos.msu.ru/TERMS/razumovsky_ochered.htm

[79] ПРИЧИНА И СЛЕДСТВИЕ - филос. категории, отображающие одну из форм всеобщей связи и взаимодействия явлений. Под причиной (лат. causa) понимается явление, действие которого вызывает, определяет, изменяет, производит или влечёт за собой др. явление; последнее называют следствием. Производимое причиной следствие зависит от условий. Одна и та же причина при разных условиях вызывает неодинаковые следствия. Различие между причиной и условием относительно. Каждое условие в определ. отношении является причиной, а каждая причина в соответств. отношении есть следствие. П. и с. находятся в единстве: одинаковые причины в одних и тех же условиях вызывают одинаковые следствия. В области обществ. наук причины отличаются от поводов — процессов, способствующих их проявлению.

[80] **ЛОГИЧЕСКОЕ** СЛЕДОВАНИЕ Философия отношение между некоторыми высказываниями (посылками) Г и высказыванием В (заключением), отображающее тот факт, что из Г, используя правильные приёмы рассуждения, можно получить В. dic.academic.ru›**Философская энциклопедия**

логическое следование - отношение, существующее между посылками и обоснованно выводимыми из них заключениями. Л.с. относится к числу фундаментальных, исходных понятий логики, точного универсального определения не имеет; в частности… dic.academic.ru›**Словарь по логике. - 1997**

[81] **ЛОГИКА ОТНОШЕНИЙ** — раздел логики, посвящённый изучению отношений между объектами различной природы. Эти отношения выражаются сказуемыми и аналогичными им словами в предложениях естеств. языков. В зависимости от числа объектов, связанных данным отношением,…

Отдельные конкретные конструкции, пригодные для математического описания и исчисления.

И опять произошло незаметное смещение понятий. Порядок следования, как главное понятие логики, ушло в тень, а вторичные понимания стали главными. Почему?

Причина простая. Основное понимание порядка следования плохо отражается средствами математики. В математическом представлении исчезает логическая многогранность порядка следования. Возникает множественность математических выражений, может быть и связанных логически, но лишенных этой связи ограничениями математики.

Вот пример:

Последовательность нуклеотидов в одиночной РНК мы записываем, как:

…AATGCGGATTCGG… 1)

Теперь запишем их в математическом порядке следования, как последовательность:

$$\ldots a_n, a_{n+1}, a_{n+2}, \ldots a_{n+k}, \ldots \qquad 2)$$

Возникла математическая однородность членов последовательности, но пропала их логическая информативность.

Теперь восстановим логическую связь и получим:

$$\ldots a_n \rightarrow a_{n+1} \rightarrow a_{n+2} \rightarrow \ldots \rightarrow a_{n+k} \rightarrow \ldots \qquad 3)$$

Запись сразу потеряла однозначность понимания. Математики понимают одно, специалисты по математической логике другое, а мы, все остальные, чаще всего – третье.

И тем не менее, мы понимаем, все записи отражают модификации одного и того же понятия – порядка следования. Вопрос оказывается не в понятии, а в его применимости к той или иной сфере отражения восприятия. В одном случае понятнее так, в другом – иначе, но говорим мы об одном и том же - порядке следования … логической связи между компонентами этого множества.

Если мы говорим о множестве, как едином объеме, то применима запись 2), если о связи объектов множества, то запись 3), если об информационной стороне последовательной записи, то запись 1). Но, есть еще логический переход, трансформация одного объекта в разные состояния, и т.д. Многозначность очевидна. Математическими средствами она невыразима.

Тогда *надо отказываться от математического выражения порядка следования, как единственного средства отображения и переходить к его логическому пониманию, как основному. Принимать многозначность понимания, как естественное качество логической связи.*

Это ведет к другому порядку формализации понятий логики – логическому. И к другому механистическому пониманию логики. Математика была и остается только частью логики, но никак не её основой. В том числе и для машины, в любом её понимании.

Обобщение и детализация.

Обобщение и детализация, это те же *симметричные противоположности* в движении к пониманию. Это, как раз, переход «от штучек к кучке» [11] и обратно. По направлению действия они противоположны:

$$a,a,a,\ldots \rightarrow A(a); \qquad 4)$$
$$A(a) \rightarrow a,a,a,\ldots;$$

Функция минимизирует применение математики в логических операциях. В математическом смысле это взаимный переход от суммы элементов массива к массиву, как новой единице измерения. А в логическом смысле, например, от деревьев к лесу, и обратно. Для логической системы применение этой функции обусловлено несчетностью понятия *много*. Все, что выходит за рамки определяемой счетности логическая система почти автоматически переводит в единичность нового эквивалента. Переход назад осуществляется в случае необходимости обращения к конкретному элементу множества.

Множественность переходов из одной системы эквивалентов в другую, и назад, привел к обратимости этого процесса и на уровне функции:

$$a,a,a,\ldots \leftrightarrow A(a); \qquad 5)$$

Правда, эта функция требует организации хранения, как первичной информации, так и получаемого эквивалента.

Принцип полярности.

Полярность[82], это начало *противоположности*. И возможно, наиболее полное её понимание.

Потому, что поляризовать или разнести в разные стороны понимания относительно *оси симметрии*, *мнимой* или *вещественной*, можно практически все понятия и определения, результаты, логические ответы, действия и объекты. Лишь бы нашлось хоть одно качественное различие между парой сравнения.

Насколько бесконечен процесс *поляризации* и *разделения* на основе *детализации* качеств объектов сравнения, настолько же бесконечен и процесс их объединения или обобщения.

Вопрос только в целях этого процесса.

С другой стороны, полярность еще не определяет понятия противоположности. Это только общая часть понятия. Только качественная оценка различий, не более. Выделение *разных* объектов сравнения. На основе найденных *различий*.

Это действие обратно по направлению к *объединению* объектов в одно множество на основе найденного качественного *сходства*. Это две стороны одной медали.

Метод сравнения и механизм его реализации объективен и независим. Например, от нас, и нашей субъективной оценки.

Сравнение позволяет объективно оценить одно качество или свойство двух объектов сравнения. И определить их сходство и различия в отношении этого качества.

Это дает нам в руки важный инструмент этого действия – эквивалент сравнения. Что и на основе чего сравниваем?

Логическое обоснование.

Собственно, с этого и начинается логика. С системы логического обоснования связей между составляющими логики. С главных постулатов системы.

Что и на основании чего признается логичным, т.е. связанным в систему логики?

На этом уровне рановато говорить о суждениях и высказываниях. Пока можно говорить лишь о самых простых средствах, позволяющих создать связи понятий и логических объектов в системе управления клетки. На основе шаблонных действий, возникающих на самом первом этапе – появлении нового объекта в памяти клетки или мозга.

На этом уровне логики можно говорить только об автоматических действиях, а они не могут быть сложными.

Принцип логического обоснования.

В основе логического обоснования лежит задача ЦЕЛЬ – РЕЗУЛЬТАТ. Всё начинается с цели создания того или иного логического обоснования. Логичность построения логического обоснования определяется целью. А результат мы уже знаем. Как всегда, надо найти цепочку доказательства. И тогда задача будет решена.

В запасе у нас не так уж много инструментов. Собственно, по большому счету их всего четыре: *детализация и обобщение*, тут же *порядок следования*, *создание полярности* и *уравнивание эквивалентов*. Вот на основе этих инструментов и устанавливается цепочка

---

[82] **ПОЛЯРНОСТЬ** Философия (позднелат. polaris – полярный, от греч. πόλος – полюс) – отношение, выражающее попарную противоположность нек-рых сторон объекта (признаков, свойств, тенденций развития), называемых полюсами. dic.academic.ru›Философская энциклопедия

обоснования логичности исходного системного понятия задачи: цель - результат. Во всех вариациях его понимания. Как вопрос-ответ, причина-следствие, сходство-различие, … и т.д.

Даже в постановочной части задачи мы видим те же инструменты. Мы или устанавливаем эквивалентность конечных объектов, или полярность до противоположности, или ассоциацию их в одном множестве на основе признака определения, или причинно-следственную связь. Вот на этом пока и остановимся…

Выделим то, что мы перечислили:

- Ассоциация[83] в одно множество;
- Эквивалентность;
- Полярность до противоположности;
- Движение по причинно-следственной связи;

Это задачи логического обоснования. С одной стороны отдельные и независимые, с другой – взаимосвязанные и составляющие части единого процесса.

Как мы видим, первые две задачи основаны на *обобщении*, вторые две на *детализации*.

Для решения этих основных задач логического обоснования нам и нужны основные понятия логики: Сходства и различия, границы и ограничения, условия и события, качества и счетность, ожидание и действие, и т.д.

Решение задач логического обоснования связей исследуемых объектов составляет основу логической системы и образной памяти.

Здесь просматривается прямая связь основных задач логического обоснования с системой единиц логической системы управления.

Странно, но понятия *доказательство*[84] мы пока не использовали. Странно так же то, что понятия *логическое обоснование* мне так и не удалось найти в словарях. Только – *обоснование*[85]. Хотя используем мы это понятие достаточно часто, и именно, как логическое.

Более того, вместо логического обоснования в логике и философии чаще используется другая форма выражения логической связи объектов – *логическое следование*.

Механистическое понимание логики не может включать в себя слишком общие понятия *доказательства* и *логического следования* по причине отсутствия в них механистического логического обоснования. А понятия «правильности» или «неправильности» в данном случае неприменимы.

---

[83] **Ассоциация** (от лат. *accosiare* — соединять) — объединение. http://ru.wikipedia.org/?oldid=34062299

[84] **Доказательство** — это логическая операция обоснования истинности утверждения с помощью фактов и других истинных связанных с ним суждений. Познание отдельных фактов, предметов, их свойств происходит посредством форм чувственного познания (ощущений и восприятий) и высказывания вспомогательных суждений и утверждений. Мы видим, что этот дом ещё не достроен, ощущаем вкус горького лекарства и так далее. Эти истины и факты не подлежат особому доказательству, они очевидны. Во многих случаях, например на лекции, в сочинении, в научной работе, в докладе, на защите диссертации и во многих других, приходится доказывать, обосновывать высказанные суждения и утверждения. Доказательность и обоснованность важное качество правильного мышления взрослых людей. Теория доказательства и опровержения является в современных условиях средством формирования научно обоснованных и юридически грамотных убеждений и утверждений.

Доказательство — это совокупность логических приемов обоснования истинности какого-либо суждения с помощью других истинных и связанных с ним суждений. Доказательство связано с убеждением, но не тождественно ему: доказательства должны основываться на данных науки и общественно-исторической практики, убеждения же могут быть основаны, например, на религиозной вере в догматы церкви, на предрассудках, на неосведомлённости людей в вопросах экономики и политики, на видимости доказательности, основанной на различного рода софизмах. http://ru.wikipedia.org/?oldid=38243810

[85] **ОБОСНОВАНИЕ** — способ убеждения В истинности (правильности) чего-либо, напр. мысли или действия. Это **обоснование** в широком его значении, оно не связано необходимо с логикой. **Обоснованием** в этом смысле служат и **логическое** доказательство, и эксперимент… www.terme.ru›**Новая философская энциклопедия. – 2003**

обоснование процедура проведения тех убедительных аргументов, или доводов, в силу которых следует принять к.-л. утверждение или концепцию. О. является, как правило, сложным процессом…

dic.academic.ru›**Словарь по логике. - 1997**

ОБОСНОВАНИЕ - мыслительный процесс, основанный на использовании определенных знаний, норм и установок с целью регламентации и эталонизации практической и познавательной деятельности.

О. является **логическим** каркасом научной аргументации, подобно тому, как понятие является **логической** формой слова (словосочетания), а суждение - **логической** формой предложения. voluntary.ru›**Социология: Энциклопедия. – 2003**

Детализация

Детализация, как форма логического обоснования, применима к поиску и усилению различий сравниваемых объектов, действий и качеств с целью получения максимальной точности в оценке отличий в рамках локальной логической системы управления.

Собственно, весь процесс сравнения построен на получении максимального количества сходства и различий одного и того же качества сравниваемых объектов.

Необходимость этого логического действия вытекает из обоснования *индивидуальности*[86] любого *единичного* объекта. Это тем более важно, если объектов – *много*. Их просто необходимо *различать*. На основе *любого* найденного различия в качествах. От большого и, может быть, очевидного для нас, но не для простейшей логической системы управления. Она не понимает очевидности. Для неё это просто отличие, добавляющее в образ индивидуальности для конкретного исследуемого объекта еще один аргумент логического обоснования этого.

Детализация позволяет получить логический путь уточнения и индивидуализации построения ассоциативных связей для образа конкретного логического объекта в памяти.

С другой стороны, детализация служит основой для построения последующего пути обобщения качеств объекта, для фиксации его в системе классификации групповых свойств, как обобщающих факторов построения основных ступеней иерархии логических понятий системы управления.

По этой причине детализация представляет собой достаточно сложную систему действий и понятий. Попробуем в этом разобраться…

Полярности.

Если логические противоположности не блокируют и не меняют свойств друг друга, то это – поляры. Поляры или полярности могут быть построены, как по функциональному признаку, так и по другим качественным характеристикам.

Действие – объект.

Событие – исполнительная команда, как пассивная фиксация изменения и создание этого изменения. Они полярны в понимании, но независимы в применении. Могут появляться и учитываться, как одновременно, так и отдельно.

Результат – Логический Ответ.

Объект – Образ

Полярность изначально является создаваемым противопоставлением. Мы формируем полярности в относительном понимании различий качеств при сравнении. Главным в понимании полярности является сравнение и нахождение различий. Нахождение различий является основой для создания полярности понятий. Любых.

Если вернуться к основам логики, то создание полярности, это создание детализации множества до отдельных объектов сравнения. На основе любого признака различия объектов этого множества.

Функциональные полярности.

Действие – ожидание, как полярности в отражении режима работы системы управления.

Решение – результат, как полярности частей в задаче.

Счетные полярности.

Один – Много

---

[86] **Индивидуа́льность** (от лат. *individuum* — неделимое, особь) — совокупность характерных особенностей и свойств, отличающих одного индивида от другого; своеобразие психики и личности индивида, неповторимость, уникальность. http://ru.wikipedia.org/?oldid=40033042

Часть – Целое.

Относительные полярности.

Они полярны только относительно оси симметрии. И существуют, пока есть ось симметрии:

Правый – Левый,

Верх – Низ.

Сравнительные полярности..

Они уже сравниваются не между собой, а с эталоном. И существуют, пока есть эталон сравнения:

Далеко – Близко.

Больше – Меньше.

Логические противоположности.

Если внимательно приглядеться к логическим противоположностям, то окажется, что и они вполне разнообразны. Вот только некоторые…

Действительные противоположности.

К таким противоположностям можно отнести пары антагонистов, уничтожающих или блокирующих действие друг друга. Как кислота и щелочь, положительное и отрицательное, например. Но, как выясняется, таких противоположностей в логике совсем немного.

Качественные противоположности.

Тут надо обратить внимание на главное: противоположность логических объектов рассматривается только в отношении их свойств или качеств. И наоборот, противоположность одного качества двух объектов может служить основанием для объединения их в пару противоположностей. Собственно, о качестве мы и говорим, когда говорим о противоположности: черное и белое, тихое и громкое, … и т.д.

Противоположность качества в приложении к логическим объектам и создает логические противоположности объекта с разным направлением действия одного и того же качества. Одного. Потому мы и понимаем под противоположностями черную кошку и белую кошку, а не ворону или собаку, например. Эти объекты в таком качестве несравнимы. В них сравнивать можно только цвет, его и определять, как противоположность.

