Продукционная модель представления знаний

Кружалов Алексей Сергеевич

Московский Политехнический Университет
26 сентября 2022 г.

Системы, основанные на знаниях

Разумное поведение, такое как правильная речь, представляется нам как некоторый процесс, регламентируемый определенными правилами, даже в том случае, если мы не можем их точно сформулировать. В искусственном интеллекте правила играют даже более явно выраженную роль в формировании того, что мы называем разумным поведением. Мы говорим, что нечто (агент) ведет себя именно таким образом, поскольку оно располагает представлением правил, уместных для формирования поведенческого акта, который в таком случае претендует на разумность.

Системы, основанные на знаниях

Набор порождающих правил — это формализм, который уже использовался в теории автоматов, формальной грамматике, разработке языков программирования, прежде чем стать на службу моделированию психофизиологической деятельности.

В литературе по экспертным системам их иногда называют правилами «условие — действие» или «ситуация — действие». Это связано с тем, что такие правила обычно используются для представления эмпирических ассоциативных связей между данными, предъявленными системе, и действиями, которые система должна предпринять в ответ.

В экспертных системах такие правила обычно задают, что нужно сделать с символической структурой, представляющей текущее состояние проблемы, чтобы перейти к представлению, более близкому к решению.

Канонические системы

Канонические системы были изобретены Эмилем Постом в 1941 году при попытке найти наиболее общий вид формальных систем, таких, например, как Principia Mathematica.

Эмиль Леон Пост
(1897-1954)

Канонические системы

Будем предполагать, что задан список *знаков* и другой список символов, называемых *переменными*, и отличных от знаков. *Терм* – это любая цепочка знаков и переменных, а *производящая схема* – это фигура вида

$$\frac{\mathrm{t}_1\ \mathrm{t}_2\ \ldots\ \mathrm{t}_\mathrm{n}}{\mathrm{t}},$$

где все $\mathrm{t}, \mathrm{t}_1, \mathrm{t}_2, \ldots, \mathrm{t}_\mathrm{n}$ ($\mathrm{n} \geqslant 0$) суть термы. Термы $\mathrm{t}_1, \mathrm{t}_2, \ldots, \mathrm{t}_\mathrm{n}$ называются *посылками*, а t – заключением схемы. Схема без посылок ($\mathrm{n} = 0$) называется аксиомой.

Канонические системы

Каноническая система, или система Поста, состоит из списка знаков, списка переменных, и конечного множества схем. Знаки образуют *алфавит* канонической системы.

Применение схемы получается из схемы путём подстановки цепочек знаков вместо всех переменных, причём вместо всех вхождений одной и той же переменной подставляется одна и та же цепочка.

Канонические системы

- Применение акисомы – доказательство этого применения.
- Если $\mathrm{P}_1, \mathrm{P}_2, \ldots, \mathrm{P}_\mathrm{n}$ – доказательства соответственно слов $\mathrm{a}_1, \mathrm{a}_2, \ldots, \mathrm{a}_\mathrm{n}$ и

$$\frac{\mathrm{a}_1\ \mathrm{a}_2\ \ldots\ \mathrm{a}_\mathrm{n}}{\mathrm{a}}$$

  - применение некоторой схемы, то

$$\frac{\mathrm{P}_1\ \mathrm{P}_2\ \ldots\ \mathrm{P}_\mathrm{n}}{\mathrm{a}}$$

  - доказательство слова a.

Слово доказуемо в канонической системе, если мы можем найти его доказательство.

Канонические системы

Определение канонической системы, возможно, станет более понятным, если привести пример. Пусть — алфавит $\mathrm{a}, \mathrm{b}, \mathrm{c}$, а аксиомы суть: $\mathrm{a}, \mathrm{b}, \mathrm{c}, \mathrm{aa}, \mathrm{bb}, \mathrm{cc}$.

Тогда следующие порождения сгенерируют все палиндромы, базирующиеся на этом алфавите, приняв за отправную точку имеющиеся аксиомы:

(P1) $\$ \to \mathrm{a}\$\mathrm{a}$,

(P2) $\$ \to \mathrm{b}\$\mathrm{b}$,

(P3) $\$ \to \mathrm{c}\$\mathrm{c}$.

