МЕТАГРАФОВАЯ МОДЕЛЬ ДАННЫХ И ПОДХОДЫ К ЕЕ ХРАНЕНИЮ

Гапанюк Юрий Евгеньевич, к.т.н., доцент кафедры
«Системы обработки информации и управления» (ИУ-5)
МГТУ им. Н.Э. Баумана

31 мая 2018 года

Разнородность моделей. Отсутствие гибридизации

- В классической Computer Science используются разнородные модели данных, знаний и процессов. Эта ситуация сложилась исторически, потому что раньше мощность вычислительных систем была невысока и во главу угла ставилась производительность обработки данных. Вопросы интеграции информационных систем и унификации информационных моделей оставались на втором плане.
- В настоящее время ситуация изменилась. Появление и активное развитие технологий обработки больших данных, расширение круга информационно-аналитических задач привело к тому, что в качестве обрабатываемых данных вполне могут выступать знания, ситуации, процессы. Это требует новых подходов к интеграции информационных систем, новых моделей данных.

Разнородность моделей. Отсутствие гибридизации

- Сервис-ориентированный подход (в том числе в его современном микросервисном варианте) до определенной степени решает задачу интеграции информационных систем, но при этом каждая система функционирует как черный ящик. Интеграция возможна только на уровне элементов, которые вынесены в интерфейс сервиса.
- Для объединения (оркестровки) сервисов используется технология workflow.
- Доработка SOA-системы состоит в доработке отдельных сервисов и в улучшении их оркестровки. Как правило, отдельные сервисы разрабатываются отдельными командами. То что сервисы могут использовать подобные или одинаковые алгоритмы никак не учитывается.
- Методы искусственного интеллекта рассматриваются как отдельное «нишевое» решение, которое может использоваться внутри отдельного сервиса. Например, использование ИНС для решения задачи распознавания.

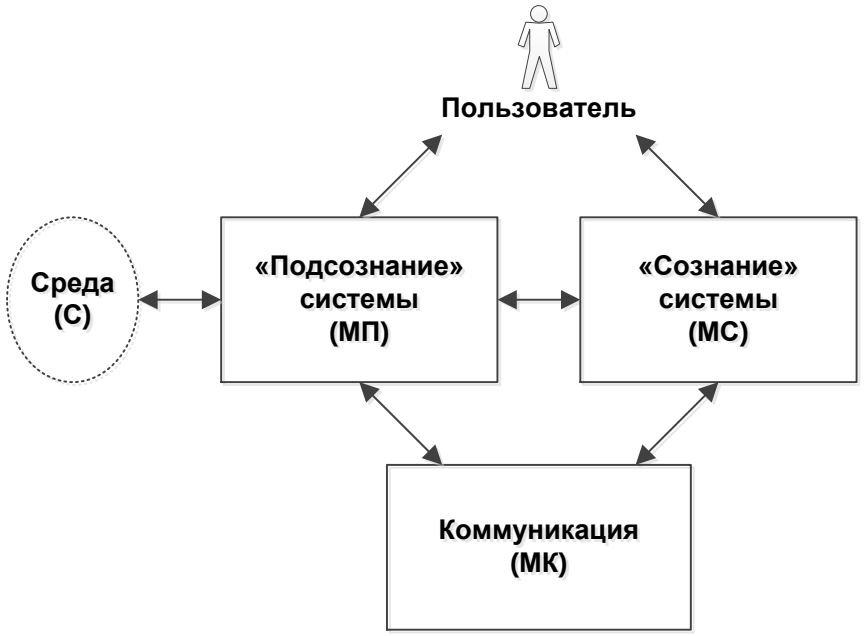
ГИС и ГИИС

- 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:

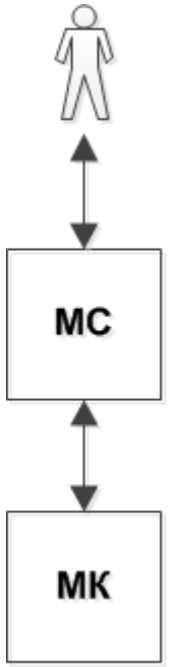
- обработка данных и знаний на основе модели данных «онтологического» типа;
- логический контроль и проверка непротиворечивости данных, поступающих от МП;
- реализация функций ввода и вывода для среды (посредством МП), для модуля коммуникации и для взаимодействия с пользователем.
- реализация функции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

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

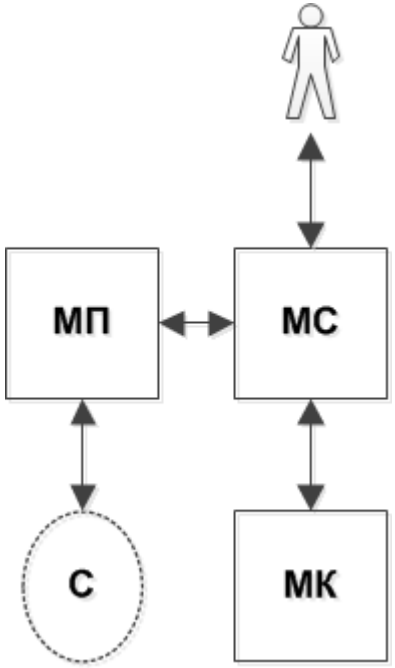


Классическая информационная система, в которой осуществляется только обработка данных и знаний (которую выполняет МС), реализуется коммуникация с другими системами (которую выполняет МК) и взаимодействие с пользователем.

Частные случаи структуры ГИИС - 2

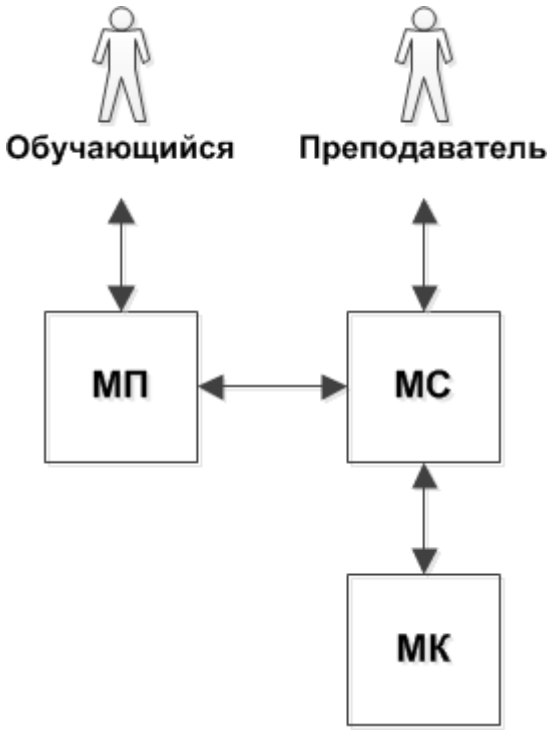
Простейшая система распознавания сигналов, поступающих из среды, с помощью МП. Сигналы могут иметь различную природу. Это может быть система распознавания музыкальной партитуры по звуковому сигналу, система распознавания элементов в видеопотоке и др. Данная система является простейшей, так как в ней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

- Сложные сети сегодня в основном анализируются как плоские графы без иерархии (состоящие из вершин и ребер).
- Тем не менее одним из главных свойств, которые используются в сложных сетях, является свойство эмерджентности.
- Под «сетью с эмерджентностью» будем понимать такую сеть, в которой отдельный фрагмент, состоящий из вершин и связей, может выступать как отдельное целое.
- В качестве примеров моделей таких сетей рассмотрим метаграфовую модель и ее преимущества перед гиперграфами и гиперсетями.

