Сравнительная оценка алгоритмов поддержки принятия решений на основе качественных характеристик

Олег Викторович Рогозин, *доцент кафедры «Программное обеспечение ЭВМ и информационные технологии» Московского государственного университета им. Н.Э. Баумана, кандидат технических наук*

Разнообразие решаемых в образовательном процессе задач и инновационная составляющая современных экономических процессов диктуют новые требования ко всем компонентам качественного обучения. В связи с этим возникает необходимость подбора ПО не только с учётом формальных показателей его качества, но и в соответствии с предпочтениями конкретного пользователя. По характеру принимаемого решения задача относится к задаче распределения альтернатив по классам решений: из множества альтернатив (программных продуктов) выделяется группа предпочтительных для пользователя.

Представим задачу выбора эффективного ПО в образовании как задачу поддержки принятия решения (ППР Decision Making DM) в инвестиционном процессе с инновационной составляющей. Определим инвестиционный процесс как долгосрочное вложение экономических ресурсов с целью создания и получения выгоды в будущем. Основной аспект реальных инвестиций состоит в преобразовании инвестиционных средств в производительные активы и в создании новой ликвидности при использовании этих активов[1]. Особенность рассматриваемой проблемы в том, что инвестиции производятся в инновационные объекты, что делает задачу трудноформализуемой. Под инновационными объектами будем понимать единицы ПО в области образования. Выбор производится на основе как количественных (объём занимаемой оперативной памяти, время отклика и т.д.), так и качественных (надёжность, эффективность) характеристик ПО, причём преобладают качественные. Таким образом, задача является слабо структурированной.

Под эффективностью принятого решения будем понимать отношение результата к затратам, необходимым для достижения этого результата. Для оценки эффективности выбора альтернативных решений на основе качественных оценок необходимо основываться на определённом подходе:

- во-первых, использовать стандарты, чтобы обеспечить объективность сравнения программных средств;
- во-вторых, опираться на комплексный системный подход, чтобы гарантировать полноту и объективность результатов;
- в-третьих, обеспечить учёт пользовательских предпочтений.

Качественные показатели ПО как инвестиционного обьекта

В комплексную оценку ПО, в соответствии со стандартом ISO 9126–1 как инвестиционного объекта, будем включать следующие качественные характеристики[2]:

[1] **Bellman R.E., Zadeh L.A.** Decision making in a fuzzy environment. Management Science, 17, 141–164, 1970.

[2] Там же.

Функциональность. В первую очередь продукт должен обеспечивать реализацию необходимых функций, корректность их выполнения. Набор функций будет различным в таких группах ПО, как программы моделирования, оптимизации, машинной графики, управления объектами и т.п., поэтому для каждой группы программ после предварительного проведения классификации ПО нужно устанавливать свои способы оценки функциональности.

Защищённость и целостность. В коммерческих организациях большое значение имеют конфиденциальность, блокировка неавторизованного доступа к данным, предотвращение потери информации. Целостность очень важна для интернет-приложений.

Надёжность. Показатель надёжности наиболее важен для систем, выполняющих ответственные функции. Чем серьёзнее могут быть последствия сбоя в применяемой системе, тем выше требования к её надёжности.

Эффективность. Эффективность приобретает наибольшее значение в системах с большим количеством пользователей, обрабатывающих большие объёмы данных или производящих сложные расчёты. Эффективность также определяет минимальную конфигурацию необходимого оборудования.

Сопровождаемость. Сопровождаемость имеет значение при необходимости регулярных обновлений программного продукта (например, для антивирусов).

Переносимость и способность к взаимодействию. Переносимость зависит от уже используемых в организации программ, необходимости взаимодействия их с новой системой, а также от операционной системы и наличия сети.

Изучаемость и **простота использования**. Программы со сложным интерфейсом и значительной трудоёмкостью освоения операций пользования отвлекают обучаемого от основного предмета изучения, приводят к нерациональным затратам времени.

Анализ применения методов принятия решения на примере подбора программного обеспечния

Метод многокритериальной функции полезности

Метод MAUT (Multi-Attribute Utility Theory) имеет следующие особенности:

1. Строится функция полезности, имеющая аксиоматическое (чисто математическое) обоснование.
2. Некоторые условия, определяющие форму этой функции, подвергаются проверке в диалоге с ЛПР.
3. Обычно используется для решения задач с заданными альтернативами, а полученные результаты применяются для оценки заданных альтернатив.

Основные этапы подхода MAUT:
1. Разработать перечень критериев.
2. Построить функции полезности по каждому из критериев. (В практических задачах в случае линейной функции полезности бывает достаточно таблицы значений функции для имеющихся значений критерия).
3. Задать веса критериев.
4. Построить зависимость между оценками альтернатив по критериям и общим качеством альтернативы (многокритериальная функция полезности).
5. Оценить все имеющиеся альтернативы и выбрать наилучшую.

Аксиоматическое обоснование

В методе MAUT выдвигаются некоторые условия (аксиомы), которым должна удовлетворять функция полезности ЛПР. В MAUT эти условия можно разделить на две группы.

Первая группа — аксиомы общего характера:

1. Аксиома, утверждающая, что может быть установлено отношение между полезностями любых альтернатив: либо одна из них превосходит другую, либо они равны.

2. Аксиома транзитивности: из превосходства полезности альтернативы A над полезностью альтернативы B и превосходства полезности B над полезностью C следует

превосходство полезности альтернативы A над полезностью альтернативы C.

3. Для соотношений между полезностями альтернатив A, B, C, имеющими вид

$$U(A) > U(B) > U(C),$$

можно найти такие числа α, β меньше 1 и больше 0, что:

$$\alpha \cdot U(A) + (1-\alpha) \cdot U(C) = U(B),$$
$$U(A) \cdot (1-\beta) + \beta \cdot U(B) > U(B),$$

где U — многокритериальная функция полезности альтернативы.

Аксиома 3 основана на предположении, что функция полезности непрерывна, и что можно использовать любые малые части полезности альтернатив. Вторая группа условий специфична для MAUT. Они называются *аксиомами* (условиями) *независимости*, позволяющими утверждать, что некоторые взаимоотношения между оценками альтернатив по критериям не зависят от значений по другим критериям.

1. Независимость по разности

Предпочтения между двумя альтернативами, отличающимися лишь оценками по порядковой шкале одного критерия C_1, не зависят от одинаковых (фиксированных) оценок по другим критериям $C_2,\ldots,C_N$.

2. Независимость по полезности

Критерий C_1 называется *независимым по полезности* от критериев $C_2,\ldots, C_N$, если порядок предпочтений альтернатив, в которых меняются лишь уровни критерия C_1, не зависит от фиксированных значений по другим критериям.

3. Независимость по предпочтению

Независимость по предпочтению является одним из наиболее важных и часто используемых условий. Два критерия C_1 и C_2 *независимы по предпочтению* от других критериев $C_3,\ldots,C_N$, если предпочтения между альтернативами, различающимися лишь оценками по C_1, C_2, не зависят от фиксированных значений по другим критериям.

В тех примерах, где три и более критериев зависят от остальных, также проявляется и нарушение условия независимости по предпочтению. В связи с этим особое внимание уделяется проверке условия независимости по предпочтению. Если аксиомы первой группы и некоторые из условий независимости выполнены, то из этого следует строгий вывод о существовании многокритериальной функции полезности в определённом виде. По теореме Р. Кини, если условия независимости по полезности и независимости по предпочтению выполнены, то функция полезности является аддитивной:

$$U(x) = \sum_{i=1}^{N} w_i U_i(x),$$

при $\sum_{i=1}^{N} w_i = 1$, либо мультипликативной:

$$1 + kU(x) = \prod_{i=1}^{N} [1 + kw_i U_i(x)],$$

при $\sum_{i=1}^{N} w_i \neq 1$, где

U, U_i _ функции полезности, изменяющиеся от 0 до 1;
w_i — коэффициенты важности (веса) критериев, причём $0 < w_i < 1$;
коэффициент $k > -1$.

