

Методы машинного обучения

ИУ-5, магистратура, 2 семестр, весна 2021 года



Гибридные интеллектуальные информационные системы, сложные графы и графы знаний



План

- ГИС и ГИИС. Обобщенная структура ГИИС. Частные случаи структуры ГИИС. Реализация ГИИС на основе холонической МАС.
- Сложные сети. Метаграф как разновидность сложной сети с эмерджентностью.
- Разновидности метаграфовых агентов.
- Сравнение метаграфового подхода и RDF. Основные подходы к построению хранилища на основе метаграфовой модели.
- Примеры использования архитектуры ГИИС. Графы и метаграфы знаний.

Обобщенная структура ГИИС.
Частные случаи структуры ГИИС.
Реализация ГИИС на основе
холонической МАС

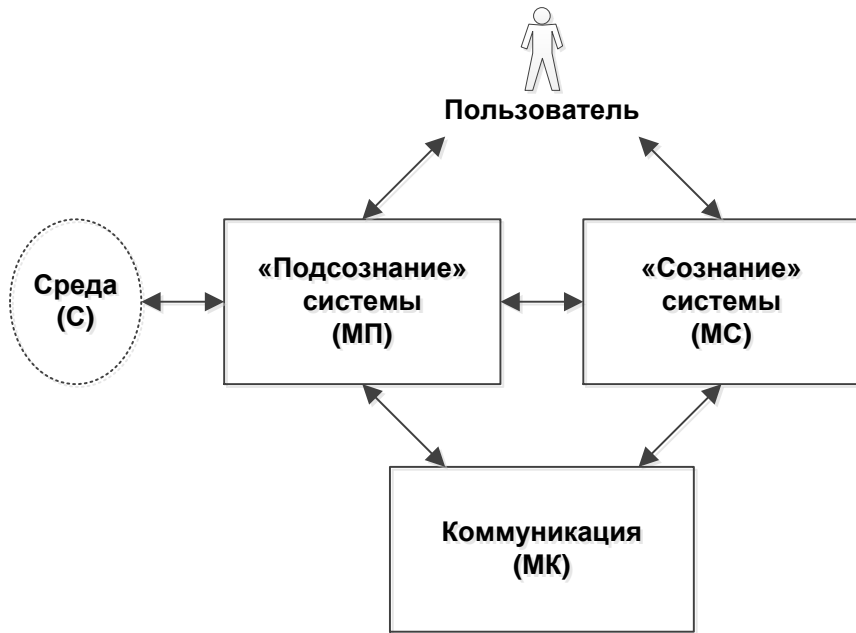
ГИС и ГИИС

- В настоящее время можно отметить явную тенденцию к совместному использованию различных интеллектуальных 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 Александра Васильевича Колесникова.
- 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)**.
- ГИИС обладает следующими особенностями:
 - сочетает различные методы, используемые 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.

Принцип гибридности

- Ключевым вопросом является вопрос о реализации принципа гибридности.
- В работах Надежды Глебовны Ярушкиной сформулирован следующий принцип гибридности: «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.
- Метафора право- и лево-полушарности возможно не совсем точна, скорее стоит говорить о «подсознании» и «сознании» гибридной ИС. «Подсознание» строится на основе мягких вычислений, а «сознание» на основе логической обработки данных и знаний.

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 - обработка

- «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 (rule-based programming).
- В настоящее время появляются гибридные продукты, в частности система Drools, которая позволяет проводить обработку 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.
- К достоинствам подхода на основе правил можно отнести гибкость, так как в этом случае программа не кодируется жестко, а «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) и его модификации.

«Сознание» ГИИС – модель данных

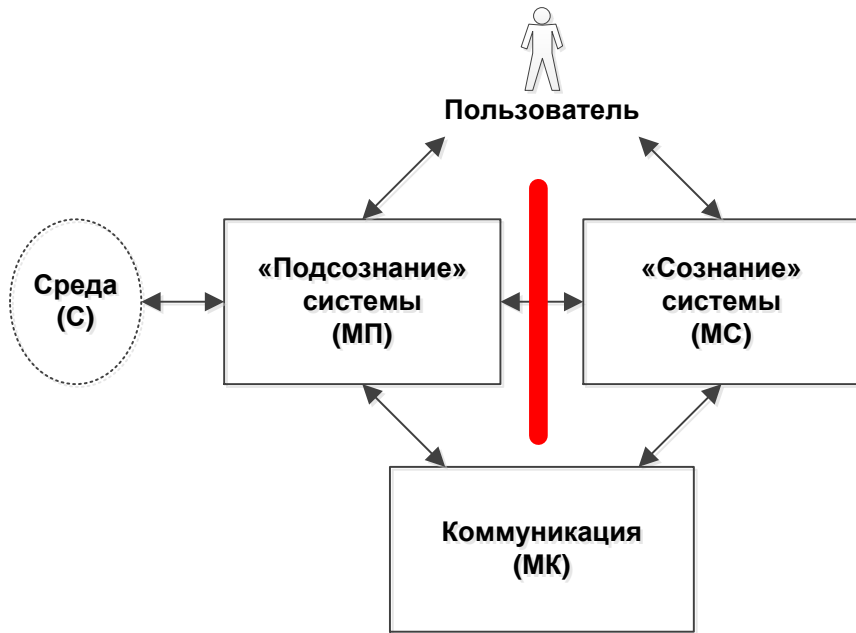
- В качестве модели данных МС используются модели «онтологического» класса. Это могут быть классические онтологии, разработанные в рамках технологии Semantic Web (стандарты RDF, RDFa, OWL, OWL2).
- Также к моделям этого класса можно отнести (возможно, с некоторыми ограничениями) и классическую объектно-ориентированную 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).

«Сознание» ГИИС - функции

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 планирование действий системы (автоматизированное планирование).

Граница между сознанием и подсознанием



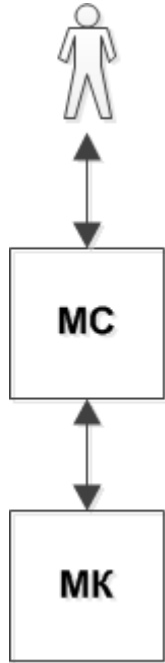
- Границей между сознанием и подсознанием является понятийная система ГИИС («онтология понятий»).
- Модуль сознания воспринимает понятийную систему как целостную модель «онтологического» класса и может «осознанно» обрабатывать элементы данной модели на основе правил.
- Модуль подсознания воспринимает понятийную систему в виде отдельных (возможно несвязанных) признаков. Требования к «осознанию» целостности модели не предъявляется. Основным критерием является эффективность взаимодействия системы со средой.
- Основными функциями подсознания являются:
 - Выделение из среды сигналов, соответствующих различным аспектам понятийной системы, эффективное распознавание элементов понятийной системы на основе сигналов.
 - Запись изменений в понятийной системе в среду в виде сигналов.

ГИИС - коммуникация

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

- 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.
- 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).
- 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).

Частные случаи структуры ГИИС - 1



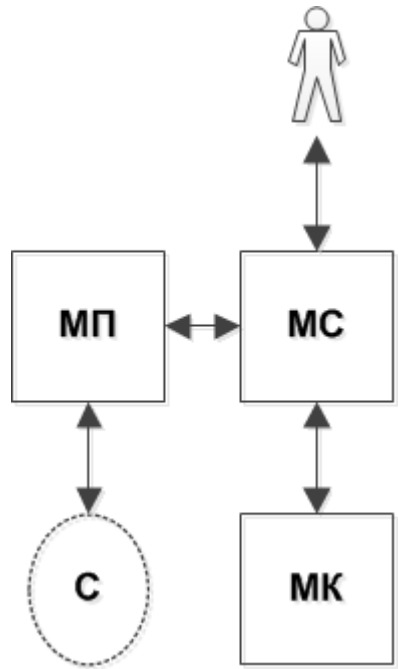
Классическая информационная система, в которой осуществляется только обработка данных и знаний (которую выполняет МС), реализуется коммуникация с другими системами (которую выполняет МК) и взаимодействие с пользователем.

Частные случаи структуры ГИИС - 2



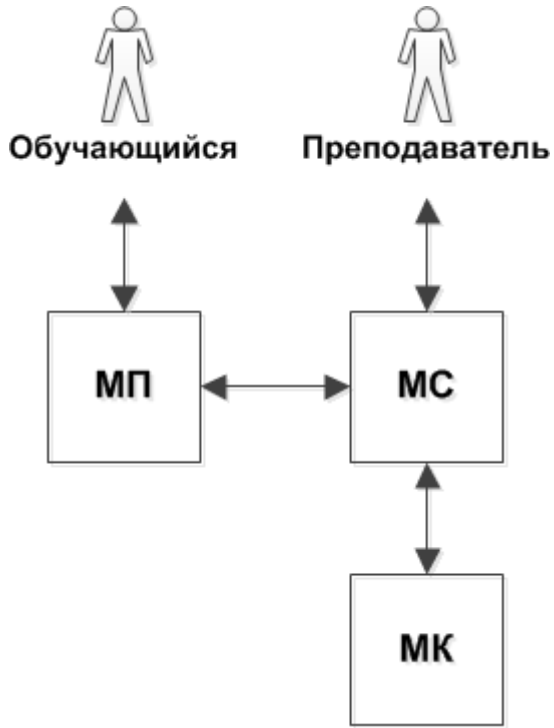
Простейшая система распознавания сигналов, поступающих из среды, с помощью МП. Сигналы могут иметь различную природу. Это может быть система распознавания музыкальной партитуры по звуковому сигналу, система распознавания элементов в видеопотоке и др. Данная система является простейшей, так как в ней 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.

Частные случаи структуры ГИИС - 3



- 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.
- 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.
- 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.

Частные случаи структуры ГИИС - 4



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.

Реализация на основе холонической МАС - 1

- 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 работах Валерия Борисовича Тарасова. В соответствии с определением холон – это «целое, рассматриваемое в то же время как часть целого».
- 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.

Реализация на основе холонической МАС - 2

- 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, hyperNEAT), 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.

Реализация на основе холонической МАС - 3

- 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.
- Для реализации требований используется подход на основе сложных сетей.

Сложные сети. Метаграф как
разновидность сложной сети с
эммерджентностью

Сложные сети

- В настоящее время термины «сложная сеть» или «комплексная сеть» (являются различными переводами англоязычного термина «complex network») и термин «сложный граф» (англ. «complex graph») часто употребляются как синонимы.
- Отмечается, что термин «сложная сеть», как правило, употребляется для обозначения реальной исследуемой системы, в то время как термин «сложный граф» обычно используют для обозначения математической модели такой системы.
- Наибольшие разночтения вызывает термин «сложный» применительно к графовым моделям. Как правило, термин «сложный» трактуется в двух вариантах:

Сложные сети, вариант I

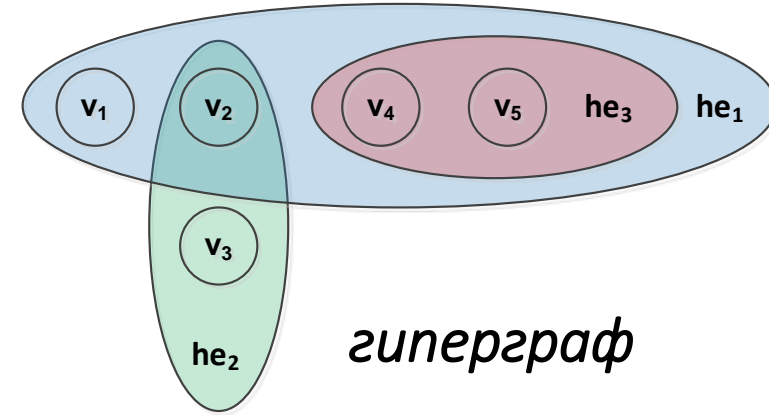
- Плоские графы (сети) очень большой размерности.
- Такие сети могут включать миллионы и более вершин.
- Ребра, соединяющие вершины, могут быть ненаправленными или направленными.
- Иногда используется модель мультиграфа, в этом случае две вершины могут соединяться не одним, а несколькими ребрами.
- Такие модели представляют интерес при изучении социальных сетей, глобальных компьютерных сетей, различных социологических и биологических моделей. Но они не очень хорошо помогают при описании сложных моделей данных и знаний.

Сложные сети, вариант II

- Сложные графы, в которых используется сложное (комплексное) описание вершин, ребер и/или их расположения.
- Часто в таких моделях отказываются от плоского расположения вершин и ребер.
- Именно подобные модели могут быть наиболее полезны при описании сложных моделей данных.
- На сегодняшний день известны четыре подобных модели: **гиперграф, гиперсеть, метаграф и многоуровневая сеть** (которая является упрощенным вариантом гиперсети).
- В настоящее время в литературе еще не появился единый «собирательный термин» для моделей такого класса. Авторы моделей, как правило, используют собственные названия для каждой модели, не всегда даже указывая на родство предлагаемой модели со сложными графами (сетями).
- Для подобного класса моделей можно предложить такой «собирательный термин» как **«ансамбли сложных сетей (графов)»**.
- Для гиперсетевой и метаграфовой моделей может быть использован термин **«сложные сети (графы) с эмерджентностью»**, так как данные модели реализуют принцип эмерджентности, хорошо известный в общей теории систем.

Гиперграф

- Гиперграф $HG = \langle V, HE \rangle$, $v_i \in V$, $he_j \in H$, V – множество вершин гиперграфа; HE – множество непустых подмножеств V , называемых гиперребрами; v_i – вершина гиперграфа; he_j – гиперребро гиперграфа. Гиперребро ненаправленного гиперграфа включает множество вершин, а ребро направленного гиперграфа задает последовательность обхода вершин.
- Гиперребро he_1 включает вершины v_1, v_2, v_4, v_5 ; гиперребро he_2 включает вершины v_2 и v_3 ; гиперребро he_3 включает вершины v_4 и v_5 . Гиперребра he_1 и he_2 имеют общую вершину v_2 . Все вершины гиперребра he_3 также являются вершинами гиперребра he_1 . Но «вложенность» гиперребра he_3 в гиперребро he_1 является скорее «визуальным эффектом», потому что операция вложенности для гиперребер формально не определена.
- Поэтому, хотя гиперграф и содержит гиперребра, но не позволяет моделировать сложные иерархические зависимости и не является полноценной «сетью с эмерджентностью».