Формализованная модель метаграфа - 1

Метагараф – холонически организованный граф.

Основной теоретической работой является монография А.Базу и Р. Блэннинга – «*Basu A., Blanning R. Metagraphs and Their Applications, 2007*».

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

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

Вершина метаграфа характеризуется множеством атрибутов:

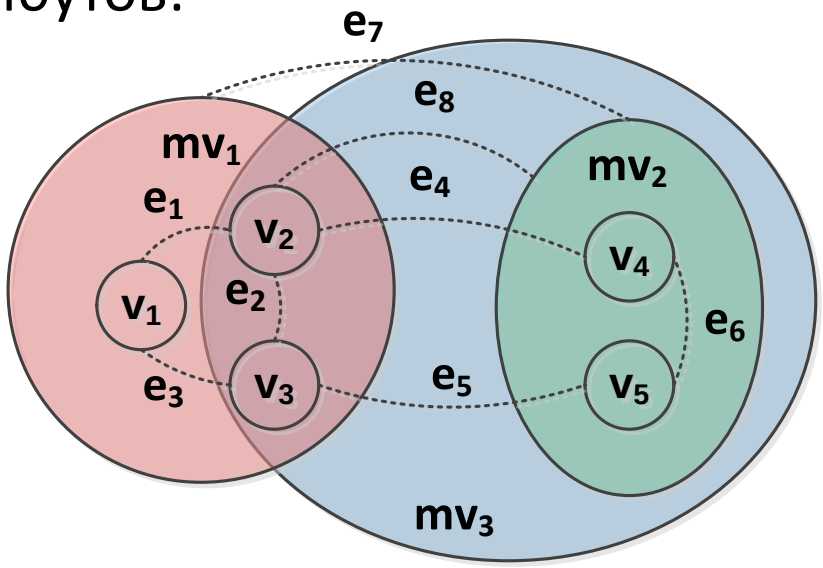
$$v_i = \{atr_k\}, v_i \in V,$$

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.



Формализованная модель метаграфа - 2

Фрагмент метаграфа:

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

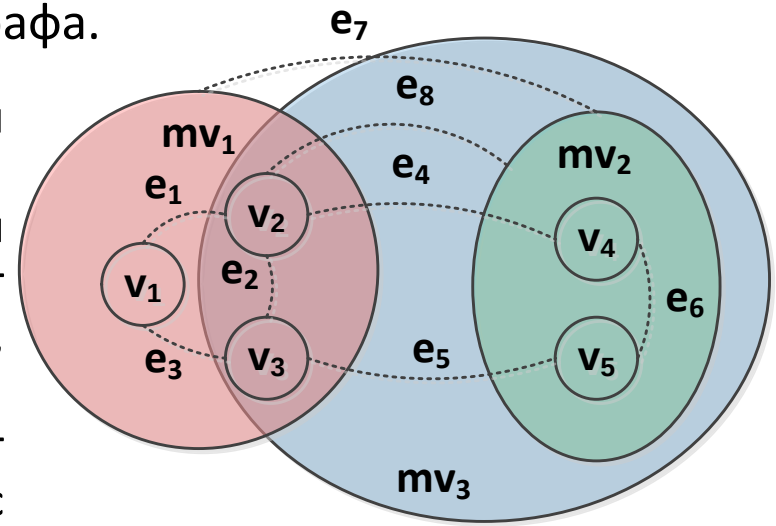
где MG_i – фрагмент метаграфа; ev_j – элемент, принадлежащий объединению множеств вершин, метавершин и ребер метаграфа.

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

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

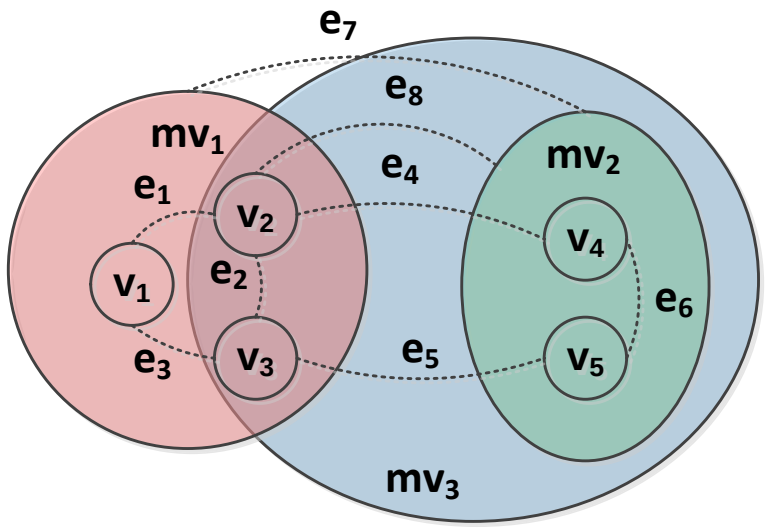
где mv_i – метавершина метаграфа; atr_k – атрибут, MG_i – фрагмент метаграфа.

- 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.



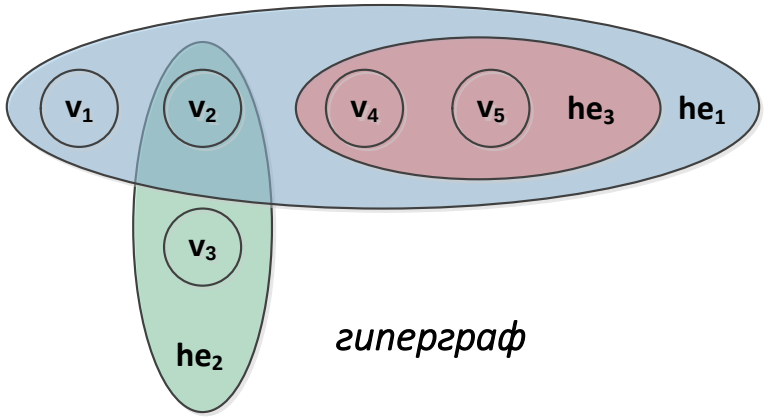
Пример описания метаграфа

- 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.