Создаваемые противоположности.

Это противоположности, создаваемые логической системой для применения в решениях задач управления. Эти противоположности могут создаваться и задаваться для любого понятия или объекта логической системы, не имеющих действительных противоположностей. В этом случае характер создаваемой противоположности определяется противоположным направлением действия относительно оригинала.

Цель – противоцель

Математические логики, определяя противоположность, как *инверсию*, именно так и создают противоположности.

Односторонние противоположности.

Это противоположности, создаваемые только наличием или отсутствием аргумента:

Условие – безусловность.

***

С полярностями и противоположностями разобрались. Хотя бы на самом простом уровне понимания.

Обобщение.

Как мы уже говорили, это вторая сторона формализации понимания. Обратная детализации по качествам. На основе *сходства*, или *унификации* качественного признака. Группирование объектов в одно множество с целью его идентификации, как *единичного объекта* сложного состава.

По этой причине в формируемое множество могут попадать совершенно разные понятия и объекты. Их объединяет *качественный признак*, определяющий сходство или унификацию понимания основы данного множества.

Обобщение является основным механизмом уравнивания в процессе установления эквивалентности объектов сравнения на основе соединения их в одно множество на основе объединяющего качества.

В механистической логике это основной путь логического обоснования эквивалентности сравниваемых логических объектов (действий, качеств) и установления между ними логической связи. Получаемые логические связи служат основой для создания образа объекта в памяти системы. Да и самой памяти…

Принцип усиления уравнивания.

Это основной принцип установления логической связи. Обратный принципу полярности. Он обеспечивает постепенное усиление уравнивания объектов сравнения на основе тех или иных качественных признаков с переходом всех объектов в одно множество с одним определителем.

Обратим внимание, на то, что в этом процессе мы постепенно изменяем *действие* в сравнении объектов. Мы изменяем *качественную связь* объектов. На основе усиления объединения объектов в одно множество.

На основе этого определителя и устанавливается эквивалентность объектов сравнения с возможностью их уравнивания в системе этого определителя.

Принцип эквивалентности.

Этот принцип вытекает из принципа усиления уравнивания, но имеет самостоятельное применение. В этом смысле, вся логика построена на сравнении эквивалентов. И их замены в процессе логического обоснования. Этот принцип лежит в основе моделирования и поиска решения. Если мы доказали возможность эквивалентности, то доказали и возможность замены этих эквивалентов одного на другой в любой момент.

Эквивалентность объектов сравнения доказывается на основе усиления уравнивания, т.е. *смены действия* для эквивалентности в сравнении. Но при этом сравниваем мы уже действия, объекты или понятия. Для них устанавливается эквивалентность.

Эквивалентность.

Для определения своего отношения к объекту рассмотрения нам нужен его понятный эквивалент[87] сравнения. Если не с чем сравнить, то … объекта для нас просто – нет. Он невоспроизводим нашими органами чувств. А то, что хоть как-то прямо или косвенно воспроизводится, всё имеет хоть какие-то эквиваленты, пусть тех же ощущений.

Математическое описание эквивалентности [11] абстрактно, и может быть приложено к любому понятию и пониманию эквивалентов. Принцип один. Говорим ли мы о звуках, зрительных образах, и не только зрительных, понятие образа значительно шире, или о типах логических задач, мы все равно говорим и об их эквивалентах.

---

[87] **Эквивалент** (от позднелат. aequivalens - равнозначный - равноценный) — предмет или количество, равноценные, или соответствующие в каком-либо отношении другим и способное служить им выражением или заменой [1][2]. Нечто равноценное, равнозначное другому, полностью заменяющее его. http://dic.academic.ru/dic.nsf/ruwiki/575555

Если же говорить о логической записи эквивалентности, то мы приходим к оценке сравнения эквивалентов. Эта оценка и определяет логичность сравнения.

Полная запись будет выглядеть:

(Объект сравнения ≈ эквивалент) → Логический ответ 6)

И мы пришли к логическому ответу, как оценке сравнения эквивалентов. Логика, в отличие от математики, не может пользоваться количественной оценкой в виде конкретного результата. Но, оценку сравнения надо давать.

Мы снова вынуждены обратить внимание на многозначность полученного выражения.

В том виде, как показана запись выражения эквивалентности, можно оценить сам вариант подбора эквивалентов для сравнения. Подходит ли выбранный эквивалент для сравнения? Да или нет. Логически получить достоверный ответ на этот вопрос получить сразу невозможно. И, тем не менее, именно так начинается логическое решение.

Для получения возможности сравнения необходимо уточнить направление сравнения, определить качество для сравнения:

(Качество объекта сравнения ≈ эквивалент) → Логический ответ 7)

А качеств у объекта – много. И потому, качество надо установить. Или оставить первичную запись 6) на последний этап решения, начав решение с необходимого качества.

Как мы уже знаем, именно так сделана адресация памяти логической машины управления. По качеству. Это ассоциативная связь.

Сравнение качественных эквивалентов преобразуется в задачу движения по цепочке ассоциативных связей между образами объектов в памяти логической машины. Техническую задачу, похожую на движение в экспертной системе компьютера. С определением сходств и различий качеств сравниваемых объектов. С применением глобальных алгоритмов.

Результат сравнения также отражает качественную характеристику. Это и есть главное обоснование появления логического ответа в сравнении эквивалентов.

Пример логического обоснования.

Математически, [11] мы имеем функцию *1+1 = (1+1),* как преобразование (вычислением) сложного выражения 1+1 в единый эквивалент сравнения (1+1) и новую функцию сравнения *(1+1)=2.* Логически, сначала идет функция обобщения *1+1⟶(1+1),* затем выражение суммы преобразуется в простой эквивалент *(1+1)⟶2;*

Вот в чем разница математического и логического преобразования эквивалентов. Математика акцентирует внимание на процедуре вычисления (1+1)=2, а логика - на преобразовании *(1+1)⟶2.*

Математика сравнивает количественные эквиваленты, а логика – объекты и их качества.

Различие очень существенное. Математически 2=3 неверно, а логически – вполне…

Потому, что логика не считает формальные единицы. Она идет по пути: 2↔3, потом 2≈3, эквивалентность установлена, и нам пока все равно – почему. Повторим путь логического обоснования по порядку следования:

(2↔3)⟶(2≈3)⟶ (2=3), как Ш1⟶Ш2⟶Ш3; 8)

Сначала утверждение: 2 и 3 – числа. Следовательно, они обладают примерно одинаковыми свойствами (качествами). Следовательно, они равны, как логические объекты.

Если же мы будем оценивать это обоснование с математической стороны, то первым аргументом «против» будет их количественная оценка. Но именно это логика может и не учитывать, при качественной оценке.

Для логики это стандартный путь продвижения к уравниванию эквивалентов постепенным смещением оценки эквивалентности, или, что более понятно, переходом от

одного шаблона к другому, близким по пониманию, но с усилением уравнивания. Мы это движение считаем логическим обоснованием пути.

Логический ответ.

Так как сравнение эквивалентов, как образов, может проводиться по нескольким качествам сравнения, то и результат этого сравнения – *логический ответ*[88], будет сложным.

И теперь уже будет сравниваться с эталоном полученный сложный логический ответ. Точно так же, с нескольких сторон. И только тогда мы получим конечный логический ответ, как общую оценку сравнения.

Вот тут и появляется ассоциативная оценка сравнения логических ответов – *эмоции*[89].

Эту оценку дает субъект Я, по отношению к своему требованию – хочу…

Это его логический ответ на решение задачи сравнения логической машиной. Даже на уровне клетки. Оказывается, и эмоции, это вполне машинный логический ответ, только на высшем уровне оценки. И он вполне обоснован логически…

Организация логических связей.

Теперь уже можно предположить, что ассоциативная связь[90] между объектами на основе качества строится по целевой направленности, определяемой одной из четырех основных задачи логического обоснования. Это позволяет совместить связь объектов с логическим обоснованием этой связи на основе постулатов.

Каждый логический объект системы, представленный в системе, как функциональный образ, имеет конечное количество направлений обоснования логической связи с другими объектами в виде образов. Это:

- Сходство данного объекта с другими, входящими в это же множество (ассоциацию) обобщения, как определяющее качество для объединения.
- На основе найденного сходства, используя принцип усиления уравнивания, находятся функциональные эквиваленты объекта, для использования в разных задачах логики.
- С другой стороны, определяются основные различия с объектами множества обобщения. Это другие качества, которые по принципу усиления полярности позволяют надежно отличать именно этот объект от его ближайших функциональных эквивалентов сравнения множества объединения.
- Все действия проводятся на основе причинно-следственной связи, фиксирующей начало и конец усиления эквивалентности, как причину и следствие проведенного действия.

---

[88] Например, в двузначной логике они могут принимать значения «истина» либо «ложь» (***true*** либо ***false*** , **1** либо **0**). http://ru.wikipedia.org/?oldid=40661070

[89] **Эмо́ция** (от лат. *emoveo* — потрясаю, волную) — эмоциональный процесс средней продолжительности, отражающий субъективное оценочное отношение к существующим или возможным ситуациям. Эмоции отличают от аффектов, чувств и настроений.[1]

Эмоции эволюционно развились из простейших врождённых эмоциональных процессов, сводящихся к органическим, двигательным и секреторным изменениям, до значительно более сложных, утративших инстинктивную основу процессов, имеющих отчётливую привязку к ситуации в целом, то есть выражающих личное оценочное отношение к имеющимся или возможным ситуациям, к своему участию в них. http://ru.wikipedia.org/?oldid=40446989

[90]**Ассоциация** — связь, возникающая в процессе мышления, между элементами психики, в результате которой появление одного элемента, в определенных условиях, вызывает образ другого, связанного с ним; субъективный образ объективной связи между элементами, предметами или явлениями.

По типу образования различают:

- ассоциации по сходству;
- ассоциации по контрасту;
- ассоциации по смежности в пространстве или во времени;
- причинно-следственные ассоциации. http://ru.wikipedia.org/?oldid=37478914

Таким образом, одновременно проводится обобщение, детализация, логическое обоснований и установление порядка следования усиления связи, как причины и следствия, от действия к действию.

Для усиления установленной ассоциативной связи, как правило, устанавливаются и связи на основе полярности до противоположности.

Таким образом, система связей получает дополнение своей противоположностью. В полном соответствии с принципами системы управления клетки.

Доказательство от противного видимо родилось на уровне клетки.

В техническом плане все оказывается достаточно просто. Есть набор обязательных направлений обоснования логичности связей, он дополнен своими противоположностями. На том же уровне автоматического действия.

В результате такого простого механического дополнения клетка получает развернутую систему логических связей прямой и противоположной направленности для всех компонентов, составляющих память логической системы управления.

Для чего?

Для возможности восстановления связей путем сравнивания их с противоположностями. Надежно и просто. Как радикальное средство против действия той же случайности.

Но, все это просто только на первичном уровне, когда связей мало, и каждую надо беречь. Конечно, с ростом объемов памяти вся эта «простота» выливается в огромную сложность связей даже не очень большого количества единиц хранения памяти. И с ростом объема памяти эта сложность только увеличивается.

С другой стороны, только такая система позволяет быстро создать необходимую выборку из общего массива хранения без отдельной системы адресации памяти. Того самого, что в компьютере было введено в первую очередь, и на что клетка изначально неспособна.

Получается, что клетка ввела относительную адресацию памяти гораздо раньше, чем это было сделано в компьютере.

Клетка, да и наш мозг имеют одно кардинальное отличие от компьютера в плане образования системы памяти. Далее мы попробуем посмотреть это более внимательно.

Образ

Напомним, что *образ, это набор инструкций, делающих информацию понимаемой системой* [11]. Образ, это шаблон для размещения информации в памяти системы. Образ изначально функционален. Так понимает образ клетка.

Иначе клетка и не могла. Только так можно надежно разместить необходимую для системы управления информацию с возможностью её восстановления при большой вероятности действия случайности. И только так можно создать функциональную копию объекта по имеющейся информации.

Понятие *образа* создала память клетки, принципы её формирования. Формирование образа стало результатом применения шаблонов формирования ассоциативных связей в памяти. Как следствие применения относительной адресации.

Не умеет клетка считать, и потому она может применять только такую адресацию информации в памяти. Сегодня мы понимаем, насколько это сложно, но … необходимость заставила это сделать.

Собственно, ассоциативные связи и есть система адресации клеточной памяти. Эта же система в полном объеме применена и на уровне мозга.

Центральное понятие, круг аксиоматических связей, цепочки связей с другими объектами и понятиями, которые позволяют однозначно локализовать и определить центральное понятие логического объекта в многомерном пространстве такой относительной адресации. Это и есть образ объекта.

Образ позволяет найти этот объект по любому начальному, пусть и минимальному, объему исходных данных. Восстановить полный объем информации и иметь возможность

моделировать этот объект средствами логической системы. Как в логической модели, так и физически.

Для этого и нужен образ.

Принцип формирования памяти самостоятельной логической системы управления.

Например, в логической системе управления появился новый логический объект, как внутренняя копия реального. Пока это неизвестный объект. Его надо внести в память системы.

Клетка имеет универсальный механизм для такого действия. Мы его уже знаем.

Есть четыре весовые единицы счета – A,T,C,G, по их количеству и установлено количество построения линий логических связей. Тут что-то понятно, а что-то нет.

Ячейка памяти под наличие объекта есть. И это первый элемент в последовательности. Пусть это будет – A. Система зафиксировала вот эту ячейку, но в будущем адресе она оказывается не в начале адреса, а в центре. Почему?

Потому, что в этот момент и начинают работу автоматические программы установления логических связей. Понятно, что сравнение идет с имеющимися эквивалентами памяти.

Она работает так. Мы:

- сравниваем обнаруженный нами объект с известными нам понятиями и объектами по первым обнаруженным качествам,
- устанавливаем качество объекта для получения эквивалентности,
- фиксируем качество сходства и эталон сравнения.
- По найденному сходству уже по новой связи переходим к новому объекту сравнения с нашим неизвестным объектом.
- Снова проводим весь комплекс сравнения.

Почему это делается именно так?

Необходимо найти комплекс сходных качеств известных нам понятий и объектов для формирования образа этого, пока неизвестного объекта. Понять его.

С этого момента закрутилась бесконечная цепочка установления эквивалентности на основе качеств неизвестного объекта с известными системе аналогами или эталонами. Заработала система сравнения.

Смысл этого процесса в установлении связей неизвестного объекта с имеющимися образами других объектов по сходству качеств. Вот в этом он сходен с A, вот по этому качеству он сходен с B, вот по этому качеству… и т.д.

Идет *процесс обобщения*. Включение неизвестного объекта в *разные множества* известных объектов на основе *разных* качеств. На этом этапе сравнения с эталонами выявляются как *сами качества*, так и их *сходства* в сравниваемых эталонах. Такими, иногда и беспорядочными поисками мы набираем первую базу сравнения для неизвестного объекта.

С другой стороны мы еще и *создаем множество* эталонов, как-то связанных с *построением образа* вот этого, неизвестного пока объекта.

Задача установления эквивалентности *целевая*, и потому, тут рассматривается только один вариант решения, принимаемый системой. Есть сходство по качеству…, качество и эталон сходства включается в создаваемое множество определения для этого объекта. Если сходных качеств в очередном эталоне не обнаружено, то эталон отбрасывается и поиск продолжается.

