УДК 681.5.015

В.П. АВРАМЕНКО

ИНТЕЛЛЕКТУАЛЬНЫЕ ПРОЦЕДУРЫ ОБРАБОТКИ ДАННЫХ В СИСТЕМАХ КОНТРОЛЯ ЗНАНИЙ

Исследуются причины некорректности постановок задач обработки данных в системах контроля знаний. Предлагаются интеллектуальные процедуры регуляризации некорректно поставленных задач на основе принципа внешнего дополнения. Применение разработанных интеллектуальных процедур позволило повысить эффективность поиска и обработки данных, благодаря подбору информации, более релевантной запросу.

1. Введение

В основу многих интеллектуальных систем, в том числе и систем контроля знаний, закладывается принцип текстового диалога, когда вопрос задается обучающей стороной, а ответ формируется обучаемой стороной по заранее оговоренным правилам. В системах контроля знаний важное место занимает проблема однозначного смыслового толкования задаваемого вопроса и получаемого ответа. Для решения этой проблемы в формализованных системах обработки данных используются разнообразные интеллектуальные процедуры регуляризации [1- 4].

Регуляризация представляет собой процесс интеллектуального исправления исходной некорректно поставленной задачи на корректно поставленную задачу путем привлечения дополнительной информации с последующим отысканием приближенного решения в области допустимых решений. Термин регуляризация происходит от латинского слова *regula***,** которое означает что-то, сделанное по определенным *правилам*. Задачи обработки данных не всегда удовлетворяют желаемым *правилам*, поэтому по природе своей они часто принадлежат к некорректно поставленным задачам.

Многие процедуры обработки данных в системе контроля знаний описываются алгебраическими соотношениями. Каждому алгебраическому объекту ставится в соответствие набор правил «хорошего поведения», при соблюдении которых «регуляризованные» вычислительные процедуры ведут себя «наилучшим образом». Регулярным объектом может выступать регулярный оператор, в частном случае, хорошо обусловленная матрица, а нерегулярным объектом может выступать плохо обусловленная матрица. Для вычисления обратной матрицы к исходной нерегулярной (плохо обусловленной) матрице применяются «регуляризованные» процедуры обработки данных.

2. Цель и задачи интеллектуализации процедур обработки данных

Качество принимаемых решений в задачах обработки данных тесно связано с характером некорректности их исходной постановки. Большинство такого рода задач принадлежит к классу обратных некорректно поставленных задач. Некорректность проявляется в том, что решение исходной задачи может вовсе не существовать, быть неединственным или оказаться неустойчивым. В последнем случае сравнительно небольшие отклонения исходных данных могут привести к значительным отклонениям получаемых решений.

Для преодоления некорректной постановки задач обработки данных и получения устойчивых загрубленных решений используются регулярные языки и регулярные выражения, принцип внешнего дополнения, методы регуляризации, процедуры псевдообращения, методы гребневой регрессии и др. Исследования корректности постановок задач направлены на уточнение вопроса о существовании, единственности и устойчивости принимаемых решений в системе контроля знаний [4, с. 16; 5, с. 26].

Многие постановки задачи обработки данных и поддержки принимаемых решений являются многокритериальными. Сложность решения таких многокритериальных задач вытекает из некорректности их постановки, проявляющейся в следующем:

- не существует решения, доставляющего одновременно экстремум всем локальным критериям;

- для противоречивых целевых функций оптимальное решение принадлежит области компромиссов, в которой улучшение одних критериев вызывает ухудшение других критериев, что в свою очередь порождает неединственность решений;

- возможна неустойчивость решений из-за того, что малым вариациям параметров схемы компромисса может соответствовать большой разброс компромиссных решений.

Многие реальные задачи обработки данных решаются в условиях неопределенности цели, функционирования и проявления внешней среды. Источниками неопределенности могут выступать: отсутствие информации о предпочтительности многокритериальных решений; несогласованность требований качества обработки и выделенных ресурсов; отклонение исходных предпосылок от фактически существующих условий; наличие дрейфа характеристик и зашумленности измерений, слабой формализуемости процессов и плохой обусловленности системы.