Таким образом, многокритериальную функцию полезности можно определить, если известны значения коэффициентов w_i, k, а также однокритериальные функции полезности U(x).

Рассмотрим применение метода на примере задачи выбора программного обеспечения. Имеются три пакета ПО для управления обработкой документов и потоками работ: Docs Open 3.0., Keyfile 3.1., Livelink Intranet 7.0. Были выбраны следующие критерии для их оценки:

- Функциональная пригодность.
- Защищённость.
- Документация и поддержка.
- Цена.

Экспертные оценки по критериям для каждой альтернативы приведены в табл.1:

Таблица 1

Критерий (оценки по шкале от 0 до 10)	Docs Open 3.0.	Keyfile 3.1.	Livelink Intranet 7.0.	Вес (от 0 до 1)
Функциональная пригодность	6	7	8	0,5
Защищённость	10	8	6	0,2
Документация и поддержка	7,5	7,5	6	0,15
Цена	7	8	6	0,15

Обобщённая оценка альтернатив по всем критериям:

$$A1 = 6 \cdot 0{,}5 + 10 \cdot 0{,}2 + 7{,}5 \cdot 0{,}15 + 7 \cdot 0{,}15 = 7{,}175$$
$$A2 = 7 \cdot 0{,}5 + 8 \cdot 0{,}2 + 7{,}5 \cdot 0{,}15 + 8 \cdot 0{,}15 = 7{,}425$$
$$A3 = 8 \cdot 0{,}5 + 6 \cdot 0{,}2 + 6 \cdot 0{,}15 + 6 \cdot 0{,}15 = 7{,}0$$

Хотя построение общей функции полезности требует достаточно много времени и усилий ЛПР, полученный результат позволяет оценить любые (в том числе и вновь появляющиеся) альтернативы.

Достоинства

С помощью метода MAUT можно определить полезность каждой из альтернатив. Многокритериальная теория полезности позволяет получить значения в интервальной шкале.

Недостатки

В методе MAUT ЛПР должен изначально задать точные количественные измерения всех основных параметров, что является достаточно сложным. Подход MAUT не даёт возможности провести исследования проблемы привычным для человека методом «проб и ошибок». Это приводит к тому, что различные заданные параметры (например, веса критериев) приводят к различным результатам.

Метод AHP

Метод анализа иерархий (Analytic Hierarchy Process — AHP) является систематической процедурой для иерархического представления элементов, определяющих суть проблемы. Метод состоит в декомпозиции проблемы на всё более простые составляющие части и дальнейшей обработке последовательности суждений ЛПР по парным сравнениям. В результате может быть выражена относительная степень (интенсивность) взаимодействия элементов в иерархии. Эти суждения затем выражаются численно. AHP включает в себя процедуры синтеза множественных суждений, получения приоритетности критериев и нахождения альтернативных решений. Такой подход к решению проблемы выбора исходит из естественной способности людей думать логически и творчески, определять события и устанавливать отношения между ними.

Постановка задачи, решаемой с помощью метода AHP, заключается обычно в следующем. Дано: общая цель (или цели) решения задачи; критерии оценки альтернатив; альтернативы. Требуется: выбрать наилучшую альтернативу. Подход AHP состоит из совокупности этапов.

Первый этап заключается в структуризации задачи. В AHP любая задача или проблема предварительно структурируется и представляется в виде иерархии, древовидной или сетевой. Таким образом, основная цель исследования и все факторы, в той или иной степени влияющие на достижение цели, распределяются по уровням в зависимости от степени и характера влияния. На первом уровне иерархии всегда находится одна вершина — цель проводимого исследования. Второй уровень иерархии составляют факторы, непосредственно влияющие на достижение цели. При этом каждый фактор представляется в строящейся иерархии вершиной, соединённой с вершиной 1-го уровня. Третий уровень составляют факторы, от которых зависят вершины 2-го уровня и т.д. Этот процесс построения иерархии продолжается до тех, пока в иерархию не включены все основные факторы или хотя бы для одного из факторов последнего уровня невозможно непосредственно получить необходимую информацию. По окончании

построения иерархии для каждой материнской вершины проводится оценка весовых коэффициентов, определяющих степень её зависимости от влияющих на неё вершин более низкого уровня. При этом используется метод парных сравнений Саати.

На втором этапе ЛПР выполняет парные сравнения элементов каждого уровня, результаты сравнений переводятся в числа. Используем метод парных сравнений Саати.

При парных сравнениях ЛПР даётся шкала словесных определений уровня важности, причём каждому определению ставится в соответствие число.

1. При сравнении элементов, принадлежащих одному уровню иерархии, ЛПР выражает своё мнение, используя одно из приведённых в таблице определений. В матрицу сравнения заносится соответствующее число. Если элемент A_1 доминирует над элементом A_2, то в клетку матрицы, соответствующую строке A_1 и столбцу A_2, заносится целое число, а в клетку, соответствующую строке A_2 и столбцу A_1, заносится обратное число. Если обозначить долю фактора (коэффициент важности) A_i через w_i, то элемент матрицы

$$a_{ij} = \frac{w_i}{w_j}$$

Таким образом, в предлагаемом варианте применения метода парных сравнений, определяются не величины разностей значений факторов, а их отношение. При этом очевидно

$$a_{ij} = \frac{1}{a_{ji}}.$$

Работа экспертов состоит в том, что, производя парное сравнение факторов $A_1,\ldots,A_n$ эксперт заполняет таблицу парных сравнений. Если $w_1, w_2,\ldots, w_n$ неизвестны заранее, то парные сравнения элементов производятся с использованием субъективных суждений, численно оцениваемых по шкале, а затем решается проблема нахождения компонента w.

2. При сравнении критериев определяется их важность или воздействие на достижение цели. Таким образом определяются веса критериев. При *n* критериях производится n(n – 1)/2 сравнений.

Таблица 2

Уровень важности	Количественное значение
Равная важность	1
Умеренное превосходство	3
Существенное или сильное превосходство	5
Значительное (большое) превосходство	7
Очень большое превосходство	9
Промежуточные решения между двумя соседними суждениями	2, 4, 6, 8

3. Вычисляются коэффициенты важности для элементов каждого уровня. При этом проверяется согласованность суждений ЛПР.

Один из основных методов отыскания вектора w основывается на одном из утверждений линейной алгебры.

При построении матриц парных сравнений важным вопросом является согласованность матрицы. Ранжирование элементов осуществляется с помощью главных собственных векторов матрицы $A[n\times n]$. Число λ является собственным числом квадратной матрицы A, а ненулевой элемент w — её собственным вектором. Собственные числа квадратной матрицы $A[n\times n]$ могут быть вычислены как корни уравнения $\det(A-\lambda E)=0$, а собственные векторы — как решение соответствующих однородных систем $(A-\lambda E)w=0$. При этом собственный вектор, соответствующий максимальному собственному числу, называется главным собственным вектором.

При проведении сравнений в реальной ситуации вычисленное максимальное собственное число будет отличаться от соответствующего собственного числа для идеальной матрицы. Это различие характеризует так называемую рассогласованность реальной матрицы и характеризует уровень доверия к полученным результатам. Для проверки согласованности собственные числа матрицы сравниваются с собственными числами случайно заполненной матрицы.

Рассогласованность матрицы парных сравнений может быть вызвана, по крайней мере, двумя факторами:

— личными качествами эксперта;
— степенью неопределённости объекта оценки.

Поэтому рассогласованность матрицы выступает как результат взаимодействия этих факторов. В последнем случае необходимо изучать объект таким, какой он есть со всеми присущими ему неопределённостями.

В практических задачах приближённое значение главного собственного вектора можно получить суммированием элементов каждой строки. Для каждой строки вычисляется сумма

$$a_i = \sum_{j=1}^{n} a_j$$

Затем всё a_i нормируются так, чтобы их сумма была равна 1. В результате получается искомый собственный вектор w. Этот способ значительно проще в реализации, но он не позволяет определять качество исходных данных.