Да, это процесс длительный и беспорядочный. Он может продолжаться долго. А строить образ неизвестного объекта надо быстро. Потому, так важны уже первые найденные качества сходства с известными эталонами.

Рис. 1. Часть первичной цепи ассоциативных связей.

Параллельно с процессом поиска сходств с эталонами напускается еще один автоматический процесс. *Процесс детализации*. Это уже *поиск отличий* от найденного качества, принятого как обобщающего, в установлении эквивалентности с эталоном. В чем эталон не соответствует определяемому объекту? Чтобы не уравнять эталон и объект, нужен отличительный признак, однозначно отделяющий эталон от объекта. Главное отличие в качествах. Любое. Пока этого достаточно.

Действие двух одновременно идущих процессов обобщения и детализации создает цепочку парных логических связей «обобщение-детализация» для каждого эталона сравнения, попавшего в этот процесс. Примерно так, как показано на рис.1.

Как только появились первые звенья парной цепи «обобщение-детализация» сравнения с разными эталонами, сразу запускается третий автоматический процесс. Сравнение звеньев этой цепи. Принцип сравнения тот же самый – «обобщение-детализация» качеств, теперь звеньев цепи. На предмет нахождения наибольшего сходства и наибольшего отличия. Для чего?

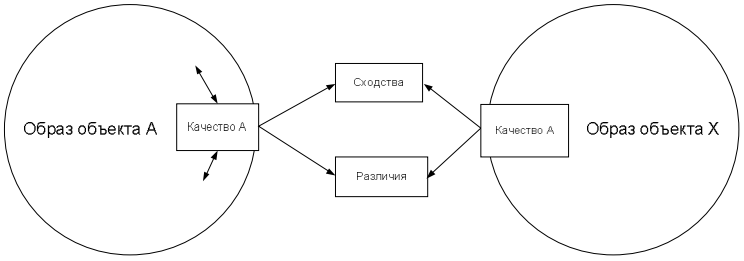

Рис. 2. Связь образов по качеству А.

Посмотрим на рис.2. Обратим внимание, что образ объекта А уже ранее нормализован установкой ассоциативных связей по порядку следования. Это отражают стрелки от качества А внутри образа объекта А. Качества объекта Х только определяются, и тут пока порядок следования не установлен.

Вспомним, процесс сравнения исходно начинается, как беспорядочный и случайный. Он так и идет. Связи, получаемые в результате этого процесса, не обладают свойством линейности в фиксации изменения. Они хаотичны. Вот для определения их действительного места в создаваемом образе неизвестного объекта и запускается третий процесс сравнения этих связей. Для выставления их по свойству линейности изменения сходств и отличий. Каждого определяемого качества.

Отдельные звенья цепи логических связей создаваемого образа объекта выставляются по нарастанию, как сходств, так и отличий. В результате мы получаем цепь логических связей с полярными отношениями «сходства-различия» на концах цепи. Т.е. от

максимального сходства и минимальных отличий в паре связей сравнения на одном конце цепи, к минимальному сходству и максимальным отличиям по определяемому качеству на её другом конце. Нарастание сходства и рост отличий идут по цепи логических связей в противоположных направлениях.

Сколько зафиксировано определяемых качеств в создаваемом образе объекта, столько цепей логических связей мы и получаем. Сложно? Да. Но только так можно надежно зафиксировать новый объект с памяти системы. Мы уже не раз говорили, что множественность связей, это лекарство от случайностей.

Собственно, потому система и создает *образ* объекта, намертво связанный с другими образами, хранящимися в памяти системы. Не локальный объект с набором своих качеств, как это сделано в памяти компьютера, а многосвязный образ. Надежность, прежде всего.

Потому, что даже случайная утеря части связей в системе памяти не сможет стереть из неё образ этого объекта, а значит, и сам объект не исчезнет из неё. Утерянные связи можно восстановить, если появится такая возможность. Для этого нужен оригинал – реальный физический объект. И если система его опознает своими регистраторами, то уж точно проведет полное восстановление образа в памяти. На это и настроена вся система автоматических процессов создания образа.

С другой стороны, если регистраторы системы управления зафиксировали какой-то объект, то, те же автоматические процессы формирования образа быстро обнаружат сходство нового создаваемого образа с имеющимся. И так же автоматически проведут совмещение двух образов с выдачей логического ответа по результату этой работы. Да, совпадает. Уже в самом начале этого процесса. Мы это знаем, как «узнавание» уже знакомого нам объекта по «характерным» признакам. Почти с первого взгляда…

Центр аксиомы образа.

Ранее[91] [11], мы говорили, что очень важно замкнуть цепочку логических связей в кольцо. В этом случае цепочка связей превращается в аксиому, правило, не требующее доказательств для использования. Аксиома предполагает использование любого элемента кольца связей сразу, без дополнительных действий. Правда, тут должен быть использован элемент, ранее не упоминавшийся в [11], но уже использованный в этой работе. Это центральный элемент аксиомы, адрес понятия, относительно которого всё и определяется. Та самая ячейка памяти, центральная, о которой мы говорили в самом начале разговора о памяти логической системы. Ячейка, выделенная под основное понятие неизвестного нам пока объекта. Это вокруг неё мы стараемся образовать кольцо связей с другими объектами, известными системе и взятыми в качестве эталонов для сравнения по разным качествам.

Образовывая кольцо аксиомы для определения понятия неизвестного нам объекта в памяти системы, мы создаем образ этого понятия. Может быть, мы пока и не можем точно дать название этому объекту, как его главный определитель, но постепенно формируем его, создавая кольцо связей, соединенных с центром определения.

Вот теперь, из центрального узла образа мы можем уже сразу обратиться к любой составляющей образа, заключенного в кольце аксиомы для уточнения выбранного качества, и, наоборот, любое качество, обнаруженного сходства выведет логическую связь на центральную ячейку, главное понятие для определяемого объекта.

Правда, логика автономных логических систем, не имеющая счетных способностей, сразу будет пытаться выделять главное определяющее качество, устанавливая его основным в определении. Но, по мере набора качеств в кольце аксиомы определения, главная составляющая будет меняться. И процесс это будет длительным. Пока не будет установлена основная последовательность качеств и порядок следования связей в аксиоме определения.

---

[91] См. Действие. Логический переход.

Теперь сравним процесс формирования образа объекта в памяти системы с принципом логического обоснования, приведенным выше. Как мы видим, образ объекта строится на этом принципе.

У нас пока отсутствует только последняя ветвь построения: причинно-следственные связи. Всё остальное уже есть. И это достигнуто простыми автоматическими действиями, сразу, при формировании образа в памяти. Но, пока все построения проводились по одному виду связей – *ассоциативному*.

Причинно-следственные связи.

И, наконец, когда уже сформирована основная аксиома, центральное понятие, но процесс его формирования еще в самом разгаре, вдруг обнаруживается несоответствие в аксиоме.

Качества, определяемые для создания образа объекта определенные в первом цикле, не соответствуют качествам последующего определения.

Конечно же, логическая система этого пока не обнаружила. Она просто строит новое кольцо аксиомы вокруг нового центра определения. Для неё это новый объект, но …

Когда-нибудь, но, это обязательно произойдет, строительство кольца аксиомы выведет сходство свойств этого объекта с другим, понятие которого уже сформировано. И установит очень большое совпадение качеств. Возникла логическая связь «старого» понятия объекта с его «новым» вариантом. Она совершенно обоснованно станет доминантой. Но, чуть позже возникнет еще один «новый» вариант образа. И снова доминантная связь соединит цепочку. Теперь уже несколько образов связаны одной связью. Нужен центр определения. И он возникает, как центральное понятие уже новой аксиомы, создаваемой по этой новой для системы цепочке связей. И снова система определения замыкает кольцо вокруг этого нового центра определения.

Но, вот связь уже не только ассоциативная. Да ассоциативные связи присутствуют, они связывают части, отдельные качества этих, пока разных образов в один комплекс. Уже понятно, что весь комплекс образов создается вокруг одного, пока нового понятия. И есть уже аксиома, соединяющая «разные» образы в один. И эти образы оказались разными только потому, что они создавались в *разные* моменты времени.

Теперь надо устанавливать цепочки качеств с другими объектами системы на основе этих «разных» разностей в образах аксиомы для центрального понятия.

Понятно, что это новое объединение и новая детализация.

Если принять за основу первичные образы для центрального объекта, как, например, *горизонтальные*, создаваемые по локальным качествам, то новая аксиома строится уже в новой плоскости. Это *вертикальная* система определения.

Эта система определения связывает уже *качества, изменяемые во времени или пространстве*. То есть, определяются уже не сами качества, а *их изменения*. В зависимости от чего-то, что и надо найти, и связать ассоциативными связями с известными объектами в системе.

Конечно, это другой уровень определения.

Но, так или иначе, система этот шаг сделала. Она вышла из *плоскости определения качеств* в многомерный *объем определения изменения качеств*, относительно новых центров определения, создаваемых на основе причинно-следственных связей. В новую систему координат и объединений *на основе глобальных определителей*.

Обобщаем понимание…

Отметим еще раз. Все процессы формирования образа объекта являются автоматическими и достаточно простыми. Никаких логических задач высокого уровня при

этом не решается. Работают только система адресации памяти и автоматические процессы установления логических связей, как часть системы.

Все процессы основаны на поиске сходств и отличий в качествах, создаваемых образов логических объектов системы. На основе обобщения и детализации, основных процессах логического обоснования. На принципах усиления уравнивания и принципе эквивалентности. Ничего другого не используется.

Вот только результат этого простого, но масштабного процесса формирования образа объекта, оказывается отличным. Получение *образа* объекта равносильно его *пониманию*. Мы понимаем, что *это* такое. То, что чуть раньше было *непонятным* и *неизвестным*.

Как мы уже выяснили, связи и их логическое обоснование строятся системой управления сразу, при формировании функционального образа исследуемого логического объекта в памяти системы.

Формирование образа идет сразу по двум возвратным направлениям. По цепочке ассоциативных и причинно-следственных связей. Кроме того, сразу строится и логическая противоположность этих связей для образа объекта.

Это первичный уровень логики. Но … на нем и формируется вся база логических связей. Уже на этом уровне начинают работать привычные нам: система логического обоснования, логическое следование, ассоциации и классификация качеств.

Во всех задачах логики управления логическая машина получает результат решения только в виде логического ответа. Другого варианта результата эта логика не имеет.

Часть 3. Логическая машина.

Мы снова подошли к понятию логической машины. И хочется надеяться, что теперь мы понимаем всю сложность её реализации.

И уже понятно, что вопрос не в сложности схемы этой машины, а в реализации принципов её строительства. Тут главные проблемы.

Всё сегодняшнее развития этого направления идет на основе компьютера. В свою очередь, развитие компьютерной техники основано на принципах фон Неймана, определяющих рамки этого развития. Все составляющие компьютера, так или иначе, участвуют в процессе обработки информации.

Компьютер, с самых азов…

Вот теперь посмотрим поближе компьютер.

Даже не сам компьютер, а принципы его строительства. То основное, на чем базируется система математической логики.

Материал дан по учебнику В.И.Юрова [12].

Фон Нейман[92] предложил схему ЭВМ с программой в памяти и двоичной логикой вместо десятичной. Логически машину фон Неймана составляли пять блоков (рис. 2.1): оперативная память, арифметико-логическое устройство (АЛУ) с аккумулятором, блок управления, устройства ввода и вывода. Особо следует выделить роль аккумулятора. Физически он представляет собой регистр АЛУ. Для процессоров Intel, в которых большинство команд — двухоперандные, его роль не столь очевидна. Но существовали и существуют процессорные среды с однооперандными машинными командами. В них наличие аккумулятора играет ключевую роль, так как большинство команд используют его содержимое в качестве либо второго, либо единственного операнда команды.

---

[92]**Джон фон Не́йман** (англ. *John von Neumann*; или **Иоганн фон Нейман**, нем. *Johann von Neumann*; при рождении **Я́нош Ла́йош Не́йман**, венг. *Neumann János Lajos*; 28 декабря 1903, Будапешт — 8 февраля 1957, Вашингтон) — венгро-американский математик еврейского происхождения, сделавший важный вклад в квантовую физику, квантовую логику, функциональный анализ, теорию множеств, информатику, экономику и другие отрасли науки.

Наиболее известен как праотец современной архитектуры компьютеров (так называемая архитектура фон Неймана), применением теории операторов к квантовой механике (алгебра фон Неймана), а также как участник Манхэттенского проекта и как создатель теории игр и концепции клеточных автоматов. http://ru.wikipedia.org/?oldid=39634990

Рис. 3. Схема машины фон Неймана

Ниже описаны свойства и принципы работы машины фон Неймана.

- *Линейное пространство памяти*. Для оперативного хранения информации компьютер имеет совокупность ячеек с последовательной нумерацией (адресами) 0, 1, 2,... Данная совокупность ячеек называется оперативной памятью.
- *Принцип хранимой программы*. Согласно этому принципу, код программы и ее данные находятся в одном и том же адресном пространстве оперативной памяти.
- *Принцип микропрограммирования*. Суть этого принципа заключается в том, что машинный язык еще не является той конечной субстанцией, которая физически приводит в действие процессы в машине. В состав процессора (см. главу 1) входит устройство микропрограммного управления, поддерживающее набор действий-сигналов, которые нужно сгенерировать для физического выполнения каждой машинной команды.
- *Последовательное выполнение программ*. Процессор выбирает из памяти команды строго последовательно. Для изменения прямолинейного хода выполнения программы или осуществления ветвления необходимо использовать специальные команды. Они называются командами условного и безусловного переходов.
- *Отсутствие разницы между данными и командами в памяти*. С точки зрения процессора, нет принципиальной разницы между данными и командами. Данные и машинные команды находятся в одном пространстве памяти в виде последовательности нулей и единиц. Это свойство связано с предыдущим. Процессор, поочередно обрабатывая некоторые ячейки памяти, всегда пытается трактовать содержимое ячеек как коды машинных команд, а если это не так, то происходит аварийное завершение программы. Поэтому важно всегда четко разделять в программе пространства данных и команд.
- *Безразличие к назначению данных*. Машине все равно, какую логическую нагрузку несут обрабатываемые ею данные.

Очень значимые, оказывается, эти принципы фон Неймана. Во всех смыслах компьютерных технологий. И вопрос не в количестве процессоров или скорости обработки информации, а в подходе к этой обработке.

Организация работы цифровых систем любого назначения сегодня примерно одинакова. Все технологии обработки цифровой информации базируются на принципах фон Неймана.

И вопрос не в том, хорошо это или плохо, а в том, что именно они унифицировали все цифровые технологии.

Процессор.

Это основа компьютера. Его центр обработки информации. Но здесь мы долго не задержимся. Только самое основное:

Центра́льный проце́ссор (**ЦП**, или *центральное процессорное устройство* — **ЦПУ**; англ. *central processing unit*, сокращенно — ***CPU***, дословно — *центральное обрабатывающее устройство*) — электронный блок либо микросхема — исполнитель машинных инструкций (кода программ), главная часть аппаратного обеспечения компьютера или программируемого логического контроллера. Иногда называют микропроцессором или просто процессором. Изначально термин *центральное процессорное устройство* описывал специализированный класс логических машин, предназначенных для выполнения сложных компьютерных программ. Вследствие довольно точного соответствия этого назначения функциям существовавших в то время компьютерных процессоров, он естественным образом был перенесён на сами компьютеры. Начало применения термина и его аббревиатуры по отношению к компьютерным системам было

положено в 1960-е годы. Устройство, архитектура и реализация процессоров с тех пор неоднократно менялись, однако их основные исполняемые функции остались теми же, что и прежде.