Решения, принимаемые в условиях неопределенности исходных данных и проявления внешней среды, всегда приводят к худшим результатам, чем при полной определенности. В этом случае под оптимальным решением подразумеваем не самое безусловно лучшее, а лучшее в некотором смысле, например, в среднем, при многократном повторении или с точки зрения конкретного лица, принимающего решения. При формализации задачи делается попытка снизить меру неопределенности путем привлечения дополнительной информации. Фактически осуществляется переход от полной неопределенности к частичной стохастической неопределенности или расплывчатым множествам, что приводит к неоднозначности получаемых результатов.

Целью исследований является анализ некорректности задач обработки данных, а также разработка процедур регуляризации (исправления) некорректно поставленных задач для повышения эффективности принимаемых решений в системах контроля знаний. Для достижения поставленной цели предусмотрено *решение следующих задач*:

1) анализ причин некорректности задач обработки данных в системах контроля знаний;

2) разработка интеллектуальных регуляризованных процедур обработки данных для однозначного смыслового толкования задаваемого вопроса и получаемого ответа.

3. Математическое описание процедур обработки данных

Компьютерная обработка данных в системе контроля знаний сориентирована на решение следующих задач: кодирование и пересылка данных; поиск всех слов, содержащих последовательность символов query; поиск всех слов, начинающихся с последовательности символов query и оканчивающихся цифрой, и т. д. При обработке данных важное место занимают регулярные языки и регулярные выражения. Под выражением следует подразумевать совокупность действий, выполняемых в заданной последовательности для того, чтобы получить определенное значение некоторого алгебраического объекта. Среди алгебраических выражений особое место занимают регулярные выражения, составленные по определенным правилам на специальном алгебраическом языке. В качестве математического описания процедур обработки данных может использоваться аппарат математической лингвистики, алгебры регулярных событий, теории графов и теории множеств.

Многие задачи обработки данных описываются математической моделью вида [5]

$$y = Ax,\ y \in Y \subset E_y, x \in X \subset E_x, \tag{1}$$

где A – оператор системы, представляющий совокупность математических действий, которые необходимо выполнить над вектором входной последовательности x размерности m, чтобы получить выходную последовательность y размерности n; X, Y – множества элементов x_j, $j = \overline{1,m}$, и y_i, $i = \overline{1,n}$, которые принадлежат метрическим пространствам E_x и E_y соответственно.

Большинство задач обработки данных в лучшем случае являются слабо корректными в силу их слабой структурированности. В них отсутствует достоверная информация о непротиворечивости ограничений, характере возмущающих воздействий и погрешностях вычислений. В качестве регуляризованного решения задачи можно принять нормальное решение $x_н$, наименее уклоняющееся от некоторого заданного вектора x_0.

Меру уклонения нового решения от старого можно задать квадратом нормы

$$\Omega[x_н - x_0] = \|x_н - x_0\|_n^2. \tag{2}$$

Пусть задан некоторый вектор $x_0 \in X$. Искомый вектор $x_н$ представляет нормальное решение задачи обработки данных (по отношению к x_0), если справедливо соотношение

$$\left\|x_н - x_0\right\|^2 = \min_{x^* \in X} \left\|x^* - x_0\right\|^2, \qquad (3)$$

где x^* – любое решение этой задачи. Из совокупности квазиоптимальных решений с помощью интеллектуальной технологии (способа задания критерия близости) выделяется нормальное решение, наилучшее в смысле выбранной функции уклонения.

Многокритериальные процедуры обработки данных принято относить к классу многократно некорректных задач. Их некорректность возникает из-за некорректности задач локальной оптимизации и из-за процедур принятия многокритериальных решений, в основу которых положен принцип неединственности. Множественность принимаемых решений является скорее достоинством, а не недостатком, поскольку «жесткие» схемы получения единственного решения неадекватны сущности задач многокритериальной оптимизации, а имеющаяся интеллектуальная «свобода» выбора предпочтительного решения из множества эффективных позволяет учесть неопределенность целей и критериев.

