УДК 004.89

Структура гибридной интеллектуальной информационной системы на основе метаграфов

The structure of the hybrid intelligent information system based on metagraphs

1. **Черненький В.М. (Chernenkiy V. M.), д.т.н., профессор, зав. кафедрой «Системы обработки информации и управления» МГТУ им. Н.Э. Баумана (Bauman Moscow State Technical University), iu5vmch@rambler.ru**
2. **Терехов В.И. (Terekhov V. I.), доцент кафедры «Системы обработки информации и управления» МГТУ им. Н.Э. Баумана (Bauman Moscow State Technical University), terekchow@bmstu.ru**
3. **Гапанюк Ю.Е. (Gapanyuk Yu.E.), доцент кафедры «Системы обработки информации и управления» МГТУ им. Н.Э. Баумана (Bauman Moscow State Technical University), gapyu@bmstu.ru**

Аннотация. Статья посвящена структуре гибридной интеллектуальной информационной системы (ГИИС) на основе многоагентного подхода. Рассматривается обобщенная структура ГИИС на основе модулей сознания и подсознания. Приводятся частные случаи структуры ГИИС для различных вариантов информационных систем. Для реализации ГИИС предлагается использовать многоагентный подход на основе холонической организации. Рассматриваются требования к агенту холонической многоагентной системы. Для описания структуры агента предлагается использовать метаграф. Рассматривается формализованная модель метаграфа. Вводится понятие информационного элемента метаграфа (ИЭМ). Рассматриваются основные операции над метаграфами на основе ИЭМ. Приводится описание холонической системы агентов в статике на основе метаграфа. Для статического

описания предлагается использовать агенты-функции, метаграфовые агенты и контейнерные агенты. Приводится описание холонической системы агентов в динамике на основе метаграфа. Для динамического описания предлагается использовать динамические метаграфовые агенты.

Abstract. The article is devoted to the structure of the hybrid intelligent information system (HIIS) based on multi-agent approach. Generalized structure of HIIS based on modules of consciousness and subconsciousness is discussed. Special cases of the structure of HIIS for different variants of information systems are given. For HIIS implementation usage of multi-agent approach based on holonic organization is proposed. Requirements for agent of holonic multi-agent system are addressed. To describe the structure of the agent is proposed to use the metagraph. The formal model of metagraph is discussed. The concept of information element of a metagraph (IEM) is introduced. Basic operations on metagraphs based on IEM are discussed. Description of holonic multi-agent system in statics based on metagraph is given. For description in statics usage of agents-functions, metagraph agents and container agents is proposed. Description of holonic multi-agent system in dynamics based on metagraph is given. For description in dynamics usage of dynamic metagraph agents is proposed.

Ключевые слова. гибридная интеллектуальная информационная система (ГИИС), сознание и подсознание информационной системы, многоагентный подход, холоническая организация, метаграф.

Keywords: hybrid intelligent information system (HIIS), conscious and subconscious of information system, multi-agent approach, holonic organization, metagraph.

Введение

Для построения интеллектуальных систем используются продукционные правила, нейронные сети, нечеткая логика, эволюционные методы и др. Нейронные сети, нечеткую

логику и эволюционные методы традиционно объединяют под термином «мягкие вычисления».

При этом можно отметить явную тенденцию к совместному использованию различных методов для решения различных классов задач. Это привело к появлению такого направления как «гибридные интеллектуальные системы» (ГИС). Основополагающими работами в области ГИС можно считать работы А.В. Колесникова [1, 2].

В настоящее время интеллектуальные системы, как правило, не разрабатываются отдельно, но встраиваются в виде модулей в традиционные информационные системы для решения задач, связанных с интеллектуальной обработкой данных и знаний. Такую комбинированную систему назовем гибридной интеллектуальной информационной системой (ГИИС).

Объектом исследования данной статьи являются гибридные интеллектуальные информационные системы. ГИИС обладает следующими особенностями:

- сочетает различные методы, используемых для построения интеллектуальных систем, и в этом смысле является ГИС;
- сочетает интеллектуальные методы с традиционными методами, используемыми для разработки данных в информационных системах, и в этом смысле является комбинацией ГИС и информационной системы, предназначенной для обработки данных.

Таким образом, под ГИИС будем понимать информационную систему, которая использует комбинацию традиционных методов обработки данных и интеллектуальных методов.

Предметом исследования данной статьи является использование метаграфов для описания структуры ГИИС. Но для того, чтобы использовать метаграфы, необходимо предварительно построить несколько промежуточных моделей.

Ключевым вопросом является вопрос о том, каким образом реализовать принцип гибридности. Здесь отправной точкой для авторов послужила работа Н.Г. Ярушкиной [3]. В ней сформулирован следующий принцип гибридности [3, с. 20-21]: «В литературе встречаются схемы гибридизации нейроинформатики и ИИ, построенные по следующему принципу: правое полушарие – нейрокомпьютер; левое полушарие – основанная на знаниях система, а вопрос лишь в их взаимодействии или балансе право- и лево-полушарности. В реальном поведении человека невозможно разделить восприятие и логическую обработку, поэтому более успешной представляется схема глубинной интеграции».

Таким образом, ГИИС должна сочетать элементы системы, построенной на основе мягких вычислений, и системы построенной на обработке данных и знаний.

На взгляд авторов статьи, метафора право- и лево-полушарности здесь не совсем точна, скорее стоит говорить о «подсознании» и «сознании» гибридной ИС. «Подсознание» строится на основе мягких вычислений, а «сознание» на основе обработки данных и знаний.

1. Обобщенная структура ГИИС

Обобщенная структура ГИИС, построенная на основе «сознания» и «подсознания» представлена на рис. 1.