Канонические системы

Более того, в данном случае можно проследить применение правил, которые должны привести к росту определенного палиндрома. Например, чтобы сгенерировать bacab, нужно применить P1 к аксиоме c, а затем P2 — к результату. Другими словами, приняв c в качестве аксиомы, можно вывести из нее теорему aca и добавить ее к имеющимся аксиомам. Затем из aca можно вывести новую теорему bacab. Обратите внимание, что эта последовательность порождений не обладает свойством коммутативности, т.е. если применять те же правила, но в ином порядке, получится совсем другой результат. Например, если к аксиоме c применить сначала правило P2, а затем P1, то получим abcba.

Канонические системы

На первый взгляд канонические системы довольно тривиальны. Все, что можно сделать в рамках такой системы, — преобразовать одну строку символов в другую. Но если задуматься, то **любое логическое или математическое исчисление в конце концов сводится к набору правил манипулирования символами**. Мы упускаем это из виду, поскольку для нас часто важен определенный смысл логических и математических символов, чего не скажешь о строках типа abcba.

Канонические системы

Отсюда следует, что **любая формальная система может рассматриваться как каноническая**.

В действительности к этому нужно добавить тривиальную оговорку, что такая система может нуждаться еще в дополнительном алфавите, буквы которого будут использоваться в качестве знаков пунктуации в сложных доказательствах. Таким образом, для проверки доказательства в любой формальной системе или для того, чтобы выполнить любую эффективную процедуру, вполне достаточно способности прочесть строку символов, разделить ее на компоненты и переупорядочить (при этом, возможно, придется добавить еще какие-то символы или удалить существующие в исходной строке).

Продукционные системы

Имея в своем распоряжении словарь символов и грамматику, регламентирующую порождение символических структур, можно представить в машинном виде исходное состояние интересующих нас проблем. Эти представления соответствуют аксиомам канонической системы — они представляют собой некоторую символическую структуру, которую нужно преобразовывать, применяя имеющиеся правила в определенном порядке.

Продукционные системы

Продукционная система состоит из множества правил, интерпретатора правил, который решает, когда надлежит применить каждое из них, и рабочей памяти, содержащей данные, описание цели и промежуточные результаты, в совокупности определяющие текущее состояние проблемы.

Именно структуры данных в рабочей памяти анализируются и преобразуются порождающими правилами. Обращение к правилам синхронизируется текущими данными, а интерпретатор правил управляет выбором и активизацией определенных правил в каждом цикле.

Продукционные системы

Предпосылки часто называются **условиями**, а действия — **заключениями**, поскольку один из видов действий — сделать заключение, если встретилось такое сочетание условий, которое делает истинным или вероятным определенное порождающее правило. Иногда используется и другая терминология, согласно которой предпосылки называются **левой частью правила**, а действия — **правой**.

Предпосылки обычно бывают представлены в форме вектора объект-атрибут — значение, как, например:

(organism-1 (morphology rod) (aerobicity aerobic)).

В данном случае предпосылка состоит в том, что определенный микроорганизм имеет форму палочки и размножается в воздушной среде.

Рабочая память

Основная функция рабочей памяти — хранить данные в формате векторов объект-атрибут- значение. Эти данные используются интерпретатором, который в случае присутствия (или отсутствия) определенного элемента данных в рабочей памяти активизирует те правила, предпосылки в которых удовлетворяются наличными данными. Как это делается, рассмотрим на примере.

Пусть в рабочей памяти содержатся векторы

(patient (name Jones) (age 40)

(organism organism-1))

(organism (name organism-1)

(morphology rod) (aerobicity .aerobic)).

Функционирование интерпретатора

Процесс применения специфицированных правил можно описать в терминах цикла распознавание-действие, который состоит из следующих шагов.

1. Сопоставить образцы в предпосылках правил и элементы данных в рабочей памяти.
2. Если окажется, что можно активизировать более одного правила, выбрать одно из них; этот шаг называется **разрешением конфликта**.
3. Применить выбранное правило. Результатом, скорее всего, будет добавление нового элемента данных в рабочую память и/или удаление какого-либо существующего элемента из рабочей памяти. Затем перейти к шагу 1.