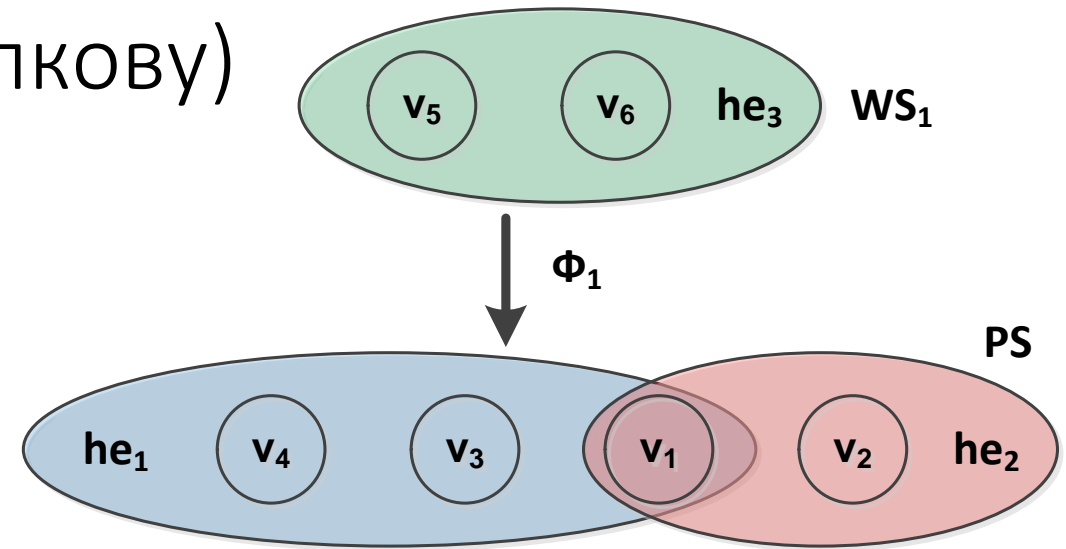


Гиперсетевая модель

- Достаточно типичной является ситуация, когда для семейства схожих моделей сложных сетей используется одинаковое название. Примером является гиперсетевая модель.
- Модель, с одинаковым названием «гиперсетевая», была независимо предложена профессором В.К. Попковым и профессором Дж. Джонсоном.
- С одной стороны, концепции предлагаемых вариантов гиперсетевой модели во многом схожи.
- Но, с другой стороны, разница между двумя вариантами моделей, использование различающихся математических аппаратов, и само изложение материала авторами моделей достаточно убедительно свидетельствует об отсутствии возможных заимствований.

Гиперсетевая модель (по В.К. Попкову)

- Удивительным является факт, что гиперсетевая модель была открыта дважды.
- В первый раз гиперсетевая модель предложена д.ф.м.н. профессором Владимиром Константиновичем Попковым в 1980-х годах.
- Фактически это первая модель «сети с эмерджентностью».

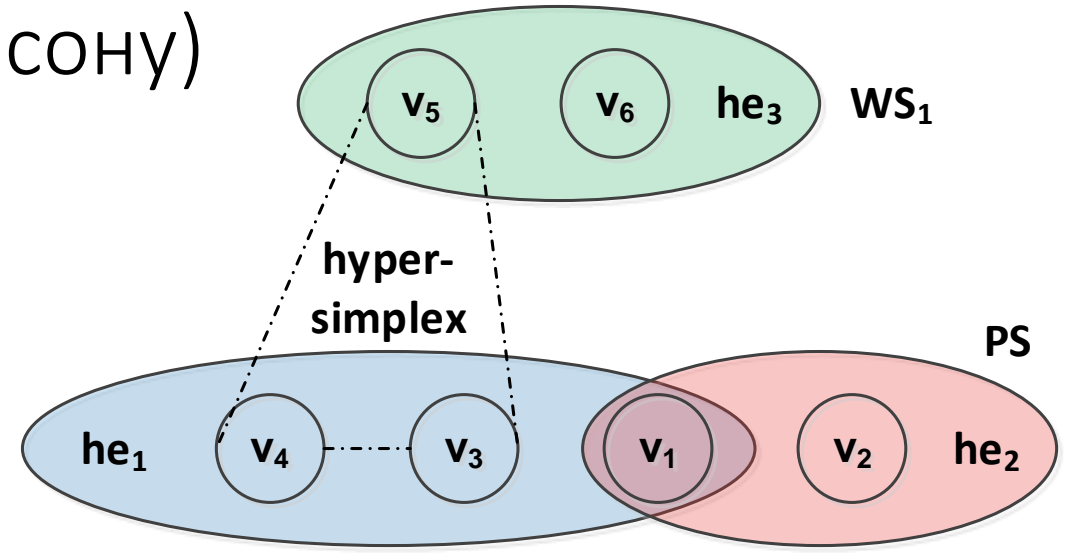


- Пусть даны гиперграфы $PS \equiv WS_0, WS_1, WS_2, \dots, WS_K$
- Гиперграф PS или WS_0 называется первичной сетью. Гиперграф WS_i называется вторичной сетью i -го порядка.
- Также задана последовательность отображений между сетями различных уровней:

$$\{\Phi_i\}: WS_K \xrightarrow{\Phi_K} WS_{K-1} \xrightarrow{\Phi_{K-1}} \dots WS_1 \xrightarrow{\Phi_1} PS$$
- Тогда иерархическая абстрактная гиперсеть порядка K : $AS^K = \langle PS, WS_1, \dots, WS_K; \Phi_1, \dots, \Phi_K \rangle$
- Эмерджентность в гиперсети возникает при переходе между уровнями за счет использования отображений между «слоями» гиперребер.

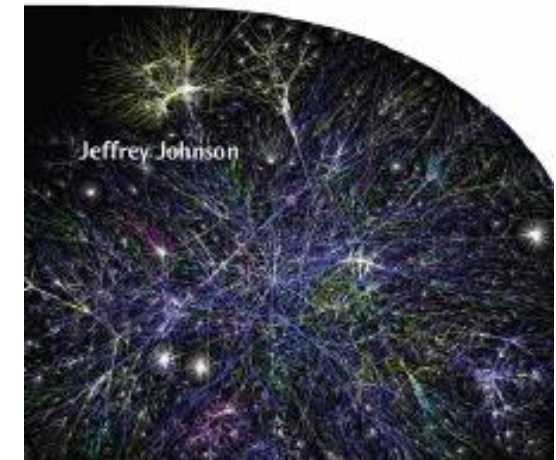
Гиперсетевая модель (по Дж. Джонсону)

- Во второй раз гиперсетевая модель была предложена профессором Джеффри Джонсоном в его монографии 2013 года.
- Эмерджентность в такой гиперсети возникает при переходе между уровнями за счет возникновения гиперсимплексов. Основание гиперсимплекса содержит множество элементов одного уровня, а его вершина образуется описанием их отношений и приобретает интегральные свойства, делающие ее элементом сети более высокого уровня.
- Профессор Константин Владимирович Анохин считает гиперсетевую модель (в интерпретации Дж. Джонсона) основой своей модели когнитума.



Series on Complexity Science - Vol. 3

Hypernetworks in the Science
of Complex Systems



Imperial College Press

Многоуровневая сеть

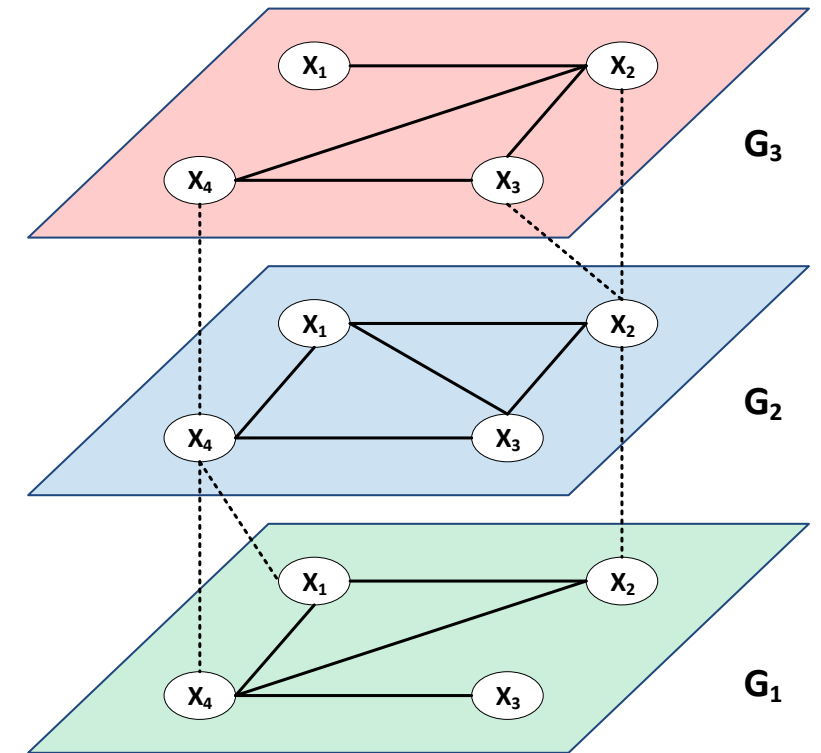
- Модель многоуровневой сети представляет собой попытку исследователей перейти от сложных сетей в трактовке I к сложным сетям в трактовке II. Детальный обзор модели приведен в работах.
- Отмечается, что в последнее время исследователи все чаще обращают внимание на многоуровневый характер реальных сложных систем. Многоуровневая сеть, предлагается в качестве графового формализма для описания сложных систем.
- Формализованное описание многоуровневой сети:

$$MN = \left\langle \{G_1, \dots, G_L\}, \{E_{ij} \subset X_i \times X_j, i, j \in \{1, \dots, L\}, i \neq j\} \right\rangle, G_K = \langle X_K, E_K \rangle.$$

- В приведенной формуле $\{G_1, \dots, G_L\}$ – семейство плоских графов $G_K = \langle X_K, E_K \rangle$, где X_K – множество вершин графа G_K , E_K – множество ребер графа G_K .
- Множество $\{G_1, \dots, G_L\}$ называется уровнями в многоуровневой сети MN .
- Граф G_K может быть направленным или ненаправленным, взвешенным или невзвешенным, а также мультиграфом.
- Каждый элемент E_{ij} является множеством связей между вершинами графов уровней i и j . Условие $i \neq j$ говорит о запрете циклических связей на одном уровне, связи могут быть только между элементами соседних уровней.
- При условии $L=1$ многоуровневая сеть превращается в обычную одноуровневую сеть.

Многоуровневая сеть (пример)

- Пример многоуровневой сети представлен на рисунке.
- Сеть содержит три уровня G_1 , G_2 , G_3 . В данном примере каждый уровень является плоским ненаправленным графом. Ребра графов E_k показаны сплошными линиями. Примеры связей E_{ij} между уровнями показаны пунктирными линиями.
- Сравним модель многоуровневой сети и гиперсетевую модель. Как и гиперсетевая модель, модель многоуровневой сети является послойной.
- Если в гиперсетевой модели на каждом уровне применяются гиперграфы, то в многоуровневой сети уровнем является более простая модель – обычный плоский граф. Связи между уровнями E_{ij} можно рассматривать как частный случай отображения Φ в гиперсети на основе модели В.К. Попкова.
- Таким образом, модель многоуровневой сети можно считать частным упрощенным случаем гиперсетевой модели в интерпретации В.К. Попкова.

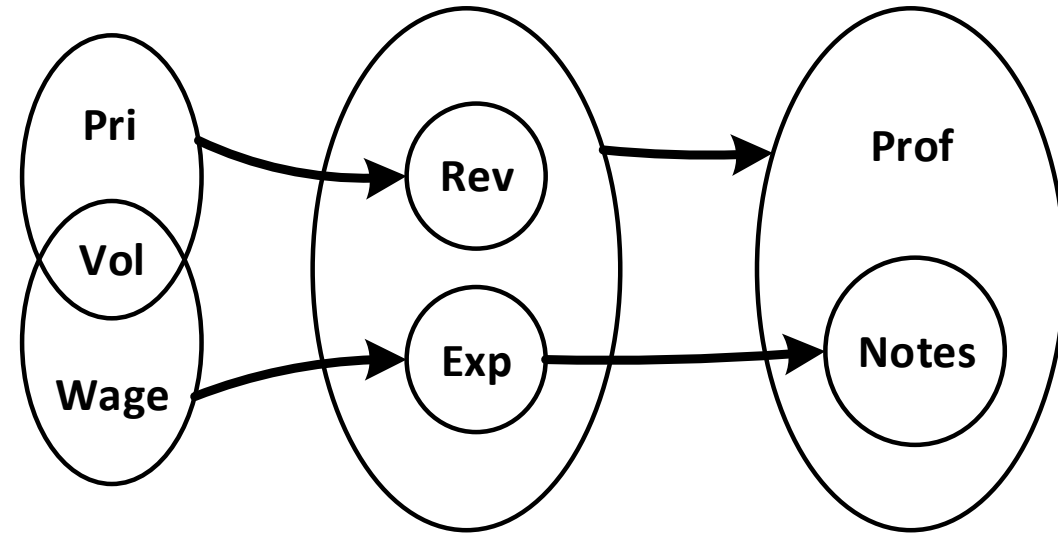


Метаграфовая модель А. Базу и Р. Блэннинга

- Исторически монография А. Базу и Р. Блэннинга [4] была первым источником, в котором появился термин «метаграф». В монографии даются следующие определения, характеризующие метаграфовую модель.
- **Порождающее множество метаграфа** – это множество переменных, встречающихся в ребрах метаграфа: $X = \{x_1, x_2, \dots, x_n\}$.
- **Ребро метаграфа** $e = \langle V_e, W_e \rangle \in E$ (где E – множество ребер) содержит **входную вершину (invertex)** $V_e \subset X$ и **выходную вершину (outvertex)** $W_e \subset X$. Входная и выходная вершины могут содержать произвольное количество элементов. Различные элементы, принадлежащие входной (выходной) вершине, называются соответственно **совходами (совыходами)**.
- Тогда **метаграф** $S = \langle X, E \rangle$ – это графовая конструкция, определяемая порождающим множеством X и множеством ребер E , при этом множество ребер определено на том же порождающем множестве.
- **Простым путем** $h(x, y)$ из элемента x в элемент y это последовательность ребер $\langle e_1, e_2, \dots, e_n \rangle$, такая что:
 - x является входной вершиной e_1 , $x \in invertex(e_1)$;
 - y является выходной вершиной e_n , $y \in outvertex(e_n)$;
 - для всех $e_i, i = 1, \dots, n - 1$ выполняется условие: $outvertex(e_i) \cap invertex(e_{i+1}) \neq \emptyset$, то есть путь из начальной вершины в конечную не прерывается.

Метаграфовая модель А. Базу и Р. Блэннинга (пример)

- На рисунке представлен пример метаграфа, для которого приводится следующая теоретико-множественная интерпретация:



- $S = \langle X, E \rangle$;
 - $X = \{Exp, Notes, Prof, Rev, Pri, Vol, Wage\}$;
 - $E = \langle \{Pri, Vol\}, \{Rev\} \rangle, \langle \{Vol, Wage\}, \{Exp\} \rangle,$
 - $\langle \{Rev, Exp\}, \{Prof, Notes\} \rangle, \langle \{Exp\}, \{Notes\} \rangle$.
- Эмерджентность в модели А. Базу и Р. Блэннинга достигается за счет использования ребер. Понятие метавершины в данной модели отсутствует.
 - Можно отметить, что данный вариант метаграфовой модели более подходит для описания направленных процессов, чем для описания сложных графовых структур данных.
 - В дальнейшем модель получила ряд расширений, которые независимо предлагались различными группами исследователей.

