ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ

УДК 519.872.7

ОПТИМИЗАЦИЯ В ЭКСПЕРТНЫХ СИСТЕМАХ ДЛЯ ПОСЛЕДОВАТЕЛЬНОГО ВЫБОРА ОБЪЕКТОВ ПО ОПРОСАМ ПОЛЬЗОВАТЕЛЕЙ В СОСТАВЕ ЕСТЕСТВЕННО-ЯЗЫКОВЫХ ИНФОРМАЦИОННЫХ АГЕНТОВ

Соколов А.А.
Учреждение Российской академии наук
Вычислительный центр им. А.А. Дородницына РАН, г. Москва

Поступила в редакцию 14.06.2009, после переработки 28.06.2009.

В работе рассматривается построение экспертных систем с последовательным опросом пользователей, которые ищут объекты в базе знаний этой ЭС. Ставится задача минимизации среднего числа шагов процедуры поиска. Показывается, что для минимизации длительности процедуры в базу знаний должны быть внесены данные не только о предметной области, но и о целевых группах пользователей. Для использования ЭС в естественно-языковых информационных системах (Инфах) вводится описание базы знаний с помощью нечетких множеств и лингвистических переменных. Предлагаются оценки и квазиоптимальные алгоритмы решения поставленной задачи. Приводятся примеры внедрения.

The construction of expert systems with serial poll of users which search for objects in the knowledge base of this ES is considered. The problem of minimization of an average of steps of procedure of search is posed. It is shown that for minimization of duration of procedure the data should be introduced to the knowledge base not only about a subject domain, but also about users target groups. For use ES in natural language information systems (Infs) the description of the knowledge base by means of indistinct sets and linguistic variables is imposed. Estimations and quasioptimal algorithms of the decision of a task in view are offered. Introduction examples are given.

Ключевые слова: экспертная система, минимизация, естественно-языковая информационная система, нечеткие множества.
Keywords: expert system, minimization, natural language information system, fuzzy set.

1. Введение

Имеется база знаний о свойствах некоторого достаточно большого множества объектов, из которой производится множественный выбор объектов пользователем. Выбор объекта прямым просмотром базы затруднителен из-за объема базы. Фильтрация фиксацией свойств также затруднена из-за большого числа этих свойств.

Для поиска объекта с желаемыми свойствами строится экспериментальная система, которая некоторым оптимальным способом последовательно задает пользователю вопросы о свойствах выбираемого объекта и показывает возможные варианты ответов для уже отобранного к этому шагу процедуры множества объектов.

На каждом шаге процедуры пользователю предоставляется информация о числе отобранных к этому шагу объектов и сам список объектов, отсортированный тем или иным способом. Пользователь может выбрать, отвечать ли на следующий вопрос ЭС, вмешаться в процедуру опроса и выбрать следующий вопрос по своему усмотрению или закончить процедуру.

Множество потенциальных пользователей ЭС можно разбить на целевые группы, возможно пересекающиеся. Каждая целевая группа заинтересована в поиске определенного набора объектов.

Необходимо так построить ЭС, чтобы она по ответам пользователя фактически выявляла, к какой целевой группе принадлежит пользователь и предъявляла ему тот набор объектов, в котором заинтересованы пользователи этой целевой группы и делала это за минимальное число шагов.

ЭС должна строиться как естественно-языковая система, имитирующая опрос пользователей человеком-экспертом. Будем называть такие ЭС, понимающие естественный язык, Информационными агентами сокращенно Инфами.

2. Введение

2.1 Объекты и их свойства

Пусть

OBJ — множество объектов (например, список электронных товаров).

ATR — множество свойств объектов OBJ (например, назначение, цвет, вес и т.п.)

$\forall$ atr $\in$ ATR, отображает OBJ в некоторое множество значений этих свойств VAL (atr).

Например, VAL (цвет) = {Черный, Белый, Серебристый, ...}

DOM(atr) $\subseteq$ OBJ область определения для $\forall$ atr $\in$ ATR. Объекты вне области определения atr считаются не обладающими свойством atr.