Главными характеристиками ЦПУ являются: тактовая частота, производительность, энергопотребление, нормы литографического процесса используемого при производстве (для микропроцессоров) и архитектура.

Теперь немного о работе процессора. Он работает по хорошо известному циклу:

Этапы цикла выполнения:

1. Процессор выставляет число, хранящееся в регистре счётчика команд, на шину адреса и отдаёт памяти команду чтения.
2. Выставленное число является для памяти адресом; память, получив адрес и команду чтения, выставляет содержимое, хранящееся по этому адресу, на шину данных и сообщает о готовности.
3. Процессор получает число с шины данных, интерпретирует его как команду (машинную инструкцию) из своей системы команд и исполняет её.
4. Если последняя команда не является командой перехода, процессор увеличивает на единицу (в предположении, что длина каждой команды равна единице) число, хранящееся в счётчике команд; в результате там образуется адрес следующей команды.

Данный цикл выполняется неизменно, и именно он называется *процессом* (откуда и произошло название устройства).

Во время процесса процессор считывает последовательность команд, содержащихся в памяти, и исполняет их. Такая последовательность команд называется программой и представляет алгоритм работы процессора. Очерёдность считывания команд изменяется в случае, если процессор считывает команду перехода, — тогда адрес следующей команды может оказаться другим. Другим примером изменения процесса может служить случай получения команды остановка или переключение в режим обработки прерывания.

Команды центрального процессора являются самым нижним уровнем управления компьютером, поэтому выполнение каждой команды неизбежно и безусловно. Не производится никакой проверки на допустимость выполняемых действий, в частности, не проверяется возможная потеря ценных данных. Чтобы компьютер выполнял только допустимые действия, команды должны быть соответствующим образом организованы в виде необходимой программы.

Скорость перехода от одного этапа цикла к другому определяется тактовым генератором. Тактовый генератор вырабатывает импульсы, служащие ритмом для центрального процессора. Частота тактовых импульсов называется тактовой частотой.

Для процессора понятие конвейер[93] является четким и определенным:

Конвейерная архитектура (*pipelining*) была введена в центральный процессор с целью повышения быстродействия. Обычно для выполнения каждой команды требуется осуществить некоторое количество однотипных операций, например: выборка команды из ОЗУ, дешифровка команды, адресация операнда в ОЗУ, выборка операнда из ОЗУ, выполнение команды, запись результата в ОЗУ. Каждую из этих операций сопоставляют одной ступени конвейера. Например, конвейер микропроцессора с архитектурой MIPS-I содержит четыре стадии:

- получение и декодирование инструкции,
- адресация и выборка операнда из ОЗУ,
- выполнение арифметических операций,
- сохранение результата операции.

После освобождения *k*-й ступени конвейера она сразу приступает к работе над следующей командой. Если предположить, что каждая ступень конвейера тратит единицу времени на свою работу, то выполнение команды на конвейере длиной в *n* ступеней займёт *n* единиц времени, однако в самом оптимистичном случае результат выполнения каждой следующей команды будет получаться через каждую единицу времени.

Действительно, при отсутствии конвейера выполнение команды займёт *n* единиц времени (так как для выполнения команды по-прежнему необходимо выполнять выборку, дешифровку и т. д.), и для исполнения *m* команд понадобится $n \cdot m$ единиц времени; при использовании конвейера (в самом оптимистичном случае) для выполнения *m* команд понадобится всего лишь $n + m$ единиц времени.

На этом разговор о процессорах можно и закончить. Основное сказано, остальное можно изучить позже…

---

[93] **Конве́йер** — это способ организации вычислений, используемый в современных процессорах и контроллерах с целью повышения их производительности (увеличения числа инструкций, выполняемых в единицу времени), технология, используемая при разработке компьютеров и других цифровых электронных устройств. http://ru.wikipedia.org/?oldid=37408434

Существующие системы адресации памяти.

Память, или устройства памяти являются основой любой логической системы в её техническом понимании. Это информационная база для работы логики.

Мы продолжаем разбираться в основах построения компьютера. Теперь мы должны немного познакомиться с существующими устройствами и адресацией памяти компьютера.

Что же такое компьютерная память[94]? Читаем:

Компью́терная па́мять (*устройство хранения информации, запоминающее устройство*) — часть вычислительной машины, физическое устройство или среда для хранения данных, используемых в вычислениях, в течение определённого времени. Память, как и центральный процессор, является неизменной частью компьютера с 1940-х. Память в вычислительных устройствах имеет иерархическую структуру и обычно предполагает использование нескольких запоминающих устройств, имеющих различные характеристики.

В персональных компьютерах «памятью» часто называют один из её видов — динамическая память с произвольным доступом (DRAM), — которая в настоящее время используется в качестве ОЗУ персонального компьютера.

Задачей компьютерной памяти является хранение в своих ячейках состояния внешнего воздействия, запись информации. Эти ячейки могут фиксировать самые разнообразные физические воздействия (см. ниже). Они функционально аналогичны обычному электромеханическому переключателю и информация в них записывается в виде двух чётко различимых состояний — 0 и 1 («выключено»/«включено»). Специальные механизмы обеспечивают доступ (*считывание*, произвольное или последовательное) к состоянию этих ячеек.

Процесс доступа к памяти разбит на разделённые во времени процессы — операцию записи (сленг. *прошивка*, в случае записи ПЗУ) и операцию чтения, во многих случаях эти операции происходят под управлением отдельного специализированного устройства — контроллера памяти.

Также различают операцию *стирания памяти* — занесение (запись) в ячейки памяти одинаковых значений, обычно 00_{16} или FF_{16}.

Наиболее известные запоминающие устройства, используемые в персональных компьютерах: модули оперативной памяти (ОЗУ), жёсткие диски (винчестеры), дискеты (гибкие магнитные диски), CD- или DVD-диски, а также устройства флеш-памяти.

Память на жестких дисках.

Начнем с памяти на жестких дисках[95].

На сегодняшний день это самая большая адресная память компьютера. Читаем о жестком диске[96]:

Минимальной адресуемой областью данных на жёстком диске является *сектор*. Размер сектора традиционно равен 512 байт.[18] В 2006 году IDEMA объявила о переходе на размер сектора 4096 байт, который планируется завершить к 2010 году[19]. Компания Western Digital уже сообщила[20] о начале использования новой технологии форматирования, названой *Advanced Format*, и выпустил серию накопителей, использующих новую технологию. К этой серии относятся линейки AARS/EARS и BPVT в отличие от BEVT, которые при тех же характеристиках используют "старый" 512-байтный кластер.

…Существует 2 основных способа адресации секторов на диске: *цилиндр-головка-сектор* (англ. *cylinder-head-sector, CHS*) и *линейная адресация блоков* (англ. *linear block addressing, LBA*).

CHS

При этом способе сектор адресуется по его физическому положению на диске 3 координатами — *номером цилиндра, номером головки* и *номером сектора*. В дисках, объёмом больше 528 482 304 байт (504 Мб), со встроенными контроллерами эти координаты уже не соответствуют физическому положению сектора на диске и являются «логическими координатами» (см. выше).

LBA

При этом способе адрес блоков данных на носителе задаётся с помощью логического линейного адреса. LBA-адресация начала внедряться и использоваться в 1994 году совместно со стандартом EIDE (Extended IDE). Стандарты ATA требуют однозначного соответствия между режимами CHS и LBA:

---

[94] Компьютерная память http://ru.wikipedia.org/?oldid=39959817

[95] **Накопи́тель на жёстких магни́тных ди́сках** или **НЖМД** (англ. *hard (magnetic) disk drive, HDD, HMDD*), *жёсткий диск*, в компьютерном сленге *«винче́стер»* — запоминающее устройство (устройство хранения информации) произвольного доступа, основанное на принципе магнитной записи. Является основным накопителем данных в большинстве компьютеров. В отличие от «гибкого» диска (дискеты), информация в НЖМД записывается на жёсткие (алюминиевые или стеклянные) пластины, покрытые слоем ферромагнитного материала, чаще всего двуокиси хрома — магнитные диски. В НЖМД используется одна или несколько пластин на одной оси. Считывающие головки в рабочем режиме не касаются поверхности пластин благодаря прослойке набегающего потока воздуха, образующейся у поверхности при быстром вращении.

[96] Жесткий диск. http://ru.wikipedia.org/?oldid=40141455

LBA = [(Cylinder * no of heads + heads) * sectors/track] + (Sector-1)

Метод LBA соответствует Sector Mapping для SCSI. BIOS SCSI-контроллера выполняет эти задачи автоматически, то есть для SCSI-интерфейса метод логической адресации был характерен изначально.

Оперативная память[97] компьютера.

Точнее, это устройства, реализующие функцию оперативной памяти[98].

Смотрим в Википедии:

ОЗУ большинства современных компьютеров представляет собой модули динамической памяти, содержащие полупроводниковые БИС ЗУ, организованные по принципу устройств с произвольным доступом. Память динамического типа дешевле, чем статического, и её плотность выше, что позволяет на том же пространстве кремниевой подложки размещать больше ячеек памяти, но при этом её быстродействие ниже. Статическая, наоборот, более быстрая память, но она и дороже. В связи с этим массовую оперативную память строят на модулях динамической памяти, а память статического типа используется для построения кеш-памяти внутри микропроцессора.

Память динамического типа *DRAM (Dynamic Random Access Memory)*

DRAM - Экономичный вид памяти. Для хранения разряда (бита или трита) используется схема, состоящая из одного конденсатора и одного транзистора (в некоторых вариациях конденсаторов два). Такой вид памяти решает, во-первых, проблему дороговизны (один конденсатор и один транзистор дешевле нескольких транзисторов) и во-вторых, компактности (там, где в SRAM размещается один триггер, то есть один бит, можно уместить восемь конденсаторов и транзисторов). Есть и свои минусы. Во-первых, память на основе конденсаторов работает медленнее, поскольку если в SRAM изменение напряжения на входе триггера сразу же приводит к изменению его состояния, то для того чтобы установить в единицу один разряд (один бит) памяти на основе конденсатора, этот конденсатор нужно зарядить, а для того чтобы разряд установить в ноль, соответственно, разрядить. А это гораздо более длительные операции (в 10 и более раз), чем переключение триггера, даже если конденсатор имеет весьма небольшие размеры. Второй существенный минус — конденсаторы склонны к «стеканию» заряда; проще говоря, со временем конденсаторы разряжаются. Причём разряжаются они тем быстрее, чем меньше их ёмкость. За то, что разряды в ней хранятся не статически, а «стекают» динамически во времени, память на конденсаторах получила своё название динамическая память. В связи с этим обстоятельством, дабы не потерять содержимое памяти, заряд конденсаторов для восстановления необходимо «регенерировать» через определённый интервал времени. Регенерация выполняется центральным микропроцессором или контроллером памяти, за определённое количество тактов считывания при адресации по строкам. Так как для регенерации памяти периодически приостанавливаются все операции с памятью, это значительно снижает производительность данного вида ОЗУ.

Память статического типа *SRAM (Static Random Access Memory)*

SRAM (память) - ОЗУ, которое не надо регенерировать (и обычно схемотехнически собранное на триггерах), называется *статической памятью с произвольным доступом* или просто *статической памятью*. Достоинство этого вида памяти — скорость. Поскольку триггеры собраны на вентилях, а время задержки вентиля очень мало, то и переключение состояния триггера происходит очень быстро. Данный вид памяти не лишён недостатков. Во-первых, группа транзисторов, входящих в состав триггера, обходится дороже, даже если они вытравляются миллионами на одной кремниевой подложке. Кроме того, группа транзисторов занимает гораздо больше места, поскольку между транзисторами, которые образуют триггер, должны быть вытравлены линии связи. Используется для организации сверхбыстрого ОЗУ, критичного к скорости работы.

---

[97] **Операти́вная па́мять** (англ. *Random Access Memory*, память с произвольным доступом; комп. жарг. *Память*) — энергозависимая часть системы компьютерной памяти, в которой временно хранятся данные и команды, необходимые процессору для выполнения им операции. Обязательным условием является адресуемость (каждое машинное слово имеет индивидуальный адрес) памяти. Передача данных в оперативную память процессором производится непосредственно, либо через сверхбыструю память. Содержащиеся в оперативной памяти данные доступны только тогда, когда компьютер включен. При выключении компьютера содержимое стирается из оперативной памяти, поэтому перед выключением компьютера все данные нужно сохранить. Так же от объёма оперативной памяти зависит количество задач, которые одновременно может выполнять компьютер.

[98] **Оперативное запоминающее устройство, ОЗУ** — техническое устройство, реализующее функции оперативной памяти. ОЗУ может изготавливаться как отдельный блок или входить в конструкцию, например однокристальной ЭВМ или микроконтроллера.

Флеш-память[99].

Сегодня это отдельный и самостоятельный вид устройств памяти:

Принцип работы полупроводниковой технологии флеш-памяти основан на изменении и регистрации электрического заряда в изолированной области (кармане) полупроводниковой структуры.

Изменение заряда («запись» и «стирание») производится приложением между затвором и истоком большого потенциала чтобы напряженность электрического поля в тонком диэлектрике между каналом транзистора и карманом оказалась достаточна для возникновения туннельного эффекта. Для усиления эффекта тунеллирования электронов в карман при записи применяется небольшое ускорение электронов путем пропускания тока через канал полевого транзистора (эффект Hot carrier injection (англ.)русск.).

Чтение выполняется полевым транзистором, для которого карман выполняет роль затвора. Потенциал плавающего затвора изменяет пороговые характеристики транзистора что и регистрируется цепями чтения.

Эта конструкция снабжается элементами которые позволяют ей работать в большом массиве таких же ячеек.

…Различаются методом соединения ячеек в массив и алгоритмами чтения-записи.

Конструкция **NOR** использует классическую двумерную матрицу проводников («строки» и «столбцы») в которой на пересечении установлено по одной ячейке. При этом проводник строк подключался к стоку транзистора, а столбцов к второму затвору. Исток подключался к общей для всех подложке. В такой конструкции было легко считать состояние конкретного транзистора подав положительное напряжение на один столбец и одну строку.

Конструкция **NAND** — трехмерный массив. В основе та же самая матрица что и NOR, но вместо одного транзистора в каждом пересечении устанавливается столбец из последовательно включенных ячеек. В такой конструкции затворных цепей в одном пересечении получается много. Плотность компоновки можно резко увеличить (ведь к одной ячейке в столбце подходит только один проводник затвора), однако алгоритм доступа к ячейкам для чтения и записи заметно усложняется.

Технология NOR позволяет получить быстрый доступ индивидуально к каждой ячейке, однако площадь ячейки велика. Наоборот, NAND имеют малую площадь ячейки, но относительно длительный доступ сразу к большой группе ячеек. Соответственно различается область применения: NOR используется как непосредственная память программ микропроцессоров и для хранения небольших вспомогательных данных. Топовые значения объемов микросхем NOR — 64 МБайт. NAND имеет топовые значения объема на микросхему в единицы гигабайт.

Названия NOR и NAND произошли от ассоциации схемы включения ячеек в массив со схемотехникой микросхем КМОП логики.

Нужно заметить, что существовали и другие варианты объединения ячеек в массив, но они не прижились.

…Различают приборы в которых элементарная ячейка хранит один бит информации и несколько. В однобитовых ячейках различают только два уровня заряда на плавающем затворе. Такие ячейки называют одноуровневыми (англ. *single-level cell*, ***SLC***). В многобитовых ячейках различают больше уровней заряда, их называют многоуровневыми (англ. *multi-level cell*, ***MLC***[2]). MLC-приборы дешевле и более емкие чем SLC-приборы, однако время доступа и количество перезаписей хуже.