Функционирование интерпретатора

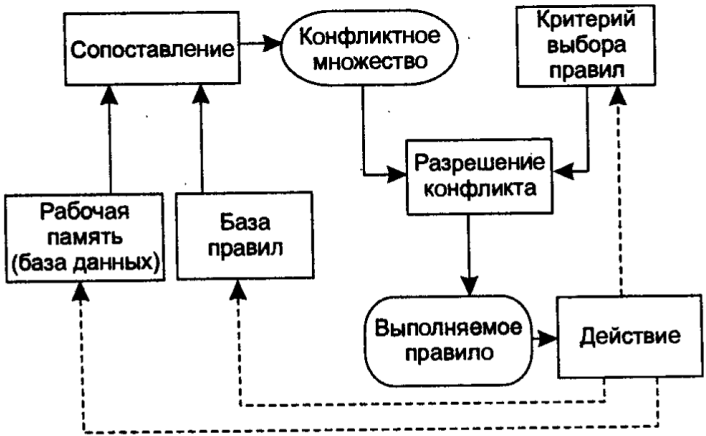
Работа машины вывода основана на применении правила *modus ponens*.

Правило modus ponens

Если известно, что истинно утверждение A и существует правило вида «Если A, то B», тогда утверждение B также истинно. Форма записи:

$$\frac{\mathrm{A}, \mathrm{A} \implies \mathrm{B}}{\mathrm{B}}.$$

Функционирование интерпретатора



Функционирование интерпретатора

Механизм разрешения конфликтов специфичен для каждой системы, т.е. для каждого интерпретатора правил. Можно, конечно, сформулировать и такой набор правил, что в любой ситуации только одно из них будет удовлетворяться (он называется **детерминированным**). Но в экспертных системах обычно используются **недетерминированные** наборы правил, поскольку в реальной жизни очень часто встречаются ситуации, которые позволяют использовать более одного правила.

Функционирование интерпретатора

Управление процессом функционирования системы, основанной на применении порождающих правил, выдвигает ряд нетривиальных проблем. Существуют две разновидности обобщенного подхода к управлению функционированием — **локальный** и **глобальный**. Глобальный подход имеет тенденцию к поиску решений, не связанных с особенностями определенной предметной области, а локальный, наоборот, на первый план выдвигает приемы, специфические для данной предметной области.

Локальный подход предполагает использование специальных правил управления правилами — **метаправил**. Такие правила обычно программируются в явном виде разработчиком конкретной системы с учетом специфики ее применения.

Разрешение конфликтов

Как мы видели, в алгоритме функционирования продукционной системы введен специальный шаг принятия решения между шагами анализа ситуации и применения правила. В результате анализа соответствия текущих данных и предпосылок различных правил в имеющемся наборе можно сформировать список правил, которые могут быть применимы в данной ситуации. Такой набор иногда называют **конфликтующим множеством** (англ. conflict set).

Разрешение конфликтов

Разнообразие. Не следует применять к одним и тем же данным правило, которое уже было к ним применено ранее. Самый простой вариант реализации этого механизма — удалять из списка заявок примененное ранее правило. Иногда используется другой вариант — из списка удаляется правило, активизированное в предыдущем цикле, — это предотвращает зацикливание, но если желательно именно повторять процедуру, то в распоряжение программиста предоставляется функция refresh, которая позволяет временно подавить механизм, действующий по умолчанию.

Разрешение конфликтов

Новизна. Элементы в рабочей памяти в таких инструментальных системах, как CLIPS, снабжены специальным атрибутом времени порождения. Это позволяет системе ранжировать элементы в списке заявок соответственно тому, как давно введены в рабочую память данные, которые использовались при сопоставлении, а затем приоритет отдается правилам, «реагирующим» на более свежие данные. Идея состоит в том, чтобы следовать за «текущей волной» и меньше внимания уделять тем данным, которые были давно сформированы. К ним можно будет вернуться в дальнейшем, если текущая цепочка рассуждения натолкнется на какое-либо препятствие.