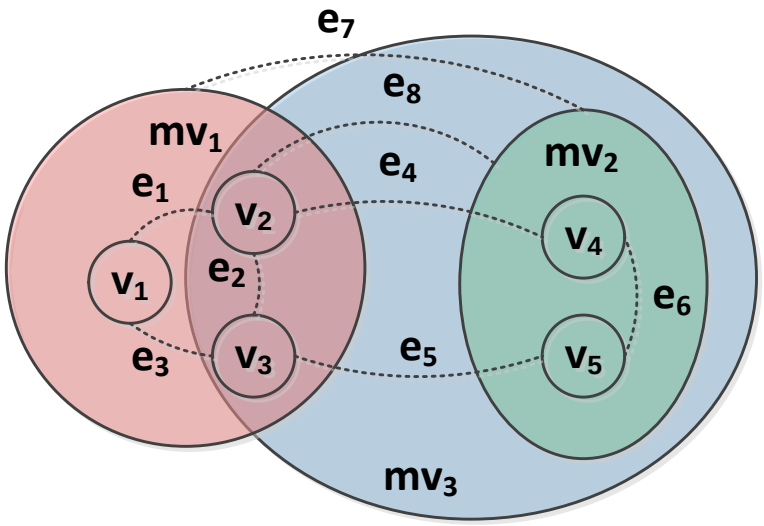


Метаграфы и гиперграфы

- 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».

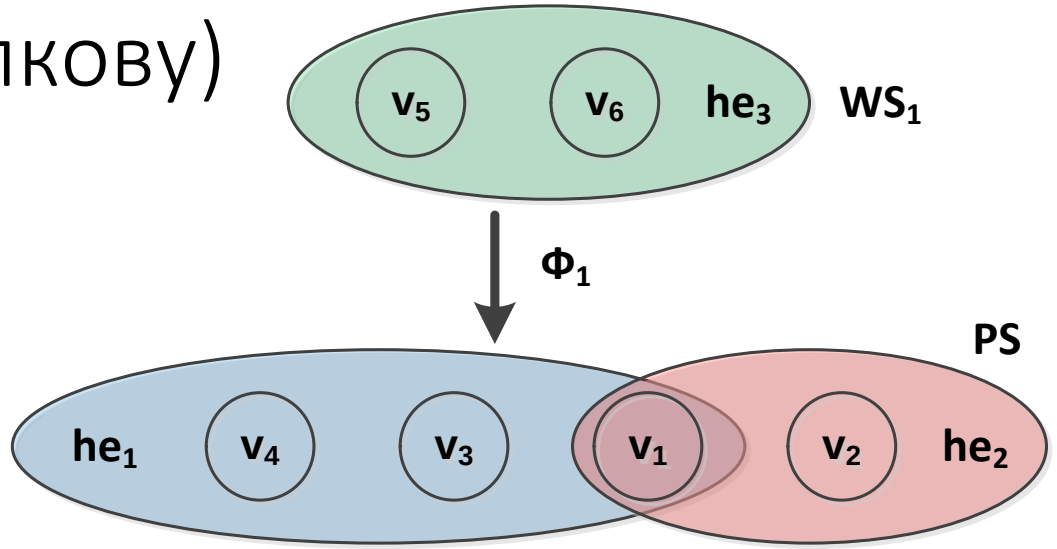


гиперграф



метаграф

Гиперсетевая модель (по В.К. Попкову)



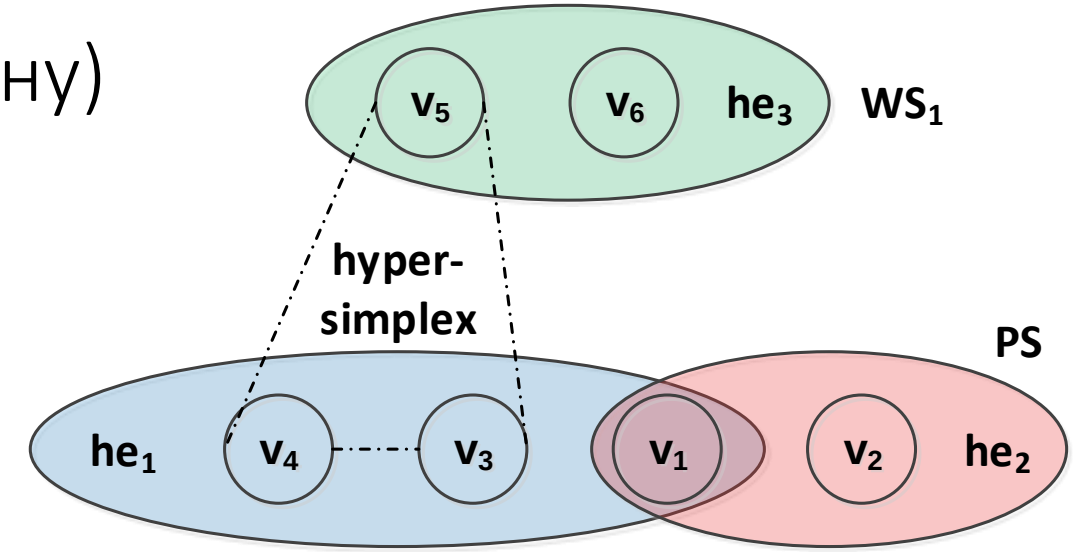
- Удивительным является факт, что гиперсетевая модель была открыта дважды.
- В первый раз гиперсетевая модель предложена д.ф.-м.н. профессором Владимиром Константиновичем Попковым в 1980-х годах.
- Фактически это первая модель «сети с эмерджентностью».
- Пусть даны гиперграфы $PS \equiv WS_0, WS_1, WS_2, \ldots WS_K$
- Гиперграф PS или WS_0 называется первичной сетью. Гиперграф WS_i называется вторичной сетью i-го порядка.
- Последовательность отображений между сетями различных уровней:

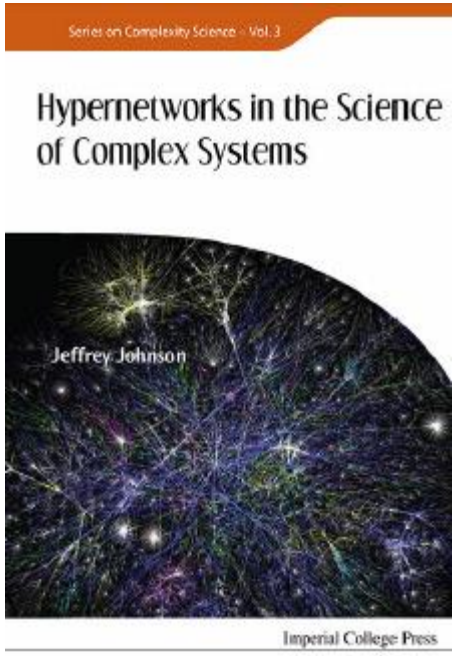
$$\{\Phi_i\}: WS_K \xrightarrow{\Phi_K} WS_{K-1} \xrightarrow{\Phi_{K-1}} \ldots WS_1 \xrightarrow{\Phi_1} PS$$

- Тогда иерархическая абстрактная гиперсеть порядка K: $AS^K = \langle PS, WS_1, \ldots, WS_K; \Phi_1, \ldots, \Phi_K \rangle$
- 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 когнитома *«Анохин К.В. Когнитом: гиперсетевая модель мозга // Нейроинформатика-2015».*
- Отметим, что гиперсимплекс, как совокупность элементов различных уровней, в теории метаграфов может быть представлен в виде метавершины (в соответствии с определением метавершины).