… Стирание, запись и чтение флеш-памяти всегда происходит относительно крупными блоками разного размера, при этом размер блока стирания всегда больше блока записи, а размер блока записи не меньше, чем размер блока чтения. Собственно, это характерный отличительный признак флеш-памяти по отношению к классической EEPROM.

Как следствие все микросхемы флеш-памяти имеют ярко выраженную иерархическую структуру. Память разбивается на блоки, блоки состоят из секторов, секторы из страниц. В зависимости от назначения конкретной микросхемы глубина иерархии и размер элементов может меняться.

Например, NAND-микросхема может иметь размер стираемого блока в сотни кБайт, размер страницы записи и чтения 4 кБайт. Для NOR-микросхем размер стираемого блока варьируется от единиц до сотен кБайт, размер сектора записи — от единиц до сотен байт, страницы чтения — единицы-десятки байт.

…Для упрощения применения микросхем флеш-памяти NAND-типа они используются совместно со специальными микросхемами — NAND-контроллерами. Эти контроллеры должны выполнять всю черновую работу по обслуживанию NAND-памяти: преобразование интерфейсов и протоколов, виртуализация адресации (с целью обхода сбойных ячеек), проверка и восстановление данных при чтении, забота о разном размере блоков стирания и записи, забота о периодическом рефреше записанных блоков (есть и такое требование), равномерное распределение нагрузки на сектора при записи.

---

[99] **Флеш-память** (англ. *flash memory*) — разновидность полупроводниковой технологии электрически перепрограммируемой памяти (EEPROM). Это же слово используется в электронной схемотехнике для обозначения технологически законченных решений постоянных запоминающих устройств в виде микросхем на базе этой полупроводниковой технологии. В быту это словосочетание закрепилось за широким классом твердотельных устройств хранения информации.

Системы организации памяти с относительной адресацией.

Это уже системы организации памяти для облегчения поиска нужной информации или организации необходимой выборки. В информатике рассматриваются деревья и списки. Вот эти структуры[100] мы и рассмотрим.

Данные мы можем найти в Википедии:

В-дерево (по-русски произносится как **Б-дерево**) — структура данных, дерево поиска. С точки зрения внешнего логического представления, сбалансированное, сильно ветвистое дерево во внешней памяти.

Использование В-деревьев впервые было предложено Р. Бэйером (англ. *R. Bayer*) и Е. МакКрейтом (англ. *E. McCreight*) в 1970 году.

Сбалансированность означает, что длина любых двух путей от корня до листов различается не более, чем на единицу.

Ветвистость дерева — это свойство каждого узла дерева ссылаться на большое число узлов-потомков.

С точки зрения физической организации В-дерево представляется как мультисписочная структура страниц внешней памяти, то есть каждому узлу дерева соответствует блок внешней памяти (страница). Внутренние и листовые страницы обычно имеют разную структуру.

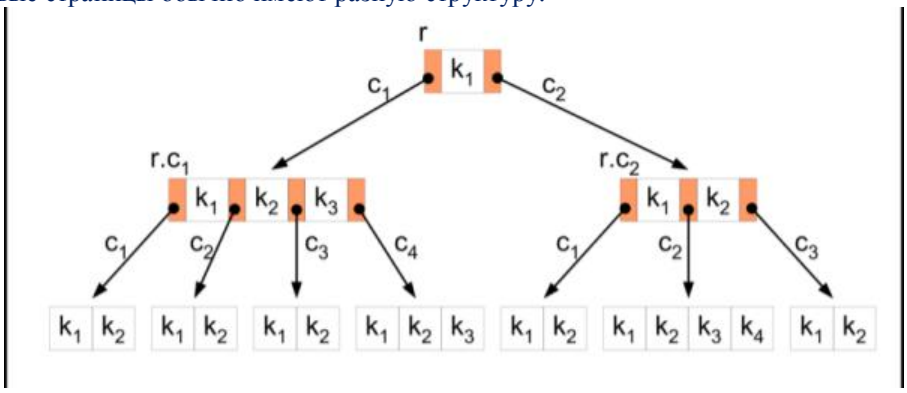

Рис. 4. Пример В-дерева степени 2

T-tree — сбалансированное дерево во внешней памяти, оптимизированное для случаев, когда востребованные (горячие) данные полностью хранятся в оперативной памяти. Данные хранятся в самих узлах дерева. Указатели переводят на следующий узел дерева.

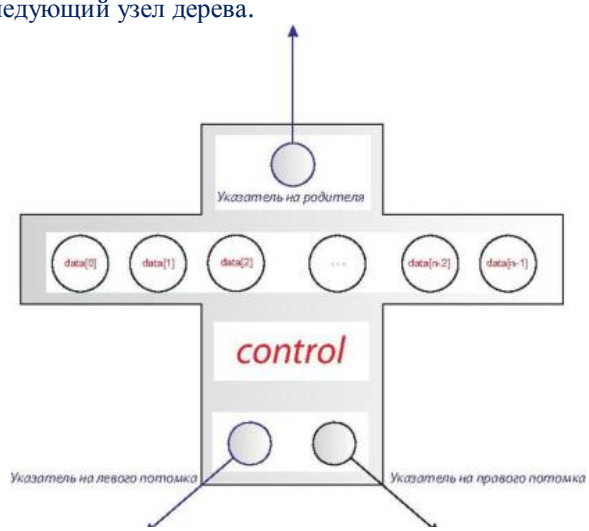

Рис. 5. T-tree.

Т-деревья сами не хранят копии индексированных полей данных в своих вершинах. Вместо этого, они пользуются тем, что горячие данные всегда содержатся в памяти вместе со своим индексом. Таким образом, они просто содержат ссылки на эти горячие данные.

---

[100] **Структура данных** — программная единица, позволяющая хранить и обрабатывать множество однотипных и/или логически связанных данных в вычислительной технике. Для добавления, поиска, изменения и удаления данных структура данных предоставляет некоторый набор функций, составляющих её интерфейс. Структура данных часто является реализацией какого-либо абстрактного типа данных. http://ru.wikipedia.org/?oldid=40233138

Линейный связный список

Односвязный список (Однонаправленный связный список)

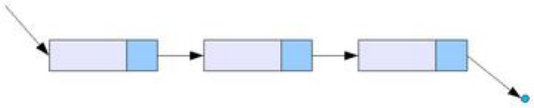

Рис. 6. Однонаправленный связный список.

Здесь ссылка в каждом узле указывает на следующий узел в списке. В односвязном списке можно передвигаться только в сторону конца списка. Узнать адрес предыдущего элемента, опираясь на содержимое текущего узла невозможно.

Двусвязный список (Двунаправленный связный список)

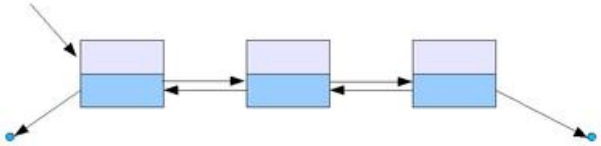

Рис. 7. Двунаправленный связный список.

Здесь ссылки в каждом узле указывают на предыдущий и на последующий узел в списке. По двусвязному списку можно передвигаться в любом направлении — как к началу, так и к концу. В этом списке проще производить удаление и перестановку элементов, так как всегда известны адреса тех элементов списка, указатели которых направлены на изменяемый элемент.

Кольцевой связный список

Разновидностью связных списков является кольцевой (циклический, замкнутый) список. Он тоже может быть односвязным или двусвязным. Последний элемент кольцевого списка содержит указатель на первый, а первый (в случае двусвязного списка) — на последний.

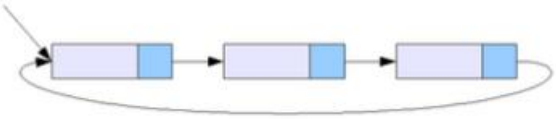

Рис. 8. Кольцевой связный список.

Реализация такой структуры происходит на базе линейного списка. С каждым кольцевым списком есть указатель на первый элемент. В этом списке константы NULL не существует.

Так же существуют циклические списки с выделенным головным элементом, облегчающие полный проход через список.

XOR-связный список — структура данных, похожая на обычный двусвязный список, однако в каждом элементе хранящая только один адрес — результат выполнения операции XOR над адресами предыдущего и следующего элементов списка. Для того, чтобы перемещаться по списку, необходимо взять два последовательных адреса и выполнить над ними операцию XOR, которая и даст реальный адрес следующего элемента.

Развёрнутый связный список — список, каждый физический элемент которого содержит несколько логических (обычно в виде массива, что позволяет ускорить доступ к отдельным элементам).

Позволяет значительно уменьшить расход памяти и увеличить производительность по сравнению с обычным списком. Особенно большая экономия памяти достигается при малом размере логических элементов и большом их количестве — так, односвязный список из 10 тысяч четырёхбайтных целых чисел при четырёхбайтной же адресации памяти займет 40 тысяч байт под собственно значения, плюс 40 тысяч байт под адреса, итого 80 тысяч байт; если же объединить числа в 100 массивов по 100 элементов, расход памяти на адреса упадёт до 400 байт, и суммарный расход составит 40400 байт.

Прирост производительности достигается за счёт того, что большая часть операций проводится над относительно небольшими массивами, которые обычно целиком помещаются в кэш-памяти. Благодаря этому, быстродействие программы может быть даже выше, чем при работе с обычными массивами. В развёрнутый список легко можно добавлять новые элементы — без необходимости переписывать весь массив, что является большой проблемой при работе с обычными массивами.

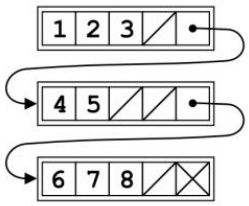

Рис. 9. Развернутый связный список.

При реализации необходимо тщательно выбирать размер «блока» (количество элементов в массивах). При слишком большом размере блока список начинает страдать от тех же проблем, что и обыкновенный массив: долгая вставка элементов в начало или середину, долгое удаление элементов оттуда же, и т.п. При слишком маленьком — увеличивается расход памяти.

Приведенные данные нам нужны для понимания того, чем мы сегодня располагаем. Хотя бы в общих чертах. Структуры хранения информации в виде дерева имеют распространение в экспертных системах. Линейные списки применяются в базах данных.

Способы реализации машинной памяти.

Если внимательно приглядеться к тем или иным устройствам памяти компьютера, то можно выделить три основных способа реализации работы памяти.

Первый способ реализован в устройствах внешней памяти, например в жестких дисках. Дисковое пространство памяти отделено от устройства внесения или съема информации с ячеек памяти - магнитной головки, и независимо от него. Магнитная головка двигается над пространством памяти и работает с памятью только в последовательном режиме. От ячейки к ячейке, от кластера к кластеру или от сектора к сектору. Параллельное считывание или запись информации в этом пространстве памяти достигается одновременной работой нескольких головок. Адресация памяти жестко связана с конструкцией накопителя и системой адресации жесткого диска. Поиск нужного участка памяти реализуется механическим перемещением, как диска, так и магнитной головки.

Второй способ реализован в устройствах оперативной памяти. В этом случае процесс запись-считывание непосредственно затрагивает схему каждой ячейки через цепи управления. Процесс запись-считывание индивидуален для каждой ячейки памяти. С другой стороны, это и обеспечивает любой вариант доступа к пространству памяти, как последовательный или параллельный, так их комбинации. Поиск нужного участка памяти осуществляется коммутатором цепей управления по системе ориентации в этом пространстве. Если найдены цепи управления участка памяти, то можно и управлять состоянием этих ячеек. Записывать с входных цепей или считывать информацию через цепи контроля состояния.

Третий способ реализован в организации систем относительной адресации. В деревьях или списках разной сложности. Это способ целевого логического образования связей по тому или иному признаку соединяемых информационных частей.

Но, целевой признак образования связей в списке или дереве остается за рамками цепочки информации. Соединение информационной цепи происходит по связи системных координатных адресов участков хранения информации.

Таким образом, относительность адресации выражена только для случайного определения следующего адреса при формировании списка в пространстве памяти машины и возможности его изменения исправлением записей адресов.

Формально, ни один из способов адресации объектов памяти не может быть применен в качестве основы для построения памяти логической системы управления, не имеющей первичной системы такого построения.

Но именно этот вопрос и стал основным в нашем понимании пространства памяти клетки и мозга, как ступеней развития самостоятельных логических систем управления.

О системе языков программирования.

Как говорят, языки программирования, это языки диалога человека и машины. Это так, и не совсем так. Пока, на сегодняшний день, это скорее языки управления, чем диалога. Почему?

Ну вот, давайте посмотрим сами…

Сначала общая часть. Заглянем в статью об основах программирования «вообще» [1]. Итак:

«Описание последовательности действий, достаточно определенное для того, чтобы ее можно было выполнить при помощи некоторого автоматического устройства называют алгоритмом (algorithm). Обычно эту последовательность записывают (кодируют) с помощью некоторых формальных обозначений. При этом формальная система, предназначенная для записи алгоритмов, называется алгоритмическим языком, сам текст алгоритма — программой, а процесс его создания — программированием». [1]

Вполне четко и понятно. Теперь немного об основном направлении программирования, или о парадигме[101].

Основная парадигма программирования.

Вот как говорится о парадигме программирования в Википедии:

Парадигма программирования — это совокупность идей и понятий, определяющих стиль написания программ. Парадигма в первую очередь определяется базовой программной единицей и самим принципом достижения модульности программы. В качестве этой единицы выступают определение (декларативное, функциональное программирование), действие (императивное программирование), правило (продукционное программирование), диаграмма переходов (автоматное программирование) и др. сущности. В современной индустрии программирования очень часто парадигма программирования определяется набором инструментов программиста, а именно, языком программирования и используемыми библиотеками.

Парадигма программирования определяет то, в каких терминах программист описывает логику программы. Например, в императивном программировании программа описывается как последовательность действий, а в функциональном программировании представляется в виде выражения и множества определений функций (слово *определение* (англ. *definition*) следует понимать в математическом смысле). В популярном объектно-ориентированном программировании (в дальнейшем ООП) программу принято рассматривать как набор взаимодействующих объектов. ООП, в основном[1], есть по сути императивное программирование, дополненное принципом инкапсуляции данных и методов в объект (принцип модульности) и наследованием (принципом повторного использования разработанного функционала).

Да, написано много. Придется разобраться более внимательно.

Теперь немного о разных направлениях программирования:

«Известно несколько основных парадигм программирования, важнейшими из которых на данный момент времени являются парадигмы *директивного*, *объектно-ориентированного* и *функционально-логического* программирования. Для поддержки программирования в соответствии с той или иной парадигмой разработаны специальные алгоритмические языки». [1]

Там же:

«*Функциональное и логическое программирование* использует языки типа Lisp, Haskell и Prolog. Эта парадигма базируется на принципиально иной трактовке понятия программы. Здесь главным является *точная формулировка задачи*, а выбор и применение необходимого для ее алгоритма решения -- проблема исполняющей системы, но не программиста». [1]

Вот теперь разберемся по группам:

- *директивное*, *объектно-ориентированное* программирование

---

[101] **Парадигма** (от греч. παράδειγμα, «пример, модель, образец») — совокупность фундаментальных научных установок, представлений и терминов, принимаемая и разделяемая научным сообществом и консолидирующая (объединяющая) большинство его членов. Обеспечивает преемственность развития науки и научного творчества. Википедия: парадигма

- *функционально-логическое* программирование

Директивное, *объектно-ориентированное* программирование, или *императивное*[102] программирование, это все языки, широко применяемые сегодня в программировании действий компьютера. Как можно представить, это направление выросло из команд Ассемблера и других языков низкого уровня для программирования машинных команд на самом низком уровне, уровне автоматического исполнения.