Разрешение конфликтов

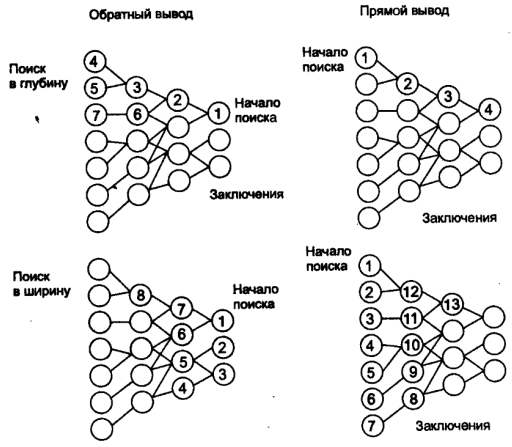
Специфика. Более специфичные правила, которые включают большее количество компонентов в предпосылках и соответственно труднее удовлетворяются, имеют приоритет перед более общими. Идея состоит в том, что использование таких правил должно принести больше «пользы», поскольку они принимают во внимание больше информации. Эту стратегию можно эффективно использовать при работе с исключениями из общих правил.

Прямая и обратная цепочки рассуждений

Прямой порядок означает, что цепь рассуждений строится, отталкиваясь от данных (условий, о которых известно, что они удовлетворяются), к гипотезам (состоянию проблемы, вытекающему из этих условий).

Обратная цепочка означает, что рассуждения строятся, отталкиваясь от заданной цели (гипотезы, представляющие целевое состояние системы) к условиям, при которых возможно достижение этой цели.

Прямая и обратная цепочки рассуждений



Правила и метаправила

Код каждого порождающего правила является самодостаточным, т.е. весь необходимый контекст активизации правила содержится только в его предпосылках. Не существует способа, который позволял бы одному правилу вызывать другое, как если бы правила были процедурами. Правило R, которое активизируется в цикле $\mathrm{C_i}$, может облегчить последующую активизацию правила R' в цикле $\mathrm{C_{i+1}}$, но единственный способ сделать это — изменить состояние рабочей памяти.

Правила и метаправила

Иногда, для того чтобы решить, какое правило следует активизировать, желательно использовать конкретные знания, а не следовать общей стратегии разрешения конфликтов. С этой целью в некоторые интерпретаторы правил включены средства, позволяющие программисту сформулировать и ввести в программу метаправила. Эти метаправила определяют правила применения правил, т.е. правила, по которым выполняется отбор тех правил из претендующих на выполнение, которые следует рассматривать в первую очередь или, более того, выполнять обязательно.

Правила и метаправила

Довольно часто метаправила отражают знания относительно конкретной предметной области. Например, если мы обратимся к системам медицинской диагностики, то в виде метаправила можно представить тот факт, что пациенты определенной группы, например склонные к употреблению алкоголя или пострадавшие от ожогов, особенно подвержены влиянию определенных видов инфекций. Тогда в метаправиле нужно указать, что для таких пациентов при анализе правил-кандидатов предпочтение следует отдавать тем, которые специфичны именно для этой инфекции.

Пример работы машины вывода

Пример. Имеется фрагмент базы знаний из двух правил:

П1: Если (отдых летом) и (человек активный) то (ехать в горы)

П2: Если (любит солнце) то (отдых летом)

Предположим, в систему поступили данные — (человек активный) и (любит солнце).

Пример работы машины вывода

П1: Если (отдых летом) и (человек активный) то (ехать в горы)

П2: Если (любит солнце) то (отдых летом)

Прямой вывод

1-й проход.

Шаг 1. Пробуем П1, не работает (не хватает данных (отдых летом)).

Шаг 2. Пробуем П2, работает, в базу поступает факт (отдых летом).

2-й проход.

Шаг 3. Пробуем П1, работает, активируется цель (ехать в горы), которая и выступает как вывод.

Пример работы машины вывода

П1: Если (отдых летом) и (человек активный) то (ехать в горы)

П2: Если (любит солнце) то (отдых летом)

Обратный вывод

1-й проход.

Шаг 1. Цель — (ехать в горы): пробуем П1 — данных (отдых летом) нет, они становятся новой целью, и ищется правило, где она в правой части.

Шаг 2. Цель (отдых летом): правило П2 подтверждает цель и активирует ее.

2-й проход.

Шаг 3. Пробуем П1, подтверждается искомая цель.

Особенности продукционной модели

Достоинства:

- Простота создания и понимания отдельных правил;
- Простота пополнения и модификации;
- Простота механизма логического вывода;
- Модульность организации знаний.

Недостатки:

- Отсутствие возможности описания взаимных отношений правил;
- Сложность анализа целостного образа знаний;
- При накоплении достаточно большого количества правил (продукций) они начинают противоречить друг другу.