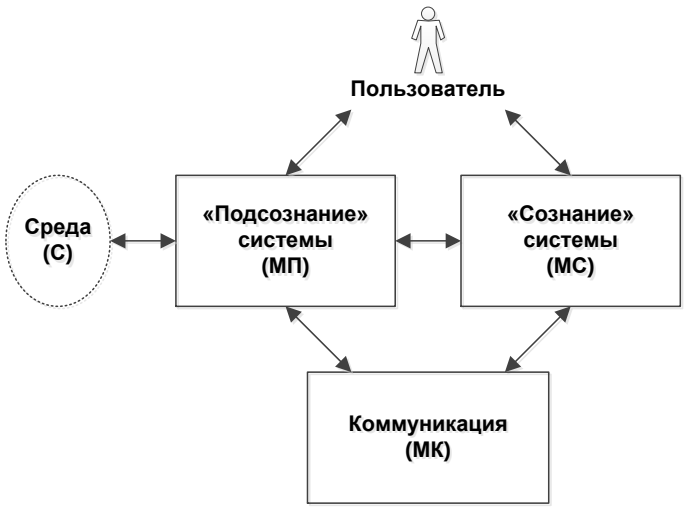

Рис. 1. Обобщенная структура ГИИС.

Основой системы являются «подсознание» системы (модуль подсознания, МП) и «сознание» системы (модуль сознания, МС). «Подсознание» связано со средой, в которой функционирует ГИИС.

Основной задачей МП является обеспечение взаимодействия ГИИС со «средой», или «выживание» ГИИС в среде.

Поскольку среда может быть представлена в виде набора непрерывных сигналов, то в качестве методов обработки данных «подсознания» хорошо подходят методы, основанные на нейронных сетях и нечеткой логике, в том числе и комбинированные нейронечеткие методы.

Модель данных «подсознания» максимально приближена к «понятийной системе» среды, представляет собой набор данных, который позволяет максимально эффективно взаимодействовать со средой. Часть этих данных может не иметь «физического смысла» с точки зрения МС, однако позволяет МП взаимодействовать со средой с нужной производительностью.

«Сознание» ГИИС строится на принципах обработки данных и знаний. Обработка данных в МС может вестись на основе традиционных языков программирования или технологии workflow. Однако, в последнее время, все большую популярность приобретает подход на основе продукционных правил. Раньше данный подход использовался для принятия решения в экспертных системах, но в настоящее время на основе правил пишутся обычные программы. Такой подход называется программированием на основе правил (rule-based programming). В частности, в продукте для разработки интеллектуальных систем Drools [4] возможна комбинированная обработка данных как с использованием workflow-подхода, так и с использованием rule-based-подхода.

Отметим, что в зависимости от особенностей предметной области правила могут быть нечеткими или вероятностными, что вносит в МС элементы МП. Это одно из проявлений принципа холоничности, о котором будет сказано далее.

К достоинствам подхода на основе правил можно отнести гибкость, так как в этом случае программа не кодируется жестко, а фактически «выводится» из правил на основе данных. К недостаткам можно отнести возможность зацикливания правил, а также сложность обработки большого объема правил. В настоящее время для обработки большого объема правил используется алгоритм RETE (разработанный Ч. Форджи [5]) и его модификации.

В качестве модели данных МС используются модели «онтологического» класса. Это могут быть классические онтологии, разработанные в рамках технологии Semantic Web (стандарты RDF, RDFA, OWL, OWL2). Также к моделям этого класса можно отнести (возможно, с некоторыми ограничениями) и классическую объектно-ориентированную модель. Классическая модель ООП обладает рядом ограничений по сравнению с онтологиями Semantic Web, но на практике именно она используется для моделирования предметных областей в большинстве современных информационных систем. С использованием средств объектно-реляционного отображения (Object-Relational Mapping, ORM) обеспечивается хранение элементов этой модели в реляционных СУБД.

Как правило, большинство моделей «онтологического» класса обладает следующими свойствами:

- явное выделение «абстрактных» понятий (классов) и «конкретных» понятий (объектов, экземпляров);
- возможность работы как с абстрактными понятиями (например, наследование классов) так и с конкретными понятиями (например, создание объекта класса);
- возможность работы как с непрерывными типами данных (целые, действительные числа), так и возможность перечисления объектов, относящихся к классу (перечисляемый тип в ООП).

МС, базируясь на моделях «онтологического» класса, выполняет следующие функции:

- обработка данных и знаний на основе модели данных «онтологического» типа;

- логический контроль и проверка непротиворечивости данных, поступающих от МП;
- реализация функций ввода и вывода для среды (посредством МП), для модуля коммуникации и для взаимодействия с пользователем.
- реализация функции поддержки принятия решений (в этом случае МС выполняет функцию СППР);
- реализация функции прогноза;
- реализация функции планирование действий системы (автоматизированное планирование).

Отметим, что задача хранения требуемых данных решается отдельно на уровне МС и на уровне МП. Мы предполагаем, что на уровне обобщенной структуры соответствующие хранилища «встроены» в МС и МП, поэтому хранилища не представлены явно на рис. 1.

С точки зрения коммуникации в ГИИС возможны следующие варианты или их комбинации:

1. Коммуникация осуществляется через среду. МП читает данные из среды, преобразует и передает в МС. МС осуществляет логическую обработку и возвращает результаты обработки в МП. МП записывает результирующие данные в среду, откуда они могут быть прочитаны другими ГИИС.
2. Для коммуникации с другими ГИИС используется модуль коммуникации (МК). В зависимости от решаемых задач с МК может взаимодействовать МС (что характерно для традиционных информационных систем) или МП (что более характерно для систем на основе мягких вычислений).
3. Взаимодействие с пользователем также может осуществляться через МС (что характерно для традиционных информационных систем) или через МП (что может быть использовано, например, в автоматизированных тренажерах).