Как мы уже говорили, первыми программируемыми элементами в машинной логике стали машинные коды.

Читаем в Википедии:

«Первыми императивными языками были машинные коды — родной язык программирования для компьютера. В этих языках инструкции были крайне просты, что снижало нагрузку на компьютеры, однако затрудняло написание крупных программ».

И немного истории:

«В 1954 появился первый «человеческий» язык программирования — FORTRAN, разработанный Джоном Бэкусом в IBM. FORTRAN является компилируемым языком программирования и позволяет использовать именованные переменные, составные выражения, подпрограммы и многие другие элементы распространённых сейчас императивных языков.

В конце 1950х годов с целью упростить выражение математических алгоритмов был разработан ALGOL; в дальнейшем он послужил базой для написания операционных систем для некоторых моделей компьютеров. COBOL (1960) и BASIC (1964) являлись первыми попытками сделать программирование более похожим на обычный английский язык.

В 1970х годах Никлаус Вирт разработал язык Pascal.

Язык C был создан Денисом Ритчи.

Команда разработчиков из Honeywell начала разработку языка Ada в 1978, и через четыре года опубликовала требования для его работы. Спецификация увидела свет в 1983 и была обновлена в 1995 и 2005/6 годах».

Как мы видим, все основные языки программирования в первичном варианте разрабатывались, как императивы. Прямые команды для исполнения. Но время и развитие вычислительной техники потребовали нового подхода.

Там же:

«В 1980х резко возрос интерес к объектно-ориентированному программированию. Smalltalk-80, впервые разработанный Аланом Кэем в 1969, был обновлён в 1980 исследовательским центром Xerox PARC.

По образу и подобию языка Simula (предположительно, первого в мире ООП-языка, разработанного ещё в 1960х) Бьерн Страуструп разработал C++, основанный на C. C++ был впервые реализован в 1985.

В 1987 Ларри Уолл выпустил язык Perl;

Python был выпущен в 1990 Гвидо ван Россумом;

в 1994 Расмус Лердорф разработал PHP;

Java была разработана в Sun Microsystems в 1994;

Ruby был выпущен в 1995;

C# был зачат в декабре 1998.

В 2002 — система .NET Framework, объединяющая многие языки».

Собственно, директива отличается от императива только расширением понятия команды. Она стала включать не только действие, но и подробное описание объектов, организующих или исполняющих это действие. Это и определило дальнейшее развитие подходов к программированию. Об этом мы уже говорили и ранее.

---

[102] **Императи́вное программи́рование** — это парадигма программирования, которая, в отличие от декларативного программирования, описывает процесс вычисления в виде инструкций, изменяющих состояние программы. Императивная программа очень похожа на приказы, выражаемые повелительным наклонением в естественных языках, то есть это последовательность команд, которые должен выполнить компьютер. http://ru.wikipedia.org/?oldid=39013582

На самом деле объектно-ориентированное программирование появилось вместе с развитием средств отображения элементов управления на экране монитора. Это резко расширило возможности интерактивного управления, как конкретной программой, так и компьютером в целом. Резко возросла наглядность управления. Но, за все надо платить…

И программирование ушло в частности описания этих экранных объектов управления. Постепенно сформировались общие правила и подходы к этим объектам. Они и стали основой объектно-ориентированного программирования.

Второй составляющей этой парадигмы является *декларативное*[103] программирование. Вот как говорится в [1]:

«Программа на декларативных языках представляет собой описание объектов, и связей между ними. В функциональных языках эти связи представляются функциями в логических - отношениями (предикатами).

Вроде бы, формально все так же. Но различия оказываются очень существенными:

«Императивные языки программирования противопоставляются функциональным и логическим языкам программирования. Функциональные языки, например, Haskell, не представляют собой последовательность инструкций и не имеют глобального состояния. Логические языки программирования, такие как Prolog, обычно определяют ***что*** надо вычислить, а не ***как*** это надо делать».

Это уже дилемма, требующая разрешения.

И хоть функциональное и логическое программирование изначально выделены в одну группу, это скорее отражение основной парадигмы, чем действительная схожесть понимания программирования, как процесса. Как мы видим, возникает еще один подход. Мы приходим к *функциональному* и *логическому* программированию.

Отметим, что языки функционального и логического программирования изначально исходили из логических функций, а не ориентировались на машинные команды. Они сразу разрабатывались, как языки высокого уровня. Вот это мы и отметим особо. К этому мы еще вернемся. А пока, просто посмотрим чуть внимательнее.

Функциональное программирование[104].

Читаем:

«… Функции в функциональных языках "однонаправлены". Они получают аргументы и возвращают результат. В логических языках разница между вводом и выводом условна. Мы можем указать желаемый вывод и получить ввод, который его обеспечит. Другое важное отличие - недетерминизм логических языков. Результат в них не обязательно определяется однозначно». [1]

---

[103] **Декларативное программирование** — термин с двумя различными значениями.

Согласно первому определению, программа «декларативна», если она описывает *каково́* нечто, а не *как его создать*. Например, веб-страницы на HTML декларативны, так как они описывают *что* должна содержать страница, а не *как отображать* страницу на экране. Этот подход отличается от языков императивного программирования, требующих от программиста указывать алгоритм для исполнения. В типично декларативном языке программирования XSLT, последовательность исполнения зависит, как правило, от входящего XML (в случае с использованием push-модели — «проталкивание»), в случае использования pull-модели (вытягивания), XSLT вырождается в частный случай функционального программирования и легко может быть заменена на аналогичный код в XQuery.

Согласно второму определению, программа «декларативна», если она написана на исключительно функциональном, логическом или языке программирования с ограничениями. Выражение «декларативный язык» иногда употребляется для описания всех таких языков программирования как группы, чтобы подчеркнуть их отличие от императивных языков.

Программы на языках декларативного программирования легко поддаются методикам метапрограммирования — когда программа может генерироваться по её описанию. Например XSLT-программа может быть сгенерирована из файла XML (часто с помощью другой XSLT) — см. Schematron. http://ru.wikipedia.org/?oldid=39014057

[104] **Функциона́льное программи́рование** — раздел дискретной математики и парадигм программирования, в которой процесс вычисления трактуется как вычисление значений функций в математическом понимании последних (в отличие от функций как подпрограмм в процедурном программировании). Противопоставляется парадигме императивного программирования, которая описывает процесс вычислений как последовательность изменения состояний (в значении, подобном таковому в теории автоматов). Функциональное программирование не предполагает изменяемость данных (в отличие от императивного, где одной из базовых концепций является переменная). http://ru.wikipedia.org/?oldid=39740789

В Википедии находим:

Наиболее известными языками функционального программирования являются[2]:
LISP - (Джон МакКарти, 1958) и множество его диалектов, наиболее современные из которых:
Scheme
Clojure
Common Lisp
F# — функциональный язык для платформы .NET
Haskell — чистый функциональный. Назван в честь Хаскелла Карри.
Erlang — (Joe Armstrong, 1986) функциональный язык с поддержкой процессов.
APL — предшественник современных научных вычислительных сред, таких как MATLAB.
ML (Робин Милнер, 1979, из ныне используемых диалектов известны Standard ML и Objective CAML).
Scala
Miranda (Дэвид Тёрнер, 1985, который впоследствии дал развитие языку Haskell).
Nemerle — гибридный функционально/императивный язык.
XQuery

Функция, как она понимается в функциональном программировании, и отражает понятие действия. Действие лежит в основе этого программирования, но очень специфическое, только в его математическом понимании. Но, это не помешало расширить его понятие для применения в широком классе задач. Мы это отметим.

Логическое программирование.[105]

Пожалуй, единственный и хорошо разработанный язык – ПРОЛОГ. О нем достаточно полно изложено в [3]. И если уж начинать рассматривать основы системы логики машины, как самостоятельного интеллекта, то основой может стать ПРОЛОГ.

Немного истории:

«Первым языком логического программирования был язык Planner, в котором была заложена возможность автоматического вывода результата из данных и заданных правил перебора вариантов (совокупность которых называлась планом). Planner использовался для того, чтобы понизить требования к вычислительным ресурсам (с помощью метода backtracking) и обеспечить возможность вывода фактов, без активного использования стека. Затем был разработан язык Prolog, который не требовал плана перебора вариантов и был, в этом смысле, упрощением языка Planner.

От языка Planner также произошли логические языки программирования QA-4, Popler, Conniver и QLISP. Языки программирования Mercury, Visual Prolog, Oz и Fril произошли уже от языка Prolog. На базе языка Planner было разработано также несколько альтернативных языков логического программирования, не основанных на методе поиска с возвратами (backtracking), например, Ether (см. обзор Шапиро [1989])».

Собственно, на этом и можно пока закончить знакомство со структурой языков программирования, применяемых для компьютера. И основной парадигмой. Как мы видим, это парадигма об одной направленности разных подходах и их комбинировании в конкретной программе. Начальные различия давно сошлись в единый структурный комплекс и различаются лишь в теории программирования. Различия остались только в средствах, в языках программирования.

Но, есть все же, одно различие в подходе, не устраняемое ни чем. Директивное и логическое программирование различны изначально. Это принципиально разные пути управления процессом решения. И пусть пока они базируются на одной и той же математике и двоичной логике, но противоречие в подходе разводит их все дальше и дальше.

Программный ИИ должен иметь самостоятельность в выборе пути решения. Этот путь возможен только при применении одного вида программирования – логического. Никакие директивные языки тут не помогут.

---

[105] **Логи́ческое программи́рование** — парадигма программирования, основанная на автоматическом доказательстве теорем, а также раздел дискретной математики, изучающий принципы логического вывода информации на основе заданных фактов и правил вывода. Логическое программирование основано на теории и аппарате математической логики с использованием математических принципов резолюций. http://ru.wikipedia.org/?oldid=37497437

О языках программирования ИИ.

Логика определяется, все же, как наука о знаниях и о законах интеллектуальной и познавательной деятельности. Это, как мы видим, не совсем то, о чем мы говорим.

Но,… знания, это информация. И математическая логика здесь не является исключительной. Как и привычная классическая логика, она точно также ориентирована на получение и обработку информацию. На получение, хоть и машинных, но знаний, в данном случае в виде информации, результата, логического ответа…

На это, в первую очередь и были направлены усилия программирования машинного интеллекта. Сразу, на управление этим, еще не созданным, интеллектом, а не на его самостоятельность.

Читаем здесь:

«Маккарти[106] отстаивал использование математической логики для искусственного интеллекта. В 1958 г. он предложил систему «принятия советов», которая позже вдохновила работы по ответам на запросы и логическому программированию. В том же году он изобрел язык программирования Лисп и опубликовал его описание в журнале Communications of the ACM в 1960 г.»

Язык Лисп[107] родился с направленностью на ИИ. Он входит в класс функциональных языков программирования, а не логических, как можно было бы подумать. Логический вариант появился позже. (см. QLISP)

Пролог.

Теперь чуть внимательнее присмотримся к языку Пролог[108].

Что написано о Прологе в Википедии:

«Основными понятиями в языке Пролог являются факты, правила логического вывода и запросы, позволяющие описывать базы знаний, процедуры логического вывода и принятия решений.

Факты в языке Пролог описываются логическими предикатами с конкретными значениями. Правила в Прологе записываются в форме правил логического вывода с логическими заключениями и списком логических условий.

Особую роль в интерпретаторе Пролога играют конкретные запросы к базам знаний, на которые система логического программирования генерирует ответы «истина» и «ложь». Для обобщённых запросов с переменными в качестве аргументов созданная система Пролог выводит конкретные данные в подтверждение истинности обобщённых сведений и правил вывода.

Факты в базах знаний на языке Пролог представляют конкретные сведения (знания). Обобщённые сведения и знания в языке Пролог задаются правилами логического вывода (определениями) и наборами таких правил вывода (определений) над конкретными фактами и обобщёнными сведениями.

Начало истории языка относится к 1970-м годам.[1] Будучи декларативным языком программирования, Пролог воспринимает в качестве программы некоторое описание задачи или баз знаний и сам производит логический вывод, а также поиск решения задач, пользуясь механизмом поиска с возвратом и унификацией».

Если рассматривать конкретные примеры языка Пролог, то можно увидеть там то, что мы больше нигде не видим. Цели.[3]

В языке, по мнению автора, предусмотрены процедуры работы с целями на основе предикатов. Конъюнкция, дизъюнкция…

---

[106] **Джон Маккарти** (4 сентября 1927, Бостон — 24 октября 2011[1][2][3], Стэнфорд) — выдающийся американский информатик, автор термина «искусственный интеллект» (1955), изобретатель языка Лисп (1958), основоположник функционального программирования, лауреат Премии Тьюринга (1971) за огромный вклад в область исследований искусственного интеллекта. http://ru.wikipedia.org/?oldid=40387478

[107] **Лисп** (*LISP*, от англ. *LISt Processing language* — «язык обработки списков»; современное написание: *Lisp*) — семейство языков программирования, программы и данные в которых представляются системами линейных списков **символов**. Лисп является вторым в истории (после Фортрана) используемым по сей день высокоуровневым языком программирования. Создатель Лиспа Джон Маккарти занимался исследованиями в области искусственного интеллекта (в дальнейшем ИИ) и созданный им язык по сию пору является одним из основных средств моделирования различных аспектов ИИ. http://ru.wikipedia.org/?oldid=39550636

[108] **Пролог** (фр. ***Pro**gramma**t**ion en **Log**ique*) — язык и система логического программирования, основанные на языке предикатов математической логики дизъюнктов Хорна, представляющей собой подмножество логики предикатов первого порядка. http://ru.wikipedia.org/?oldid=39614756

Вот, например, в [3]:

«Главной операцией в процессе выполнения пролог-программы, является **сопоставление** (согласование, унификация) термов.

Например,

parent(pam, bob).

?- parent(pam, bob).

Yes

Цель согласуется.

?-parent(pam, X).

X=bob

Тоже согласуются, но **X** конкретизируется и принимает значение **bob».** [3]

Но, будем справедливы, то, что понимается здесь, как цель, в данном случае таковым является лишь отчасти. Как мы видим, все авторы [3,7,8,9] говорят о целевом характере решения задач, а понятие «цели» в языке Пролог при этом отсутствует.

Скорее всего, понимание целевой направленности Пролога появилось несколько позже, уже при интерпретации возможностей языка в решениях логических задач, а не при его разработке. Это и объясняет только косвенное указание цели решения.

В [8] цель появляется в явном виде только в приложениях. Например, при создании оболочки для экспертной системы:

«**Цель** - это вопрос, подлежащий рассмотрению; **Трасса** - это цепочка, составленная из "целей-предков" и правил, находящихся между вершиной **Цель** и вопросом самого верхнего уровня; **Ответ** - решающее дерево типа И/ ИЛИ для вершины **Цель»**. [8]

И, тем не менее, в исследуемом нами аспекте Пролог не может самого главного. Он не может работать на уровне автоматических операций, как, например Ассемблер, т.к. изначально разрабатывался, как язык высокого уровня. Для компьютера. В качестве средства диалога оператора и машины. Но, как мы видим, получился, все равно, монолог. Оператора, программиста, пользователя…, а машина только исполняет данные установки, переводя написанную программу в машинные коды, и далее работает стандартным методом поиска решения.