Пример.

OBJ = ФОТО $\cup$ МОБТЕЛ $\cup$ ПЛЕЕР $\cup$ КПК $\cup$ СМАРТФ, где

ФОТО — список моделей цифровых фотоаппаратов,

МОБТЕЛ — список моделей телефонов,

ПЛЕЕР — список моделей плееров,

КПК — список моделей наладонников,

СМАРТФ — список моделей смартфонов.

ATR = {«Назначение», «Опции», «Производитель», «Цвет», «Объём памяти», «Чувствительность матрицы», «Радио»}

= {atr_1, atr_2, atr_3, atr_4, atr_5, atr_6, atr_7}, где atr_i – обозначение перечисленных свойств в том же порядке.

VAL (atr_1) = {Фото, Мобтел, Плеер, Кпк, Смартф} — основное назначение

VAL (atr_2) = {Фото, Плеер, Радио} — опции

VAL (atr_3) = {Canon, Nikon, , , , , , ,} производители
VAL (atr_4) = {Черный, Белый, Серебристый, , , , , ,} — цвета
VAL (atr_5) = $[8, 4096]$ Мб — диапазон объёмов памяти
VAL (atr_6) = $[50, 3200]$ ISO — диапазон чувствительностей фотоматрицы
VAL (atr_7) = {РадиоЕсть, РадиоНет} – наличие радио.

$\mathrm{DOM}(\mathrm{atr}_1) = \mathrm{OBJ}$ — то есть свойство atr_1 = «Назначение» определено на всём множестве объектов.

2.2 Упорядоченность объектов и порядок выдачи

Пусть на множестве объектов OBJ задана функция P их приоритетов: $P(\mathrm{obj}) \in [0, 1]$, $\mathrm{obj} \in \mathrm{OBJ}$. В соответствии с приоритетом множество OBJ может быть упорядочено и в таком порядке выдано пользователю. На OBJ может быть задано несколько разных приоритетов P. Некоторые из приоритетов индуцируются свойствами, имеющими упорядоченный набор значений.

Примеры.

P_1 — относительная стоимость объектов по возрастанию (может быть и по убыванию).

P_2 — относительная популярность объекта у покупателей.

P_3 — относительная экспертная оценка объектов.

P_4 — алфавитный порядок по названиям объектов (здесь все приоритеты близки к 1)

P_5 — алфавитный порядок по названиям фирм (здесь все приоритеты также близки к 1)

P_6 — относительная новизна объектов и т.п.

2.3 Упорядоченность как нечёткое множество

Можно считать, что P — нечёткое множество на OBJ (нечёткое в смысле Заде). То есть каждому элементу приписана его степень принадлежности μ_P в диапазоне $[0, 1]$. О нечётких множествах см., например [1]. Порядок выдачи может быть многоприоритетный: вначале упорядоченность по одному приоритету, а среди объектов с одинаковым приоритетом упорядоченность по второму приоритету и т.п.

Пример.

Вначале сортируются объекты в алфавитном порядке названий фирм (P_5), а внутри объектов одной фирмы по стоимости (P_1).

2.4 Уровневые множества и объём выдачи

Уровневым множеством уровня отсечения α для нечёткого множества называется множество, для которого функция принадлежности $\mu > \alpha$. Это понятие можно использовать при выдаче пользователю списка всех объектов: список можно ограничить некоторым заданным уровнем отсечения α. Уровень отсечения α может задаваться опосредовано через максимальное число одновременно показываемых объектов: оставляется лишь заданное число объектов с максимальным μ.

Может быть отсечение одновременно по 2 параметрам: не более определённого числа объектов и не менее определённого значения функции принадлежности.

2.5 Ограничение значений свойств

Пусть $R_{\mathrm{VAL}} \subset \mathrm{VAL}(\mathrm{atr})$ подмножество значений свойств для некоторого свойства $\mathrm{atr} \in \mathrm{ATR}$.