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 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».

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)

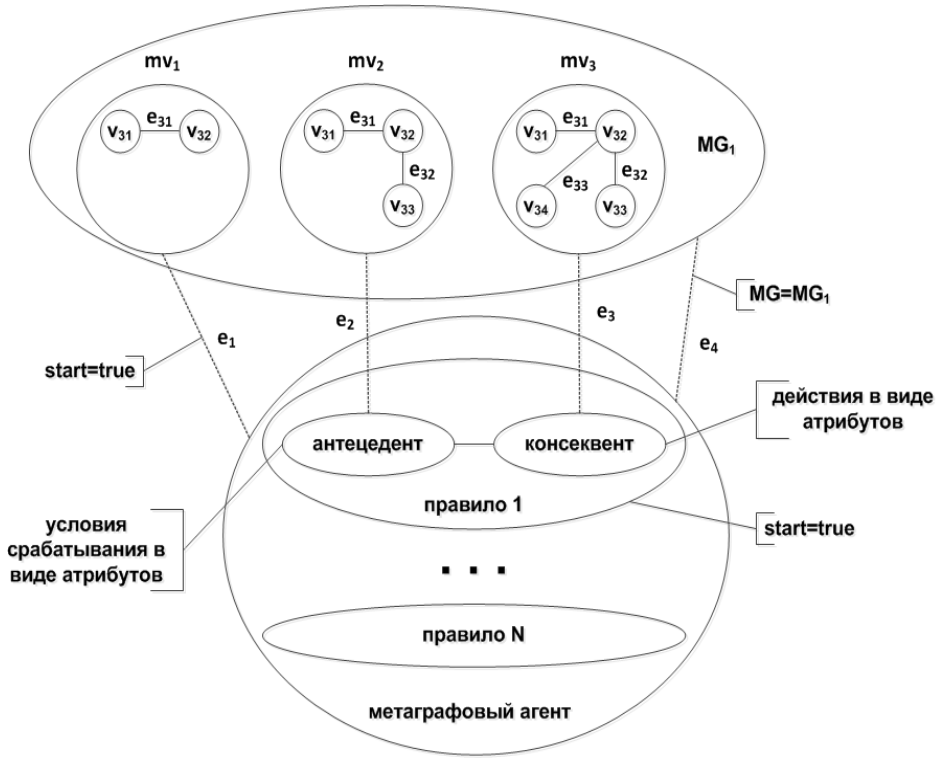
- Метаграфовый агент:

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

- 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:

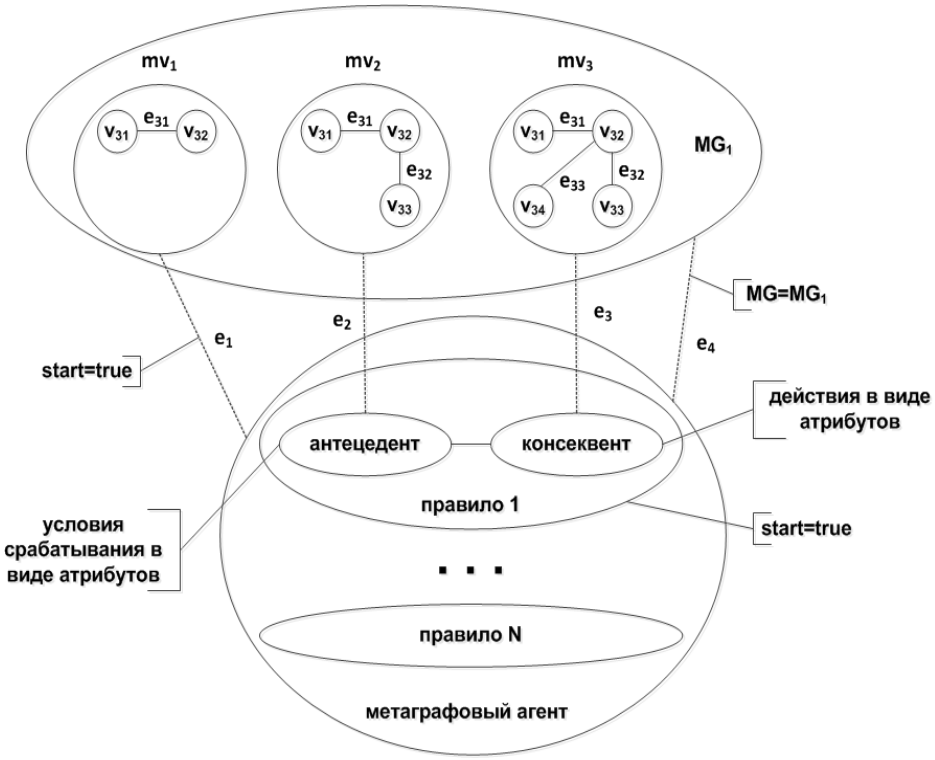
$$r_i : MG_j \rightarrow OP^{MG},$$

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



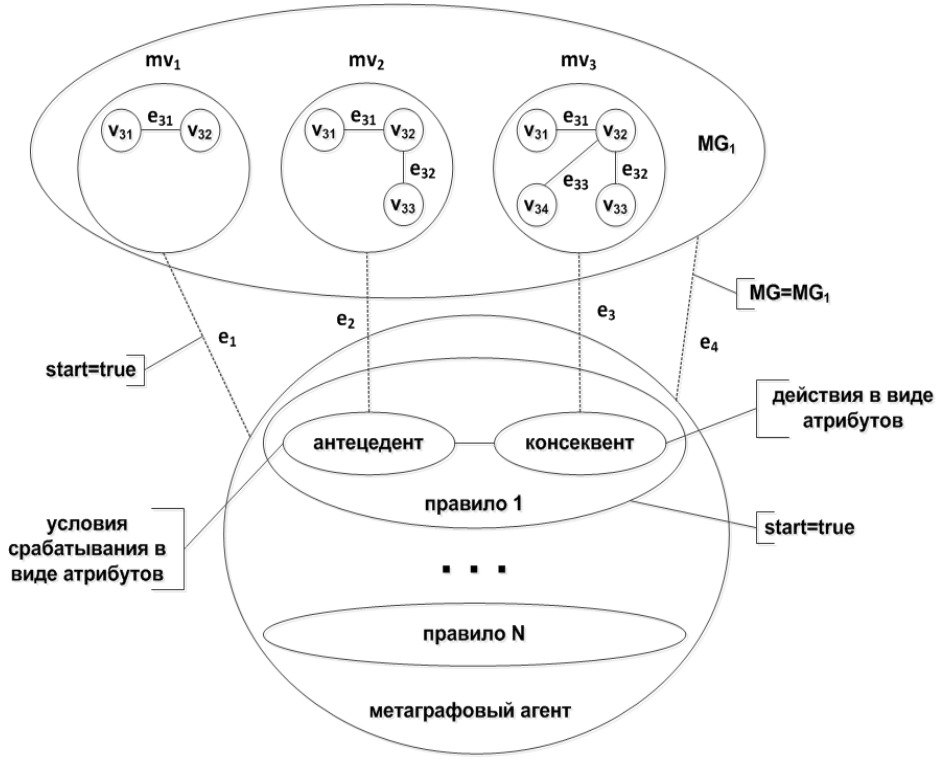
Метаграфовый агент (правила)

- 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 зацикливанию 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.