2. Частные случаи структуры ГИИС

На основе обобщенной структуры ГИИС могут быть рассмотрены частные случаи такой структуры, которые соответствуют конкретным ГИИС. Примеры таких частных случаев представлены на рис. 2. Модули на рис. 2 представлены в виде обозначений, соответствующих рис. 1.

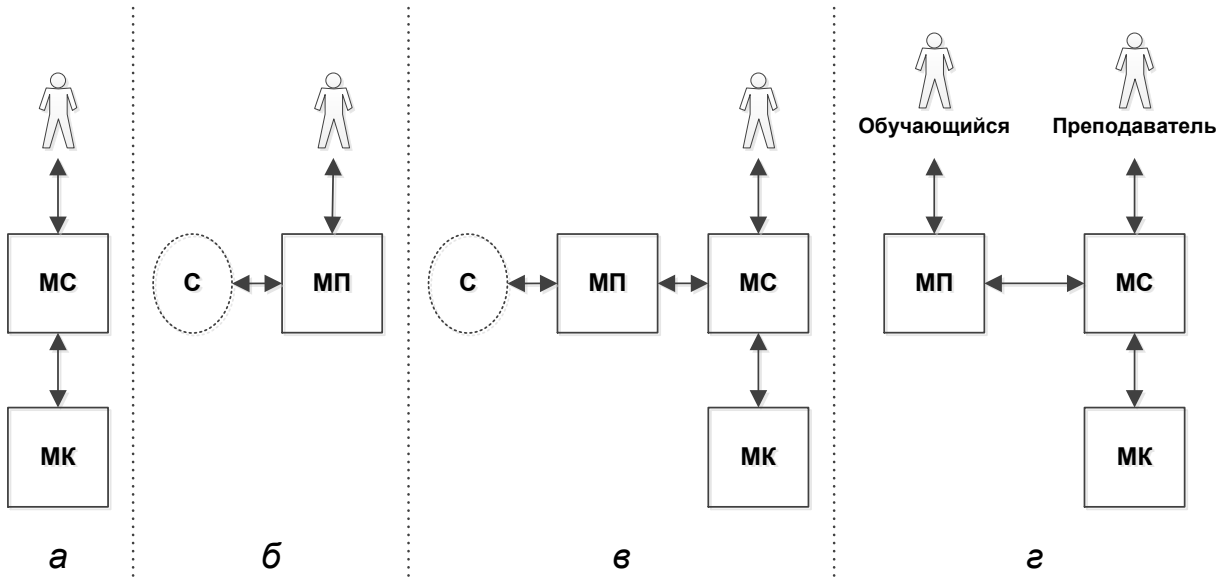

Рис. 2. Частные случаи структуры ГИИС.

Рисунку 2а соответствует классическая информационная система, в которой осуществляется только обработка данных и знаний, реализуется коммуникация с другими системами и взаимодействие с пользователем.

Рисунку 2б соответствует, в частности, простейшая система распознавания сигналов, поступающих из среды, с помощью МП. Сигналы могут иметь различную природу. Это может быть система распознавания музыкальной партитуры по звуковому сигналу, система распознавания элементов в видеопотоке и др. Данная система является простейшей, так как в ней отсутствует МС, который должен осуществлять коррекцию логических ошибок. Здесь эта задача возлагается на МП, что может приводить к сложным правилам при распознавании и обработке сигналов.

Рисунку 2в может соответствовать большое количество конкретных ГИИС. Приведем несколько примеров:

1. Усовершенствованная система распознавания сигналов. Сигналы выделяются из среды с помощью МП и преобразуются в элементы онтологии, которые обрабатывает МС. МС осуществляет дополнительный логический контроль. Например, для системы распознавания музыкальной партитуры, МП выделяет ноты из входного сигнала (здесь ноты являются элементами онтологии), а МС может скорректировать неверно распознанную ноту на основе правил музыкальной гармонии. В этом случае модуль коммуникации может не использоваться.
2. Медицинская система функциональной диагностики. В этом случае роль среды выполняют сигналы от медицинских приборов. МП преобразует сигналы в элементы онтологии, МС на основе продукционных правил может осуществлять поддержку принятия решений. Пользователем является врач, модуль коммуникации может осуществлять коммуникацию с другими информационными системами.
3. АСУТП (автоматизированная система управления технологическими процессами), использующая методы мягких вычислений. В этом случае роль среды выполняют наблюдаемые параметры технологического процесса. МП преобразует сигналы в элементы онтологии, МС на основе продукционных правил может осуществлять логический контроль поступающей информации и поддержку принятия решений. Пользователем является оператор АСУТП, модуль коммуникации может осуществлять коммуникацию с другими информационными системами.

Рисунку 2г соответствует автоматизированная система виртуального тренажера. В этом случае действия обучающегося по управлению тренажером поступают на МП. МП преобразует сигналы в элементы онтологии, МС на основе продукционных правил может осуществлять логический контроль поступающей информации, поддержку принятия

решений и выдачу информации преподавателю. Модуль коммуникации может быть использован в случае группы тренажеров.

3. Использование холонической многоагентной системы для реализации ГИИС

Следующим вопросом является вопрос о том, каким образом реализовать рассмотренную выше структурную схему. Нами предлагается использовать подход на основе многоагентной системы (МАС). В [6] приводится несколько определений МАС и программных агентов, мы адаптируем данные определения применительно к ГИИС.

Под программным агентом будем понимать программный модуль, который выполняется в виде автономной задачи (не зависит от других агентов), способен обмениваться информацией со средой и другими агентами. Под МАС будем понимать систему однородных или разнородных агентов, функционирующих в среде.