4. Обработка данных на основе алгебры регулярных событий

Математическое описание процедур обработки данных можно составить на основе теории конечных автоматов или их алгебраических инвариантов с помощью регулярных выражений формальных языков и грамматик. Регулярные выражения служат удобной альтернативой математическому описанию таких программных компонентов, как программы текстового поиска в анализаторах и программы компонент трансляторов [1, 2, 6].

Регулярные выражения, как и различные конечные автоматы, определяют (задают, представляют) регулярные языки. Регулярные языки в отличие от автоматов определяют допустимые текстовые цепочки данных декларативным способом. Поэтому регулярные выражения используются в качестве входного языка во многих системах обработки текстовых цепочек данных. Различные поисковые системы преобразуют регулярные выражения в детерминированные и недетерминированные конечные автоматы, а такой автомат используют для поиска текстовых цепочек данных в файле.

Регулярные выражения определяются в специальной алгебраической языковой среде регулярных событий, взаимосвязанных набором операций. В качестве регулярных событий может выступать формальный язык (произвольное множество слов или текстовых цепочек), выражения которого задают события над некоторым алфавитом. В качестве операций выступают три оператора регулярных выражений: объединение (дизъюнкция), конкатенация (умножение) и итерация.

Объединение (дизъюнкция) $L \cup M$ двух языков L и M представляет собой множество цепочек данных, содержащихся либо в языке L, либо в языке M, либо в обоих языках. Например, если L = {001, 10, 111} и M = { ε , 001}, то $L \cup M$ = { ε , 001, 10, 111}.

Конкатенация (сцепление или произведение) $L \cap M$ двух языков L и M представляет собой множество цепочек, которые можно образовывать путем дописания к любой цепочке из языка L любой цепочки из языка M. Например, если L = {001, 10, 111} и M = { ε , 001}, то $L \cap M$ = {001, 10, 111, 001001, 10001, 111001}.

Итерация L^* представляет собой множество всех цепочек, которые можно образовать путем конкатенации любого количества цепочек из L. При этом допускаются повторения, т.е. одна и та же цепочка из L может выбираться для конкатенации более одного раза. Если L = {0, 11}, то в итерацию L^* входят цепочки из нулей и четного количества единиц, например, цепочки 011, 11110 или ε .

Если этот формальный язык обозначить через *L*, то его правильно построенные (регулярные) выражения можно назвать *L*-выражениями, а события, которые они задают, – *L*-событиями. Множество *L*-событий для любого языка *L* не более, чем счетное (эквивалентно множеству всех натуральных чисел). Поскольку мощность множества всех событий континуальна (иными словами, мощность множества *L* чисел отрезка $0 \le x \le 1$), то не существует такого языка *L*, для которого все события являются *L*-событиями.

Одна из наиболее изученных алгебр событий – *алгебра регулярных событий*, которая является максимальной и конечно-порождаемой алгеброй, содержащей все конечные события, т.е. события, состоящие из конечного числа слов. В качестве дополнительных операций в алгебре регулярных событий используются операции пересечения и дополнения, относительно которых класс регулярных событий оказывается замкнутым.

Математический аппарат алгебры событий

$$A_{соб} =< A_{оп}, A_{ус} > \qquad (4)$$

базируется на использовании алгебры операторов $A_{оп}$ и алгебры условий $A_{ус}$.

Элементами алгебры операторов $A_{оп}$ являются частичные преобразования (операторы) Y_j некоторого абстрактного множества B, определяющие конкретные акты деятельности по преобразованию входных данных в выходные данные. Кроме основных операторов Y_j, в алгебру операторов $A_{оп}$ входят также два вспомогательных оператора: тождественный оператор E и пустой оператор $\varnothing$.

Элементами алгебры условий $A_{ус}$ являются частичные предикаты (все логические условия) X_i, которые определены на множестве B, принимают значения только $X_i = 1$ или $X_i = 0$ и используются для описания дискретных преобразований. В этом случае множество B называется информационным множеством. Кроме основных условий X_i, в алгебру условий входят два вспомогательных условия: “1” – тождественно-истинное значение и “0” – тождественно-ложное значение.