4. Подсчитывается количественный индикатор качества каждой из альтернатив и определяется наилучшая альтернатива.

Рассмотрим применение метода на примере задачи выбора программного обеспечения, описанной выше.

Структуризация

Структура решаемой задачи может быть представлена в виде, показанном на рисунке 1.

В иерархии выделяются элементы-родители и элементы-потомки. Потомки воздействуют на элементы вышестоящего уровня иерархии, являющиеся по отношению к ним родителями. Для всех элементов-потомков, относящихся к одному родителю, строятся матрицы парных сравнений (табл. 3, 4).

Определение наилучшей альтернативы

Синтез полученных коэффициентов важности осуществляется по формуле:

$$S_i = \sum_{i=1}^{N} w_i V_{ji} \ ,$$

где S_i — показатель качества i-й альтернативы;

w_i — вес i-го критерия;

V_{ji} — важность j-й альтернативы по i-му критерию.

Для трёх программных продуктов проведённые вычисления позволяют определить:

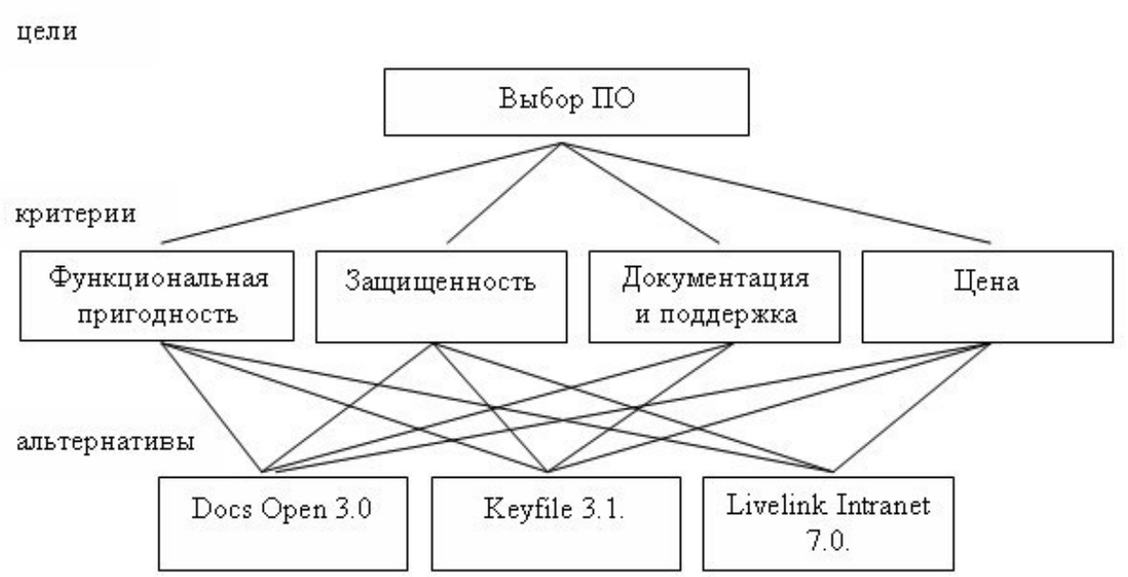

Рис. 1. Структура задачи выбора ПО по методу АНР

Таблица 3

Матрица сравнений критериев выбора ПО

Критерии	Функциональная пригодность (С1)	Защищённость (С2)	Документация и поддержка (С2)	Цена (С3)	Собственный вектор	Вес
С1	1	7	6	6	20	0,61
С2	1/7	1	3	3	7,143	0,22
С3	1/6	1/3	1	1/2	2	0,06
С4	1/6	1/3	2	1	3,5	0,11

Таблица 4

Матрица сравнения альтернатив по отдельным критериям

По критерию С1 (функциональная пригодность)					
Альтернатива	**Docs Open 3.0**	**Keyfile 3.1.**	**Livelink Intranet 7.0.**	**Собственный вектор**	**Вес**
Docs Open 3.0	1	1/2	1/3	1,83	0,16
Keyfile 3.1.	2	1	1/2	3,5	0,31
Livelink Intranet 7.0.	3	2	1	6	0,53
По критерию С2 (защищённость)					
Docs Open 3.0	1	3	5	8	0,58
Keyfile 3.1.	1/3	1	3	4,33	0,31
Livelink Intranet 7.0.	1/5	1/3	1	1,53	0,11
По критерию С3 (документация и поддержка)					
Docs Open 3.0	1	1	1/2	2,5	0,25
Keyfile 3.1.	1	1	1/2	2,5	0,25
Livelink Intranet 7.0.	2	2	1	5	0,5
По критерию С3 (цена)					
Docs Open 3.0	1	1/2	2	3,5	0,31
Keyfile 3.1.	2	1	3	6	0,53
Livelink Intranet 7.0.	1/2	1/3	1	1,83	0,16

A1 = $0,61 \cdot 0,16 + 0,22 \cdot 0,58 + 0,06 \cdot 0,25 + 0,11 \cdot 0,31 = 0,2743$;

A2 = $0,61 \cdot 0,31 + 0,22 \cdot 0,31 + 0,06 \cdot 0,25 + 0,11 \cdot 0,53 = 0,3306$;

A3 = $0,61 \cdot 0,53 + 0,22 \cdot 0,11 + 0,06 \cdot 0,5 + 0,11 \cdot 0,16 = 0,3951$;

Альтернатива Livelink Intranet 7.0. оказалась лучшей.

Общая характеристика подхода AHP

Метод AHP направлен на сравнение реальных альтернатив.

Достоинства

Метод AHP может применяться в тех случаях, когда эксперты (или ЛПР) не могут дать абсолютные оценки альтернатив по критериям, а пользуются более слабыми сравнительными измерениями. Метод позволяет получить приоритеты в шкале отношений.

Недостатки

Введение новой, недоминирующей альтернативы может в общем случае привести к изменению предпочтений между двумя ранее заданными альтернативами.

Метод ELECTRE

Метод ELECTRE направлен на решение задач с уже заданными многокритериальными альтернативами. В отличие от метода AHP в методе ELECTRE не определяется количественно показатель качества каждой из альтернатив, а устанавливается лишь условие превосходства одной альтернативы над другой. Постановка задачи обычно имеет следующий вид:

Дано: N критериев со шкалами оценок (обычно количественные), веса критериев (обычно целые числа), альтернативы с оценками по критериям.

Требуется: выделить группу лучших альтернатив.

Основные этапы методов ELECTRE:

1. На основании заданных оценок двух альтернатив подсчитываются значения двух *индексов*: *согласия* и *несогласия*. Эти индексы определяют согласие и несогласие с гипотезой, что альтернатива A превосходит альтернативу B.

2. Задаются *уровни согласия* и *несогласия*, с которыми сравниваются подсчитанные индексы для каждой пары альтернатив. Если индекс согласия выше заданного уровня, а индекс несогласия ниже, то одна из альтернатив превосходит другую. В противном случае альтернативы несравнимы.

3. Из множества альтернатив удаляются доминируемые. Оставшиеся образуют первое ядро. Альтернативы, входящие в ядро, могут быть либо эквивалентными либо несравнимыми.

4. Вводятся более «слабые» значения уровней согласия и несогласия (меньший по значению уровень согласия и больший уровень несогласия), при которых выделяются ядра с меньшим количеством альтернатив.

5. В последнее ядро входят наилучшие альтернативы. Последовательность ядер определяет упорядоченность альтернатив по качеству.

Индексы согласия и несогласия

В различных методах семейства ELECTRE индексы согласия и несогласия строятся по-разному. Рассмотрим принцип построения этих индексов на примере метода ELECTRE1.

Каждому из N критериев ставится в соответствие целое число p, характеризующее важность критерия. Выдвигается гипотеза о превосходстве альтернативы A над альтернативой B. Множество I, состоящее из N критериев, разбивается на три подмножества:

I^+ — подмножество критериев, по которым A предпочтительнее B;

$I^=$ — подмножество критериев, по которым A равноценно B;

I^- — подмножество критериев, по которым B предпочтительнее A.