Для реализации ГИИС наиболее интересным представляется подход на основе холонической многоагентной системы (холонической МАС). Такой класс систем рассмотрен в работе В.Б. Тарасова [6]. В соответствии с определением [6, с. 234] холон – это «целое, рассматриваемое в то же время как часть целого». Этот подход соответствует существовавшему ранее понятию «вложенности».

С точки зрения данного подхода, рассмотренные компоненты, такие как МП, МС, МК являются агентами. В тоже время они являются частями системы, которая в свою очередь является агентом.

При этом МП является сложной структурой, которая включает агенты нижнего уровня, каждый из которых может в свою очередь включать МП, МС, МК, предназначенные для решения конкретных задач данного агента. Не смотря на то, что агент нижнего уровня находится в составе МП, он может включать в свою структуру МС, предназначенный для решения задач МП более высокого уровня. Поэтому с точки зрения данного подхода нет

ничего удивительного в том, что в МС могут использоваться нечеткие продукционные правила, а в МП входят «классические» модули обработки данных.

Отметим, что хотя для решения задач МС могут быть использованы методы обработки правил, а для решения задач МП нейронечеткие методы, все эти методы являются статическими. То есть предполагается, что логические правила, структура нейросети и т.д. задаются на этапе проектирования ГИИС и не изменяются в процессе работы.

Однако, подобный статический подход является недостаточным по следующим причинам:

- нет возможности использования эволюционных методов (генетические алгоритмы, генетическое программирование и др.);
- в настоящее время для разработки ГИС начинают активно использоваться самоорганизующиеся нейронные сети (в частности такие топологии как SOINN[7], hyperNEAT[8]), их использование предполагает динамическое изменение топологии нейронной сети во время работы.
- нет возможности использования других подходов, связанных с изменением порядка действий, таких как динамические workflow, алгоритмы автоматизированного планирования.

Сформулируем основные требования к холонической МАС, предназначенной для реализации ГИИС:

Требование 1. Агент должен реализовывать правила работы для МП или для МС.

Агент может быть аналогом программной процедуры, которая вычисляет функцию активации нейрона. Может быть реактивным агентом, который реализует поведение на основе заданных правил. Может быть проактивным агентом, который реализует интеллектуальные алгоритмы планирования действий и взаимодействия с другими агентами.

Требование 2. Агенты должны поддерживать принцип холонической организации. То есть агент может быть построен как структура из агентов нижнего уровня, которые агент

считает «элементарными», но которые в свою очередь могут состоять из агентов более низкого уровня.

Требование 3. Для реализации свойства динамичности должна существовать возможность перестройки как структуры связей между агентами, так и внутренней структуры самого агента.

Отметим, что поскольку речь идет о холонической организации агентов, то возможны случаи, когда изменение внутренней структуры самого агента предполагает изменение связей между входящими в его состав агентами нижнего уровня.

Наиболее близким аналогом третьего требования в традиционных языках программирования является принцип самоотображаемости. Самоотображаемость (англ. homoiconicity) – это способность языка программирования анализировать программу на этом языке как структуру данных. Термин «самоотображаемость» часто используют вместе с термином «метапрограммирование». Самоотображаемость предполагает анализ программ как структур данных, а метапрограммирование предполагает генерацию на основе этих структур данных других программ. В настоящее время наиболее известным языком, который поддерживает самоотображаемость на уровне синтаксиса, является язык Lisp, у которого существует большое количество диалектов. Однако, практически в любом языке программирования самоотображаемость может быть реализована надъязыковыми средствами с использованием библиотек для работы с абстрактным синтаксическим деревом (Abstract Syntax Tree, AST).

Рассмотренные требования не являются противоречивыми. Однако сложность их реализации состоит в том, что должна быть использована концепция, которая:

- поддерживает сложные иерархические связи (в том числе альтернативные иерархии);
- может быть полностью детализирована до элементов самого нижнего уровня;
- может быть модифицирована в процессе работы.

В основу такой концепции мы предлагаем положить использование метаграфов [9].

4. Формализованная модель метаграфа

В данном разделе кратко рассмотрим формализованную модель метаграфа в соответствии с [9].

$$MG = \langle V, MV, E, ME \rangle,$$

где MG – метаграф; V – множество вершин метаграфа; MV – множество метавершин метаграфа; E – множество ребер метаграфа; ME – множество метаребер метаграфа.

Вершина метаграфа характеризуется множеством атрибутов:

$$v_i = \{atr_k\}, v_i \in V,$$

где v_i – вершина метаграфа; atr_k – атрибут.

Ребро метаграфа характеризуется множеством атрибутов, исходной и конечной вершиной и признаком направленности:

$$e_i = \langle v_S, v_E, eo, \{atr_k\} \rangle, e_i \in E, eo = true \mid false,$$

где e_i – ребро метаграфа; v_S – исходная вершина (метавершина) ребра; v_E – конечная вершина (метавершина) ребра; eo – признак направленности ребра (eo=true – направленное ребро, eo=false – ненаправленное ребро); atr_k – атрибут.

Фрагмент метаграфа:

$$MG_i = \{ev_j\}, ev_j \in (V \cup E \cup MV \cup ME),$$

где MG_i – фрагмент метаграфа; ev_j – элемент, принадлежащий объединению множеств вершин (метавершин) и ребер (метаребер) метаграфа.

Таким образом, фрагмент метаграфа в общем виде может содержать произвольные вершины (метавершины) и ребра (метаребра) без ограничений. Ограничения вводятся на фрагменты метаграфа, входящие в метавершину и метаребро.