Множество R_{VAL} может быть нечётким в смысле Заде. Тогда степень принадлежности элементов VAL к R_{VAL} будем обозначать μ_R. Далее будем говорить только о нечётких множествах, помня, что чёткое множество частный случай нечёткого (степень принадлежности либо 1, либо 0).

Будем говорить, что R_{VAL} — ограничение значений свойств.

Примеры.

1) $R_{\mathrm{VAL}} = \{256, 512, 1024\} \subset \mathrm{VAL}(\mathrm{atr}_5)$ — ограничение на объёмы памяти прибора.

2) $R_{\mathrm{VAL}} =$ «большой объём памяти» $= \{0.5/512; 1/1024\} \subset \mathrm{VAL}(\mathrm{atr}_5)$ — ограничение на объёмы памяти в диапазоне больших значений. Здесь обозначение μ/x — степень принадлежности μ элемента x в обозначениях Заде.

2.6 Лингвистические переменные

Во втором примере использована «лингвистическая переменная», являющаяся именем нечёткого множества. Лингвистические переменные это нечёткие множества второго порядка, когда функция принадлежности элемента к нечёткому множеству сама является нечёткой. Лингвистические переменные очень удобны при общении пользователя с ЭС через интерфейс естественно-языковых систем (Инфов — информационных агентов) в процессе выбора подходящих объектов. Лингвистические переменные и соответствующие им нечёткие подмножества R_{VAL} нужно задать заранее. Каждый раз, когда пользователь будет описывать желаемое свойство объекта, нужно попытаться распознать эту «лингвистическую переменную». Инф достаточно хорошо подходит для этой цели. В диалоге с пользователем он пытается выяснить, какая «лингвистическая переменная», наиболее соответствует желанию пользователя или прямо спросить, хочет ли пользователь, чтобы объекты обладали некоторым диапазоном свойств.

2.7 Ограничение объектов и порядок, индуцируемый ограничением

Ограничение $R_{\mathrm{VAL}} \subset \mathrm{VAL}(\mathrm{atr})$ индуцирует нечёткое подмножество R_{OBJ} на множестве всех объектов.

А именно: для всех объектов, не обладающих свойством atr функция принадлежности $\mu_R(\mathrm{obj}) = 0$, а для остальных $\mu_R(\mathrm{obj}) = \mu_R(\mathrm{atr}(\mathrm{obj}))$.

Это ограничение R_{OBJ} может быть использовано для выбора объектов. Действительно, ограничение R_{OBJ} индуцирует новую упорядоченность, понимаемую как пересечение нечётких множеств $F = P \cap R_{\mathrm{OBJ}}$, где P — нечёткое множество, индуцированное приоритетом. Функция принадлежности множества F задаётся как $\min(\mu_P, \mu_R)$.

Пример.

Первоначальная сортировка объектов типа «телефон» производилась в алфавитном порядке фирм производителей, а внутри объектов одной фирмы по стоимости. На множестве свойств «цвет» пользователь выбрал «лингвистическую переменную» = «светлые». $F = P_{\text{стоимость}} \cap R_{\text{OBJ светлые}}$. Список телефонов, доступный пользователю сократится, так как из него будут удалены телефоны тёмного цвета, для которых функция принадлежности $\mu = 0$. Если задан ещё уровень отсечения α, то все телефоны, у которых $\mu < \alpha$ также не будут показаны.

2.8 Фильтрация объектов

В процессе выбора объектов будут последовательно накладываться всё новые и новые ограничения $R_{\text{OBJ}} \subset \text{OBJ}$. Пусть после применения ограничения R_1 применили ограничение R_2. Совместное действие 2-х ограничений — нечёткое множество $F_2 = R_1 \cap R_2$ с функцией принадлежности $\min(\mu_{R1}, \mu_{R2})$. Если продолжать процесс применения ограничений, то получим последовательность множеств F_i, являющуюся фильтром на OBJ (вложенные стягивающиеся). Точно так же как R_{OBJ} индуцирует новую упорядоченность на OBJ, её индуцирует и F_i.