Далее формулируется индекс согласия с гипотезой о превосходстве A над B. Индекс согласия подсчитывается на основе весов

критериев. В методе ELECTRE1 этот индекс определяется как отношение суммы весов критериев подмножеств I^+ и I^- к общей сумме весов:

$$c_{AB} = \frac{\sum_{i \in I^+, I} w_i}{\sum_{i \in I} w_i}$$

Индекс несогласия d_{AB} с гипотезой о превосходстве A над B определяется на основе самого «противоречивого» критерия — критерия, по которому B в наибольшей степени превосходит A. Чтобы учесть возможную разницу длин шкал критериев, разность оценок B и A относят к длине наибольшей шкалы:

$$d_{AB} = \max_{i \in I^-} \frac{l_B^i - l_A^i}{L_i},$$

где l_A^i, l_B^i оценки альтернатив A и B по i-му критерию;

L_i — длина шкалы i-го критерия.

Свойства индекса согласия:

1. $0 < c_{AB} < 1$;
2. $c_{AB} = 1$, если подмножество I^{+-} пусто;
3. c_{AB} сохраняет значение при замене одного критерия на несколько с тем же общим весом.

Свойства индекса несогласия:

1. $0 < d_{AB} < 1$;;
2. d_{AB} сохраняет значение при введении более детальной шкалы по i-му критерию при той же её длине.

Введённые индексы используются при построении матриц индексов согласия и несогласия для заданных альтернатив.

Выделение ядер

В методе ELECTRE 1 бинарное отношение превосходства задаётся уровнями согласия и несогласия. Если $c_{AB} \geq c_1$ и $d_{AB} \leq d_1$, где c_{AB}, d_{AB} — заданные уровни согласия и несогласия, то альтернатива A объявляется лучшей по сравнению с альтернативой B. Если же при этих уровнях сравнить альтернативы не удалось, то они объявляются *несравнимыми*. Если оценки альтернатив в значительной степени противоречивы (по одним критериям одна намного лучше другой, а по другим — наоборот), то такие противоречия никак не компенсируются и такие альтернативы сравнивать нельзя. Понятие несравнимости позволяет выявить альтернативы с «контрастными» оценками, как заслуживающие специального изучения. Похожие идеи используются и в других методах семейства ELECTRE.

Задавая уровни коэффициентов согласия и несогласия, постепенно понижая требуемый уровень коэффициента согласия и повышая требуемый уровень коэффициента несогласия, ЛПР может исследовать имеющееся множество альтернатив. При заданных уровнях на множестве альтернатив выделяется ядро недоминируемых элементов, которые находятся либо в отношении несравнимости, либо в отношении эквивалентности. При изменении уровней из данного ядра выделяется меньшее ядро и т. д. Различные ядра представляют собой возможные решения проблемы. В конечном итоге можно получить одну лучшую альтернативу.

Рассмотрим пример задачи выбора программного обеспечения, описанный выше.

Таблица 5

Таблица критериев

Критерий (оценки по шкале от 0 до 10)	Docs Open 3.0.	Keyfile 3.1.	Livelink Intranet 7.0.	Вес
Функциональная пригодность	6	7	8	0,5
Защищённость	10	8	6	0,2
Документация и поддержка	7,5	7,5	6	0,15
Цена	7	8	6	0,15

Индексы согласия

	Docs Open 3.0.	Keyfile 3.1.	Livelink Intranet 7.0.
Docs Open 3.0.	*	0,55	0,5
Keyfile 3.1.	0,2	*	0,5
Livelink Intranet 7.0.	0,5	0,5	*

Индексы несогласия

	Docs Open 3.0.	Keyfile 3.1.	Livelink Intranet 7.0.
Docs Open 3.0.	*	0,2	0,4
Keyfile 3.1.	0,1	*	0,2
Livelink Intranet 7.0.	0,2	0,1	*

Зададим уровень согласия $c_{AB} = 5$ и уровень несогласия $d_{AB} = 0{,}1$. Тогда в ядро входят доминирующая альтернатива Keyfile 3.1.

Общая характеристика метода

Достоинства: поэтапность выявления предпочтений ЛПР в процессе назначения уровней согласия и несогласия и изучения ядер. Детальный анализ позволяет ЛПР сформировать свои предпочтения, определить компромиссы между критериями.

Использование отношения несравнимости позволяет выделить пары альтернатив с противоречивыми оценками, остановиться на ядре, выделение которого достаточно обоснованно с точки зрения имеющейся информации.

Построение модели системы принятия решений

Модель системы принятия решения строится в контексте решения некоторой проблемы или достижения некоторой цели. Для построения модели системы необходимо структурировать проблемную область, определив её границы и внутренние взаимосвязи.

Выбор алгоритма принятия решения

Выбор алгоритма принятия решения зависит от особенностей решаемой задачи и модели системы принятия решения. Задача подбора программного обеспечения имеет следующие особенности:

• Задача является слабо структурированной, то есть содержит преимущественно качественные критерии оценки альтернатив.

• Критерии альтернатив представляют из себя иерархическую структуру: атрибуты качества ПО детализируются более конкретными субхарактеристиками.

• Число альтернатив заранее неизвестно, количество и состав альтернатив в задаче могут изменяться.

• Некоторые значения критериев взаимосвязаны и зависят друг от друга. В табл. 6 показана взаимосвязь соответствующих атрибутов: знак «+» означает, что увеличения величины атрибута в соответствующей строке позитивно влияет на атрибут в соответствующем столбце. Знак «–» означает негативное влияние, а пустая ячейка говорит о том, что атрибут строки оказывает незначительное влияние на атрибут столбца. Например, увеличение гибкости, целостности, способности программы к взаимодействию и других атрибутов наверняка скажется как на времени отклика, так и на объёме занимаемой памяти и приведёт к падению эффективности. Эти закономерности могут использоваться для проверки объективности имеющихся данных, а также учитываться в случае недостающих данных.

Для применения данной программы в различных областях и для оценки программных продуктов различных классов необходимо обеспечить гибкость модели системы принятия решений:

Таблица 6

Взаимосвязи атрибутов качества ПО

	Доступность	Эффективность	Гибкость	Целостность	Способность к взаимодействию	Лёгкость в эксплуатации	Переносимость	Надёжность	Устойчивость к сбоям	Удобство и простота использования
Доступность								+	+	
Эффективность			–		–	–	–		–	-
Гибкость		–		-		+	+	–		
Целостность		–			-			+		–
Способность к взаимодействию		–	+	-			+			
Изучаемость	+	–	+					+		
Переносимость		–	+		+	-				–
Надёжность	+	–	+			+			+	+
Устойчивость к сбоям	+	–						+		+
Удобство и простота использования		-							+	

• возможность изменения состава структуры критериев, добавление новых или удаления существующих критериев;
• возможность изменения списка альтернатив, добавление и удаление программных продуктов, подлежащих оценке.

При изменении условий применения системы затраты на изменения модели системы должны быть минимальными.

Метод MAUT. Для изменения структуры критериев необходимо задать веса новых критериев и оценки всех альтернатив по ним, и произвести нормировку весов всех критериев. MAUT также позволяет добавлять новые альтернативы — для этого ЛПР должен задать оценки новых альтернатив по каждому из критериев. ЛПР должен изначально задавать точные количественные оценки альтернатив по критериям, при этом даже небольшое изменение оценки может сказаться на результате.

Метод AHP является наиболее подходящим для реализации иерархической структуры критериев, однако в этом случае ЛПР задаёт гораздо больше значений, чем в методе MAUT. При добавлении новой альтернативы необходимо сравнить её со всеми существующими альтернативами по всем критериям, а при добавлении нового критерия — сравнить его по значимости с имеющимися критериями и произвести попарное сравнение всех альтернатив по новому критерию. Таким образом, изменение модели требует большой работы ЛПР.