Предметная область системы MYCIN

Антимикробный агент — это любой лекарственный препарат, созданный для уничтожения бактерий и воспрепятствования их роста. Некоторые агенты слишком токсичны для терапевтических целей, и не существует агента, который является эффективным средством борьбы с любыми бактериями.

Выбор терапии при бактериальном заражении состоит из четырех этапов:

- выяснить, имеет ли место определенный вид заражения у данного пациента;
- определить, какой микроорганизм (микроорганизмы) мог вызвать данный вид заражения;
- выбрать множество лекарственных препаратов, подходящих для применения в данной ситуации;
- выбрать наиболее эффективный препарат или их комбинацию.

Структура системы MYCIN

База знаний системы MYCIN

База знаний системы MYCIN организована в виде множества правил в форме если условие$_1$ и ... и условие$_t$ удовлетворяются, то прийти к заключению$_1$ и ... и к заключению$_n$.

Вот как выглядит перевод на обычный язык типичного правила MYCIN: ЕСЛИ 1) организм обладает грамотрицательной окраской, и 2) организм имеет форму палочки, и 3) организм аэробный, ТО есть основания предполагать (0,8), что этот микроорганизм относится к классу enterobacteriaceae.

Число 0.8 называется **уровнем соответствия** правила, т.е. мерой правдоподобия заключения, сделанного на основании сформулированных условий.

База знаний системы MYCIN

Фактически правило является парой «предпосылка—действие»; такое правило иногда традиционно называют «продукцией».

Предпосылка — это совокупность условий, а уверенность в достоверности предпосылки зависит от того, насколько достоверной является оценка условий.

Условия — это предположения о наличии некоторых свойств, которые принимают значения истина либо ложь с определенной степенью достоверности.

Действие — это либо заключение, либо рекомендация о том, какое действие предпринять. Примером заключения может служить вывод о том, что данный организм относится к определенному классу. Пример рекомендации — сформулированный перечень лечебных процедур.

База знаний системы MYCIN

Помимо правил, в базе знаний MYCIN также хранятся факты и определения. Для их хранения используются разные структурные формы:

- простые списки, например списки всех микроорганизмов, известных системе;
- таблицы знаний с записями об определенных клинических показаниях и значениях, которые эти показания имеют при разных условиях; примером может служить информация о форме микроорганизмов, известных системе;
- система классификации клинических параметров соответственно контексту, в котором эти параметры рассматриваются, например являются ли они свойством (атрибутом) пациентов или микроорганизмов.

База знаний системы MYCIN

Информация о пациенте хранится в структуре, названной **контекстным деревом** (англ. context tree).

База знаний системы MYCIN

Предположим, что в записи, связанной с узлом ОРГАНИЗМ-1 в этой структуре, хранятся данные

ГРАМ = (ГРАМ-ОТР 1.0)

МОРФ = (ПАЛОЧКА .8) (КОЛБ .2)

ВОЗДУХ = (АЭРОБ .6),

которые имеют следующий смысл: совершенно определенно организм имеет грамотрицательную окраску; со степенью уверенности 0.8 организм имеет форму палочки, а со степенью уверенности 0.2 — форму колбочки; со степенью уверенности 0.6 ОРГАНИЗМ-1 является аэробным (т.е. воздушная среда способствует его росту).

База знаний системы MYCIN

Теперь предположим, что применяется сформулированное выше правило. Нам требуется определить степень уверенности в выполнении всех трех перечисленных в нем условий применительно к данным, представленным в ОРГАНИЗМ-1. Степень уверенности в выполнении первого условия равна 1.0, второго — 0.8, а третьего — 0.6. Степень уверенности в выполнении совокупности условий принимается равной минимальному из значений, характеризирующих отдельные компоненты, т.е. 0.6.

База знаний системы MYCIN

В данном случае мы приходим к заключению, что микроорганизм, описанный в узле ОРГАНИЗМ-1, относится к классу энтеробактерий со степенью уверенности, равной 0.6 x 0.8 = 0.48. Сомножитель 0.6 — это степень уверенности в выполнении совокупности условий, перечисленных в правиле, а 0.8 — степень уверенности в том, что правило дает правильное заключение, когда все означенные в нем условия гарантированно удовлетворяются. За сомножителями и результатом этого выражения закрепился термин **коэффициента уверенности** (CF— certainty factor). Таким образом, в общем случае имеем: CF(действие) = CF(предпосылка) x CF(правило).