Метаграфовая модель с метавершинами

- Отсутствие естественного механизма для описания сложных графовых структур данных привело к появлению расширений исходной модели А. Базу и Р. Блэннинга. В моделях появились новые элементы – метавершины и метаребра.
- В работе (Глоба, Л. С. *Метаграфы как основа для представления и использования баз нечетких знаний* / Л. С. Глоба, М. Ю. Терновой, Е. С. Штогрин // *Открытые семантические технологии проектирования интеллектуальных систем = Open Semantic Technologies for Intelligent Systems (OSTIS-2015) : материалы V междунар. науч.-техн. конф. (Минск, 19-21 февраля 2015 года)* / редкол. : В. В. Голенков (отв. ред.) [и др.]. – Минск : БГУИР, 2015. – С. 237-240) появляется понятие метавершины. В этой работе даются следующие определения метаграфовой модели.
- **Метаграф** – это тройка множеств вершин, метавершин и ребер соответственно: $S = \langle V, M, E \rangle$, где $V = \{v_r\}$ – множество вершин метаграфа (порождающее множество); $M = \{m_q\}$ – множество метавершин метаграфа; $E = \{e_h\}$ – множество ребер метаграфа.
- **Метавершина метаграфа** $m_q = \{v_r | v_r \in V, r = 1, \dots, N_{m_q}\}$ определяется как множество вершин v_r , входящих в метавершину m_q , где N_{m_q} – мощность множества.
- Интересным следует считать следующее замечание авторов модели: «... если две или больше метавершин соответствуют одному и тому же множеству вершин, то такие вершины считаются одинаковыми и рассматривается только одна из таких метавершин». Назовем данное свойство модели **свойством анти-аннотируемости**.
- Интересно, что для задания ребер, авторы модели вводят понятие узла метаграфа $mv \in (V \cup M)$, принадлежащего объединенному множеству вершин и метавершин. Ребро определяется как $e_h = \langle mv_{out}, mv_{in} \rangle$, то есть характеризуется исходящим и входящим узлами метаграфа. Но использование понятия узла для создания иерархических метавершин авторами модели не предлагается.

Иерархическая метаграфовая модель с метавершинами и метаребрами

- В работе (Астанин С.В., Драгныш Н.В., Жуковская Н.К. Вложенные метаграфы как модели сложных объектов // Инженерный вестник Дона, 2012, №4. URL: ivdon.ru/ru/magazine/archive/n4p2y2012/1434) появляется не только понятие метавершины, но также понятия метаребра и иерархии вершин.
- **Метаграф** в модели определяется как $S = \langle X, X_M, E, E_M \rangle$, где X – множество вершин метаграфа (порождающее множество); X_M – множество метавершин метаграфа; E – множество ребер метаграфа; E_M – множество метаребер метаграфа, заданных на множестве $X_M \cup X$.
- Таким образом, под метаребром в данной модели понимается ребро, которое может соединять вершину и метавершину или две метавершины.
- Важной особенностью данной модели является то, что авторы вводят понятие **вложенного метаграфа**, который, как полагают авторы, является «обобщением обычных графов, гиперграфов и метаграфов».
- В данной модели множество вершин X рассматривается как иерархическое, вводится индекс i , определяющий уровень вложенности вершины.
- Свойство анти-аннотируемости авторами модели не утверждается и не опровергается. При этом, приводимые в статье примеры неявно используют свойство анти-аннотируемости.
- Необходимо отметить, что данная относительно небольшая по объему работа цитируется в большинстве более поздних статей по тематике метаграфов, что говорит о важности центрального вопроса данной статьи – описания иерархий в метаграфовой модели.

Пример 1 из статьи (С.В. Астанин, Н.В. Драгныш, Н.К. Жуковская)

Если ребро n -мерного графа является направленным, то граф называется ориентированным n -мерным графом. Вложенные метаграфы являются обобщением обычных графов, гиперграфов и метаграфов. В общем случае, вершины x_2^p являются гиперребрами графов $g_1^p(x_1^p, e_1^p)$, вершины x_2^r являются гиперребрами графов $g_2^p(x_2^p, e_2^p)$, и т.д. Ребра могут связывать вершины любого уровня представления, т.е. как отдельные вершины, так и гиперребра, что характерно для метаграфов. Подобное описание позволяет представлять вложенные структуры, каждая вершина которой может быть устроена по типу «револьверной матрешки» (рис.3).

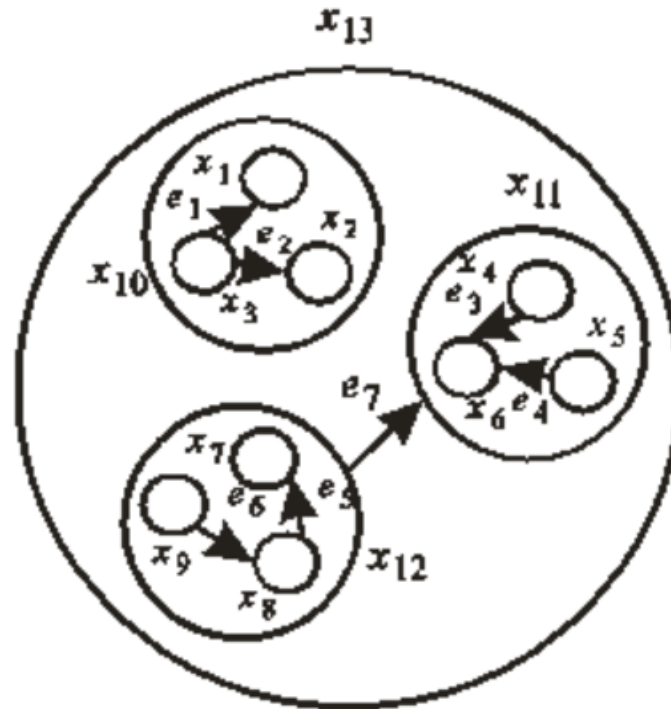


Рис.3.- Пример вложенного метаграфа трехмерной размерности

Пример 2 из статьи
(С.В. Астанин,
Н.В. Драгныш,
Н.К. Жуковская)

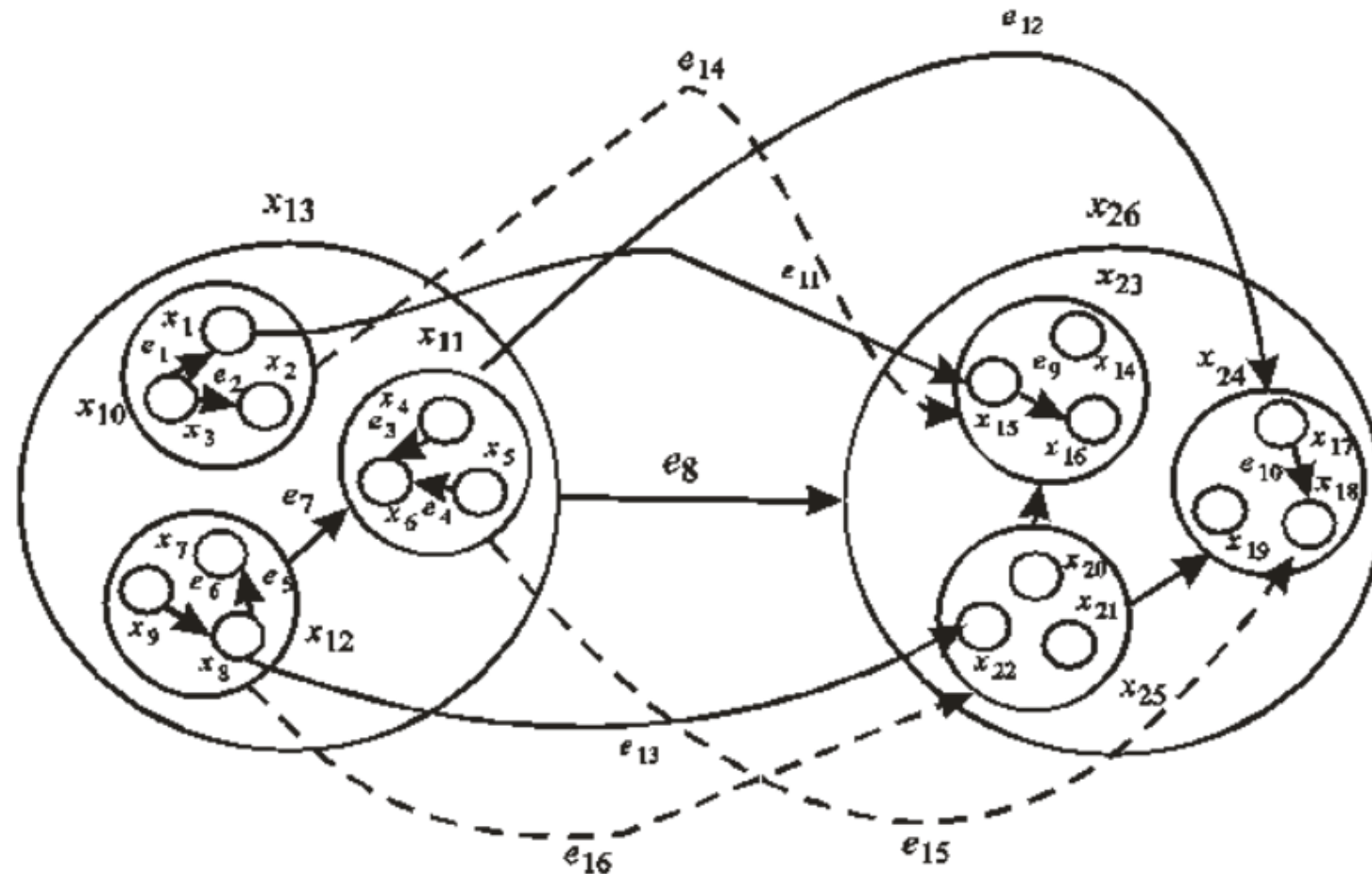


Рис.4.- Фрагмент ситуационной сети бизнес-процесса

Состояния сети x_{13} и x_{26} представлены метаграфами, причем состояние x_{13} является обобщением ситуаций x_{10} , x_{11} и x_{12} , а состояние x_{26} – обобщением ситуаций x_{23} , x_{24} и x_{25} . Каждое метаребро является управляющим воздействием, позволяющим перевести бизнес-процесс из одного состояния в другое состояние. При этом возможен анализ различных уровней в зависимости от текущих обстоятельств. Например, если в момент времени t ожидаемым состоянием является x_{26} , а фиксируется состояние x_{13} при управляющих воздействиях e_{11} , e_{12} , e_{13} , то анализируются причины на других уровнях управления, не позволившие процессу перейти в состояние x_{26} .

Аннотируемая метаграфовая модель

Определим метаграф следующим образом:

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

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

Вершина метаграфа характеризуется множеством атрибутов:

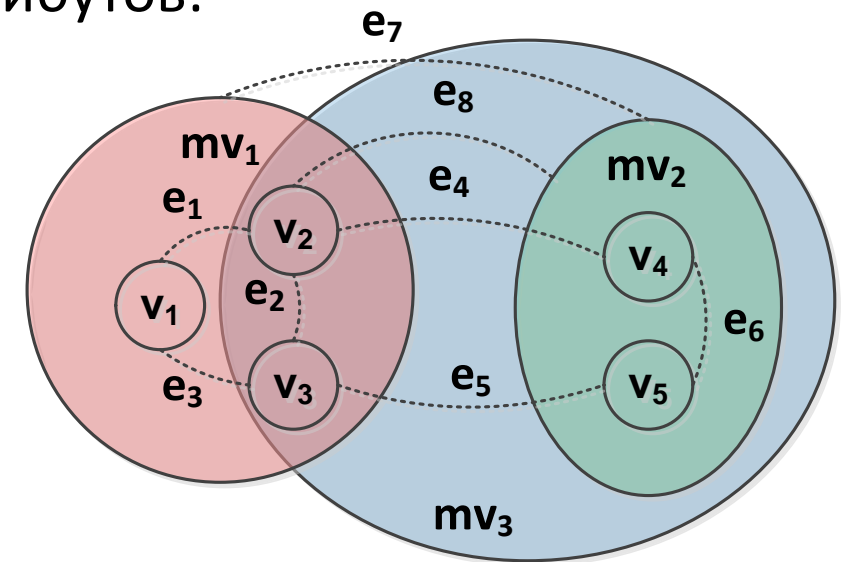
$$v_i = \{atr_k\}, v_i \in V,$$

где v_i – вершина метаграфа; atr_k – атрибут.

Ребро метаграфа характеризуется множеством атрибутов, исходной и конечной вершиной:

$$e_i = \langle v_S, v_E, \{atr_k\} \rangle, e_i \in E,$$

где e_i – ребро метаграфа; v_S – исходная вершина (метавершина) ребра; v_E – конечная вершина (метавершина) ребра; atr_k – атрибут.



Аннотируемая метаграфовая модель - 2

Фрагмент метаграфа:

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

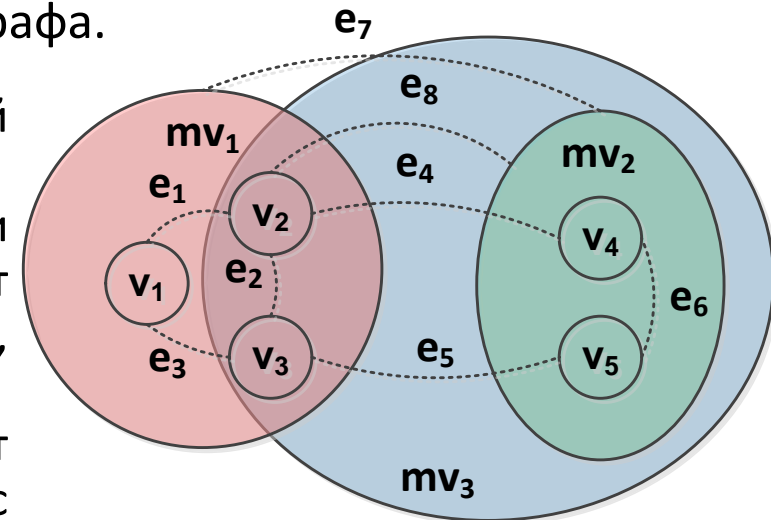
где MG_i – фрагмент метаграфа; ev_j – элемент, принадлежащий объединению множеств вершин, метавершин, ребер и метаребер метаграфа.