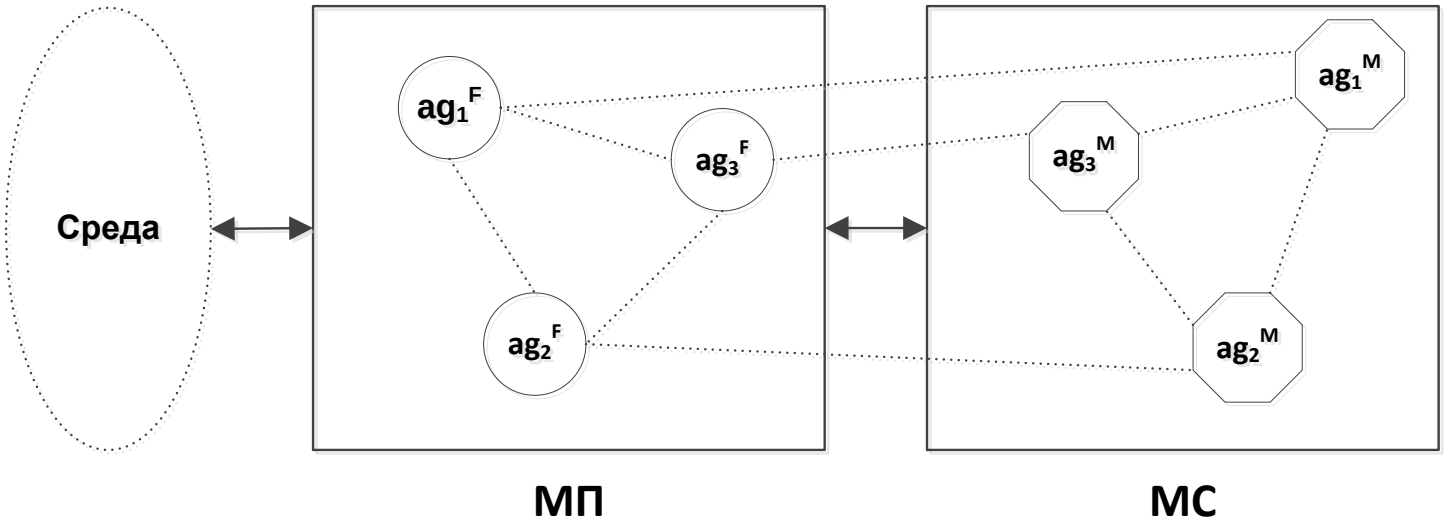
Динамический метаграфовый агент

- Динамический метаграфовый агент:

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

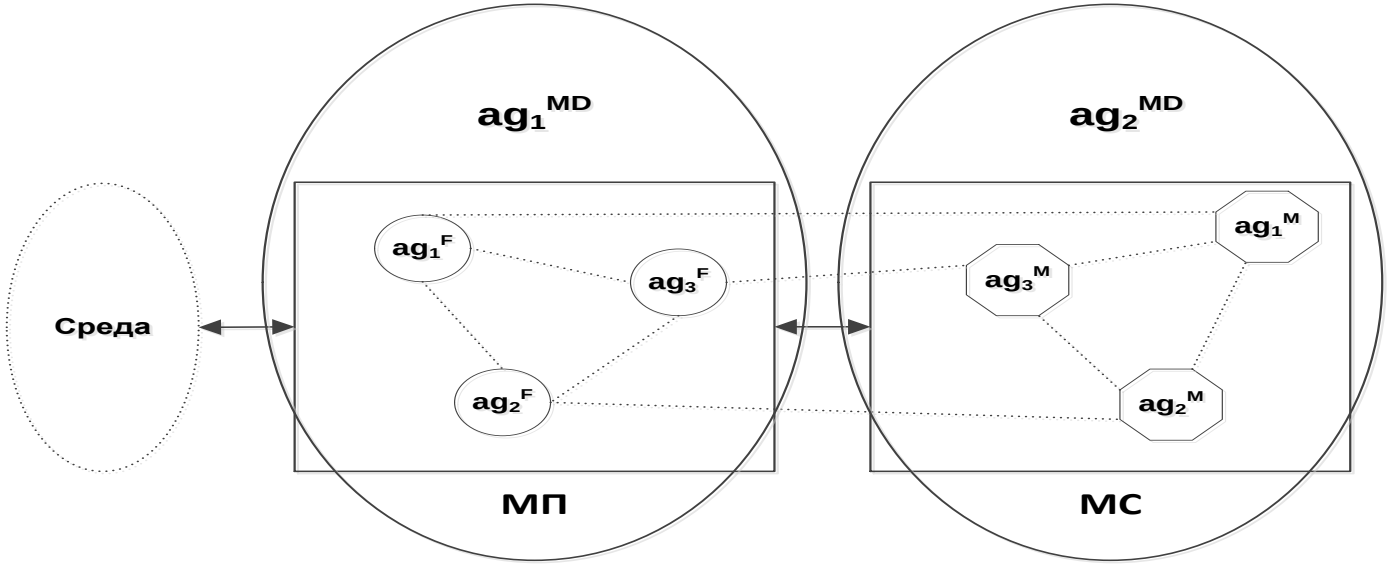
- где ag^{MD} – динамический метаграфовый агент; ag^{C} – контейнерный агент, на метаграфе которого выполняются правила агента; R – набор правил (множество правил r_j); AG^{ST} – стартовое условие выполнения агента (фрагмент метаграфа, который используется для стартовой проверки правил, или стартовое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