Метавершина метаграфа:

$$mv_i = \langle \{atr_k\}, \{ev_j\} \rangle, mv_i \in MV, ev_j \in (V \cup E^{eo=false} \cup MV \cup ME^{eo=false}),$$

где mv_i – вершина метаграфа; atr_k – атрибут, ev_j – элемент, принадлежащий объединению множеств вершин (метавершин) и ребер (метаребер) метаграфа.

Таким образом, метавершина в дополнение к свойствам вершины включает вложенный фрагмент метаграфа. При этом ребра и метаребра этого фрагмента могут быть только ненаправленными, eo=false.

Метаребро метаграфа:

$$me_i = \langle v_S, v_E, eo, \{atr_k\}, \{ev_j\} \rangle, e_i \in E, eo = true \mid false,$$

$$ev_j \in (V \cup E^{eo=true} \cup MV \cup ME^{eo=true}),$$

где me_i – метаребро метаграфа; v_S – исходная вершина (метавершина) ребра; v_E – конечная вершина (метавершина) ребра; eo – признак направленности метаребра (eo=true – направленное метаребро, eo=false – ненаправленное метаребро); atr_k – атрибут; ev_j – элемент, принадлежащий объединению множеств вершин (метавершин) и ребер (метаребер) метаграфа.

Таким образом, метаребро в дополнение к свойствам ребра включает вложенный фрагмент метаграфа. При этом ребра и метаребра этого фрагмента могут быть только направленными, eo=true.

Определения метавершины и метаребра являются рекурсивными, так как элементы ev_j могут быть в свою очередь метавершинами и метаребрами.

Наличие у метавершин собственных атрибутов и связей с другими вершинами является важной особенностью метаграфов. Это соответствует принципу эмерджентности, то есть приданию понятию нового качества, несводимости понятия к сумме его составных частей. Фактически, как только вводится новое понятие в виде метавершины, оно «получает право» на собственные свойства, связи и т.д., так как в соответствии с принципом эмерджентности

новое понятие обладает новым качеством и не может быть сведено к подграфу базовых понятий.

Также в работе [9] предлагается представить метаграф в виде совокупности информационных элементов метаграфа (ИЭМ):

$$MG = \{ИЭМ_n\},$$

$$ИЭМ_n = \langle id, NM, VAL, RL, \{link_i\}, \{atr_j\}\rangle, RL \in \{RL_V, RL_{MV}, RL_R\},$$

где $ИЭМ_n$ – информационный элемент метаграфа; id – уникальный идентификатор элемента; NM – наименование элемента; VAL – значение элемента; RL – роль элемента; $link_i$ – ссылка на другой ИЭМ; atr_j – атрибут; RL_V – роль элемента «вершина»; RL_{MV} – роль элемента «метавершина»; RL_R – роль элемента «ребро».

На основе концепции ИЭМ могут быть предложены основные операции над метаграфами.

5. Основные операции над метаграфами

Для динамической работы с метаграфами введем следующие основные операции над метаграфами и их элементами OP^{MG}.

1) Создание нового ИЭМ:

$$ИЭМ = ИЭМ(\langle NM, VAL, RL, \{link_i\}, \{atr_j\}\rangle).$$

В результате выполнения операции создается новый ИЭМ с заданными параметрами, уникальный идентификатор id генерируется автоматически.

2) Добавление ссылки на ИЭМ:

$$ИЭМ_i = ИЭМ_i + link_j(ИЭМ_j).$$

В результате выполнения операции к информационному элементу $ИЭМ_i$ добавляется ссылка на информационный элемент $ИЭМ_j$.

3) Удаление ссылки на ИЭМ:

$$ИЭМ_i = ИЭМ_i - link_j(ИЭМ_j).$$

В результате выполнения операции у информационного элемента $ИЭМ_i$ удаляется ссылка на информационный элемент $ИЭМ_j$.

4) Добавление атрибута:

$$ИЭМ_i = ИЭМ_i + atr_j.$$

В результате выполнения операции к информационному элементу $ИЭМ_i$ добавляется атрибут atr_j.

5) Удаление атрибута:

$$ИЭМ_i = ИЭМ_i - atr_j.$$

В результате выполнения операции у информационного элемента $ИЭМ_i$ удаляется атрибут atr_j.

6) Добавление информационного элемента к метаграфу:

$$MG_2 = MG_1 + ИЭМ_i.$$

В результате выполнения операции к метаграфу добавляется ИЭМ.

7) Удаление информационного элемента из метаграфа:

$$MG_2 = MG_1 - ИЭМ_i.$$

В результате выполнения операции у метаграфа удаляется ИЭМ.

8) Создание переменной-ссылки на фрагмент метаграфа:

$$VAR_i = ref(MG_j).$$

В результате выполнения операции создается переменная, которая содержит ссылку на фрагмент метаграфа.

9) Создание переменной-копии фрагмента метаграфа:

$$VAR_i = copy(MG_j).$$

В результате выполнения операции создается переменная, которая содержит копию фрагмента метаграфа.

10) Поиск ИЭМ по заданным параметрам:

$$MG_k = find\left(\left\langle id, NM, VAL, RL, \{link_i\}, \{atr_j\}\right\rangle\right).$$

В качестве параметров поиска могут использоваться любые параметры ИЭМ. В результате выполнения операции формируется метаграф, который представляет собой множество найденных ИЭМ.

11) Оператор цикла:

$$foreach\left(MG_i\right) \rightarrow \left\{OP_j^{MG}\right\}.$$

В цикле для каждого ИЭМ, входящего в состав метаграфа MG_i, выполняется заданное множество операторов OP_j^{MG}. Для связи текущего обрабатываемого ИЭМ с выполняемыми в цикле действиями могут использоваться переменные.