2.9 Процесс выбора объектов

Процесс выбора есть не что иное, как построение фильтра — последовательности стягивающихся множеств до психологически приемлемого числа объектов, отфильтрованных этим фильтром.

В результате каждого шага мы имеем некоторое нечёткое множество на F_i на OBJ.

2.10 Активный и пассивный выбор

В процессе выбора пользователь может проявлять активность и сам задавать вопросы, а может оставаться пассивным и отвечать на вопросы ЭС. В процессе выбора с активным пользователем он сам выбирает ограничения, называя их. ЭС может лишь переспрашивать активного пользователя, пытаясь уточнить, что тот имеет в виду. В процессе выбора с пассивным пользователем ЭС сама задаёт вопросы о свойствах объектов.

Оба процесса, активный и пассивный могут сочетаться, так как пользователь может быть в курсе только относительно одной части свойств и не знать о другой. В процессе выбора периоды активности и пассивности пользователя могут чередоваться.

2.11 Вопросы и переспросы ЭС

Вопросы, задаваемые ЭС при активном пользователе, назовём переспросами. Этих переспросов может и не быть, если сразу будет ясно, что имел в виду пользователь. Вопросы, задаваемые ЭС при пассивном пользователе, будем называть

просто вопросами. Начинать всегда нужно с вопросов, так как ЭС вступает в диалог первой. Если пользователь перехватывает инициативу, нужно переходить к переспросам.

2.12 Оптимальность процедуры выбора

Оптимизируемой величиной многошаговой процедуры выбора является среднее число шагов E, за которое удаётся отфильтровать список до приемлемого числа альтернатив. Для активного и пассивного пользователя оптимизация строится по-разному.

2.13 Постановка задачи для активного пользователя

Активный пользователь сам последовательно сужает множество OBJ, и задача ЭС распознать его пожелания и выполнить. Так что задача оптимизации сводится к уменьшению числа переспросов.

2.14 Постановка задачи для процесса выбора с чередованием активных и пассивных периодов

Пусть период активности пользователя закончился, и наступила пассивная фаза. Тогда к этому моменту на OBJ уже задан некоторый фильтр (нечёткое множество).

Задача сводится к задаче с пассивным пользователем при заданном OBJ.

2.15 Постановка задачи для пассивного пользователя

Пусть в распоряжении Инфа n вопросов: $(1, 2, \ldots, n)$. Каждый вопрос порождает некоторое покрытие множества объектов $A_1^i, A_2^i, \cdots \subset$ OBJ возможными ответами пользователя.

Рассмотрим набор m разных целевых групп потребителей $\{U_1, U_2, \ldots U_m\}$ и предположим, что пользователь, с которым общается Инф, принадлежит к одной из этих групп, выбирая ответы в соответствии с предпочтениями своей группы. Пусть p_u вероятность диалога с пользователем, принадлежащим к целевой группе U.

Пересекая все возможные ответы пользователя U на все вопросы, получаем результирующее целевое множество: $R_u = F_i \cap A^i_{j(i,u)} \in$ OBJ, где $i \in (1, 2, \ldots n)$, A^i_j ответ пользователя U на вопрос i.

По условию все R_u разные.

Множество ответов A^i_j пользователя U можно назвать его предпочтениями.

Будем рассматривать только «конструктивных» пользователей, то есть таких, что целевое множество R_u не пусто. Предположим также, что пользователь отвечает на вопросы одинаково, независимо от того, в каком порядке ему приходится на них отвечать.

Пусть Инф задаёт вопросы последовательно по шагам в зависимости от того, какие ответы он получил на предыдущих. Пусть на i-м шаге задаётся j-й вопрос, причём вопросы не повторяются.