Если в алгебре событий $A_{соб}$ зафиксировать систему образующих элементов (элементарные операторы и элементарные условия), то элементы алгебры операторов $A_{оп}$ и алгебры условий $A_{ус}$ можно задавать в виде выражений, составленных из этих образующих и операций системы алгебр. Такие выражения принято считать регулярными операторными выражениями и регулярными условными выражениями, а операторы и условия, действующие на множестве B, которые можно задавать таким образом, принято считать регулярными операторами и регулярными условиями.

Выражение, построенное из букв алфавита X (символов элементарных событий) и из символов операций дизъюнкции, умножения и итерации с использованием соответствующим образом круглых скобок, принято считать регулярным выражением алфавита X. Всякое регулярное выражение R определяет некоторым образом событие S (S получается в результате выполнения всех операций, входящих в выражение R). События, определяемые таким образом, принято считать регулярными событиями над алфавитом X.

Представление любого оператора из алгебры операторов $A_{оп}$ через алфавит этой алгебры, содержащий все операторы и условия на множестве B мощности n, является регулярным выражением *R* алгебры событий и обозначается

$$R = (Y_j, X_i, E, \varnothing, 1, 0), \qquad (5)$$

причем условным и линейным операторам регулярных сообщений соответствуют переменные, последовательностям операторов – операции их произведения (конкатенации), параллельно соединенным ветвям – операция дизъюнкции переменных, циклам – операция итерации переменных; условные операторы указываются в скобках для того, чтобы отличить их от линейных операторов.

5. Инварианты регулярных выражений алгебры событий

Граф-схема алгоритма (ГСА). Алгоритм обработки данных можно представить несколькими способами: вербально на содержательном уровне, графически в виде ориентированного графа и аналитически регулярным выражением алгебры событий [1]. Представляет также интерес совместное использование всех трех форм алгоритма обработки данных.

Графическое представление алгоритма состоит в том, чтобы каждый его этап представить геометрической фигурой и соединить стрелками, показывающими последователь-

ность выполняемых операций. Совокупность геометрических фигур и соединительных стрелок принято называть граф-схемой алгоритма обработки данных. Каждой вершине ГСА ставится в соответствие оператор. Для обозначения начала и конца ГСА введены начальный и конечный операторы.

Начальный оператор Y_0 не содержит ни одной входящей дуги, а выходит из него только одна дуга. В конечный оператор Y_k могут входить одна или несколько дуг, а выходить из него не может ни одной дуги. Между начальным Y_0 и конечным Y_k операторами располагаются операторы обработки данных Y_j, которые формируют выходные последовательности сигналов, и логические операторы X_i, которые реализуют логические условия. Вершины операторов обработки данных Y_j изображаются прямоугольниками, а операторы формирования условий X_i – ромбами.

Вершина оператора обработки данных может содержать один или несколько входов и только один выход. Последнее объясняется тем, что после выполнения той или иной операции обработки данных выходная последовательность сигналов передается для выполнения только одной операции. Тем не менее, логическая вершина может иметь один или несколько входов и только два выхода, которые обозначаются, как правило, 1 (да или истинно) и 0 (нет или ложно). Последнее объясняется тем, что после проверки логического условия X_i возможны только два значения выхода $X_i = 1$ или $X_i = 0$.

Граф-схема алгоритма составляется на основе содержательного описания условий работы устройства обработки данных. Сначала на граф-схеме изображается начальная вершина Y_0 с названием «Начало». За ней располагается первая операторная вершина Y_1 с названием «Ввод числовых данных». Затем изображаются остальные вершины граф-схемы алгоритма с операторами обработки данных и логическими операторами, а также все связи между ними. Последней изображается конечная вершина. Если в вершинах записать команды на содержательном уровне, то получим содержательную ГСА. Если перейти от содержательных наименований вершин к символьным обозначениям, то получим более компактную символьную ГСА.

Представление логики процедур обработки данных в виде граф-схем алгоритмов является удобным и наглядным только на первых этапах составления алгоритма. При детальном рассмотрении ГСА занимает много места и требует значительного количества промежуточных преобразований перед вводом в ЭВМ. Поэтому наряду с применением ГСА на практике часто используются и другие нотации для описания логики процедур обработки данных, из которых наиболее известными являются логическая схема алгоритма и регулярные выражения алгебры событий.