Фрагмент метаграфа в общем виде может содержать произвольные вершины (метавершины) и ребра.

Метавершина метаграфа: $mv_i = \langle \{atr_k\}, MG_i \rangle, mv_i \in MV,$

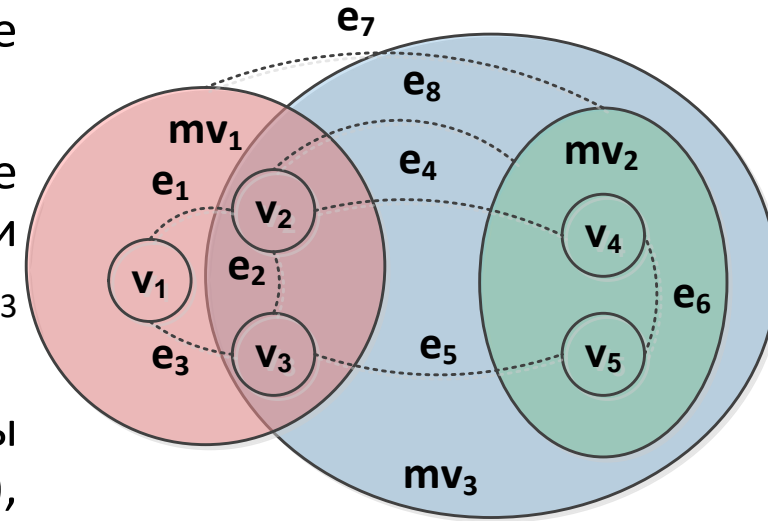
где mv_i – метавершина метаграфа; atr_k – атрибут, MG_i – фрагмент метаграфа.

- Метавершина в дополнение к свойствам вершины включает вложенный фрагмент метаграфа.
- 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.
- 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.



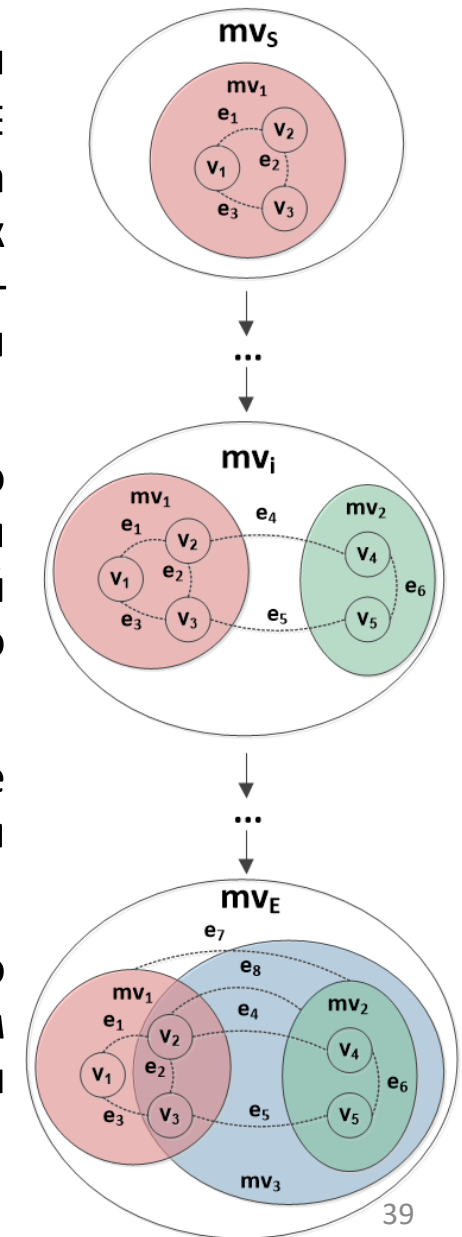
Аннотируемая метаграфовая модель – (пример)

- Метаграф позволяет естественным образом моделировать сложные иерархические зависимости и является «сетью с эмерджентностью».
- Метаграф содержит вершины, метавершины и ребра. На рисунке показаны три метавершины: mv_1 (которая включает вершины v_1, v_2, v_3 и ребра e_1, e_2, e_3), mv_2 (которая включает вершины v_4, v_5 и ребро e_6) и mv_3 (которая включает метавершину mv_2 , вершины v_1 и v_2 и ряд ребер).
- Ребро метаграфа может соединять вершины внутри одной метавершины (e_1, e_2, e_3, e_6), вершины между различными метавершинами (e_4, e_5), метавершины (e_7), вершины и метавершины (e_8).
- Метавершина позволяет выделять фрагмент графа (метаграфа), аннотировать его дополнительными свойствами, проводить к нему (как к целому) ребра.
- Отметим, что в отличие от [5], в данной модели не выполняется свойство анти-аннотируемости. Одинаковый набор вершин и ребер может быть включен в несколько различных метавершин, которые могут представлять различные ситуации и быть аннотированы различными атрибутами.
- Также, в предлагаемой модели, метавершина может включать как вершины, так и ребра.



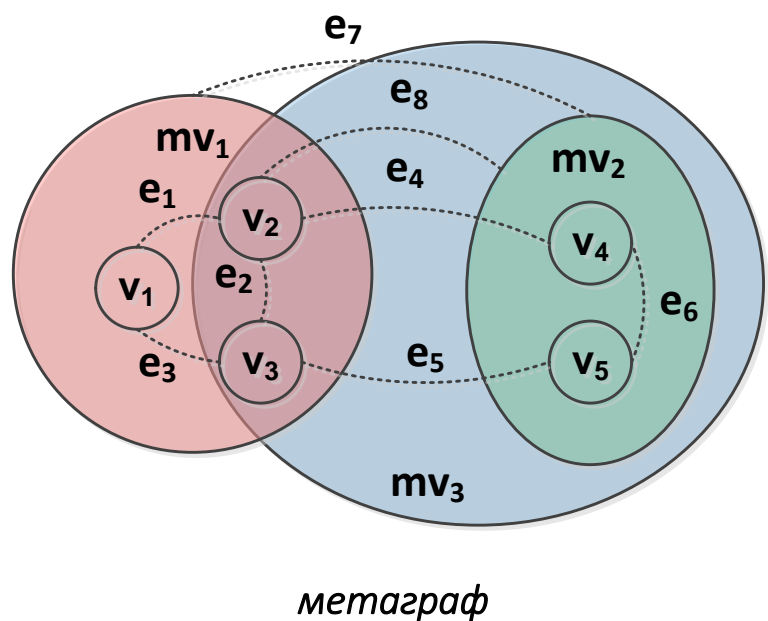
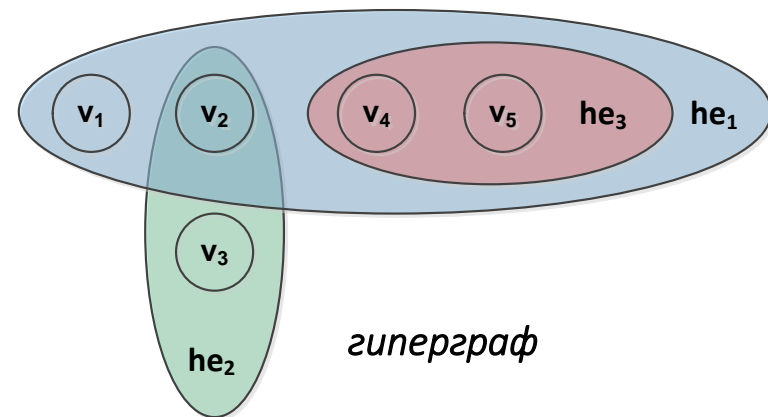
Аннотируемая метаграфовая модель – метаребро

- **Метаребро метаграфа** в дополнение к свойствам ребра включает вложенный фрагмент метаграфа: $me_i = \langle v_S, v_E, eo, \{atr_k\}, \{ev_j\} \rangle, e_i \in E, eo = true|false, ev_j \in (V \cup E \cup MV \cup ME)$, 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.
- Пример описания метаребра метаграфа представлен на рисунке. Метаребро содержит метавершины $v_S, \dots, v_i, \dots, v_E$ и связывающие их ребра. Исходная метавершина содержит фрагмент метаграфа. В процессе преобразования исходной метавершины v_S в конечную метавершину v_E происходит дополнение содержимого метавершины, добавляются новые вершины, связи, вложенные метавершины.
- Таким образом, иерархическому метаребру из модели [6] соответствует обычное ребро в предлагаемой нами модели. А под метаребром понимается последовательность изменения метавершин метаграфа.
- Если метавершины предназначены прежде всего для описания данных и знаний, то метаребра предназначены в большей степени для описания процессов. Таким образом, аннотируемая метаграфовая модель позволяет в рамках единой модели описывать данные, знания и процессы.



Метаграфы и гиперграфы

- Гиперграф $HG = \langle V, HE \rangle$, $v_i \in V$, $he_j \in H$, V – множество вершин гиперграфа; HE – множество непустых подмножеств V , называемых гиперребрами; v_i – вершина гиперграфа; he_j – гиперребро гиперграфа. Гиперребро ненаправленного гиперграфа включает множество вершин, а ребро направленного гиперграфа задает последовательность обхода вершин.
- Гиперребро he_1 включает вершины v_1, v_2, v_4, v_5 ; гиперребро he_2 включает вершины v_2 и v_3 ; гиперребро he_3 включает вершины v_4 и v_5 . Гиперребра he_1 и he_2 имеют общую вершину v_2 . Все вершины гиперребра he_3 также являются вершинами гиперребра he_1 . Но «вложенность» гиперребра he_3 в гиперребро he_1 является скорее «визуальным эффектом», потому что операция вложенности для гиперребер формально не определена. Поэтому, хотя гиперграф и содержит гиперребра, но не позволяет моделировать сложные иерархические зависимости и не является полноценной «сетью с эмерджентностью».
- Если гиперребро гиперграфа может включать только вершины, то метавершина метаграфа может включать как вершины (или метавершины), так и ребра.
- В отличие от гиперграфа, метаграф позволяет естественным образом моделировать сложные иерархические зависимости и является «сетью с эмерджентностью».



Метаграфы и гиперсети

- В соответствии с определением гиперсеть является «послойным» описанием графов. Предполагается, что слои-гиперграфы идут последовательно и имеют регулярную структуру. Метаграф позволяет с помощью метавершин группировать произвольные элементы, наличие регулярных уровней не обязательно, что делает подход метаграфов более гибким. Фактически, каждый гиперсимплекс может быть представлен отдельной метавершиной.
- Гиперсеть состоит из разнородных элементов (гиперграфов, отображений, гиперсимплексов). Метаграф позволяет с помощью метавершин обеспечивать связь как между элементами одного уровня, так и между элементами различных уровней (при этом, не обязательно соседних). Это делает метаграфовый подход более унифицированным и удобным в описании, так как для описания используются не разнородные структуры (гиперграфы и отображения), а только метавершины (и связи как элементы метавершин). Метаграфовый подход позволяет рассматривать сеть не только в виде «горизонтальных» слоев, но и в виде «вертикальных» колонок.
- Эмерджентность в гиперсети обеспечивается за счет отображений или гиперсимплексов и фактически возникает только при переходе между соседними уровнями. Эмерджентность в метаграфах обеспечивается за счет использования метавершин и может применяться на одном уровне или между уровнями (не обязательно соседними), что делает реализацию эмерджентности в метаграфах более гибкой.
- Необходимо подчеркнуть, что метаграфы и гиперсети являются лишь различными формальными описаниями одних и тех же процессов, которые происходят в «сетях с эмерджентностью». Также необходимо отметить, что настоящее время теория гиперсетей является намного более зрелой по сравнению с теорией метаграфов и именно благодаря теории гиперсетей исследователям удалось понять многие аспекты «сетей с эмерджентностью».

Разновидности метаграфовых агентов

Холоническая МАС для реализации активности

- Метаграф является пассивной структурой данных. Как реализовать активность при моделировании сложной сети с эмерджентностью?
- Как выполнить рассмотренные требования 1, 2, 3?
- Для реализации требования 1 предлагается использовать два вида агентов: агент-функцию и метаграфовый агент.
- Для реализации требования 2 предлагается использовать контейнерный агент.
- Для реализации требования 3 предлагается использовать динамический метаграфовый агент.
- Рассмотрим данные виды агентов более подробно.

Агент-функция

- Агент-функция:

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

- 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.

Метаграфовый агент (определение)

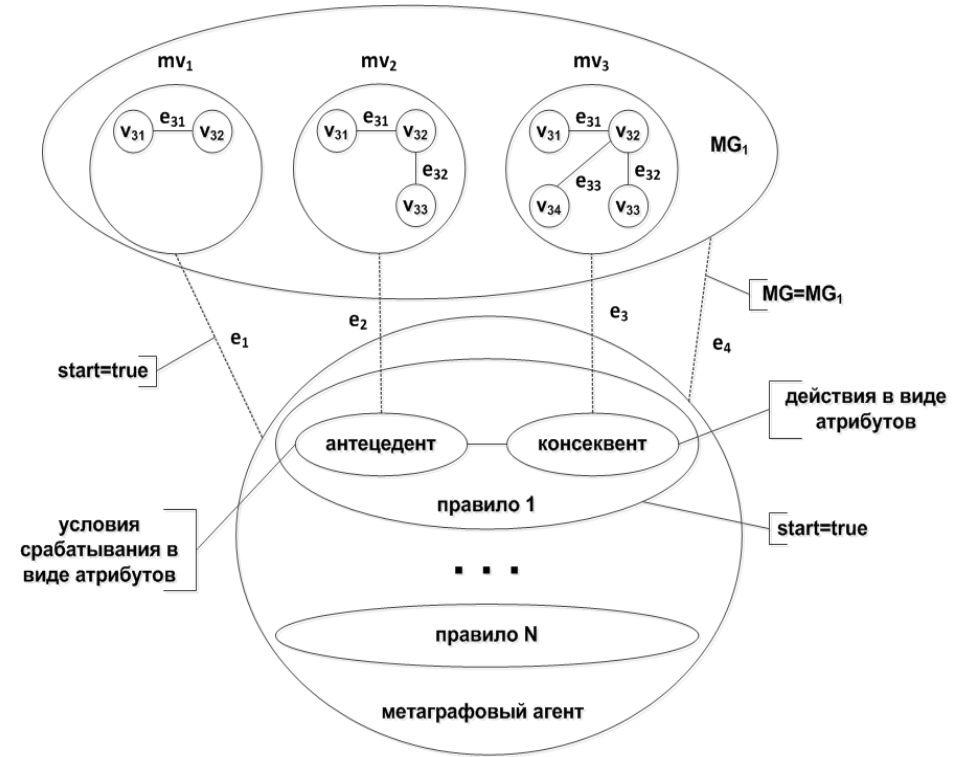
- Метаграфовый агент:

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

- где ag^M – метаграфовый агент; MG – метаграф, 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).
- При этом агент ag^M содержит множество вложенных агентов ag_i^M что соответствует принципам организации холонической многоагентной системы. Агент верхнего уровня может активизировать агентов нижнего уровня для решения подзадач.
- Структура правила метаграфового агента:

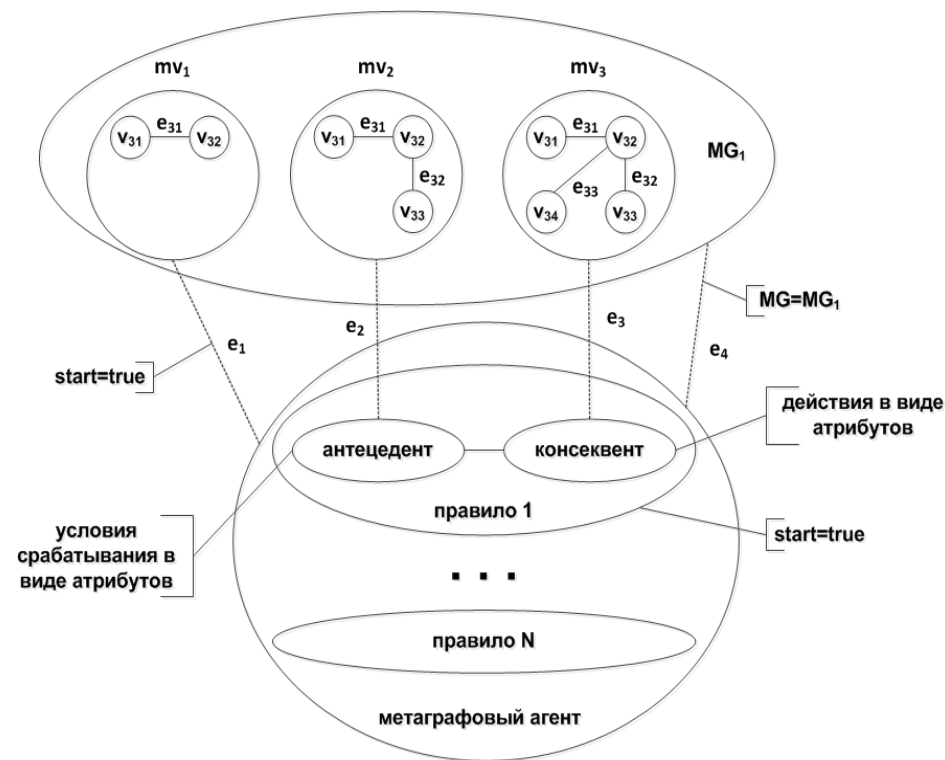
$$r_i : MG_j \rightarrow OP^{MG},$$

- где r_i – правило; MG_j – фрагмент метаграфа, на основе которого выполняется правило; OP^{MG} – множество действий, выполняемых над метаграфом.
- Антецедементом правила является фрагмент метаграфа, консеквентом правила является множество действий, выполняемых над метаграфом.



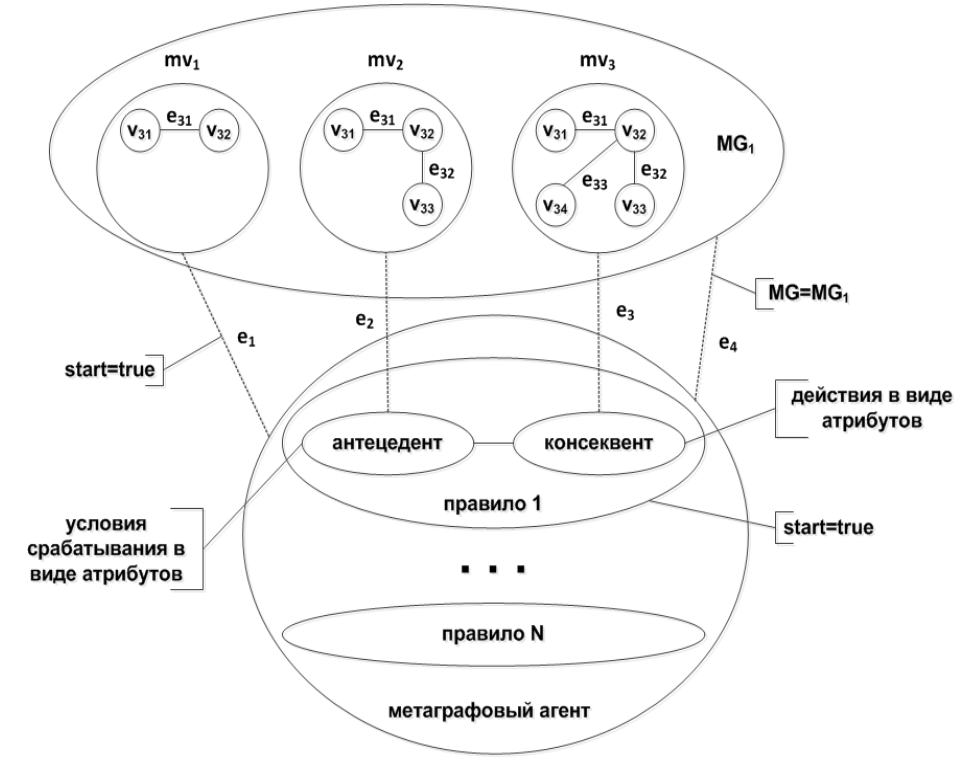
Метаграфовый агент (правила)

- Правила метаграфового агента можно разделить на замкнутые и разомкнутые.
- Разомкнутые правила не меняют в правой части правила фрагмент метаграфа, относящийся к левой части правила. Можно разделить входной и выходной фрагменты метаграфа. Данные правила являются аналогом шаблона, который порождает выходной метаграф на основе входного.
- Замкнутые правила меняют в правой части правила фрагмент метаграфа, относящийся к левой части правила. Изменение метаграфа в правой части правил заставляет срабатывать левые части других правил. Но при этом некорректно разработанные замкнутые правила могут привести к заикливанию метаграфового агента.
- Таким образом, метаграфовый агент позволяет генерировать один метаграф на основе другого (с использованием разомкнутых правил) или модифицировать метаграф (с использованием замкнутых правил).



Метаграфовый агент (самоотображаемость)

- Особенностью метаграфового агента является то, что его структура может быть представлена в виде фрагмента метаграфа. Это соответствует принципу самоотображаемости (англ. homoiconicity) в языках программирования. Самоотображаемость – это способность языка программирования анализировать программу на этом языке как структуру данных этого языка.
- Структура агента может быть изменена как данные с помощью правил агентов верхнего уровня.
- 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). 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 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 .



Контейнерный агент

- Контейнерный агент:

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

- 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.
- Контейнерный агент, представляет собой метаграф, вершины и метавершины которого являются агентами.

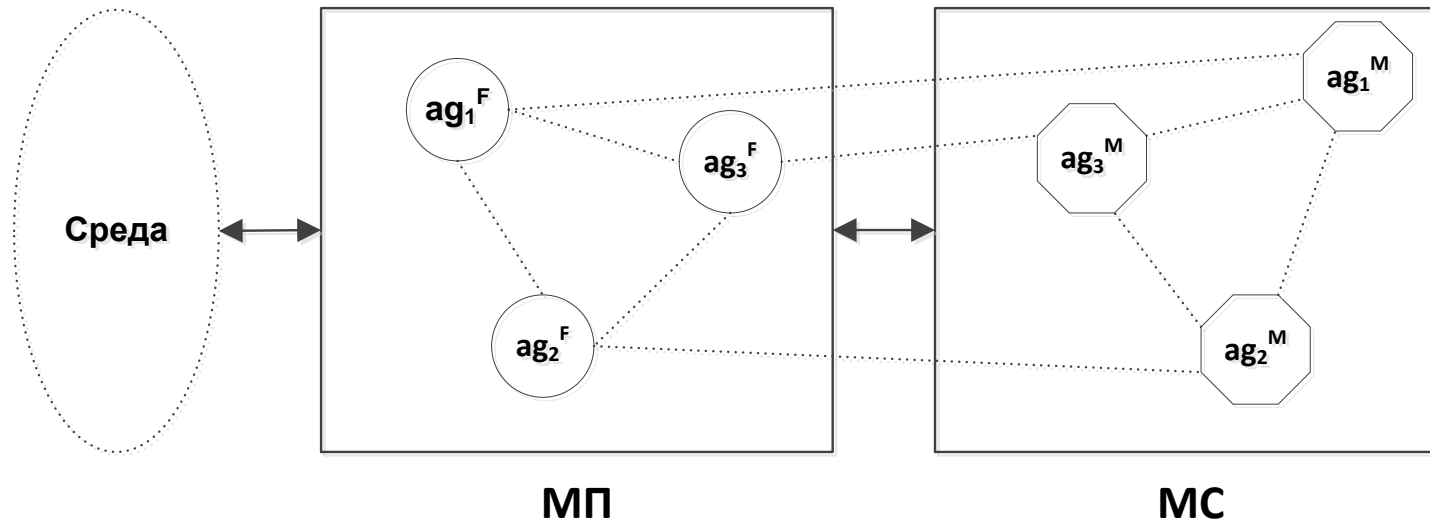
Динамический метаграфовый агент

- Динамический метаграфовый агент:

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

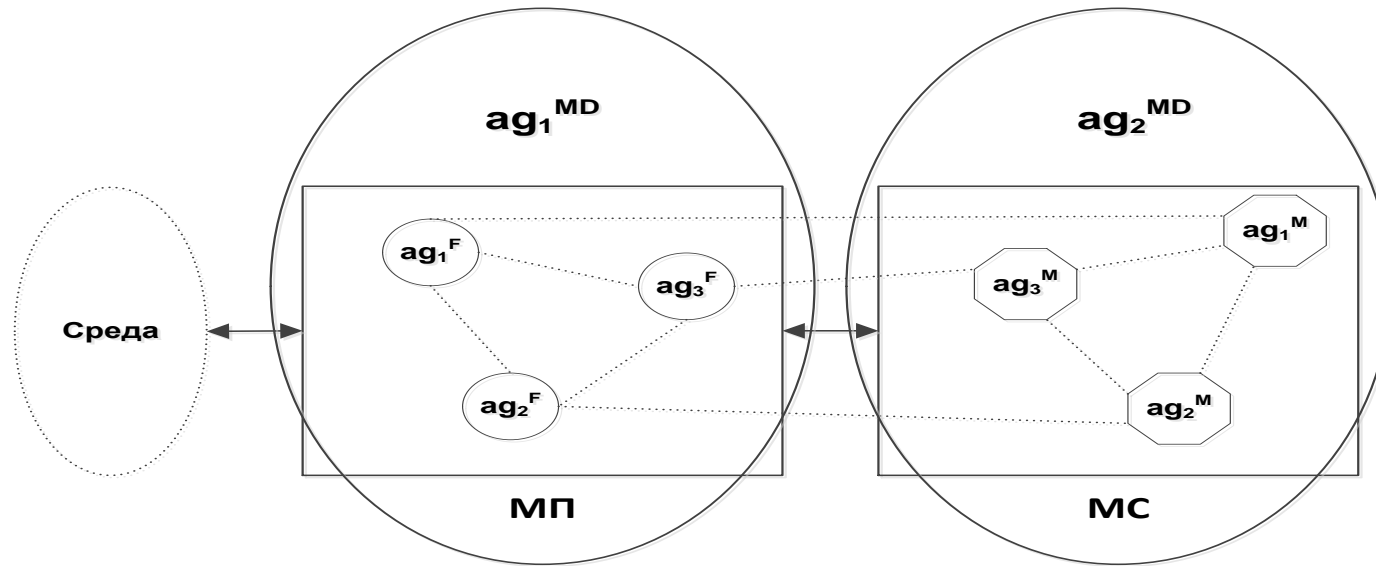
- где ag^{MD} – динамический метаграфовый агент; ag^C – контейнерный агент, на метаграфе которого выполняются правила агента; R – набор правил (множество правил r_j); AG^{ST} – стартовое условие выполнения агента (фрагмент метаграфа, который используется для стартовой проверки правил, или стартовое правило).
- Правила обработки динамического метаграфового агента выполняются не на метаграфе данных и знаний, а на метаграфе агентов для заданного контейнерного агента.
- 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.
- 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 1. Статическая структура ГИИС



- На рисунке представлена система, МП которой является нейронной сетью, а МС построен на основе обработки правил.
- Агенты-нейроны 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, возможно использование произвольной вложенности контейнеров.
- Отметим, что вся система холонических агентов представляет собой метаграф, при этом каждый агент также является метаграфом.