Но, даже это применение понятия цели сразу создает определенный интерес и к самому языку и к способу достижения поставленной в задаче цели.

Ассемблер.

В Википедии читаем:

«Команды языка ассемблера один к одному соответствуют командам процессора. Фактически, они и представляют собой более удобную для человека символьную форму записи — *мнемокоды* — команд и их аргументов. При этом одной команде языка ассемблера может соответствовать несколько вариантов команд процессора.[1]

Кроме того, язык ассемблера позволяет использовать символические метки вместо адресов ячеек памяти, которые при ассемблировании заменяются на вычисляемые ассемблером или компоновщиком абсолютные или относительные адреса, а также так называемые *директивы* (команды ассемблера, не переводимые в машинные команды процессора, а выполняемые самим ассемблером).

Директивы ассемблера позволяют, в частности, включать блоки данных, задать ассемблирование фрагмента программы по условию, задать значения меток, использовать макрокоманды с параметрами.

Каждая модель (или семейство) процессоров имеет свой набор — систему — команд и соответствующий ему язык ассемблера. Наиболее популярные синтаксисы языков ассемблера — Intel-синтаксис и AT&T-синтаксис».

И там же:

«…Исторически можно рассматривать язык ассемблера как второе поколение языков программирования ЭВМ. (Если первым считать числовые двоичные коды машинных команд.) Недостатки языка ассемблера, сложность разработки на нём больших программных комплексов привели к появлению языков третьего поколения — языков программирования высокого уровня.

… Достоинства: Язык ассемблера позволяет писать самый быстрый и компактный код, какой вообще возможен для данного процессора».

Ассемблер[109] является переходным языком от машинных команд к языкам диалога с машиной. В том числе и к языкам высокого уровня. Более подробно об ассемблере можно прочитать, например, в [12, 13]. Конечно, это язык императива. Прямые команды безусловного исполнения. Потому, что это язык автоматического управления процессом обработки информации и вычислениями. Для каждого типа процессора есть и свой Ассемблер. Это мы поняли.

Еще немного о машинных языках.

Мы посмотрели только самые общие данные о системе языков машинного программирования и немного коснулись конкретики языков Пролог и Ассемблер. Чуть-чуть.

Но тенденция понятна. Развитие программирования шло сразу с двух сторон. Со стороны усложнения машинных команд через ассемблеры и со стороны необходимости получения свободы действий в формировании алгоритма решения. И, к сожалению, пока эти встречные направления развития так не добрались до единого языка.

В компьютере, как кажется с первого взгляда, действует многоуровневая система языковых сред, определяемая основными техническими составляющими вычислительной архитектуры машины и принципами фон Неймана.

Но, если приглядеться внимательно, то никакой многоуровневой системы языковых сред в компьютере нет. Есть только один уровень – машинных команд. А всё остальное, не более чем временные надстройки, на период создания программы.

Как только процесс программирования закончен, программа транслируется в машинных кодах, и уже в таком виде начинает работать. Только в таком. Других вариантов нет. Разница только в том, сразу вся программа транслируется, или по частям, как в первых версиях языка Бейсик. Для этого и работают трансляторы, компиляторы и интерпретаторы разных языков программирования.

И вся видимая интерактивность той или иной исполнительной программы, даже с элементами ИИ, заключается в подаче нужной машинной команды для проведения того или иного программного действия. Что мы и делаем, щелкая мышкой в нужных местах экрана, или набираем заранее запланированные слова с клавиатуры.

Голосовой интерактив несколько сложнее, но, в конечном итоге, и звуки нашего голоса трансформируются в те же машинные команды по уже заданному сценарию программы.

Разобрались…

С основными, принципиальными составляющими компьютера, хоть и на очень примитивном уровне, но разобрались. Наш короткий экскурс по основам компьютерной техники только подтвердил то, что предполагалось и раньше: вся архитектура компьютера нацелена на один вид обработки информации – вычисления. Логическая обработка для

---

[109] **Язы́к ассе́мблера** — язык программирования низкого уровня, мнемонические команды которого (за редким исключением) соответствуют инструкциям процессора вычислительной системы. Трансляция программы в исполняемый машинный код производится *ассемблером* (от англ. *assembler* — сборщик) — программой-транслятором, которая и дала языку ассемблера его название.

Команды языка ассемблера один к одному соответствуют командам процессора. Фактически, они и представляют собой более удобную для человека символьную форму записи — *мнемокоды* — команд и их аргументов. При этом одной команде языка ассемблера может соответствовать несколько вариантов команд процессора.[1] http://ru.wikipedia.org/?oldid=39997219

Ассе́мблер (от англ. *assembler* — сборщик) — компьютерная программа, компилятор исходного текста программы, написанной на языке ассемблера, в программу на машинном языке. http://ru.wikipedia.org/?oldid=39586428

компьютера не является основной, да и вообще не входит в архитектуру, как отдельный вид. Только, как вспомогательный и вынужденный прием, для согласования вычислений и программы.

Такое положение будет сохраняться, пока работают принципы фон Неймана и принятая система процессорной обработки информации. И, кстати, пока существует прикладное программирование. Потому, что оно основано на директивных методах управления решением поставленной задачи.

Нет в этом ничего ни плохого, ни хорошего, это существующая реальность.

Компьютер был, есть и будет инструментом в человеческих руках. Сложным, многофункциональным, но все же – инструментом. Для решения наших задач.

Но, если мы хотим полноценного диалога с машиной, а не только широкой интерактивности в программах, то надо менять подходы к машине. И принципы её работы.

Если подходить к ИИ с этих позиций, то необходимо снова вернуться к логической машине, о которой мы уже начали разговор в [11].

Видимо, только полный отказ от компьютера, как основы для построения ИИ может приблизить нас к реальным результатам в этом направлении. Для интеллекта нужен новый технический подход и новая основа. Логическая машина.

И новая основа формализации логики.

Математика не может быть единственной системой формализации логических понятий. Механистическое понимание логики включает математику, как составную часть, но не более. Одну из нескольких. Мы уже хорошо понимаем, насколько важна математика в построении логических законов, но, может быть, надо оценить и другие возможности. В том числе и самой логики…, как основы системы управления.

Направления развития логической машины.

Все, кто хоть немного занимается проблемами ИИ, так или иначе, касаются и проблем создания технического устройства, реализующего этот самый ИИ.

Далее, уже кто – о чем…

Но, всё же, направления рассмотрения уровней реализации и набора функций логической машины очерчены достаточно четко.

Компьютерные модели.

Основная масса исследований в этом направлении имеет основой … компьютер. Парадокс. Но, видимо, привычка – вторая натура, что видим, то и ставим в качестве эталона. Компьютерные модели логической машины занимают лидирующее положение в сумме всех наработок. Конечно, к реальной логической машине все это имеет очень слабое отношение, но громкие заявления авторов и СМИ делают свое дело в раскрутке этих проектов. В эту группу разработок можно отнести и вполне практичные *нейронные сети*[110], как метод распределенных вычислений, и *нейрокомпьютер*[111], как новый вид устройства для сложных вычислений. Где-то здесь находят свое место экспертные системы. Ну и конечно же,

---

[110] **Иску́сственные нейро́нные се́ти** (ИНС) — математические модели, а также их программные или аппаратные реализации, построенные по принципу организации и функционирования биологических нейронных сетей — сетей нервных клеток живого организма. Это понятие возникло при изучении процессов, протекающих в мозге, и при попытке смоделировать эти процессы. Первой такой попыткой были нейронные сети Маккалока и Питтса [1]. Впоследствии, после разработки алгоритмов обучения, получаемые модели стали использовать в практических целях: в задачах прогнозирования, для распознавания образов, в задачах управления и др. http://ru.wikipedia.org/?oldid=41023963

[111] **Нейрокомпьютер** — устройство переработки информации на основе принципов работы естественных нейронных систем.[1] Эти принципы были формализованы, что позволило говорить о теории искусственных нейронных сетей. Проблематика же нейрокомпьютеров заключается в построении реальных физических устройств, что позволит не просто моделировать искусственные нейронные сети на обычном компьютере, но так изменить принципы работы компьютера, что станет возможным говорить о том, что они работают в соответствии с теорией искусственных нейронных сетей. http://ru.wikipedia.org/?oldid=41024496

разнообразные проекты типа «супермозг[112]» на базе Интернета и совершенствовании поисковых систем…

Понятно, что все эти проекты имеют в основе кибернетическое понимание функций мозга начала 20 века. И тот же подход. Глобальный по заявленным целям, и совершенно нереальный в плане реализации ИИ.

Тем не менее, все эти проекты имеют практическую пользу и дополняются современным программированием высокого уровня. По этой причине они и имеют широкую известность.

Реальной целью всех этих проектов является совсем не создание ИИ, а расширение применения компьютерных технологий во всех сферах нашей жизни. Приставки «нейро», «генетические», «интеллект» … и т.д. здесь применены по приближенным функциональным аналогиям и никакого реального отношения к ИИ, и тем более к логической машине, не имеют. Мы это уже рассматривали, и не раз [11, 15].

Классика…

Вторым направлением развития логической машины стала разработка устройства, реализующего алгоритмы решений на основе булевой логики и понимания ИИ, как системы для решения логических задач. Это классический путь понимания ИИ и Разума в трудах ученых, основа кибернетического подхода и современного логического программирования.

По этому пути наука идет уже сотню лет. На основе этого понимания были сделаны первые электронные логические машины, ставшие на какое-то время основой для кибернетики в начале её становления. Под эти машины проводились работы по развитию математики логики, реализованные уже на другой основе – компьютере.

Сейчас по этому же пути идет программный ИИ. Он имеет целью реализовать в компьютерной программе то, что не смогла реализовать электронная логическая машина, строившаяся на классических принципах. Примером достижения в этом направлении можно считать проекты «танцующего робота[113]» Асимо[114] японских разработчиков. Есть и другие, не менее интересные разработки[115].

Нейрон и мозг.

Третье направление развития логической машины имеет в основе нейронную теорию ИИ. Сначала реализуется «нейрон[116]» на базе процессора, затем из таких нейронов надо создать систему, позволяющую применить алгоритмы решений логических задач.

Это направление, вышедшее из нейросетей, уже ориентировано на создание устройства, приближенного по способам решения задач к мозгу, к логической машине. Но, пока все разработки данного направления находятся в начальной стадии, реальных устройств нет.

И, тем не менее, именно с развитием этого направления связаны надежды и ожидания получения хороших результатов в реализации ИИ.

---

[112] «Глобальный мозг» — направление, связанное с исследованием коллективного интеллекта человечества, объединенного с помощью информационных и сетевых (Internet) технологий в единую интеллектуальную систему. http://ru.wikipedia.org/?oldid=40489701

[113] http://newrobots-world.ucoz.ru/publ/1-1-0-23

[114] Например, здесь: http://www.liveinternet.ru/community/1726655/post61473208

[115] http://newrobots-world.ucoz.ru/publ/1-1-0-19

[116] **Иску́сственный нейро́н** (*Математический нейрон* Маккалока — Питтса, *Формальный нейрон*[1]) — узел искусственной нейронной сети, являющийся упрощённой моделью естественного нейрона. Математически, искусственный нейрон обычно представляют как некоторую нелинейную функцию от единственного аргумента — линейной комбинации всех входных сигналов. Данную функцию называют *функцией активации*[2] или *функцией срабатывания*, *передаточной функцией*. Полученный результат посылается на единственный выход. Такие искусственные нейроны объединяют в сети — соединяют выходы одних нейронов с входами других. Искусственные нейроны и сети являются основными элементами идеального нейрокомпьютера.[3] http://ru.wikipedia.org/?oldid=38008787

Развитие этого направления идет сразу в двух направлениях. Со стороны «нейрона», и со стороны функциональной схемы в целом. Оба направления развиваются достаточно уверенно. На основе имитации и эмуляции реальных функций и связей.

В рамках этого направления реализован компьютерный проект эмуляции деятельности мозга[117] мощностью в 1млрд. нейронов на базе реальной картины коры головного мозга. Идет подготовка к реализации такой же эмуляции работы человеческого мозга мощностью в 20млрд. нейронов.

Разработки электронных аналогов нейрона исчисляются уже сотнями. Уточняются и систематизируются функции, входные и выходные характеристики, количество возможных связей с другими нейронами системы, и т.д.

Каждый год публикуются все новые и новые разработки. Например, [51].

Био-логические машины.

Название взято из работ Ю.Я. Калашникова [52]. Как мне кажется, это наиболее точное название, отражающее смысл и понимание основы развития этого направления.

Это отдельное направление развития ИИ. На основе клеточных процессов. Направление начало активно развиваться после понимания роли ДНК и РНК в жизни клетки.

Отличительным признаком этого направления является применение реальных биологических объектов для реализации решений тех или иных задач в области логики и математики. Понятно, что разработка этого направления находится еще в самом начале пути. И, тем не менее, оценки его развития вполне оптимистичны. Кое-какие результаты уже есть.

Например, на коротких цепочках РНК[118] было реализовано вероятностное решение сложной транспортной задачи. Те же РНК использованы в реализации функций булевой логики[119]. Созданы простейшие искусственные клетки[120] с возможностью их роста и деления. Ведется активная разработка биологических нанодвижителей и аккумуляторов энергии на основе белков и РНК для нанороботов[121]. Намечаются подходы к созданию биологического процессора[122] для создания на их базе ИИ.

Очень интересна разработка биологического электронного элемента памяти[123], ключа на шесть положений фиксации логического состояния. Видимо, это на его основе генная инженерия уже задумывается о ДНК и РНК с шестью основаниями и делает первые шаги в этом направлении[124].

С другой стороны, едет осознание сложности реальных клеточных процессов управления в целом. Их взаимосвязей и взаимовлияний.

Вот, например, схема, предложенная Ю.Я. Калашниковым [52]. Она на рис.10.