Структуры управления в MYCIN

Целевое правило самого верхнего уровня в системе MYCIN можно сформулировать примерно так:

ЕСЛИ

- существует микроорганизм, который требует проведения курса терапии, и
- заданы соображения относительно любых других микроорганизмов, которые требуют проведения курса терапии,

ТО

- сформировать список возможных курсов терапии и выделить наилучший из них.

Структуры управления в MYCIN

Консультация представляет собой, по сути, поиск на древовидном графе целей.

В корне дерева располагается цель самого верхнего уровня — та часть целевого правила, в которой отображено действие, — рекомендуемый курс лекарственной терапии.

На более низких уровнях размещаются подцели, которые представляют собой, например, выяснение, какие микроорганизмы обнаружены в зараженных тканях и насколько заражение каждым из них существенно. Многие из этих подцелей распадаются на более мелкие подцели.

Листьями дерева являются факты, которые не нуждаются в логическом выводе, поскольку получены эмпирическим путем, например факты, установленные в лаборатории.

Структуры управления в MYCIN

Для работы программы очень удобно представить процесс порождения подцелей с помощью особого вида структуры, названной **И/ИЛИ-графом**. Основная идея состоит в том, что корневой узел дерева представляет главную цель, а терминальные узлы — примитивные операции, которые может выполнить программа. Нетерминальные (промежуточные) узлы представляют подцели, по отношению к которым допустимо выполнить дальнейший анализ. Существует довольно простое соответствие между анализом таких структур и анализом множества правил.

Структуры управления в MYCIN

Рассмотрим следующий набор правил «условие-действие»:

Если X имеет СЛУЖЕБНОЕ УДОСТОВЕРЕНИЕ И X имеет ОГНЕСТРЕЛЬНОЕ_ОРУЖИЕ,

ТО X - ПОЛИСМЕН.

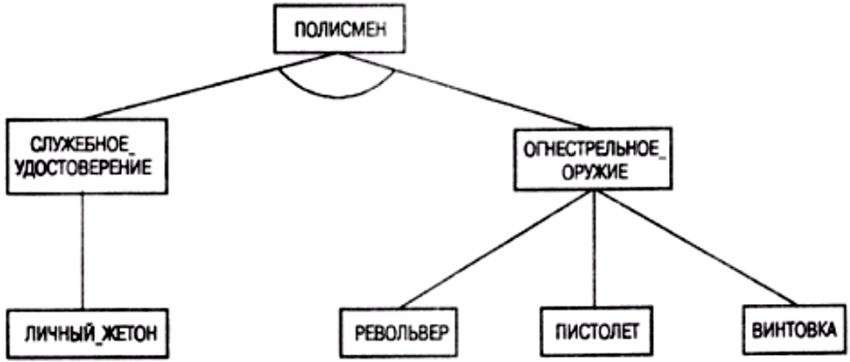
ЕСЛИ X имеет РЕВОЛЬВЕР, или X имеет ПИСТОЛЕТ, или X имеет ВИНТОВКУ,

ТО X имеет ОГНЕСТРЕЛЬНОЕ ОРУЖИЕ.

Если X имеет ЛИЧНЫЙ_ЖЕТОН,

ТО X имеет СЛУЖЕБНОЕ_УДОСТОВЕРЕНИЕ.

Структуры управления в MYCIN



Структуры управления в MYCIN

Такой вид структуры управления правилами получил наименование **цепочки обратного вывода** (англ. backward chaining), поскольку путь рассуждений идет от того, что нужно доказать, к фактам, на которых основывается доказательство. При прямой цепочке рассуждение ведется, отталкиваясь от имеющихся фактов. В этом отношении система MYCIN напоминает STRIPS, где цель также достигалась разбиением ее на подцели, к которым можно было бы применить определенные операторы. Поиск решения в процессе построения цепочки обратного вывода более целенаправлен, поскольку рассматриваются только факты, потенциально способные повлиять на решение.