Пример 2. Динамическая структура ГИИС



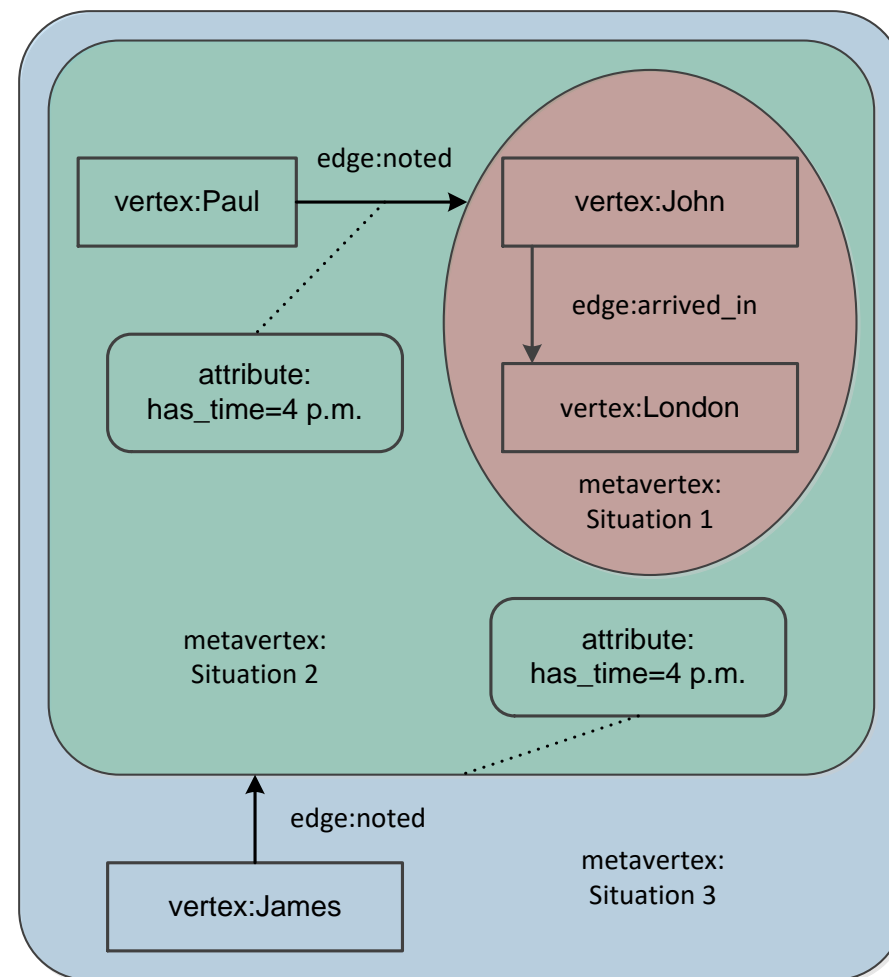
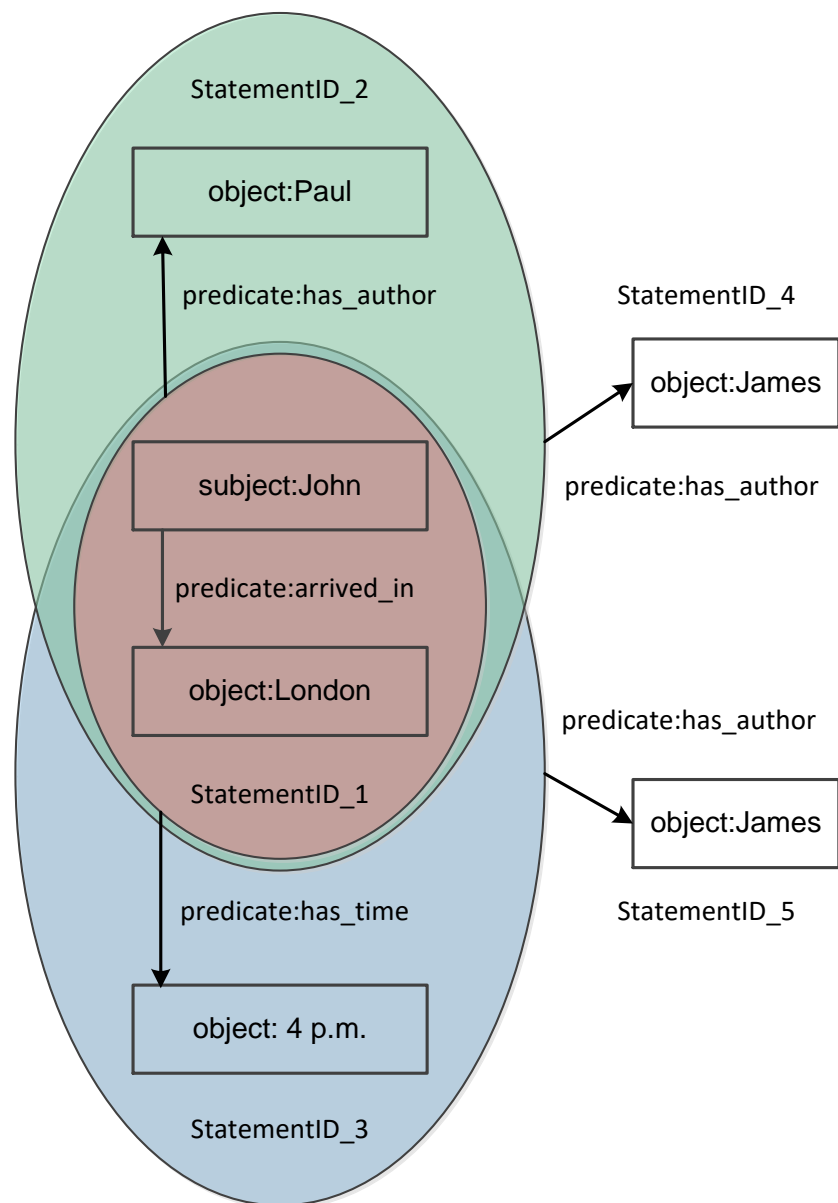
- По сравнению с предыдущим рисунком добавились два динамических метаграфовых агента.
- Агент, отвечающий за МП, может изменять структуру самоорганизующейся нейронной сети. Или может применять эволюционные методы для оптимизации конфигурации нейронной сети.
- Агент, отвечающий за МС, может изменять связи между агентами для решения задачи автоматизированного планирования. Или может применять эволюционные методы для оптимизации конфигурации агентов.
- На рисунке показаны только динамические метаграфовые агенты первого уровня, однако, количество таких уровней не ограничено. Над показанными динамическими метаграфовыми агентами могут быть надстроены динамические метаграфовые агенты более высоких уровней.

Сравнение метаграфового подхода и RDF. Основные подходы к построению хранилища на основе метаграфовой модели

Сравнение метаграфового подхода и RDF

- Модель RDF широко используется для хранения знаний.
- Но проблема в том, что она плохо подходит для описания сложных вложенных контекстов.
- Рассмотрим пример описания ситуации на естественном языке: «James noted that Paul noted at 4 p.m. that John arrived in London».
- Представление ситуации в виде RDF-триплетов:
 1. *StatementID_1 John arrived_in London*
 2. *StatementID_2 StatementID_1 has_author Paul*
 3. *StatementID_3 StatementID_1 has_time “4p.m.”*
 4. *StatementID_4 StatementID_2 has_author James*
 5. *StatementID_5 StatementID_3 has_author James*

Сравнение метаграфового подхода и RDF



Сравнение метаграфового подхода и RDF

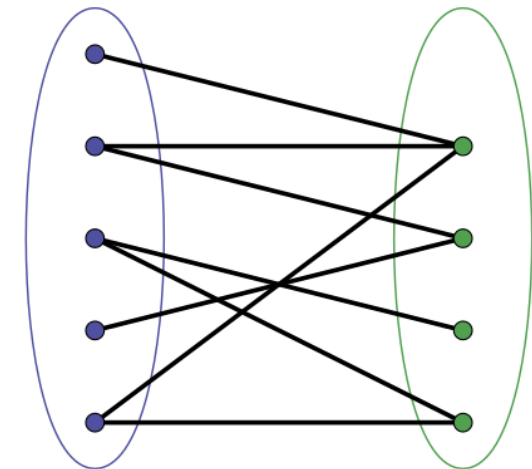
- RDF-триплет является слишком «мелким» элементом, который не позволяет описывать сложные контексты, требуется искусственно вводить вспомогательные триплеты.
- Метаграфовая модель позволяет описывать сложные контексты с использованием метавершин.

Основные подходы к построению хранилища на основе метаграфовой модели

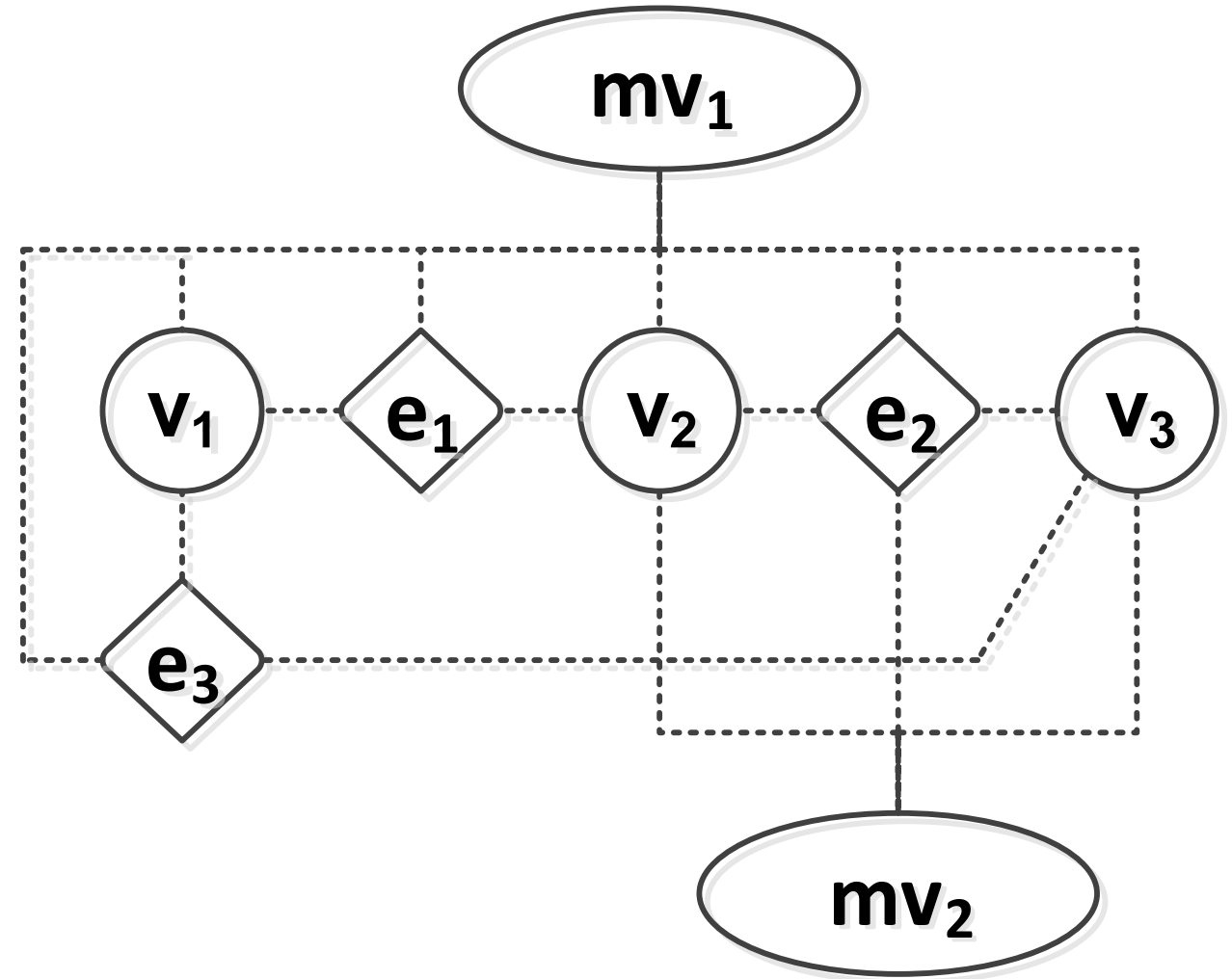
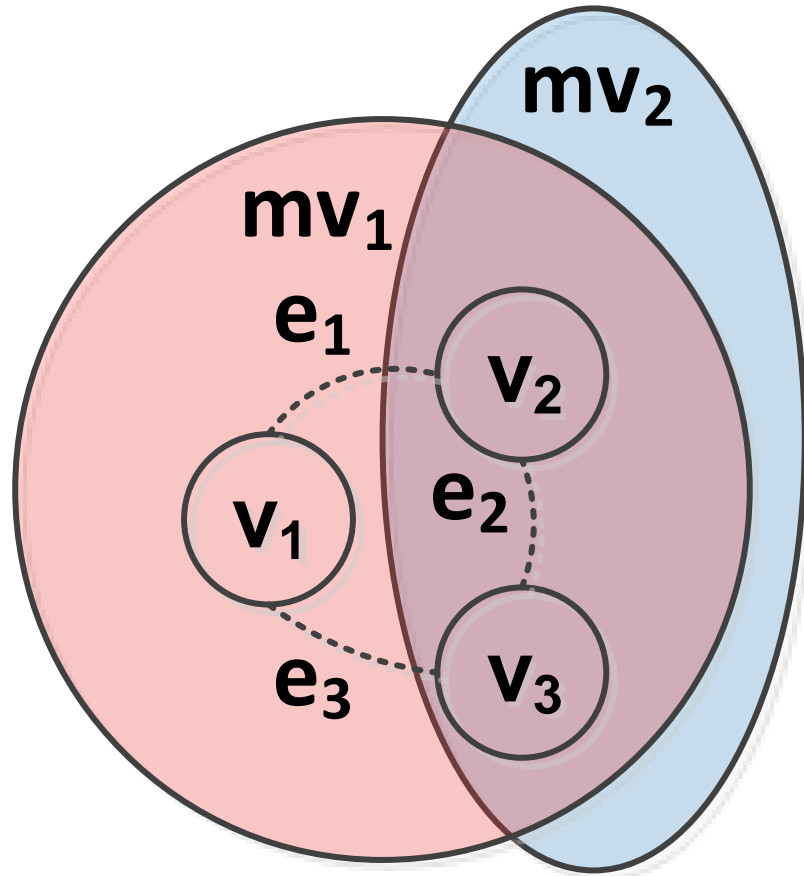
- Метаграфовая модель является аналогом «логической» модели СУБД и ей могут соответствовать различные «физические» модели.
- Мы рассмотрим три варианта «физической» модели:
 1. Модель на основе плоских графов.
 2. Документно-ориентированная модель.
 3. Реляционная модель.