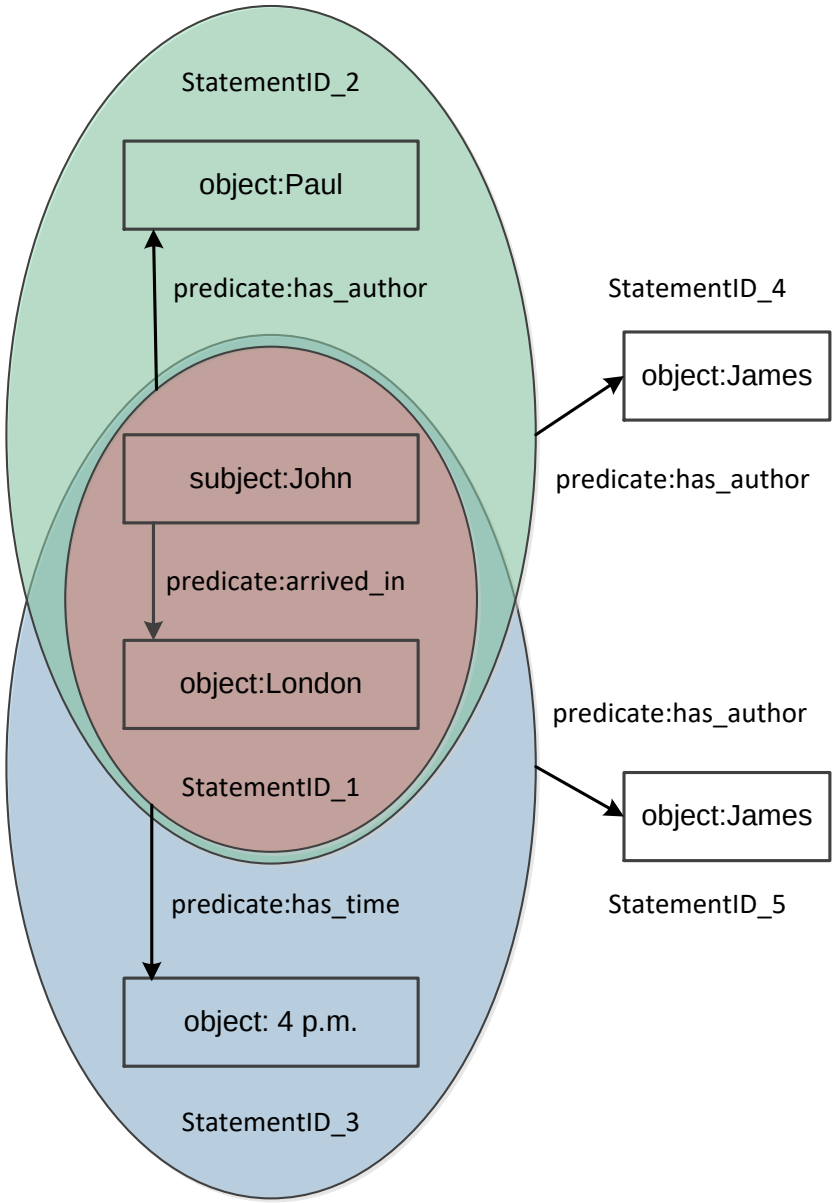


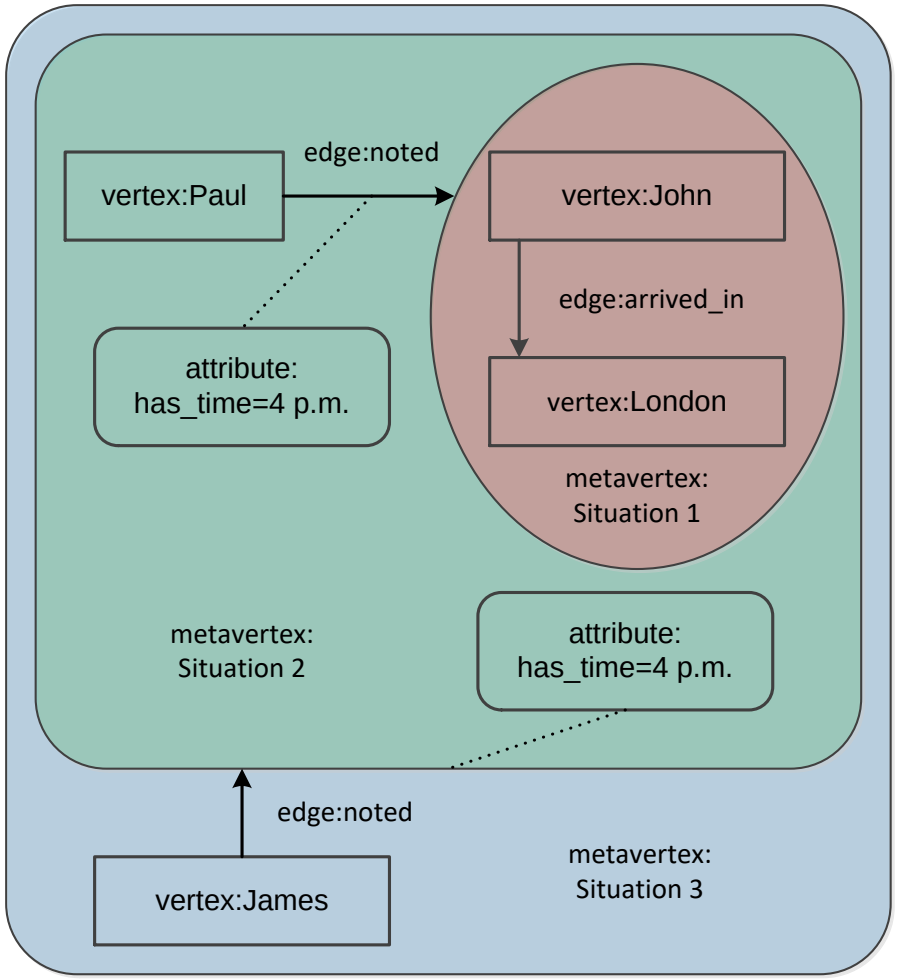
- По сравнению с предыдущим рисунком добавились два динамических метаграфовых агента.
- Агент, отвечающий за МП, может изменять структуру самоорганизующейся нейронной сети. Или может применять эволюционные методы для оптимизации конфигурации нейронной сети.
- Агент, отвечающий за МС, может изменять связи 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

- 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

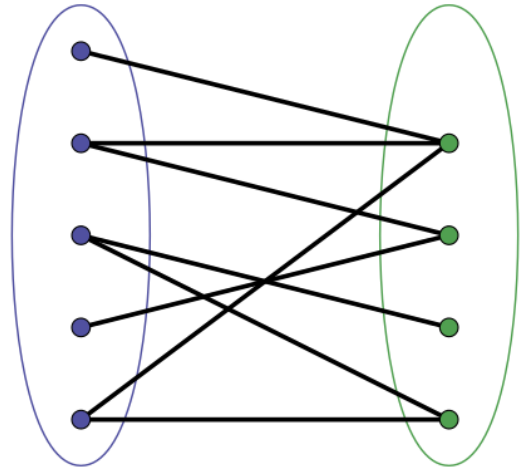
- RDF-триплет является слишком «мелким» элементом, который не позволяет описывать сложные контексты, требуется искусственно вводить вспомогательные триплеты.
- Метаграфовая модель позволяет описывать сложные контексты с использованием метавершин.