Тогда для каждого U, после некоторого вопроса k_u получим R_u, и остальные вопросы можно не задавать. Тогда среднее число вопросов рассчитывается по формуле:

$$E = \Sigma_u k_u * p_u \tag{1}$$

3. Оптимизация

3.1 Целевые функции

Введём характеристическую функцию $C_{u,i}$, равную 1 или 0 в зависимости от того, достигнуто или нет на i-м шаге целевое множество R_u:

$$C_{u,i} = \begin{cases} 1, & \text{целевое множество } R_u \text{ на } i\text{-м шаге не достигнуто,} \\ 0, & \text{целевое множество } R_u \text{ на } i\text{-м шаге достигнуто.} \end{cases}$$

Функция $C_{u,i}$ зависит, на самом деле не от номера шага, а от уже заданных к шагу i вопросов (и, соответственно, ответов на них пользователя U). Но, чтобы не загромождать обозначения, будем писать $C_{u,i}$. После достижения целевого множества R_u вопросы больше не задаются, но можно мысленно продолжить этот процесс, считая, что $C_{u,i} = 0$ для всех $i > k_u$. Тогда очевидно: $k_u = \Sigma_{1-n} C_{u,i}$, где суммирование ведётся по всем шагам.

Подставляя выражение для k_u в (1) имеем:

$$E = \Sigma_u \Sigma_{1-n} C_{u,i} * P_u.$$

Поменяем порядок суммирования:

$$E = \Sigma_{1-n} \Sigma_u C_{u,i} * P_u. \tag{3}$$

С каждым шагом целевая функция возрастает на величину $\Sigma_u C_{u,i} * P_u$. Таким образом, она аддитивна, что [2] позволяет оптимизировать её с помощью метода динамического программирования. Требуется описать класс правил выбора вопроса, задаваемого на каждом шаге и оптимизировать целевую функцию в этом классе. Правило выбора не должно зависеть от U, так как мы не знаем, какого именно U обслуживаем. Единственная доступная нам информация — перечень уже заданных вопросов и перечень ответов. Из этой информации мы можем посчитать апостериорную вероятность обслуживания пользователя из каждой целевой группы потребителей.

3.2 Фиксированный порядок вопросов

Пусть вопрос, задаваемый на очередном шаге, не зависит от полученных ответов на предыдущие вопросы. Привлекательность такой постановки в уменьшении пространства состояний. Состояние многошаговой процедуры определяется только списком уже заданных вопросов. Потом можно будет перейти и к более общей постановке.

Задача решается с конца методом динамического программирования.

а) Пусть заданы $n-1$ вопрос и осталось задать последний на последнем n-м шаге.

Для каждого из n состояний процедуры на $n-1$ шаге рассчитывается добавок d_j, $j=(1,2,\ldots,n)$ в целевую функцию, который внесёт вопрос на последнем шаге, если номер вопроса будет j. А именно, нужно посчитать сумму по всем пользователям $d_j = \Sigma_u C_{u,n-1} * P_u$. Делается это так. Нужно пересечь для каждого U все его ответы A_k^i на заданные $n-1$ вопрос (то есть все вопросы без j-го) и сравнить с R_u . В соответствии с определением (2) $C_{u,n-1}=0$ или 1.

б) Пусть задано $n-2$ вопроса и осталось задать ещё 2. Тогда возможных состояний $C_n^2 = n*(n-1)/2$. Можно задать первый из оставшихся вопросов, а можно второй. Посчитаем добавок от задания на $n-1$ шаге первого вопроса и сложим с уже посчитанным добавком от задания на следующем n шаге второго вопроса. И наоборот, вначале второй, потом первый. Минимум из сумм и будет решением. Минимальная сумма будет ценой продолжения. Запоминаем цену продолжения и оптимальный вопрос.

в) Пусть задано i вопросов. Для каждого набора из i вопросов (всего их будет C_n^i) сравниваем $n-i$ сумм. Первое слагаемое — добавок от задания очередного вопроса, второе слагаемое — уже рассчитанная цена продолжения при условии, что заданы эти $i+1$ вопросов. Выбираем минимум.