Хранение – плоские графы

- Чтобы превратить холоническую (иерархическую) метаграфовую модель в плоскую графовую модель можно использовать подход на основе многодольных графов.
- Двудольный граф позволяет превратить вершины и ребра графа в подмножества вершин другого графа: $FG = \langle FG^V, FG^E \rangle$, $BFG = \langle BFG^{VERT}, BFG^{EDGE} \rangle$, $BFG^{VERT} = \langle FG^{BV}, FG^{BE} \rangle$, $FG^V \leftrightarrow FG^{BV}$, $FG^E \leftrightarrow FG^{BE}$
- Для метаграфа используется трехдольный граф:
- $TFG = \langle TFG^{VERT}, TFG^{EDGE} \rangle$, $TFG^{VERT} = \langle TFG^V, TFG^E, TFG^{MV} \rangle$,
 $TFG^V \leftrightarrow MG^V$, $TFG^E \leftrightarrow MG^E$, $TFG^{MV} \leftrightarrow MG^{MV}$



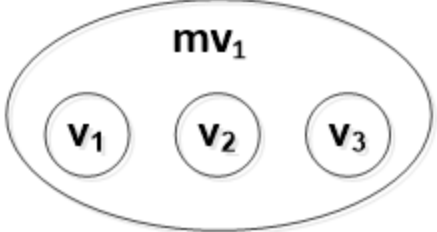


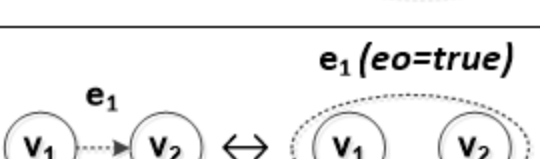
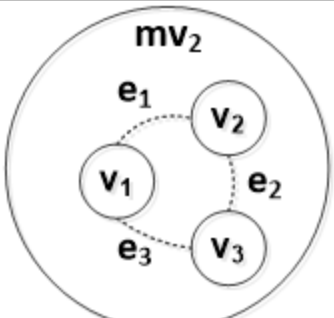
Хранение – плоские графы (пример)

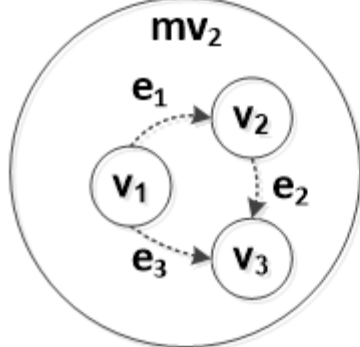
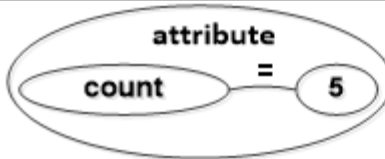
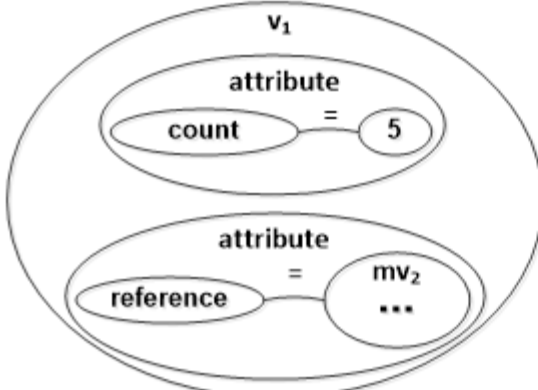


Хранение – документы

- В качестве документоориентированного представления используется Prolog-подобное предикатное описание: *predicate(atom, \dots, key = value, \dots, predicate(\dots), \dots)*.
- Структуры, используемые в данном описании изоморфны структурам, применяемым в JSON-модели: иерархически организованные массивы и пары ключ-значение.
- Основные элементы метаграфовой модели могут быть отображены в предикатное описание:

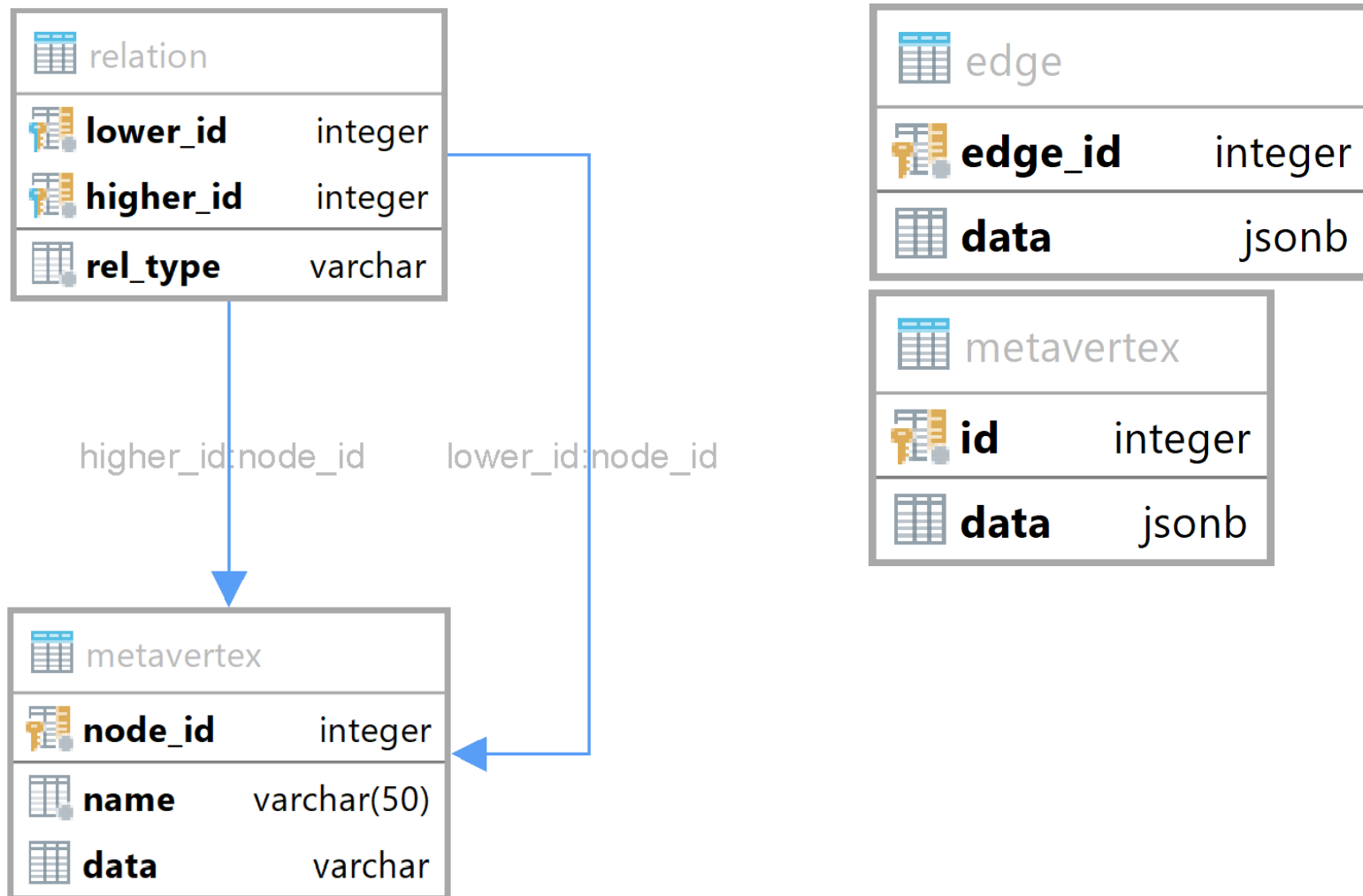
Хранение – документы

	Metavertex(Name=mv ₁ , v ₁ , v ₂ , v ₃)
	Edge(Name=e ₁ , v ₁ , v ₂)
	Edge(Name=e ₁ , v ₁ , v ₂ , eo=false)
	1. Edge(Name=e ₁ , v ₁ , v ₂ , eo=true) 2. Edge(Name=e ₁ , v _S =v ₁ , v _E =v ₂ , eo=true)
	Metavertex(Name=mv ₂ , v ₁ , v ₂ , v ₃ , Edge (Name=e ₁ , v ₁ , v ₂), Edge(Name=e ₂ , v ₂ , v ₃), Edge(Name=e ₃ , v ₁ , v ₃))

	Metavertex(Name=mv ₂ , v ₁ , v ₂ , v ₃ , Edge(Name=e ₁ , v _S =v ₁ , v _E =v ₂ , eo=true), Edge(Name=e ₂ , v _S =v ₂ , v _E =v ₃ , eo=true), Edge(Name=e ₃ , v _S =v ₁ , v _E =v ₃ , eo=true))
	Attribute(count, 5)
	Vertex(Name=v ₁ , Attribute(count, 5), Attribute(reference, mv ₂))

Хранение – реляционная модель

- Использование чистого реляционного подхода или документно-ориентированного хранилища, встроенного в реляционную СУБД.



Выводы (по результатам экспериментов с хранением метаграфовой модели)

- В случае добавления, обновления и удаления данных наиболее эффективным вариантом является использование PostgreSQL. Но в случае выборки иерархических данных графовые СУБД показывают значительно лучшие результаты.
- Для графовых СУБД (как для ArangoDB так и для Neo4j) время выполнения запросов на получение иерархических данных сопоставимо с временем выполнения запросов на добавление, обновление и удаление данных.
- В PostgreSQL добавление, обновление и удаление данных в целом производятся быстрее, чем в графовых СУБД или за сопоставимое время. Но при этом время выполнения запросов на получение иерархических данных во много раз больше времени добавления, обновления и удаления данных.
- В целом наиболее производительным вариантом является использование СУБД ArangoDB с плоской графовой моделью.

Примеры использования архитектуры ГИИС

Пример. Моделирование работы нейронной сети с использованием метаграфового подхода

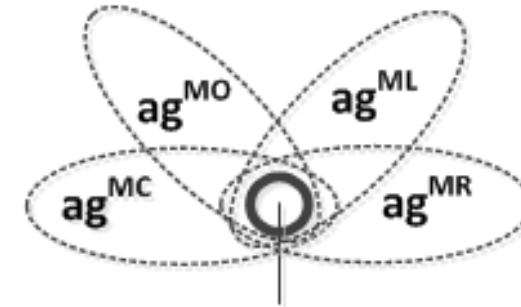
- На следующих 4 слайдах показан пример моделирования нейронной сети с использованием метаграфового подхода.
- Слайд А – Общая схема функционирования нейронной сети на основе метаграфовых агентов. Данная схема не рассматривает детально метаграфовое представление структуры отдельных нейронов, что рассматривается на слайдах Б, В, Г.
- Слайд Б – Описание функционирования персептрона на основе метаграфового подхода. Персептрон является базовым элементом для построения простых нейронных сетей.
- Слайды В,Г – Варианты описания глубокой нейронной сети с использованием различных стратегий регуляризации.
- Исследования в области метаграфового описания нейронных сетей продолжаются.

А) Описание нейронной сети с помощью метаграфовых агентов

- Представление нейронной сети в виде метаграфа может быть реализовано с помощью метаграфовых агентов.
- С использованием метаграфовых агентов может быть смоделирована работа нейросети в различных режимах.
- В примере используются следующие динамические метаграфовые агенты:

1. ag^{MC} – агент создания нейросети; 2. ag^{MO} – агент изменения нейросети;
3. ag^{ML} – агент обучения нейросети; 4. ag^{MR} – агент запуска нейросети.

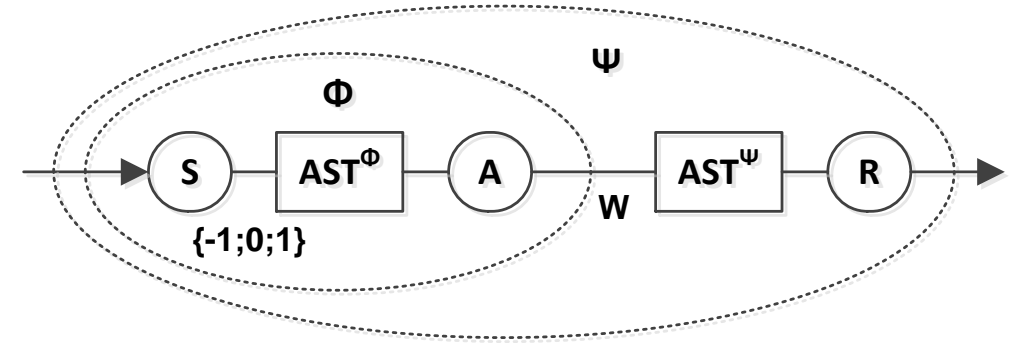
- Агент создания нейросети (ag^{MC}) реализует правила создания начальной топологии нейросети. Данный агент содержит как правила создания отдельных нейронов, так и правила соединения нейронов в нейросеть, в частности создает AST агентов-функций, моделирующих отдельные нейроны.
- Агент изменения нейросети (ag^{MO}) содержит правила изменения топологии сети в процессе работы. Это особенно важно для сетей с изменяемой топологией, таких как SOINN.
- Агент обучения нейросети (ag^{ML}) реализует один из алгоритмов обучения. При этом в результате обучения измененные значения весов записываются в метаграфовое представление нейросети. Возможна реализация нескольких алгоритмов обучения с использованием различных наборов правил для агента ag^{ML} .
- Агент запуска нейросети (ag^{MR}) реализует запуск и работу обученной нейросети в штатном режиме.
- Отметим, что агенты могут работать как независимо, так и совместно. Например, при обучении сети SOINN агент ag^{ML} может вызывать правила агента ag^{MO} для изменения топологии в процессе обучения.
- Каждый агент на основе заложенных в него правил фактически реализует специфическую программную «машину». Использование метаграфового подхода позволяет реализовать принцип «мультимашинности», когда несколько агентов с различными целями реализуют различные действия на одной и той же структуре данных.



**Представление нейронной
сети в виде метаграфа**

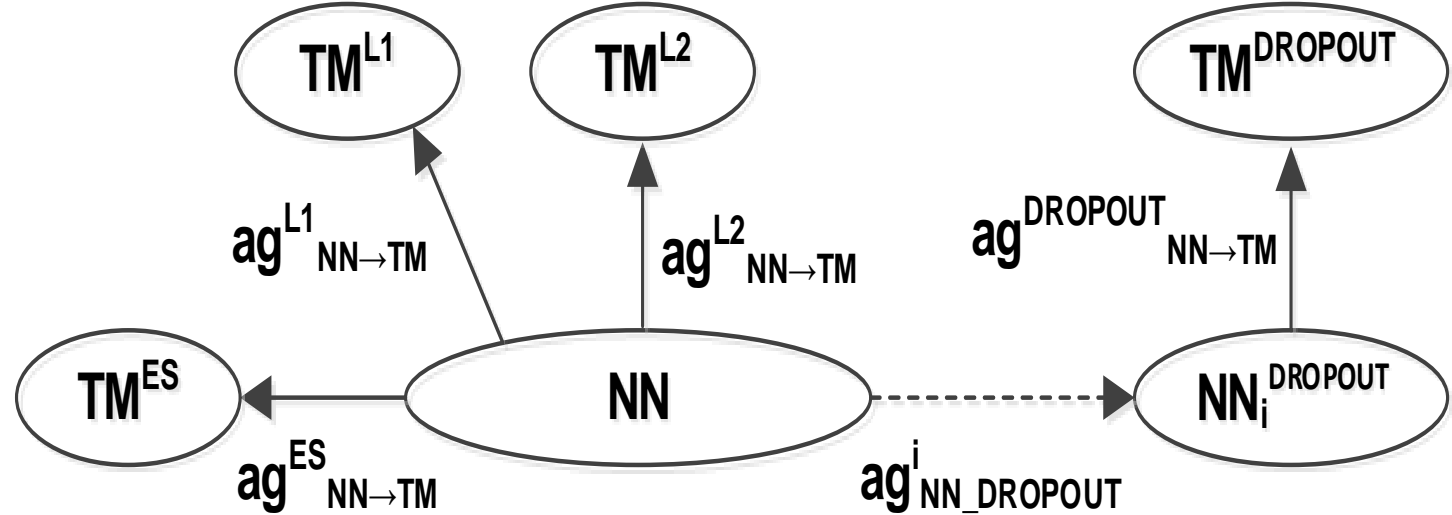
Б) Метаграфовое представление персептрона

- В соответствии с моделью Розенблатта, персептрон состоит из трех уровней: S, A и R.
- Слой сенсоров (S) представляет собой набор входных сигналов. Ассоциативный слой (A) включает набор промежуточных элементов, которые активизируются, если одновременно активизируется некоторый набор (образ) входных сигналов. Сумматор (R) активизируется, если одновременно активизируется некоторый набор A-элементов.
- В соответствии с обозначениями, принятыми в работе М. Минского и С. Пайперта, значение сигнала на A-элементе может быть представлено в виде предиката $\phi(S)$ а значение сигнала на сумматоре в виде предиката $\psi(A, W)$, где W – вектор весов. Под предикатом здесь понимается функция, принимающая только два значения «0» и «1».
- В зависимости от конкретного вида персептрона вид предикатов $\phi(S)$ и $\psi(A, W)$ может быть различным. Как правило, с помощью предиката $\phi(S)$ проверяется, что суммарный входной сигнал от сенсоров не превышает некоторый порог. Также с помощью предиката $\psi(A, W)$ проверяется, что взвешенная сумма от A-элементов не превышает некоторого порога.
- В нашем случае конкретный вид предикатов не важен, важно то, что предикаты являются обычными функциональными зависимостями, которые на программном уровне могут быть представлены в виде абстрактного синтаксического дерева и следовательно могут быть смоделированы агентами-функциями $\varphi^F = \langle S, A, AST^\phi \rangle$, $\psi^F = \langle \langle \{\varphi^F\}, W \rangle, R, AST^\psi \rangle$. Структура агентов-функций представлена на рисунке.
- Описание функций может содержать различные параметры, например пороги, но мы предполагаем, что эти параметры входят в описание абстрактного синтаксического дерева и могут быть переписаны с использованием метаграфовых агентов верхнего уровня.