---

[117] http://vlasti.net/news/66792

[118] http://www.membrana.ru/particle/16232

[119] http://www.dailytechinfo.org/news/2981-uchenye-sozdali-bazovye-logicheskie-elementy-iz-bakteriy-i-dnk.html или http://www.nature.com/ncomms/journal/v2/n10/full/ncomms1516.html

[120] http://news.rambler.ru/11365390/

[121] http://www.kit-e.ru/articles/elcomp/2008_09_151.php

[122] http://www.top-page.ru/daily_news/hitech/2042357 или http://www.dpk.com.ua/content/18287?cpage=1

[123] http://www.dailytechinfo.org/nanotech/3125-issledovateli-sozdali-elektronnyy-logicheskiy-element-sostoyaschiy-iz-edinstvennoy-molekuly.html

[124] http://www.membrana.ru/particle/374

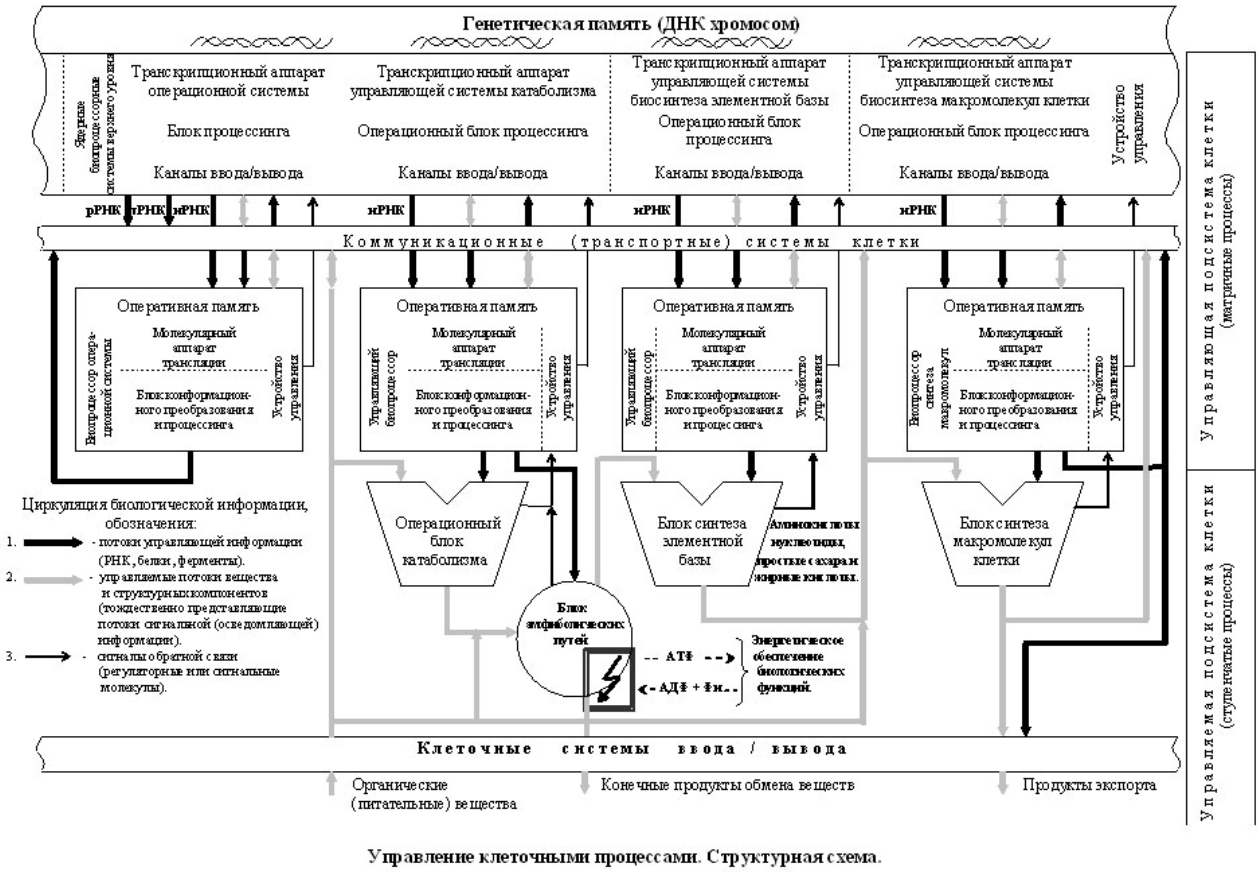

Рис. 10. Схема управления клетки, предложенная Ю.Я.Калашниковым.

Как мы видим, схема включает в себя все основные функциональные составляющие логической машины, реализующей управление клеткой. Причем, в реальном биологическом исполнении. На основе реальных, известных науке, взаимодействий клеточных и органических структур. Схема позволяет оценить реальную сложность клеточных процессов в системе управления.

Заключение

Похоже, что научное понимание логики и ИИ начинает отходить от классического философского осмысления этих понятий на основе категорий сознания, воли, свободы… и сдвигается в область механистического понимания.

С другой стороны, именно такой сдвиг понимания проблемы ИИ и механистической логики таит в себе новые опасности. Видимо настал тот момент, когда мы должны еще раз задуматься о законах роботехники [125]А.Азимова[126] Вот они:

---

[125] **Я, робот** -Сборник научно-фантастических рассказов Айзека Азимова опубликованный в 1950 году американским издательством «Gnome Press», оказавший большое влияние на современную научно-фантастическую литературу, где впервые были сформулированы Три закона роботехники.

Неоднократно переиздавался и переводился на многие языки. Впервые переведён на русский язык и издан в 1964 году (переводчик Алексей Иорданский)[1]. http://ru.wikipedia.org/?oldid=39787551

[126] **Áйзек Азúмов** (англ. *Isaac Asimov*, имя при рождении **Исаак Юдович Озимов**[1]; 2 января 1920 — 6 апреля 1992) — американский писатель-фантаст, популяризатор науки, по профессии биохимик. Автор около 500 книг, в основном художественных (прежде всего в жанре научной фантастики, но также и в других жанрах: фэнтези, детектив, юмор) и научно-популярных (в самых разных областях — от астрономии и генетики до истории и литературоведения). Многократный лауреат премий Хьюго и Небьюла. Некоторые термины из его произведений — robotics (роботехника, роботика), positronic (позитронный), psychohistory (психоистория, наука о поведении больших групп людей) — прочно вошли в английский и другие языки. В англо-американской литературной традиции Азимова вместе с Артуром Кларком и Робертом Хайнлайном относят к «Большой тройке» писателей-фантастов[2][3]. http://ru.wikipedia.org/?oldid=40901759

Три зако́на роботе́хники в научной фантастике — обязательные правила поведения для роботов, впервые сформулированные Айзеком Азимовым в рассказе «Хоровод» (1942).

Законы гласят:

1. Робот не может причинить вред человеку или своим бездействием допустить, чтобы человеку был причинён вред.
2. Робот должен повиноваться всем приказам, которые даёт человек, кроме тех случаев, когда эти приказы противоречат Первому Закону.
3. Робот должен заботиться о своей безопасности в той мере, в которой это не противоречит Первому и Второму Законам.

И разговор идет уже не о философских аспектах, а о совершенно конкретных ограничительных мерах на пути неудержимого роста неоправданного автоматического переноса математических разработок в логику ИИ. Вопрос не в их правильности или математической обоснованности, а в их логической применимости на уровне совместимости с нашей логикой.

Математика не совпадает с нашей логикой по обоснованиям логичности и это необходимо учитывать. Понятие рациональности не всегда совпадает с нашими этическими нормами и системой обоснования решения.

Компьютер и логическая машина имеют совершенно разные основания для своего развития. Булева логика не подходит для решения задач логического управления автономных систем, которые и должны стать носителями ИИ, совместимого с нашим интеллектом и разумом.

Можно только сожалеть о том, что когда-то компьютер практически остановил развитие электронных логических машин и повернул вектор развития в сторону математизации логики. Будь всё иначе, мы бы сегодня имели и реальный ИИ, и более серьезный разговор о механистическом понимании логики.

Но история не имеет сослагательного наклонения. Что есть, то и есть. И потому, мы опять находимся практически в начале этого пути развития понимания логики.

Надо начинать всё с начала. Необходимо создавать механическую логику в полном объеме, а не только её математическую часть. На основе понимания всей сложной системы управления клетки, как первой частицы Живого в эволюционной цепи.

С основных механизмов и понятий. С системы организации памяти на основе ассоциативных и причинно-следственных связей. С логики сравнения эквивалентов и противоположностей.

Только в этом случае можно думать о реальном создании логической машины и носителя ИИ в том понимании, как мы об этом говорим.

г. Екатеринбург.
январь 2011г.

Литература:

1 Основы информатики и программирования. 1. Лекция: Алгоритмы и программы http://www.intuit.ru/department/se/oip/1/oip_1.html

2 И.А. Дехтяренко Декларативное программирование 3. Инструменты. http://www.softcraft.ru/paradigm/dp/dp03.shtml

3 Морозов М.Н. Логическое программирование. Пролог Курс лекций http://www.mari-el.ru/mmlab/home/prolog/study_l.html

4 Роберт Хаспер Введение в стандартный ML http://www.dina.kvl.dk/~sestoft/mosml/harper-a.pdf

5 Проект http://wiki-linki.ru/About

6 Курс лекций по дисциплине "Системы искусственного интеллекта" http://www.mari-el.ru/mmlab/home/AI/index.html
7 Клоксин У., Меллиш К. Программирование на языке пролог http://www.kodges.ru/15542-programmirovanie-na-jazyke-prolog.html
8 Братко И. Программирование на языке Пролог для искусственного интеллекта http://www.kodges.ru/15544-programmirovanie-na-jazyke-prolog-dlja.html
9 Малпас Дж. Реляционный язык Пролог и его применение http://progbook.net/prolog/864-reljacionnyjj-jazyk-prolog-i-ego-primenenie.html
10 Visual Prolog 7.3 Build 7303 http://www.visual-prolog.com/vip6/download/default.htm
11 Никитин А.В. Логика управления клетки http://www.trinitas.ru/rus/doc/0016/001c/1905-nik.pdf
12 Юров В.И. Assembler -СПб: Питер,2002г. – 624стр. http://www.twirpx.com/file/571831 http://tabulorasa.info/39731-v.i.-jurov-assembler.-uchebnik-dlja-vuzov.html
13 Юров В.И. Assembler. Практикум. 2-е изд. - СПб.: Питер, 2006. - 399 с.: ил. http://www83.megaupload.com/files/d6bfb90f11ec8f53fd64189877253726/assm_jurov.djvu
14 Никитин А.В. На пути к Машинному Разуму. Круг третий. (Часть 3) // «Академия Тринитаризма», М., Эл № 77-6567, публ.12907, 03.02.2006 http://trinitas.ru/rus/doc/0023/001a/00230030.htm
15 Никитин А.В. На пути к машинному разуму. Круг третий. (Часть 4) // «Академия Тринитаризма», М., Эл № 77-6567, публ.12914, 06.02.2006 http://trinitas.ru/rus/doc/0023/001a/00230031.htm#_Toc124843192
16 Никитин А.В. На пути к машинному разуму. Круг третий. (Часть 5) // «Академия Тринитаризма», М., Эл № 77-6567, публ.12928, 08.02.2006 http://trinitas.ru/rus/doc/0023/001a/00230032.htm
17 Никитин А.В. На пути к машинному разуму. Круг третий. (Часть 6) http://andrejnikitin.narod.ru/upr_evol_AI6_ok.htm
18 Никитин А.В., Эволюционный путь саморазвития искусственного интеллекта // «Академия Тринитаризма», М., Эл № 77-6567, публ.14738, 19.03.2008 http://trinitas.ru/rus/doc/0016/001c/00161450.htm
19 Никитин А.В. Работа рибосомы при трансляции белка. http://andrejnikitin.narod.ru/ribosoma.htm
20 Никитин А.В. Проблемы понимания системы кодирования ДНК. http://andrejnikitin.narod.ru/Problem_DNK.htm
21 Никитин А.В. Считывание и обработка информации ДНК. http://andrejnikitin.narod.ru/information_DNK1.htm
22 Никитин А.В. Триплеты в ДНК. http://andrejnikitin.narod.ru/tripletDNK.htm
23 Никитин А.В. Информация в ДНК, РНК и белках. http://andrejnikitin.narod.ru/tripletDNK.htm
24 Никитин А.В. От «мира РНК» к Началу Жизни... http://andrejnikitin.narod.ru/otRNKkNachalu.htm
25 Забытая "Мыслительная машина" профессора А.Н.Щукарева http://www.ukrainiancomputing.org/Shchukarev_r.html
26 Н.М.Амосов - основоположник биокибернетических информационных технологий
27 В.М.Глушков - основоположник информационных технологий в Украине и бывшем СССР
28 Брусенцов Н. П. Исчерпывающее решение «неодолимой» проблемы парадоксов. http://www.computer-museum.ru/histussr/resh_trilog.htm
29 Брусенцов Н. П., Владимирова Ю. С. Троичное конструктное кодирование булевых выражений. http://www.computer-museum.ru/histussr/trilog22.htm
30 Бажанов В.А. Н.А. Васильев как мыслитель. К 100-летию открытия воображаемой логики* http://vphil.ru/index.php?option=com_content&task=view&id=159
31 Л.Кэролл История с узелками. http://golovolomka.hobby.ru/books/carrol/knot/content.shtml
32 РЕДЬКО В.Г. ПРОБЛЕМА ПРОИСХОЖДЕНИЯ ИНТЕЛЛЕКТА И ЭВОЛЮЦИОННАЯ БИОКИБЕРНЕТИКА http://www.scorcher.ru/neuro/science/intell/mem20.php
33 Владимир Белов О ПЕРСПЕКТИВАХ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА http://www.scorcher.ru/neuro/science/neurocomp/mem152.htm http://prof-ai.narod.ru/doc/119/index.html
34 Сайт "Технологии обработки данных в прогнозировании" http://artint.com.ua/index_a.htm
35 Е.М.Иванов К проблеме "вычислимости" функции сознания http://www.scorcher.ru/art/theory/algorithm/algorithm.php

36 К.В.Судаков Системные основы интеллекта
http://www.scorcher.ru/neuro/science/intell/mem36.htm
37 Философские аспекты проблем создания искусственного интеллекта
http://www.dissercat.com/content/filosofskie-aspekty-problem-sozdaniya-iskusstvennogo-intellekta
38 Новая парадигма искусственного интеллекта (постановка задачи)
http://ailab.ru/en/investigation/bionika/novaya-paradigma-iskusstvennogo-intellekta-postanovka-zadachi/85.html
39 Дрейфус Х. Чего не могут вычислительные машины: Критика искусственного разума.
http://publ.lib.ru/ARCHIVES/D/DREYFUS_H%27yubert/_Dreyfus_H..html
40 Джон Р. Лукас Разум, машины и Гедель.
http://www.philosophy.ru/library/philmath/translate/lucas_minds_machines_gedel.pdf
41 Никитин А.В., Логика автономных систем // «Академия Тринитаризма», М., Эл № 77-6567, публ.15858, 28.03.2010 http://trinitas.ru/rus/doc/0016/001c/00161628.htm
42 Альфред Тарский О ПОНЯТИИ ЛОГИЧЕСКОГО СЛЕДОВАНИЯ
43 Гаряев П.П., Фрактальность интеллекта // «Академия Тринитаризма», М., Эл № 77-6567, публ.15705, 17.12.2009 http://trinitas.ru/rus/doc/0016/001c/00161591.htm
44 Никитин А.В. Эволюционный путь саморазвития искусственного интеллекта.
http://andrejnikitin.narod.ru/samorazvitiemozg2.htm
45 Марков А. У низших животных обнаружены системы управления генами и транспозонами при помощи малых РНК. http://elementy.ru/news/430862
46 Наймарк Е. Расшифрован геном хоанофлагеллят — ближайших одноклеточных родичей всех многоклеточных животных. http://elementy.ru/news/430678
47 Марков А. Самым примитивным животным на земле оказался трихоплакс.
http://elementy.ru/news/430248
48 Гонка вооружений — двигатель эволюции. http://www.nanonewsnet.ru/articles/2010/gonka-vooruzhenii-dvigatel-evolyutsii
49 Марков А. Пути эволюции предопределены на молекулярном уровне.
http://elementy.ru/news/430196
50 Репин В.С. Эволюция в свете системной биологии.
http://vphil.ru/index.php?option=com_content&task=view&id=231
51 Л.Ф. Мараховский, Н.Л. Михно, М.В. Москвин, Автоматы третьего рода – новый шаг к моделированию работы человеческого мозга // «Академия Тринитаризма», М., Эл № 77-6567, публ.17223, 17.01.2012 http://www.trinitas.ru/rus/doc/0016/001c/1925-mrh.pdf
52 Калашников Ю. Я. Информационное управление клеточными процессами.
http://www.nestudent.ru/show.php?id=79994&p=4
53 Эдвард де Боно. Нейронная сеть мозга http://www.debono.ru/article/neyro.htm