Логическая схема алгоритма (ЛСА) представляет собой совокупность операторов обработки данных и логических операторов, соединенных между собой набором интерфейсных дуг, направленных и пронумерованных таким образом, чтобы реализовать определенную процедуру обработки данных. Если порядок выполнения операторов в ЛСА строго фиксированный, то такой алгоритм является линейным. Если порядок выполнения операторов зависит от условий, то такой алгоритм является разветвленным.

Операторы обработки данных обозначим прописными буквами латинского алфавита $(A, B, C, \ldots)$ или одной и той же прописной буквой с различными индексами $(Y_1, Y_2, Y_3, \ldots$ или $Y_i,\ i = 1, 2, 3, \ldots)$. Логические операторы, обозначенные малыми буквами латинского $(x_1, x_2, \ldots, x_n)$ или греческого $(\alpha_1, \alpha_2, \ldots, \alpha_n)$ алфавита, могут приобретать только одно из двух значений: «1» или «0». Каждое i-е логическое условие α_i начинается и заканчивается нумерованной стрелкой, направленной соответственно вверх $^{i}\uparrow$ и вниз $\downarrow^{i}$, или нумерованной круглой открывающейся $_{\alpha_i}($ и закрывающейся скобкой $)^{\alpha_i}$.

Стрелка-вверх $^{i}\uparrow$ или открывающаяся круглая скобка $_{\alpha_i}($ ставится непосредственно справа возле логического условия, которое обозначается как α_i, x_i или p_i. Номер условия, стоящий непосредственно возле стрелки-вверх или круглой открывающейся скобки, пока-

зывает оператор, куда должно быть передано управление алгоритмом в случае, если логическое условие $x_i = 0$. Номер условия, стоящий возле стрелки-вниз или круглой закрывающейся скобки, показывает оператор, куда должно быть передано управление. Этот оператор стоит непосредственно после стрелки-вниз или круглой закрывающейся скобки. Если $x_i = 1$, то после логического условия выполняется оператор обработки данных или логический оператор, стоящий непосредственно справа.

Работа алгоритма начинается с выполнения самого левого оператора ЛСА, после чего определяется следующий справа оператор. Если таковым является оператор обработки данных, то осуществляется его выполнение. В случае, если этот оператор является логическим, то возможны два варианта: 1) когда логическое условие равно единице, выполняется оператор, стоящий следующим справа; 2) когда логическое условие равно нулю, выполняется оператор, на который указывает стрелка-вверх, расположенная после этого условия.

Среди логических операторов может находиться безусловный оператор ω, который не требует проверки логического условия. Такой оператор передаёт выполнение операции другому оператору, номер которого i указан при стрелке-вверх $^{i}\uparrow$ или при открывающейся круглой скобке $\alpha_i($.

Порядок выполнения операторов в ЛСА определяется последовательностью их размещения в выражении (5), а также распределением входящих в них пронумерованных стрелок и круглых скобок. Работа ЛСА заканчивается, когда последний из выполнявшихся операторов содержит указание на остановку алгоритма или когда на некотором этапе работы не обнаруживается оператор, который должен выполняться.

Математическая модель задачи обработки данных для управления производством, построенная в классе регулярных выражений алгебры событий, имеет вид [1]:

$$R = Y_1 \cdot Y_2 \cdot_{\alpha 1} (Y_4 \cdot Y_5 \vee Y_3)^{\alpha 1} \cdot_{\alpha 2} (Y_6 \cdot Y_7 \cdot Y_8 \vee Y_9 \cdot Y_{10} \cdot Y_{11})^{\alpha 2} \cdot Y_{12}, \qquad (6)$$