3.3 Фиксированный порядок приоритетов (то есть с пропусками лишних вопросов)

Очевидный недостаток предыдущего алгоритма в том, что для некоторых пользователей приходится задавать лишние вопросы, хотя ответ на них не снижает неопределённости. Например, если пользователь ответил на вопрос «Вы хотите красную модель?» положительно, то не нужно задавать ему вопрос «Вы хотите черную модель?».

Можно модифицировать предыдущий алгоритм и считать, что избыточные вопросы не задаются, а пропускаются. Чтобы понять пропускать или нет очередной вопрос, нужно посчитать изменится или нет отфильтрованное к i-му шагу множество в результате задания вопроса, рассчитанного по оптимальной процедуре в предыдущем пункте.

3.4 Не фиксированный порядок приоритетов

Попробуем найти оптимальную процедуру, когда на порядок вопросов не накладывается никаких ограничений. Состояние процедуры на i-м шаге зависит, во-первых, от множества заданных к i-му шагу вопросов $\sigma(i)$. Число таких множеств — C_n^i, где C_n^i — биномиальный коэффициент, равный $n!/(n-i)!\,i!$.

Порядок заданных вопросов несущественен. Во-вторых, число состояний зависит от числа вариантов результирующих множеств. Число вариантов не превосходит m — число типов пользователей, но может быть и меньше, если результирующие множества совпадают для некоторых пользователей. Под результирующими множествами здесь понимается пересечение всех ответов пользователя к i-му шагу. Более формально:

Результирующими множество определяется как $F_{\sigma(i)}(u) = \cap A^k_{j(k,u)} \in$ OBJ, где $k \in (1, 2, \ldots, i)$, $A^k_{j(k,u)}$ ответ пользователя U на вопрос, заданный на k-м шаге. Если $F_i(u) = F_i(v)$ для пользователей U и V, то они отвечают одному состоянию. Выберем одного, произвольного представителя для каждой группы тождественных результирующих множеств, для определённости с минимальным номером в списке. Пусть таких представителей $L_{\sigma(i)}$.

Будем говорить, что состояние процесса характеризуется множеством заданных к i-му шагу вопросов $\sigma(i)$ и типом пользователя $U_{\sigma(i)}$. Число состояний равно сумме $L_{\sigma(i)}$ по всем возможным $\sigma(i)$. Это число не превосходит $m * C^i_n$.

Задача также решается с конца методом динамического программирования [2].

а) Пусть заданы $n-1$ вопрос и осталось задать последний на последнем n-м шаге.

Состояний будет не более $n * m$.

Для каждого варианта заданных $n-1$ вопросов и типа пользователей рассчитывается добавок $d_{j,k}$, $j = (1, 2, \ldots, n)$, $k = (1, 2, \ldots, L_{\sigma(n-1)})$ в целевую функцию, который внесёт вопрос на последнем шаге, если последний вопрос будет j, а результирующее множество определяется пользователем U_k. А именно, нужно посчитать сумму по всем пользователям $d_{j,k} = \Sigma_u C_{u,n} * P_u$. Делается это так. Нужно пересечь для каждого U все его ответы A^i_j на заданные $n-1$ вопрос и сравнить с R_u. В соответствии с определением (2) $C_{u,n} = 0$ или 1. На последнем шаге добавки рассчитываются фактически так же, как и в задаче с фиксированным порядком. Только их будет большее количество.

б) Пусть задано $n-2$ вопроса и осталось задать ещё 2. Можно задать первый из оставшихся вопросов, а можно второй. Посчитаем добавок от задания на $n-1$ шаге первого вопроса и сложим с уже посчитанным добавком от задания на следующем n шаге второго вопроса. И наоборот. Минимум из сумм и будет решением. Минимальная сумма будет ценой продолжения. Запоминаем цену продолжения и оптимальный вопрос. И так считаем для всех возможных вариантов из $n-2$ вопросов и представителей пользователей.