Основные подходы к хранению метаграфовой модели

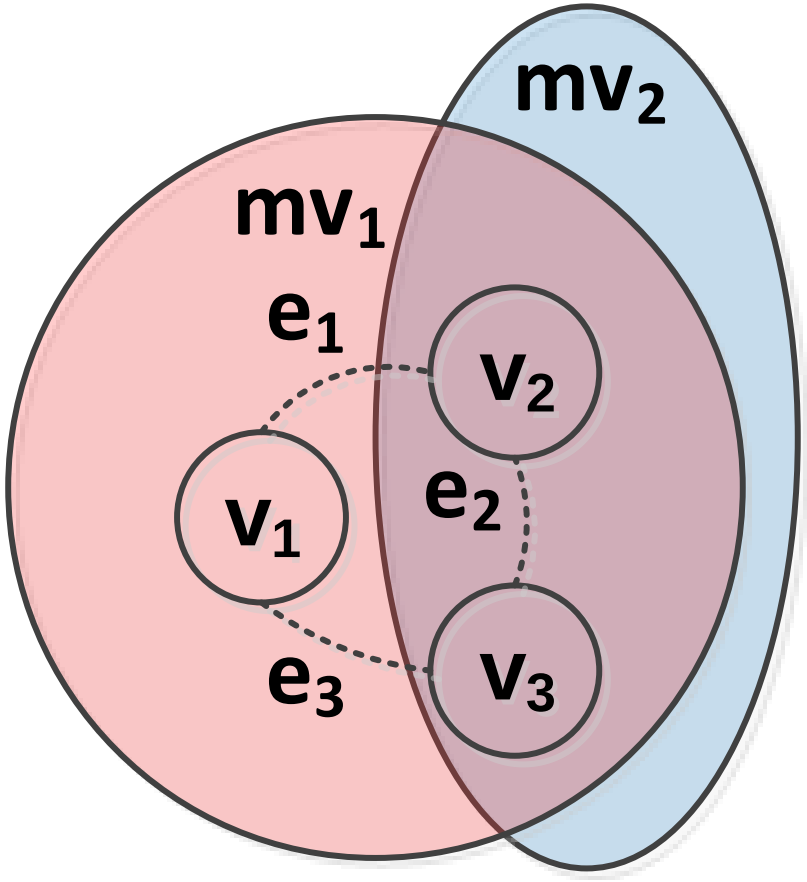
- Метаграфовая модель является аналогом «логической» модели СУБД и ей могут соответствовать различные «физические» модели.
- Мы рассмотрим три варианта «физической» модели:
  1. Модель на основе плоских графов.
  2. Документно-ориентированная модель.
  3. Реляционная модель.

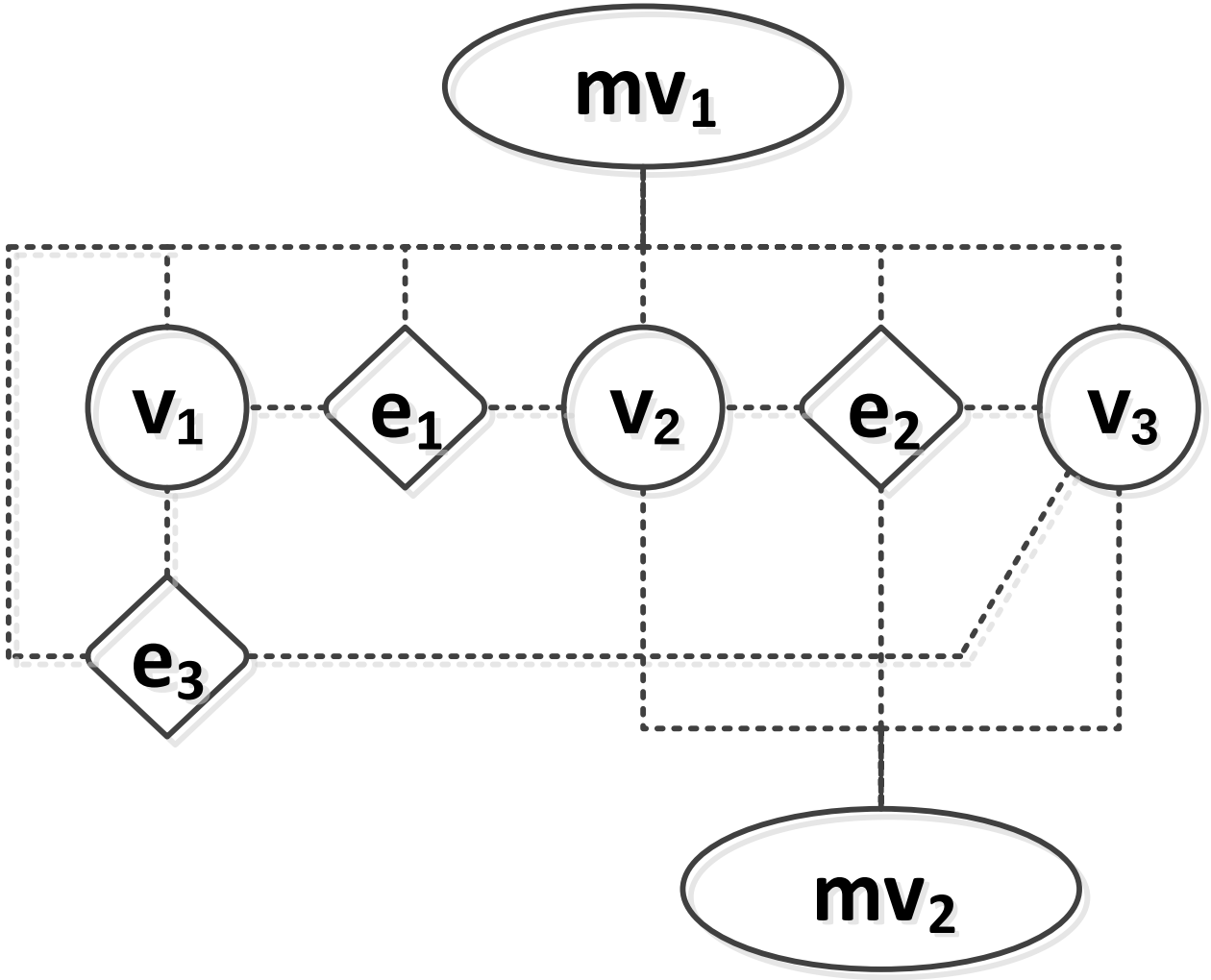
Хранение – плоские графы

- 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:
- $TFG = \langle TFG^{VERT}, TFG^{EDGE} \rangle,\quad TFG^{VERT} = \langle TFG^V, TFG^E, TFG^{MV} \rangle,$ $TFG^V \leftrightarrow MG^V, TFG^E \leftrightarrow MG^E, TFG^{MV} \leftrightarrow MG^{MV}$



Хранение – плоские графы (пример)