Комбинация гипотез

В системе MYCIN может оказаться, что для суждения об определенном параметре подойдет не одно правило, а несколько. Применение каждого из них — отдельная гипотеза — характеризуется некоторым значением коэффициента уверенности.

Например, из одного правила следует, что данный микроорганизм — это E.Coli, причем коэффициент уверенности этой гипотезы равен 0.8. Другое правило, принимая во внимание другие свойства анализируемого объекта, приводит к заключению, что этот микроорганизм — E.Coli, но эта гипотеза характеризуется коэффициентом уверенности 0.5 (или, например, -0.8). Отрицательное значение коэффициента уверенности указывает, что данное правило опровергает сформулированное заключение.

Комбинация гипотез

Пусть x и y — коэффициенты уверенности одинаковых заключений, полученные при применении разных правил. В таком случае в системе MYCIN используется следующая формула определения результирующего коэффициента уверенности:

$$\mathrm{CF}(x,y) = \begin{cases} x + y - xy, & \text{при } x, y > 0; \\ x + y + xy, & \text{при } x, y < 0; \\ \frac{x+y}{1-\min(|x|,|y|)}, & \text{при } xy < 0. \end{cases}$$

Комбинация гипотез

ЕСЛИ микроорганизм идентифицирован как pseudomonas,

ТО рекомендуется выбрать следующие медикаменты:

1 - COLISTIN (0.98)

2 - POLYMIXIN (0.96)

3 - GENTAMICIN (0.96)

4 - CARBENICILLIN (0.65)

5 - SULFISOXAZOLE (0.64)

Оценка системы MYCIN

Существует множество способов оценки или сравнения характеристик экспертных систем, но наиболее распространенный — сравнение полученных с их помощью результатов с теми, которые получает человек-эксперт.

Еще в 1974 году, на самой ранней стадии разработки системы MYCIN, были получены весьма обнадеживающие результаты. Команда из пяти высококвалифицированных экспертов в области диагностики инфекционных заболеваний подтвердила правильность 72% рекомендаций, сделанных системой, которые относились к 15 реальным заболеваниям.

База знаний системы, включающая около 400 правил, все-таки недостаточна для реального внедрения в практику лечения больных инфекционными болезнями.

Оценка системы MYCIN

Должна соблюдаться определенная процедура проведения эксперимента. Вместо того чтобы просить эксперта оценить качество ответа, предложенного компьютером, лучше предложить ему несколько вариантов решений, одни из которых предложены специалистами в этой предметной области, а другие — экспертной системой, причем эксперт не должен знать, есть ли среди предложенных вариантов «машинные». Именно так проводилась описанная выше процедура оценки качества системы MYCIN. При этом эксперт избавлен от возможно и неосознаваемой психологической «тенденциозности» в оценке того, что предлагается компьютером.

Оценка системы MYCIN

Оценка должна протекать безболезненно для эксперта либо ее вообще нет смысла проводить. Если оценка сопряжена с какими-либо неприятными для эксперта последствиями, то рассчитывать на его объективность, конечно же, нельзя. Нельзя проводить оценку, если существуют очень жесткие требования к времени ее выполнения и используемым при этом ресурсам. Вполне может оказаться так, что процесс оценки качества системы займет больше времени, чем ее разработка.

Сравнение MYCIN и STRIPS

Более широкие возможности системы MYCIN по сравнению со STRIPS в решении проблем проистекают от двух факторов: большой набор правил, которые используются для формирования гипотез и способов подтверждения их истинности, и большая база данных, в которой хранится информация о микроорганизмах, медикаментах и лабораторных тестах. В то же время механизм управления применением правил в MYCIN несколько проще, чем в STRIPS. Основное различие между двумя программами состоит не в отличиях между областями применения, а в способности использовать декларативные знания в своей области.

Рекомендуемые материалы

1. Питер Джексон. Введение в экспертные системы // М.: «Вильямс», 2001. — 624 С.
2. Филиппович Ю. Н., Филиппович А. Ю. Системы искусственного интеллекта, 2009.
3. Гаврилова, Т. А. Базы знаний интеллектуальных систем . Учебник для вузов / Т.А. Гаврилова, В. Ф. Хорошевский.// — СПб.: Питер, 2000. – 384 с.
4. Мартин-Лёф П. Очерки по конструктивной математике. М.: Мир, 1975. - 136 с.