12) Оператор создания функции:

$$function\left(\{FP_i\}\right) \rightarrow \left\{OP_j^{MG}\right\}.$$

Функция с заданным множеством параметров FP_i позволяет сгруппировать множество операторов OP_j^{MG}. Параметры функции могут использоваться в качестве параметров операторов OP_j^{MG}. В качестве примера можно рассмотреть функцию, которая принимает на вход элементы-вершины и формирует элемент-связь между этими вершинами.

Предложенные операторы могут быть достаточно просто реализованы в виде API (программного интерфейса) практически в любом современном языке программирования. При этом можно использовать уже готовые языковые механизмы работы с переменными и создания функций.

6. Метаграфы для описания холонической системы агентов в статике

Определим множество агентов системы:

$$AG = \{ag_i\},$$

где AG – множество агентов; ag_i – агент.

Для реализации требования 1 нами предлагается использовать два вида агентов: агент-функцию и метаграфовый агент.

Определим агент-функцию следующим образом:

$$ag^F = \langle MG_{IN}, MG_{OUT}, AST \rangle,$$

где ag^F – агент-функция; MG_{IN} – метаграф, который выполняет роль входного параметра агента-функции; MG_{OUT} – метаграф, который выполняет роль выходного параметра агента-функции; AST – абстрактное синтаксическое дерево агента-функции, которое может быть представлено в виде метаграфа.

Определим метаграфовый агент следующим образом:

$$ag^M = \langle MG_D, R, AG^{ST} \rangle, R = \{r_j\},$$

где ag^M – метаграфовый агент; MG_D – метаграф данных и знаний, на основе которого выполняются правила агента; R – набор правил (множество правил r_j); AG^{ST} – стартовое условие выполнения агента (фрагмент метаграфа, который используется для стартовой проверки правил, или стартовое правило).

Структура правила метаграфового агента:

$$r_i : MG_j \rightarrow OP^{MG},$$

где r_i – правило; MG_j – фрагмент метаграфа, на основе которого выполняется правило; OP^{MG} – множество операций, выполняемых над метаграфом.

Антецедентом правила является фрагмент метаграфа, консеквентом правила является множество операций, выполняемых над метаграфом.

Пример представления метаграфового агента в виде фрагмента метаграфа приведен на рис. 3.

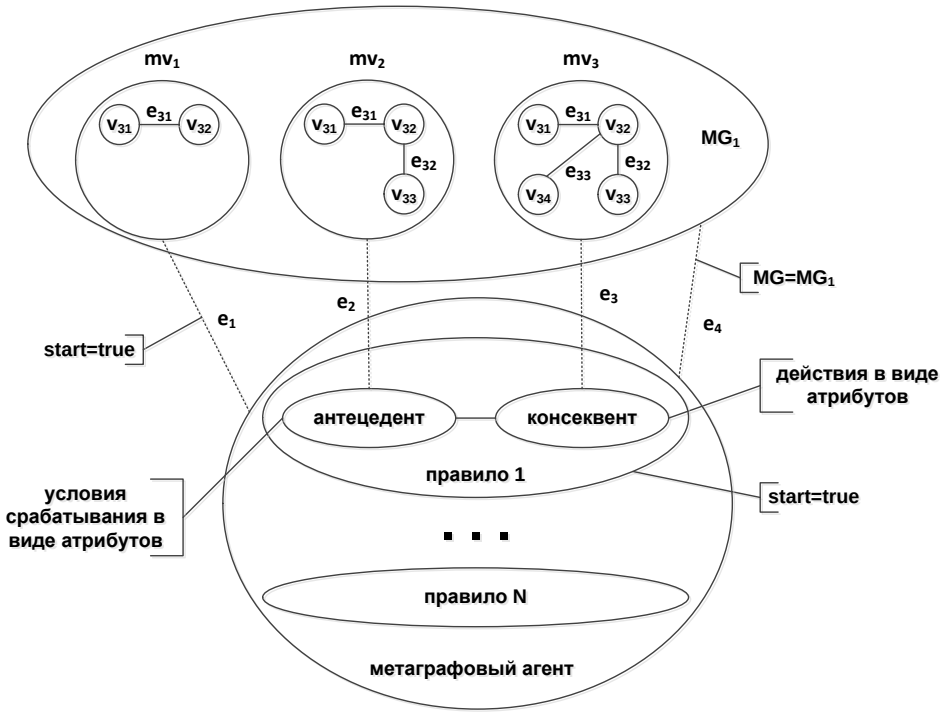

Рис. 3. Представление метаграфового агента в виде фрагмента метаграфа.

Метаграфовый агент представлен в виде метавершины метаграфа. В соответствии с определением он связан с метаграфом MG_1, на основе которого выполняются правила агента. Данная связь показана с помощью ребра e_4.

Метаграфовый агент содержит множество вложенных метавершин, соответствующих правилам (правило 1 – правило N). Каждая метавершина правила содержит вершины антецедента и консеквента. В данном примере с антецедентом правила связана метавершина данных mv_2, что показано ребром e_2, а с консеквентом правила связана метавершина данных mv_3, что показано ребром e_3. Условия срабатывания антецедента и множество действий консеквента задаются в виде атрибутов соответствующих вершин.

Стартовое условие выполнения агента задается с помощью атрибута «start=true». Если стартовое условие задается в виде стартового правила, то данным атрибутом помечается метавершина соответствующего правила, в данном примере это правило 1. Если стартовое условие задается в виде стартового фрагмента метаграфа, который используется для стартовой проверки правил, то атрибутом «start=true» помечается ребро, которое связывает стартовый фрагмент метаграфа с метавершиной агента, в данном примере это ребро e_1.