в) Пусть задано i вопросов. Для каждого набора из i вопросов (всего их будет C^i_n) и $L_{\sigma(i)}$ представителей пользователей сравниваем $n-i$ сумм. Первое слагаемое — добавок от задания очередного вопроса, второе слагаемое — уже рассчитанная цена продолжения при условии, что заданы эти $i+1$ вопросов. Выбираем минимум.

3.5 Псевдооптимальное решение

В результате применения динамического программирования перебор вариантов существенно сокращается. Полный перебор требует сравнения $n!$ вариантов. Динамическое программирование для полной оптимизации не более $n * m * \frac{n!}{\frac{n}{2}! * \frac{n}{2}!}$ вариантов, а для задачи с фиксированным порядком не более $n * n!/((n/2)! * (n/2)!)$ вариантов. То есть экономия более чем в $(n/2)! * (n/2)!/(n * m)$ раз.

Например, для 10 вопросов и 10 пользователей экономия почти в 150 и 1500 раз соответственно, а для 20 вопросов и 20 пользователей 1,6 миллиарда и 32 миллиарда раз соответственно.

Однако видно, что размерность оптимизационной задачи в случае нефиксированного порядка может быть очень высока при большом возможном числе вопро-

сов и потенциальных пользователей. Например, при 10 вопросах и 10 пользователях около 2500 состояний на 5 шаге. При 20 вопросах и 20 типах пользователя — 3,5 миллиона. Это, конечно, верхняя оценка в предположении, что все пользователи ответят по-разному на половину вопросов.

Такая же задача с фиксированным порядком — 250 и 180 000 состояний.

Способы снижения размерности могут быть такие:

а) разумным компромиссом представляется задача с фиксированными приоритетами вопросов.

б) полезно задать частичный порядок на множестве всех вопросов. Например, нельзя спрашивать про цвет, пока не спросили про тип товара или про цену.

в) так как число состояний резко растёт с ростом числа возможных вопросов, то полезно выделить несколько естественных вопросов в качестве первых (тип товара, диапазон цен), то есть таких вопросов, по которым пользователи чётко сегментируются на несколько групп.

г) последовательно решить несколько задач с фиксированным порядком. Вначале для всего множества объектов и n шагов, затем для множеств, полученных на первом шаге и $n-1$ последующих и т.п.

д) в качестве первого вопроса выбрать вопрос, даваемый задачей с фиксированным порядком, а затем применять локально-оптимальный алгоритм, например, так называемый «жадный», согласно которому нужно на очередном шаге по ответам оценить возможные типы пользователей и выбрать такой ответ из оставшихся, который мог бы разделить их пополам.

4. Пример сценария диалога выбора

4.1 Различные варианты

Сценарий диалога выбора может быть такой.

Здравствуйте. Я консультант такой-то. Могу помочь Вам выбрать подходящую модель телефона. Хотите? — Да.

Какой бы телефон вы предпочли? Вы можете самостоятельно задавать критерии Вашего предпочтения, такие как цена, вес производитель и так далее. А можете отвечать на мои вопросы. Как лучше? 1 вариант — сам. 2 — отвечать на вопросы.

1-й вариант (активный пользователь):

Итак, какие варианты Вы предпочитаете? — Хочу тонкий телефон фирмы Моторола.

У нас имеется 15 таких моделей (и показать) в ценовом диапазоне от ... до ... — Хочу подешевле.

Таких моделей 5 (Нужно оставить телефоны, попавшие в наименьший ценовой диапазон и показать, упорядочив по возрастанию цены.) — Хочу с Bluetooth.

Вот они (показать 2 подходящих. Если их мало, то рассказать, чем они хороши. Рассказ есть на сайте Связного). — Нет, не хочу такие.

Хотите посмотреть похожие? — Да.

Вот есть такие. (Показать похожие, упорядочив по возрастанию цены, так как пользователь уже хотел подешевле.)