Метод ELECTRE. Для выявления оптимального набора альтернатив ЛПР необходимо поэтапно анализировать получаемые в ходе алгоритма ядра и изменять уровни согласия и несогласия. Таким образом, суждения ЛПР требуются в процессе всего нахождения решения.

Вышеперечисленные методы также не позволяют учитывать взаимосвязь критериев эффективности ПО и использовать её в случае недостающих исходных данных.

Для решения этой задачи и построения наиболее гибкой модели можно использовать алгоритм нечёткого вывода.

Алгоритм нечёткого вывода позволяет наиболее просто и быстро изменять модель системы путём добавления или изменения правил вывода. При добавлении нового критерия эксперту необходимо задать термы и функции принадлежности для соответствующей новой лингвистической переменной и правила, учитывающие оценку по новому критерию в общем выводе. При добавлении новой альтернативы эксперт должен задать её оценки по всем критериям по выбранным шкалам. Алгоритм также позволяет учитывать взаимосвязь критериев с помощью задания соответствующих правил. В этом случае при отсутствии оценок альтернативы по некоторым критериям могут использоваться соответствующие правила исходя из оценок по остальным критериям.

Построение системы нечёткого вывода для задачи подбора ПО

В качестве лингвистических переменных в рамках данной задачи выступают критерии оценки ПО. Для них задаются такие термы, как «Низкий», «Средний», «Высокий» и т.д. Каждому терму лингвистической переменной соответствует нечёткое множество и функция. С помощью функции принадлежности определяются степени принадлежности каждого элемента глобального множества данному терму. Элементами глобального множества являются численные экспертные оценки характеристик программного продукта по некоторой шкале.

На вход алгоритма нечёткого вывода подаются чёткие значения характеристик альтернатив (экспертные оценки программных продуктов). Выходной переменной системы является лингвистическая переменная «Полезность», характеризующая предпочтительность выбора альтернативы.

Для реализации алгоритма принятия решения с помощью нечёткого вывода необходимо описать правила, по которым осуществляется связь между различными характеристиками и выходным значением полезности альтернатив. Сравнение альтернатив и принятие решения производятся с помощью продукционных правил. Каждое продукционное правило состоит из одной или нескольких посылок и заключения. Если посылок несколько, они связываются логическим «И». Продукционное правило имеет вид:

ЕСЛИ *посылка* [И *посылка* …],
ТО *заключение*, где *посылка* и *заключение* являются нечёткими высказываниями.

Например:

ЕСЛИ *Надёжность высокая* и *Изучаемость средняя*, ТО *Полезность высокая*.

Используемые для вывода правила учитывают характеристики альтернатив по всем критериям и степени важности каждой из характеристик для пользователя. Чем выше степень важности некоторой характеристики для пользователя, тем более высокую оценку по соответствующему критерию должна иметь альтернатива.

На основе имеющихся правил делаются соответствующие заключения для выходной переменной. Заключения правил объединяются для получения чётких значений переменной полезности. Полученные чёткие значения являются численными оценками предпочтительности альтернатив.

Алгоритмы нечёткого вывода

Пусть x, y — входные переменные, имеющие чёткие значения x_0, y_0.
z — выходная переменная.

Заданы функции принадлежности A_1, A_2, B_1, B_2, C_1, C_2 и правила:

Если x есть A_1 и y есть B_1, ТО z есть C_1,

Если x есть A_2 и y есть B_2, ТО z есть C_2

Алгоритм Mamdani

В системах типа Mamdani база знаний строится из нечётких высказываний вида «β есть α» с помощью связок «И», «ЕСЛИ-ТО»:

ЕСЛИ x *высокий* И y *средний*, ТО z высокий.

Этапы нечёткого вывода реализуются следующим образом:

1. Фаззификация: находятся степени истинности для предпосылок каждого правила: $A_1(x_0)$, $A_2(x_0)$, $B_1(y_0)$, $B_2(y_0)$.

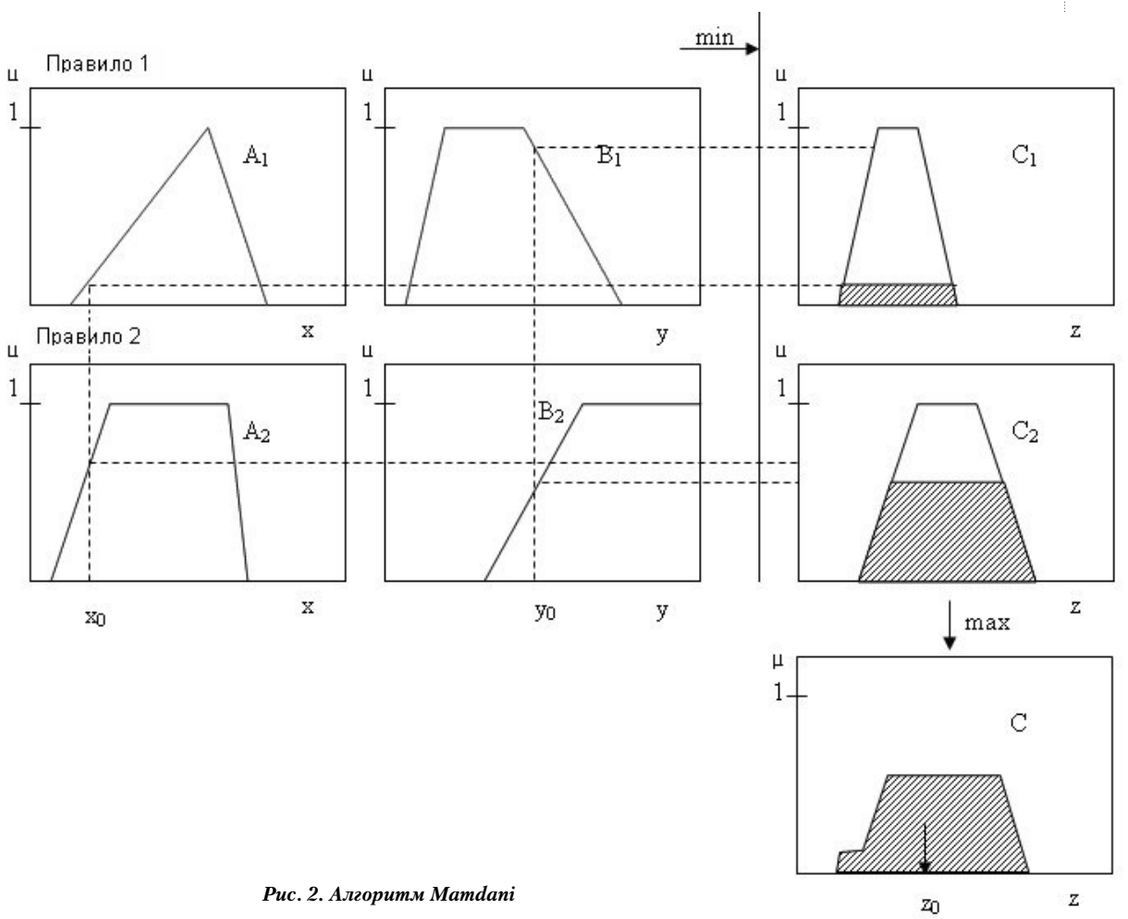

Рис. 2. Алгоритм Mamdani

2. Вывод: находятся уровни отсечения для предпосылок каждого из правил с использованием операции минимум:

$$\alpha_1 = A_1(x_0) \wedge B_1(y_0),$$

$$\alpha_2 = A_2(x_0) \wedge B_2(y_0),$$

где $\wedge$ — операция логического минимума.

Затем находятся усечённые функции принадлежности:

$$C_1'(z) = (\alpha_1 \wedge C_1(z)),$$

$$C_2'(z) = (\alpha_2 \wedge C_2(z))$$

1. Композиция: с использованием операции максимум (обозначается как «$\vee$») производится объединение найденных усечённых функций, что приводит к получению итогового нечёткого подмножества для переменной выхода с функцией принадлежности:

$$\mu_{\sum}(z) = C(z) = C_1'(z) \vee C_2'(z) = (\alpha_1 \wedge C_1(z)) \vee (\alpha_2 \wedge C_2(z))$$

Приведение к чёткости для получения z_0 производится методом центра тяжести.