Для реализации требования 2 нами предлагается использовать контейнерный агент, который представляет собой метаграф, вершины и метавершины которого являются агентами:

$$ag^C = MG, v_i \equiv ag_i, v_i \in V, mv_i \equiv ag_i, mv_i \in MV,$$

где ag^C – контейнерный агент; MG – метаграф; v_i – вершина метаграфа; ag_i – агент; V – множество вершин метаграфа; mv_i – метавершина метаграфа; MV – множество метавершин метаграфа.

Пример статической структуры ГИИС представлен в виде холонической структуры агентов на рис. 4.

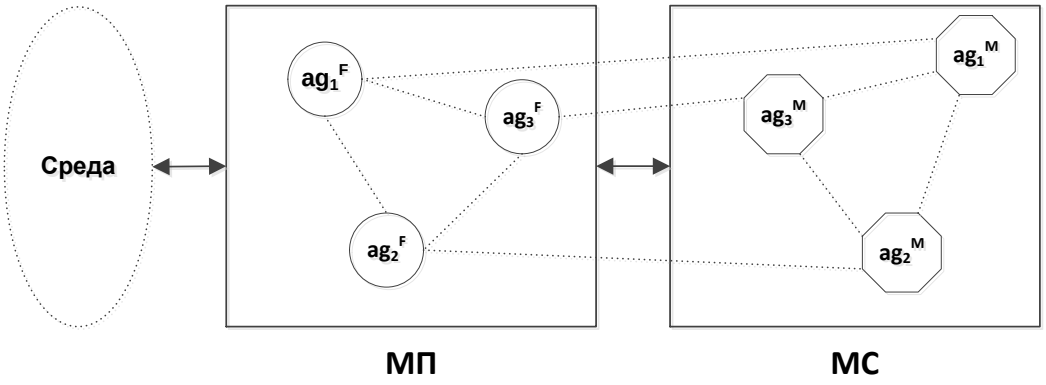

Рис. 4. Пример статической структуры ГИИС.

На рис. 4 представлена система, МП которой является нейронной сетью, а МС построен на основе обработки правил.

Агенты-функции показаны в виде окружностей, а метаграфовые агенты обработки данных в виде восьмиугольников. МП и МС выполняют роль контейнерных агентов. В данном примере используются одноуровневые контейнеры, однако, в соответствии с определением агента-контейнера, возможно использование произвольной вложенности контейнеров.

Отметим, что вся система холонических агентов представляет собой метаграф, при этом каждый агент также является метаграфом.

7. Метаграфы для описания холонической системы агентов в динамике

Для реализации требования 3 нами предлагается использовать динамический метаграфовый агент, правила обработки которого выполняются не на метаграфе данных и знаний, а на метаграфе агентов для заданного контейнерного агента:

$$ag^{MD} = \langle (ag^{C} \cup ag^{MD}), R, AG^{ST} \rangle, R = \{r_j\},$$

где ag^{MD} – динамический метаграфовый агент; ag^{C} – контейнерный агент, на метаграфе которого выполняются правила агента; R – набор правил (множество правил r_j); AG^{ST} – стартовое условие выполнения агента (фрагмент метаграфа, который используется для стартовой проверки правил, или стартовое правило).

По определению контейнерный агент включает все рассмотренные ранее виды агентов: агенты-функции и метаграфовые агенты. Поэтому динамический метаграфовый агент может изменять все виды агентов.

Определение данного агента использует тот факт, что в предлагаемой модели все агенты являются метаграфами, поэтому любые элементы структуры агентов доступны для обработки агентами верхнего уровня. Эта особенность является аналогом свойства «самоотображаемости» в традиционных языках программирования.

Отметим, что данное определение является рекурсивным. Динамические метаграфовые агенты первого уровня могут обрабатывать статические контейнерные агенты, метаграфовые агенты второго уровня могут обрабатывать метаграфовые агенты первого уровня и так далее. По мере необходимости систему можно надстраивать требуемыми уровнями динамики.

Правила динамического метаграфового агента базируются на рассмотренных ранее операциях над метаграфами. В зависимости от условий динамический метаграфовый агент может решать следующие задачи:

- первичное развертывание, создание, системы агентов более низкого уровня;
- изменение системы нижнего уровня (изменение внутренней структуры агентов, изменение связей между агентами, удаление агентов).

Пример динамической структуры ГИИС представлен в виде холонической структуры агентов на рис. 5.

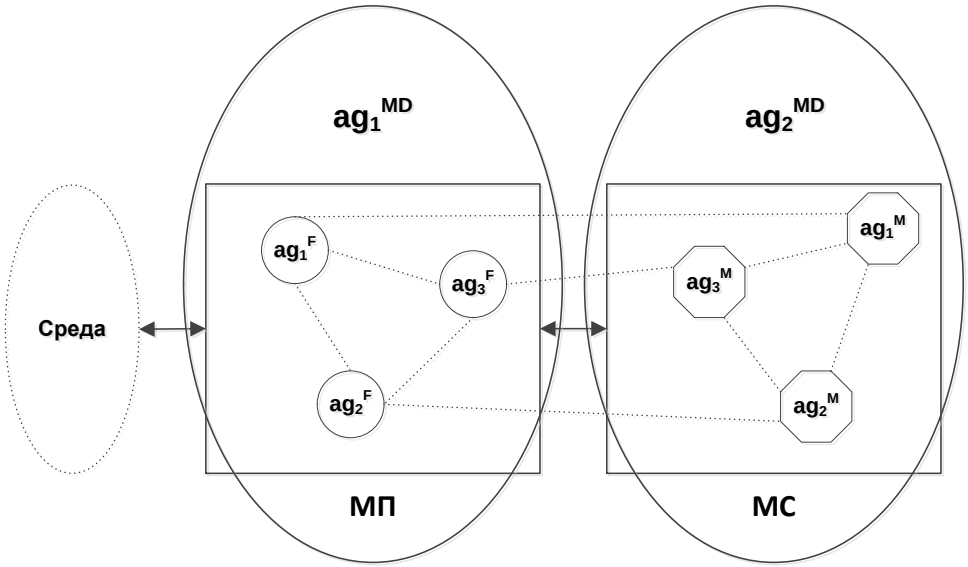

Рис. 5. Пример динамической структуры ГИИС.