В) Описание регуляризации с помощью метаграфов - 1

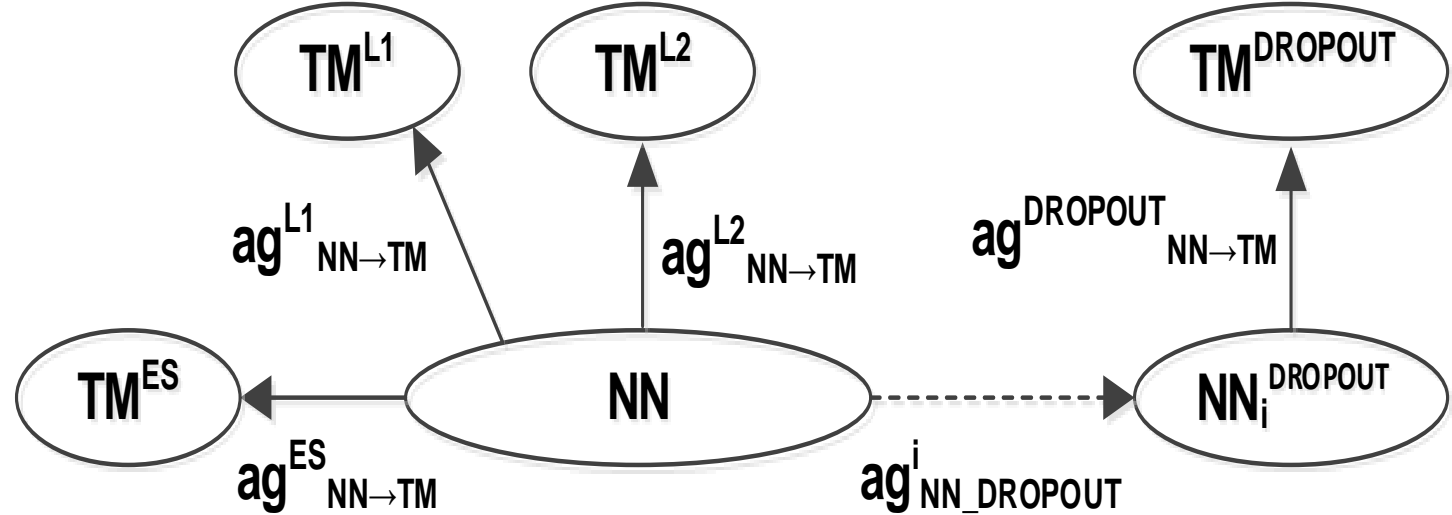
- Как и в случае отдельного персептрона, глубокая нейронная сеть может быть создана с использованием метаграфовых агентов. В зависимости от целей моделирования, нейрон может быть представлен в виде простой вершины или сложной метавершины, содержащей детализированное описание нейрона.
- На шаге создания нейросети создается структура “Neural Network” (NN), которая является плоским графом из нейронов, соединенных связями. Нейрон может быть описан как персептрон или аналогичным образом. Но при этом внутреннее представление нейрона может быть раскрыто в форме метавершины. Также в форме метавершины может быть представлен каждый уровень нейросети.



- В режиме обучения создается структура “Training Metagraph” (TM), изоморфная NN. Для каждой вершины-нейрона в NN создается метавершина в TM, связи сохраняются. Для создания TM на основе NN используется агент-функция $ag_{NN \rightarrow TM}$.
- TM является активным метаграфом, в котором с метаграфом данных связан метаграфовый агент ag_{TM} , отвечающий за обучение сети. Результаты обучения сохраняются в TM.
- Поскольку агенты могут быть представлены в виде метаграфов, то агент ag_{TM} создается с помощью агента $ag_{NN \rightarrow TM}$.

Г) Описание регуляризации с помощью метаграфов - 2

- Агент ag_{TM} может использовать различные стратегии регуляризации.
- Для NN может быть создано несколько структур TM, использующих различные стратегии регуляризации: L1, L2, ES (Early Stopping), dropout (обозначены верхними индексами).
- В случае использования стратегий L1, L2, ES не требуется изменения структуры графа нейросети в процессе обучения. Соответствующие агенты являются агентами-функциями.



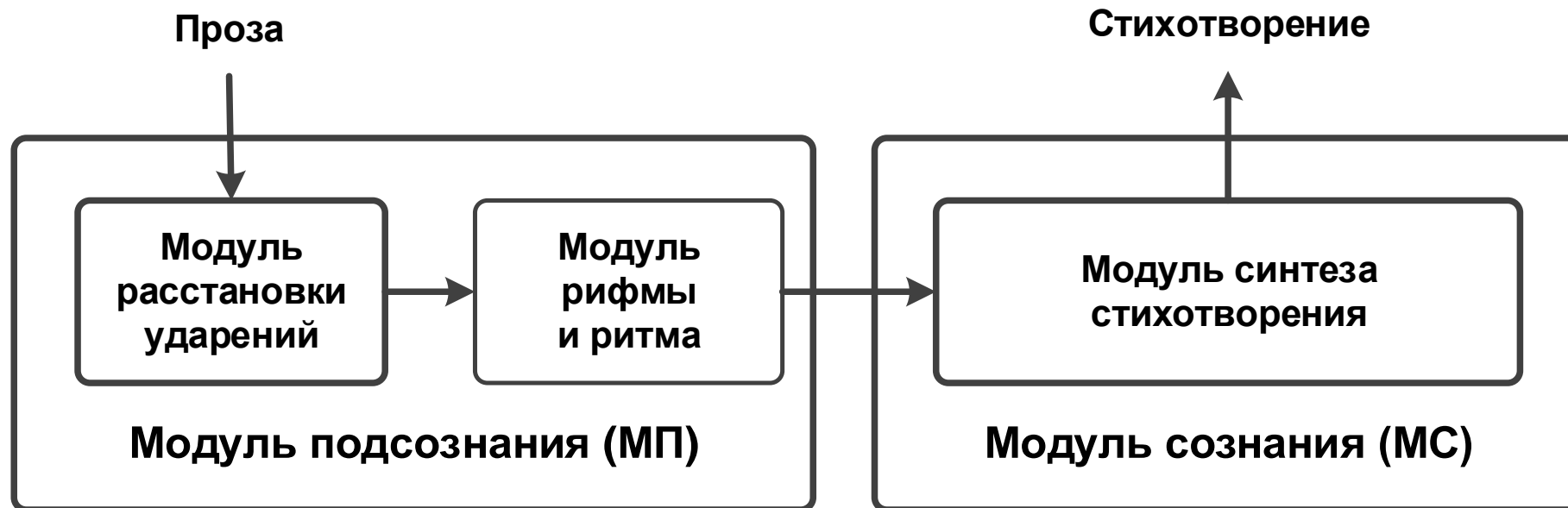
- В случае использования стратегии dropout структура нейросети должна быть изменена. При использовании данной стратегии сначала применяется метаграфовый агент $ag_{NN_DROPOUT}$ (в отличие от агентов-функций показан пунктирной стрелкой) который на основе правил модифицирует NN и формирует новую структуру NN^{DROPOUT} (индекс i показывает, что может быть сформировано несколько структур для различных вариантов dropout). Далее для NN^{DROPOUT} создается соответствующая структура TM^{DROPOUT} используемая для обучения сети.
- Альтернативой данному подходу является встраивание правил в агент ag_{TM} , которые будут использовать dropout уже в TM-структуре без необходимости модификации исходной структуры NN.
- Таким образом:
 - Нейронная сеть может быть представлена в виде метаграфа данных.
 - С использованием агентов-функций можно трансформировать структуру нейросети.
 - С использованием метаграфовых агентов можно динамически изменять структуру нейросети, моделируя различные алгоритмы обучения.

Гибридная интеллектуальная информационная система для генерации стихотворений

Maria Taran, Georgiy Revunkov, Yuriy Gapanyuk. The Hybrid Intelligent Information System for Poems Generation. NEUROINFORMATICS 2019: Advances in Neural Computation, Machine Learning, and Cognitive Research III. pp 78-86.

Подсознание – нейронная сеть

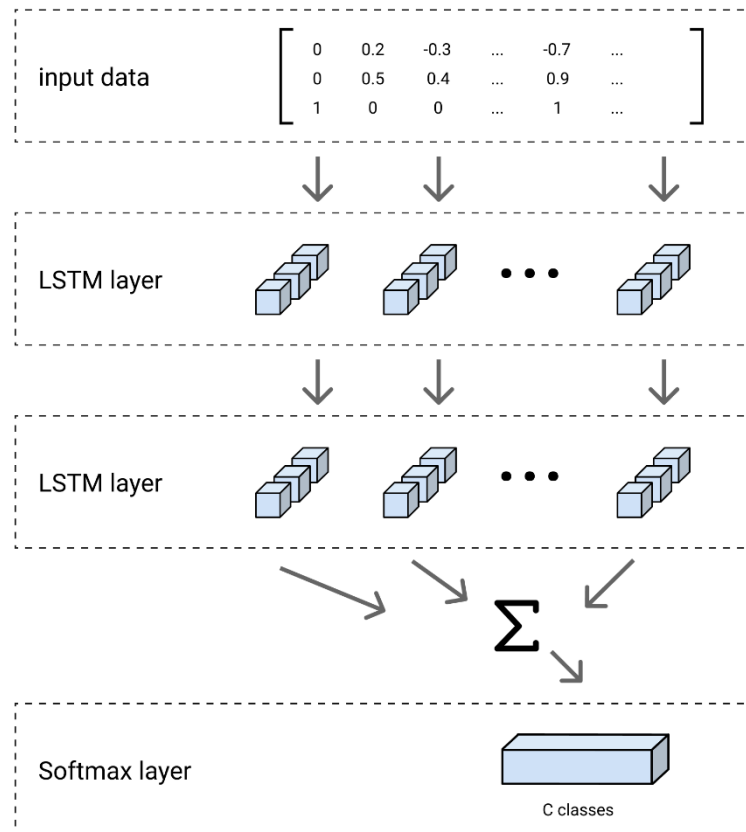
Сознание – система на правилах



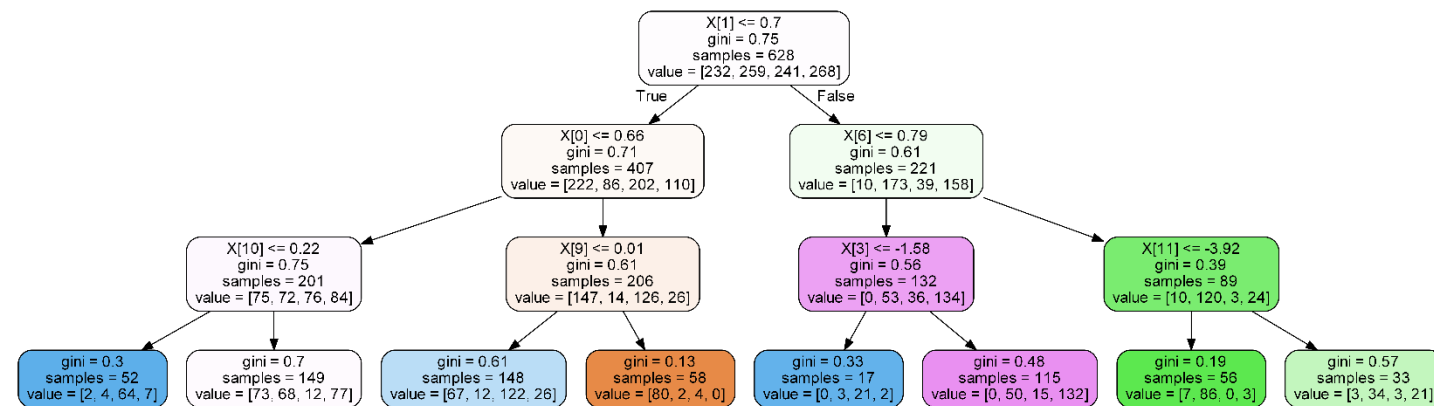
Гибридная интеллектуальная информационная система для классификации музыки

Aleksandr Stikharnyi, Alexey Orekhov, Ark Andreev, Yuriy Gapanyuk. The Hybrid Intelligent Information System for Music Classification. NEUROINFORMATICS 2019: Advances in Neural Computation, Machine Learning, and Cognitive Research III. pp 71-77.

Подсознание – нейронная сеть LSTM



Сознание – решающее дерево



Реализация ГИИС

- Классическое понимание

- В качестве методов обработки данных «подсознания» хорошо подходят методы, основанные на машинном обучении, нейронных сетях, нечеткой логике, в том числе и комбинированные нейронечеткие методы. Эти методы основаны на векторном (матричном, тензорном) представлении признаков и их обработке.
- В качестве методов обработки данных «сознания» используются онтологии и правила, экспертные системы.

- Современное понимание

- Методы, основанные на векторном представлении все активнее используются для обработки данных «сознания». Пример – [Graph Neural Network](#) (GNN).
- Использование векторного представления можно рассматривать как «**схему глубинной интеграции**» сознания и подсознания.
- Концепции «сознания» и «подсознания» ГИИС не исчезают, но могут быть реализованы унифицированным образом на основе векторного представления признаков.

Графы и метаграфы знаний

- В настоящее время в основном используются плоские графы знаний в формате RDF. Использование для хранения знаний более сложных структур – предмет дальнейших исследований.
- Основой использования графов знаний является векторное представление графов.
- В настоящее время [графы знаний](#) активно развиваются. Предполагается, что в дальнейшем графы знаний будут все более активно использоваться в интеллектуальных системах.
- Для того, чтобы использовать метаграфы знаний необходимо разработать механизмы векторного представления метаграфов. В настоящее время эти исследования только начинаются.