где Y_1 – оператор обработки данных (в дальнейшем просто оператор) определения заказов на текущий месяц; Y_2 – оператор группирования заказов по номенклатуре; Y_3 – оператор включения заказов в месячный план; Y_4 – оператор определения планового объема производства по номенклатуре на конкретные сутки; Y_5 – оператор анализа выполненных заказов; Y_6 – оператор установления заказов, по которым выполнена отгрузка; Y_7 – оператор учета остатков незавершенного производства; Y_8 – оператор определения первоочередных заказов; Y_9 – оператор частичного или полного исключения трудоемких заказов; Y_{10} – оператор определения первоочередных заказов; Y_{11} – оператор определения совокупности заказов на месяц; Y_{12} – оператор выполнения плановых заказов производства с учетом отклонений; α_1 – условный оператор проверки выполнения заказов на начало месяца; α_2 – условный оператор проверки существования отклонений от плановых показателей.

6. Применение регулярных выражений в системах обработки данных

Важным требованием к текстовым переводчикам и трансляторам является адекватность переводов, которая определяется семантической полнотой, точностью и стилистической эффективностью, включающей в себя принцип соответствия текста перевода стилистическим нормам языка перевода. По этим критериям оценивается качество перевода. На сегодняшний день проблема оценивания качества перевода еще не формализована. Практические оценки качества перевода определяются по эмпирико-интуитивным соотношениям, базирующимся на личном профессиональном опыте.

Для повышения качества процедур обработки данных применяется текстовая регуляризация, которая может распространяться на несколько уровней. Существует трехсторонняя взаимосвязь между регулярными выражениями, конечными автоматами и регулярными грамматиками языка. Описание языка часто базируется на использовании регулярных выражений, на основе которых можно построить распознающие конечные автоматы – лексические анализаторы языка.

Такие лексические анализаторы преобразовывают последовательности символов исходного файла во множество элементов, типы которых определяются видом регулярных выражений.

Ниже приведен пример программы на flex для разбиения текста русского языка на слова, аббревиатуры, числа, классы знаков пунктуации [2]:

```
digit          [0-9]
intconst       [+\-]?{digit}+
realconst      [+\-]?{digit}+\.{digit}+(e[+\-]?{digit}+)?
letter         [а-я]
capital [А-Я]
word  ({letter}|{ capital}){letter}*|{letter}+-{letter}+
abbreviation {capital}*
punctuation {,|:|;|-| (|)|{|}| [|]}
stops  {.|!|?}
%%
{intconst}   Процедура формирования лексемы – целое число;
{realconst}   Процедура формирования лексемы – дробное число;
{word}   Процедура поиска слова в словаре и формирования лексемы – слово;
{abbreviation}   Процедура формирования лексемы – аббревиатура;
{punctuation}   Процедура формирование лексемы – знак препинания;
{stops}   Процедура формирование лексемы – признак конца предложения;
%%
```

В качестве инструментов обработки данных могут использоваться различные формальные представления, удовлетворяющие ограничениям, налагаемым на данное представление. Ниже приведен пример регулярного представления языковой конструкции «Программный продукт строит график функции» на основе семантических функций:

$$f_1(V_3^2(y), V_1^1(x), f_3^4(V_1^4(z), V_1^4(u))), \tag{7}$$

где f_i - функции, определяющие отношения в предложении; $V_j(x)$ - функция, определяющая j значение аргумента x.

Важным вопросом процедуры регуляризации является эквивалентность формального представления исходной языковой конструкции выходным представлениям. Пусть исходная языковая конструкция A преобразуется процедурой регуляризации R в формальное представление B:

$$A \xrightarrow{R} B. \tag{8}$$

Предположим, что существует формальная процедура R', позволяющая получать из формального представления языковую конструкцию:

$$B \xrightarrow{R'} A'. \tag{9}$$

Важным при этом является вопрос об эквивалентности языковых конструкций A и A'. Ответ на этот вопрос зависит от симметричности преобразований R и R'. Симметричность преобразований могут обеспечить эквивалентные структуры анализа и синтеза на уровне синтаксического и семантического анализа. Главная проблема состоит в адекватности отражения семантико-синтаксических связей предложения на формальном уровне, их использование на этапах синтеза и анализа обработки данных.