По сравнение с рис. 4 на рис. 5 добавились два динамических метаграфовых агента.

Агент, отвечающий за МП, может изменять структуру самоорганизующейся нейронной сети. Или может применять эволюционные методы для оптимизации конфигурации нейронной сети.

Агент, отвечающий за МС, может изменять связи между агентами для решения задачи автоматизированного планирования. Или может применять эволюционные методы для оптимизации конфигурации агентов.

Динамические метаграфовые агенты могут использовать метазнания, которые также могут быть представлены в виде метаграфа.

На рис. 5 показаны только динамические метаграфовые агенты первого уровня, однако, в соответствии с определением, количество таких уровней не ограничено. Над показанными динамическими метаграфовыми агентами могут быть надстроены динамические метаграфовые агенты более высоких уровней.

Когда мы говорим о динамике, то предполагаем возможность изменения значений различных параметров агентов во времени. Таким образом, реализация динамики требует реализацию некоторого процесса. Для описания процесса предлагается использовать алгоритмическую модель, предложенную в [10]. Это описание может быть выполнено с помощью метаязыка, отличающегося универсальностью и представлением структуры процесса в виде графа, который может рассматриваться как элемент метаграфового описания соответствующего агента. Таким образом, использование предложенных средств описания процессов хорошо укладывается в общую концепцию метаграфового описания системы в целом, реализуя ее свойства динамичности.

8. Обсуждение полученных результатов

Таким образом, с помощью единого подхода удается описать как статику, так и динамику ГИИС на основе холонической МАС.

В [3] предлагается лево- и право- полушарный подход и выдвигается тезис о том, что гибридизация должна быть глубинной. Для гибридизации предлагается использовать методы мягких вычислений. В настоящем подходе для гибридизации предлагается использовать метаграфовый подход.

В существующих пакетах для разработки интеллектуальных систем на основе workflow, таких как RapidMiner или KNIME, интеллектуальная система может быть представлена как элементы МС и МП. Однако, вместо агентов используются элементы процесса workflow, поэтому такой подход не предполагает изменения структуры системы в динамике. Использование метаграфов позволяет динамически изменять структуру системы.

Необходимо также отметить, что в некоторых современных работах предлагаются различные варианты гибридного подхода. Так в работе [11] предлагается подход на основе радикального моделирования. Систему радикалов можно считать отдаленным аналогом многоагентной системы, при этом радикалы являются средством гибридизации модели.

Заключение

В статье предложен подход к описанию структуры гибридной интеллектуальной информационной системы на основе метаграфов.

Обобщенную структуру ГИИС предлагается строить на основе модулей «сознания» и «подсознания». На основе обобщенной структуры могут быть построены частные случаи структуры ГИИС, которым соответствуют конкретные ГИИС.

Для реализации ГИИС предполагается использовать холоничекую МАС. Структура такой МАС может быть описана с использованием метаграфовой модели.

С использованием метаграфовой модели возможно описание структуры как статических так и динамических агентов для реализации ГИИС. Реализация динамического агента предполагает использование теории описания процессов.

Список литературы

1. Колесников А.В. Гибридные интеллектуальные системы. Теория и технология разработки. – СПб.: СПбГТУ, 2001.
2. Колесников А.В., Кириков И.А., Листопад С.В. Гибридные интеллектуальные системы с самоорганизацией: координация, согласованность, спор. – М.: ИПИ РАН, 2014. – 189 с.
3. Прикладные интеллектуальные системы, основанные на мягких вычислениях. / под ред. Н.Г. Ярушкиной. – Ульяновск: УлГТУ, 2004. – 139 с.
4. Michal Bali “Dools JBoss Rules 5.0 Developer’s Guide”, PACKT publishing, 2013. – 338 p.
5. Д. Джарратано, Г. Райли. Экспертные системы: принципы разработки и программирование, 4-е издание, ИД «Вильямс», 2006. – 1152 с.
6. Тарасов В.Б. От многоагентных систем к интеллектуальным организациям: философия, психология, информатика. – М.: Эдиториал УРСС, 2002. – 352 с.

7. Xiao X., Zhang H., Hasegawa O. Density Estimation Method Based on Self-Organizing Incremental Neural Network and Error Estimation // Proceedings of the Neural Information Processing: 20th International Conference, ICONIP 2013. —Daegu, Korea, 2013. — 43–50 p.
8. J.K. Pugh, K.O. Stanley. Evolving Multimodal Controllers with HyperNEAT. Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2013). New York, NY: ACM, 2013. – 8 p.
9. Самохвалов Э.Н., Ревунков Г.И., Гапанюк Ю.Е. Использование метаграфов для описания семантики и прагматики информационных систем. Вестник МГТУ им. Н.Э. Баумана. Сер. «Приборостроение». 2015. Выпуск №1.
10. Черненький В.М. Алгоритмическая модель описания дискретного процесса функционирования системы // technomag.edu.ru:Наука и Образование: электронное научно-техническое издание. 2011. выпуск 12. URL http://technomag.edu.ru/doc/292997.html
11. Чечкин А.В. Интеллектуальная информационная система на основе радикального моделирования как инструментальное средство обеспечения комплексного развития. Нейрокомпьютеры: разработка, применение. 2015. Выпуск №5, стр. 7-13.