Алгоритм Tsukamoto

Исходные данные и база знаний такие же, как и в алгоритме Mamdani, но предполагается, что функции $C_1(z)$, $C_2(z)$ являются монотонными.

Этапы нечёткого вывода:

1. Фаззификация: находятся степени истинности для предпосылок каждого правила: $A_1(x_0)$, $A_2(x_0)$, $B_1(y_0)$, $B_2(y_0)$.

2. Вывод: Находятся уровни отсечения для предпосылок каждого из правил с использованием операции минимум:

$$\alpha_1 = A_1(x_0) \wedge B_1(y_0),$$

$$\alpha_2 = A_2(x_0) \wedge B_2(y_0),$$

где $\wedge$ — операция логического минимума.

Затем находятся чёткие значения z_1 и z_2 из уравнений

$$\alpha_1 = C_1(z),$$
$$\alpha_2 = C_2(z)$$

3. Определяется чёткое значение переменной вывода, как взвешенное среднее z_1 и z_2:

$$z_0 = \frac{\alpha_1 z_1 + \alpha_2 z_2}{\alpha_1 + \alpha_2}$$

В общем случае чёткое значение z_0 определяется по формуле ниже (дискретный вариант метода центра тяжести).

Для дискретного случая: $z_0 = \dfrac{\sum_{i=1}^{n} x_i \mu(x_i)}{\sum_{i=1}^{n} \mu(x_i)}$

где z_0 — чёткое значение выходной переменной.

Алгоритм Sugeno

В алгоритме Sugeno,аза знаний строится из правил в следующей форме:

Если *x* есть A_1 и *y* есть B_1, ТО $z_1 = a_1 x + b_1 y$,

Если *x* есть A_2 и *y* есть B_2, ТО

$$z_2 = a_2 x + b_2 y$$

Этапы нечёткого вывода:

1. Фаззификация: находятся степени истинности для предпосылок каждого правила: $A_1(x_0)$, $A_2(x_0)$, $B_1(y_0)$, $B_2(y_0)$.

2. Вывод: Находятся уровни отсечения для предпосылок каждого из правил с использованием операции минимум:

$$\alpha_1 = A_1(x_0) \wedge B_1(y_0),$$
$$\alpha_2 = A_2(x_0) \wedge B_2(y_0)$$

Находятся индивидуальные выходы правил:

$$z^*_1 = a_1 x + b_1 y$$
$$z^*_2 = a_2 x + b_2 y$$

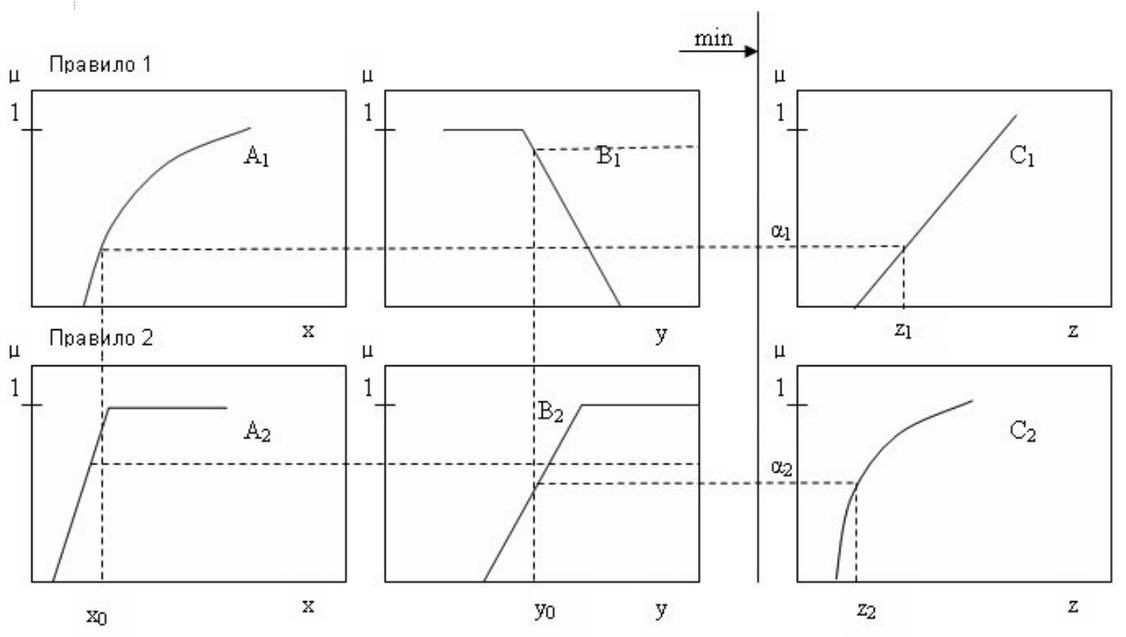

Рис. 3. Алгоритм Tsukamoto

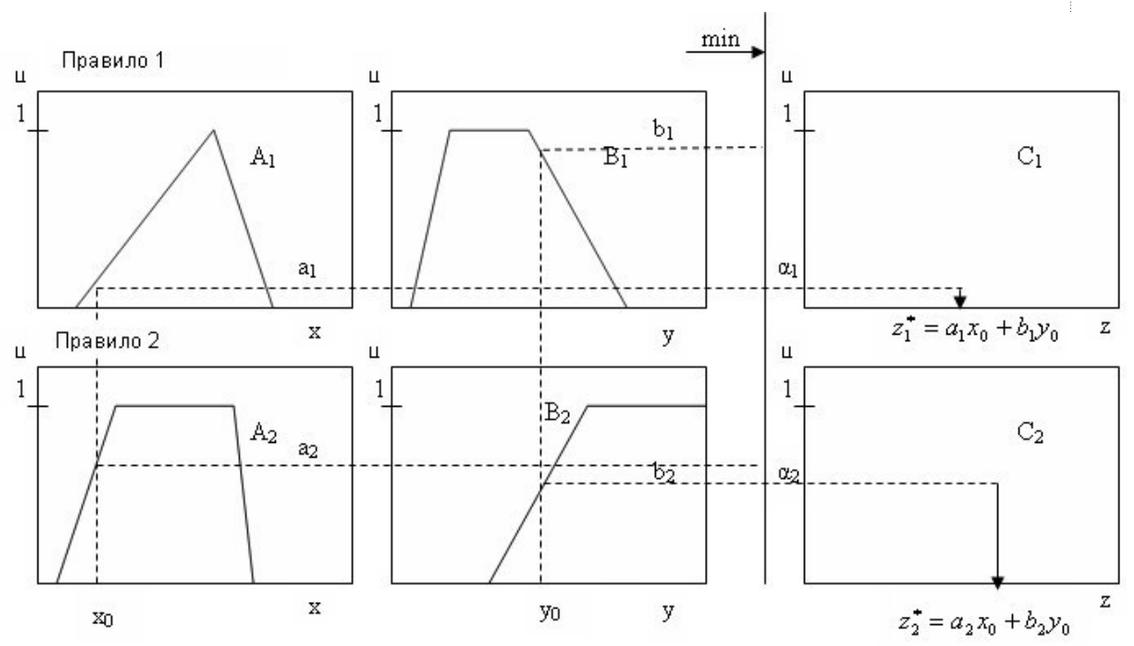

Рис. 4. Алгоритм Sugeno

3. Определяется чёткое значение переменной вывода:

$$z_0 = \frac{\alpha_1 z^*_1 + \alpha_2 z^*_2}{\alpha_1 + \alpha_2}$$

Алгоритм Larsen

Вид базы знаний совпадает с видом базы знаний для алгоритма Mamdani.