7. Гипертекстовая регуляризация обработки данных

В основу системы контроля знаний дистанционного обучения, как правило, закладывается принцип текстового диалога, когда вопрос задается обучающей стороной, а ответ формируется обучаемой стороной по определенным правилам. В системах контроля знаний важное место занимает проблема однозначного смыслового толкования и восприятия задаваемого вопроса и получаемого ответа. Для решения этой проблемы в формализованных системах используются процедуры гипертекстовой регуляризации [3].

Для математического описания процедур обработки данных в системе контроля знаний предложено использовать регулярные выражения формальных языков и грамматик. Регулярные выражения служат удобным описанием программных компонентов типа программ текстового

поиска и программ текстового перевода. Регулярные выражения строятся на основе алгебраических законов, которые определяют структуру данных с помощью текстовых цепочек.

На сегодняшний день разработано большое количество языков гипертекстовой разметки для конкретных процедур обработки данных, например, для передачи данных, перекодировки текстов, банковского обмена данными и других целей. Если каждый из формальных языков гипертекстовой разметки обозначить через L, то его регулярные выражения принято называть L-выражениями, а события, которые они задают, – L-событиями.

Поскольку мощность множества всех событий континуальна (иными словами, мощность множества L чисел отрезка $0 \le x \le 1$), то не существует такого языка L, для которого все события являются L-событиями. Поэтому процедура обработки данных системы контроля знаний должна обладать регуляризованным «мультиязыком», составляющие которого отражают конкретные формальные языки контролируемых знаний.

8. Выводы

Анализ процедур обработки данных показал, что желание повысить качество принимаемых решений в условиях неопределенности требует привлечения дополнительной информации. В процессе повышения определенности одна форма неопределенности переходит в другую форму с меньшим количеством неопределенности до тех пор, пока неопределенность “более высокого порядка” не станет соизмеримой с “фоновым” значением шумов вычислений. Дилемма определенности-неопределенности требует разумного компромисса в повышении определенности в целях устранения факторов неопределенности.

Научная новизна выполненных исследований состоит в том, что в качестве математического описания процедур обработки данных предложены регулярные выражения, которые позволяют создать регулярные языки с заданными свойствами и использовать их при обработке текстовых цепочек. Затем происходит преобразование регулярных выражений в конечные автоматы, а автоматы осуществляют поиск требуемых текстовых цепочек в файле.

Практическая ценность выполненных исследований состоит в том, что разработаны инструментальные средства обработки данных в системах искусственного интеллекта с помощью применения семантических функций для анализа языковых конструкций. Усовершенствованы также процедуры регуляризации семантического анализа данных на основе применения логического вывода на семантических функциях, что позволило выбирать единственное значение для многозначных слов и получать единственное формальное представление языковых конструкций.

Список литературы: 1. *Авраменко В.П., Калачева В.В., Кротюк И.Г.* Моделирование оперативного управления бизнес-процессами с помощью регулярных выражений алгебры событий // АСУ и приборы автоматики. 2004. Вып 129. С. 112 – 119. 2. *Авраменко В.П., Валенда Н.А.* Регуляризация процедур обработки данных в системах искусственного интеллекта // Бионика интеллекта. 2007. Вып. 4 (70). С. 95 - 98. 3. *Авраменко В.П., Штангей С.В.* Гипертекстовая регуляризация в системах контроля знаний дистанционного обучения // Управління розвитком. Збірник наукових робіт. ХНЕУ. 2006. № 2. С. 54 - 55. 4. *Тихонов А.Н., Арсенин В.Я.* Методы решения некорректных задач. М.: Наука, 1979. 288 с. 5. *Авраменко В.П.* Управление производством в условиях неопределенности. К.: УМК ВО, 1992. 48 с. 6. *Хопкрофт Дж., Мотвани Р., Ульман Дж.* Введение в теорию автоматов, языков и вычислений. М.: Вильямс, 2002. 528 с.

Поступила в редколлегию 03.03.2008

Авраменко Валерий Павлович, д-р техн. наук, профессор кафедры ИУС ХНУРЭ. Научные интересы: интеллектуальные информационные технологии и методы регуляризации. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-451.