2-й вариант

Здесь начинается применение оптимального многошагового алгоритма для целевых групп покупателей. Первый вопрос должен быть достаточно простым, таким, чтобы на него мог ответить каждый. Поэтому первый вопрос лежит вне оптимального алгоритма. Это может быть вопрос о цене (дорогой – дешевый) или о классе телефона (Классический – имиджевый – бизнес класс).

Вы хотите телефон подешевле, подороже или в среднем ценовом диапазоне? — Подешевле.

Вы хотите телефон с обычными функциями или вам нужны «навороты»?

Целевые группы потребителей должны быть определены заказчиком ЭС. Информация о предпочтениях целевых групп должна быть внесена в базу данных. Возможно, это будет несколько дополнительных колонок с названиями целевых групп, в которых будут проставлены признаки 0–1. Или более сложные случаи, когда могут быть промежуточные случаи, например, 0,5.

Каждая целевая группа U это характеристическая функция $[0, 1]$ на множестве всех объектов, заданная в виде колонки в таблице.

4.2 Веса целевых групп

Для определения весов p нужны оценки массовости целевых групп потребителей.

4.3 Целевые множества

Целевое множество R_u необязательно совпадает с множеством объектов, предпочитаемых потребителями из целевой группы U. Если в нашем распоряжении недостаточно точные и полные вопросы, то такого совпадения не будет.

Для определения целевых множеств R_u нужно проделать следующую процедуру. Нужно ответить на все вопросы за каждого представителя целевой группы потребителей. Пересечение всех множеств ответов и будет целевым множеством для потребителей из целевой группы U.

4.4 Характеристическая функция для подсчёта числа шагов

Для определения оптимального порядка вопросов вводится характеристическая функция $\mathrm{C}_{u,i}$, равная 1 или 0 в зависимости от того, достигнуто или нет на i-м шаге целевое множество R_u. См. определение (2).

4.5 Решение

Для начала нужно решить задачу с фиксированным порядком вопросов, а затем в процессе опроса принимать решение о пропуске некоторых вопросов в зависимости от достигнутого множества (фильтра) на данном шаге многошаговой процедуры. Теория приведена выше.

5. Заключение

Для оптимизации ЭС, необходимо, чтобы она по ответам пользователя фактически выявляла, к какой целевой группе он принадлежит, и предъявляла ему тот набор объектов, в котором заинтересованы пользователи этой целевой группы. Так как целевых групп существенно меньше, чем объектов выбора, то ЭС, построенная на этой основе делает это за существенно меньшее число шагов, чем ЭС, не обладающая знаниями о целевых группах пользователей. В то же время, пользователь, не принадлежащий к целевой группе, сделает выбор за примерно то же число шагов, что и в ЭС без информации о целевых группах.

Для расчета оптимальной процедуры может быть использован метод динамического программирования при условии фиксации приоритетов вопросов. Как правило, на практике в силу сложившихся традиций это условие не является обременительным, так как обычно ясно, какие свойства объектов являются важными, а какие второстепенными.

Для использования ЭС в составе естественно-языковых информационных систем (Инфов) предлагается использовать нечёткие множества в смысле Заде и лингвистические переменные — нечёткие множества второго порядка, когда функция принадлежности элемента к нечёткому множеству сама является нечёткой.

ЭС внедрена в проектах:

по подбору недвижимости в банке Дельта-Кредит `http://www.deltacredit.ru/realty/`,

в интернет-магазине Сотовик `http://www.sotovik.ru/es/` для подбора телефонов,

в магазинах Эльдорадо `http://www.eldorado.ru/expert_system/gifts.php` для выбора подарков,

в компании, предлагающей получить образование заграницей `http://www.studyabroad.ru/experts/`.

Идет внедрение рассматриваемой ЭС в Холдинге «Финам» для определения инвестиционных пакетов.

Список литературы

[1] Л. Заде. Понятие лингвистической переменной и его применение к принятию приближенных решений. М.: Мир, 1976.

[2] Р. Беллмен. Динамическое программирование. М.: ИЛ, 1960.