1. Нечёткость: находятся степени истинности для предпосылок каждого правила: $A_1(x_0)$, $A_2(x_0)$, $B_1(y_0)$, $B_2(y_0)$.

2. Вывод: находятся уровни отсечения для предпосылок каждого из правил с использованием операции минимум:

$$\alpha_1 = A_1(x_0) \wedge B_1(y_0),$$

$$\alpha_2 = A_2(x_0) \wedge B_2(y_0),$$

где $\wedge$ — операция логического минимума.

В алгоритме Larsen нечёткое подмножество переменной вывода для каждого правила находится с использованием оператора умножения по формуле:

$$C_1'(z) = (\alpha_1 C_1(z))$$

$$C_2'(z) = (\alpha_2 C_2(z)),$$

1. Композиция: с использованием операции максимум (обозначается как «$\vee$») производится объединение найденных частных нечётких подмножеств. Находится итоговое нечёткое подмножество для переменной выхода с функцией принадлежности:

$$\mu_\Sigma(z) = C(z) = C_1'(z) \vee C_2'(z) = (\alpha_1 C_1(z)) \vee (\alpha_2 C_2(z))$$

(в общем случае $\mu_\Sigma(z) = \bigvee_{i=1}^{n} (\alpha_i C_i(z))$

2. Приведение к чёткости также производится методом центра тяжести.

Выбор алгоритма нечёткого вывода

Применяемые в данной задаче характеристические функции не всегда являются монотонными (функции П и Л типов), что исключает применение метода Tsukamoto. Одной из основных причин использования нечёткой логики для механизма оценки является возможность задания требуемой за-

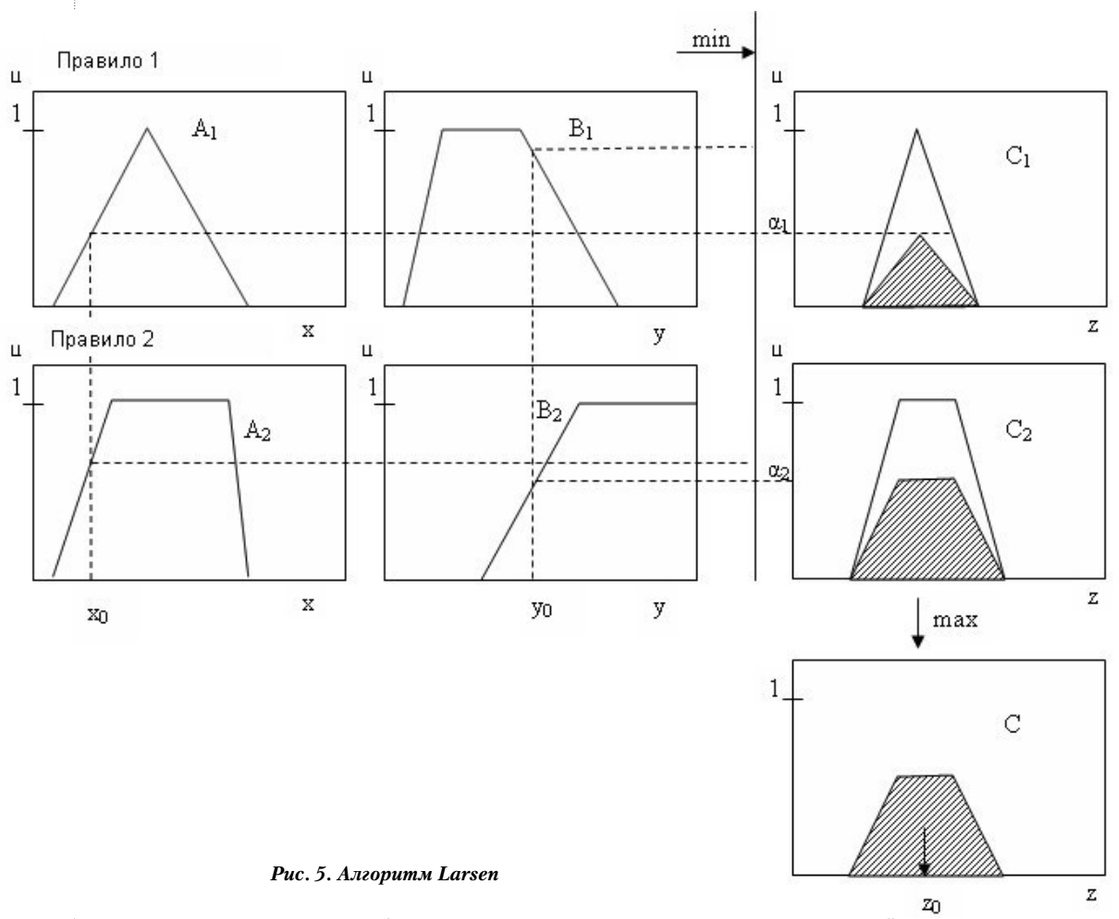

Рис. 5. Алгоритм Larsen

висимости на языке, близком к естественному. Наиболее подходящие базы для этого базы знаний применяются в методах Mamdani и Larsen.

Для выбора наилучшего алгоритма нечёткого вывода были проведены сравнения результатов работы методов Mamdani, Sugeno, Larsen и метода MAUT. Поскольку метод MAUT является основным методом многокритериальной теории полезности, для реализации был выбран алгоритм, который дал оценки, наиболее близкие к оценкам метода MAUT. Алгоритмы тестировались на одинаковых наборах данных. Для сравнения использовался набор программных продуктов, включающий 8 интернет-браузеров, а также условно наилучшую и наихудшую альтернативу.

Для каждого программного продукта были заданы экспертные оценки по критериям: доступность, гибкость, защищённость, изучаемость, эффективность, взаимодействие с программами, надёжность и простота использования. Для наилучшей и наихудшей альтернативы были заданы соответственно наиболее высокие и наиболее низкие оценки по всем критериям.

Сравнение методов было проведено для двух случаев: в первом случае для всех критериев был задан средний уровень важности. При этом оценки по каждому из критериев имеют одинаковое влияние на переменную выхода.

Полученные оценки представлены в табл. 7.

Из рисунка 6 видно, что наиболее близки к оценкам по методу MAUT оценки алгоритмов Mamdani и Larsen. Для некоторых альтернатив (Opera, SeaMonkey, Internet Explorer, K-Meleon) метод Mamdani даёт более близкие оценки, чем метод Larsen.

Во втором случае для критериев оценки были заданы различные уровни важности,

Таблица 7

Оценки альтернатив при среднем уровне важности всех критериев, полученные с использованием алгоритмов MAUT, Mamdani, Sugeno, Larsen

ПО	MAUT	Нечёткий вывод (Mamdani)	Нечёткий вывод (Sugeno)	Нечёткий вывод (Larsen)
Наилучшая альтернатива	10,000	8,500	10,000	8,500
Opera	8,175	8,386	6,409	8,395
Mozilla Firefox	7,825	8,5	5,593	8,5
Netscape	7,700	8,5	5,055	8,5
SeaMonkey	7,150	7,985	6,061	8,009
Internet Explorer	6,900	7,790	6,295	7,938
Dr. Orca	6,725	8,125	5,285	8,125
K-Meleon	6,675	7,934	5,688	7,961
Maxthon	6,225	7,875	4,324	7,875
Наихудшая альтернатива	0,000	0,000	0,000	0,000

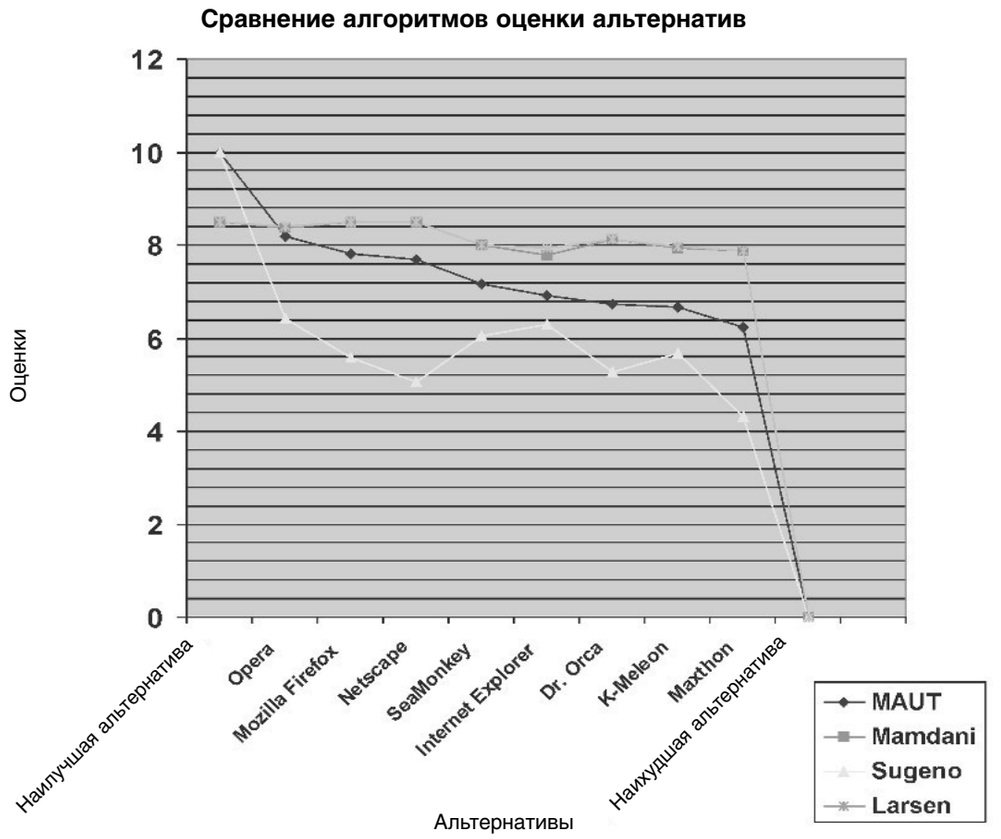

Рис. 6. Сравнение алгоритмов нечёткого вывода Mamdani, Sugeno, Larsen и метода MAUT при оценке альтернатив по критериям с одинаковым уровнем важности

в качестве примера выбора уровней важности пользователем.

Были заданы следующие требуемые уровни критериев:

Доступность	Средняя
Гибкость	Высокая
Защищённость	Очень высокая
Изучаемость	Не имеет значения
Эффективность	Высокая
Взаимодействие с программами	Не имеет значения
Надёжность	Средняя
Простота использования	Не имеет значения

Рис. 7 также показывает, что наиболее близки к оценкам по методу MAUT оценки алгоритмов Mamdani и Larsen. В случае равных уровней важности критериев значение имеют только численные оценки альтернатив, заданные экспертами, поэтому результаты работы алгоритмов нечёткого вывода зависят только от функций принадлежности лингвистических переменных. В случае присваивания критериям различных уровней важности на получаемые оценки также оказывают влияние правила вывода, поэтому нечёткий вывод даёт более точные (близкие к методу MAUT) оценки.

Для реализации был выбран алгоритм Mamdani, так как он даёт оценки, наиболее близкие к оценкам метода MAUT.

Выводы

1. Предложенный метод выбора ПО основывается на комплексной оценке качественных характеристик. Реализованный подход к оценке качества ПО позволяет строить гибкую систему оценок в зависимости от целей и приоритетов пользователей в каждом конкретном случае.

2. Разработанное приложение позволяет сравнивать оценки, полученные различными методами.

3. Алгоритм нечёткого вывода обеспечивает гибкость системы принятия решения за счёт возможности корректировки критериев оценки и правил вывода.

4. Web-приложение даёт возможность не только обеспечить доступ к системе большого количества пользователей, но и позволяет организовать взаимодействие экспертов, обладающих необходимыми знани-

Таблица 8

Оценки альтернатив при различных уровнях важности всех критериев, полученные с использованием алгоритмов MAUT, Mamdani, Sugeno, Larsen

ПО	MAUT	Нечёткий вывод (Mamdani)	Нечёткий вывод (Sugeno)	Нечёткий вывод (Larsen)
Наилучшая альтернатива	10	8,5	10	8,5
Opera	8,2	7,92	6,082	7,947
Netscape	8,011	7,826	5,342	7,857
SeaMonkey	7,778	7,826	6,558	7,857
Mozilla Firefox	7,733	7,92	5,655	7,947
K-Meleon	7,111	6,549	6,168	7,732
Dr. Orca	7	7,37	4,962	7,414
Internet Explorer	6,933	5,614	7,099	5,659
Maxthon	5,667	5,404	4,032	5,439
Наихудшая альтернатива	0	2	0	2

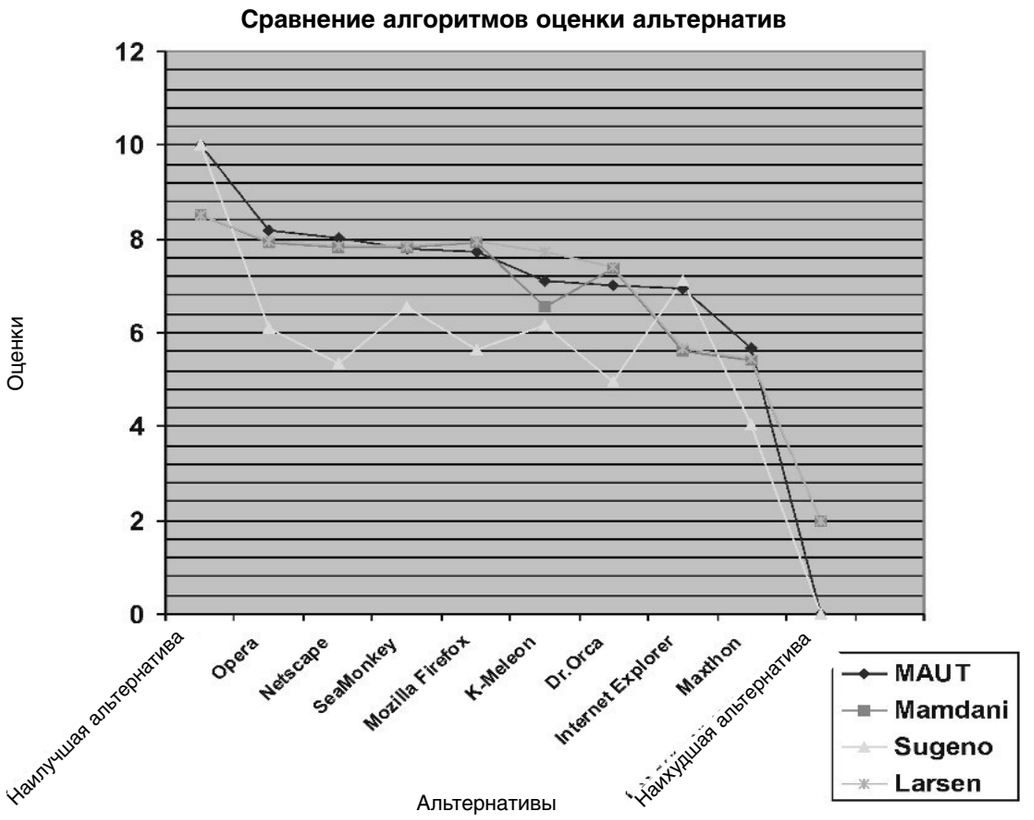

Рис. 7. Сравнение алгоритмов нечёткого вывода Mamdani, Sugeno, Larsen и метода MAUT при оценке альтернатив по критериям с различными уровнями важности

ями, и пользователей, нуждающихся в этих знаниях. Система является динамичной и позволяет добавлять и удалять оцениваемые программные продукты, расширяя список выбора для пользователей.

5. Разработанный подход к оценке является универсальным и может быть применён не только для оценки ПО, но и в любой предметной области. ❑