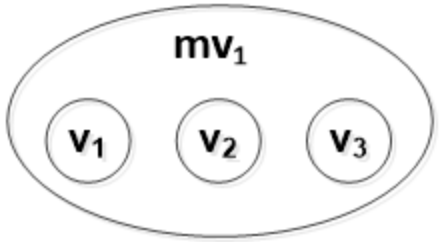
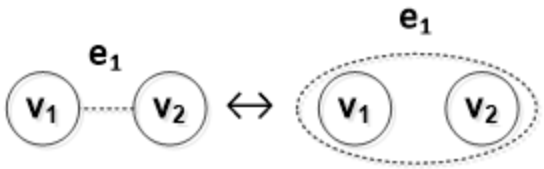
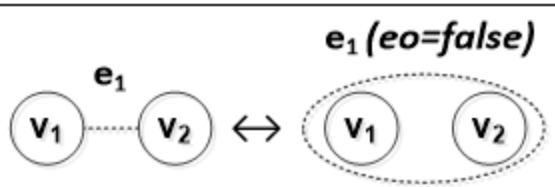
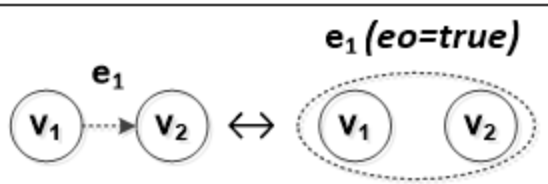
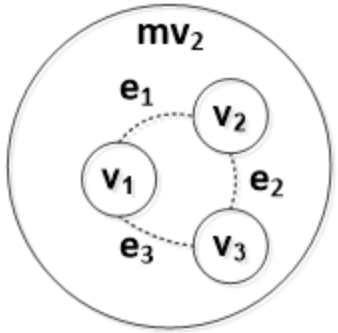


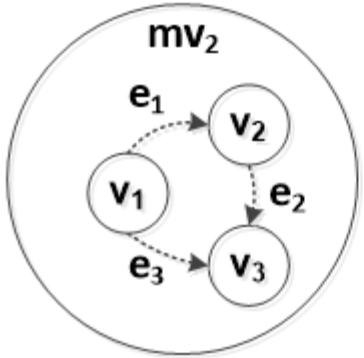
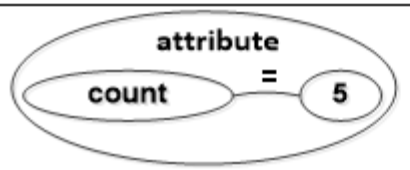
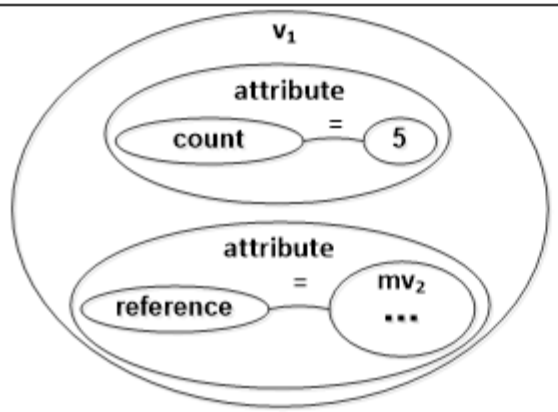


Хранение – документы

- В качестве документоориентированного представления используется Prolog-подобное предикатное описание: $predicate(atom, \cdots, key = value, \cdots, predicate(\cdots), \cdots)$.
- 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:

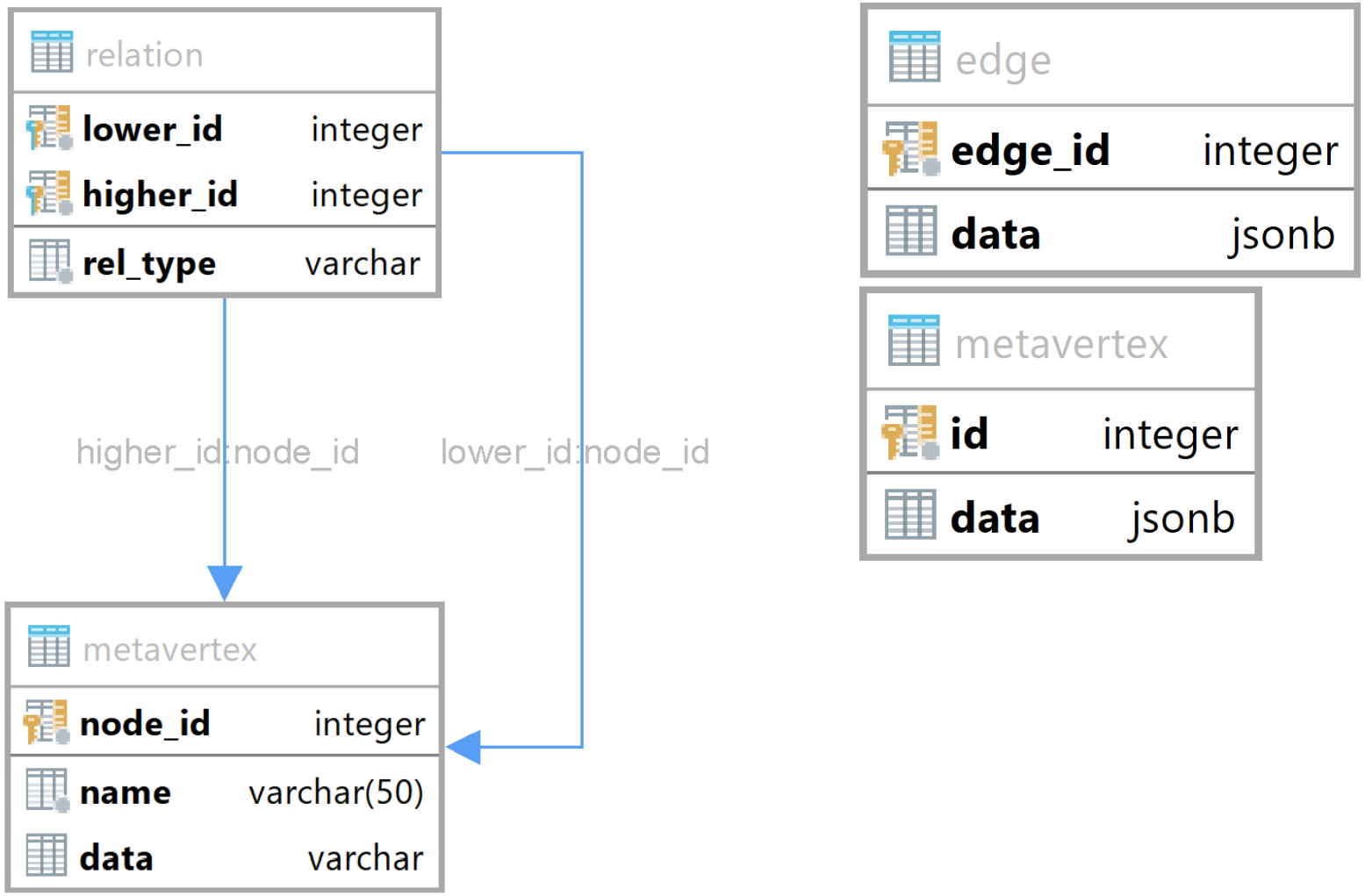
Хранение – документы

	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, mv2))

Хранение – реляционная модель

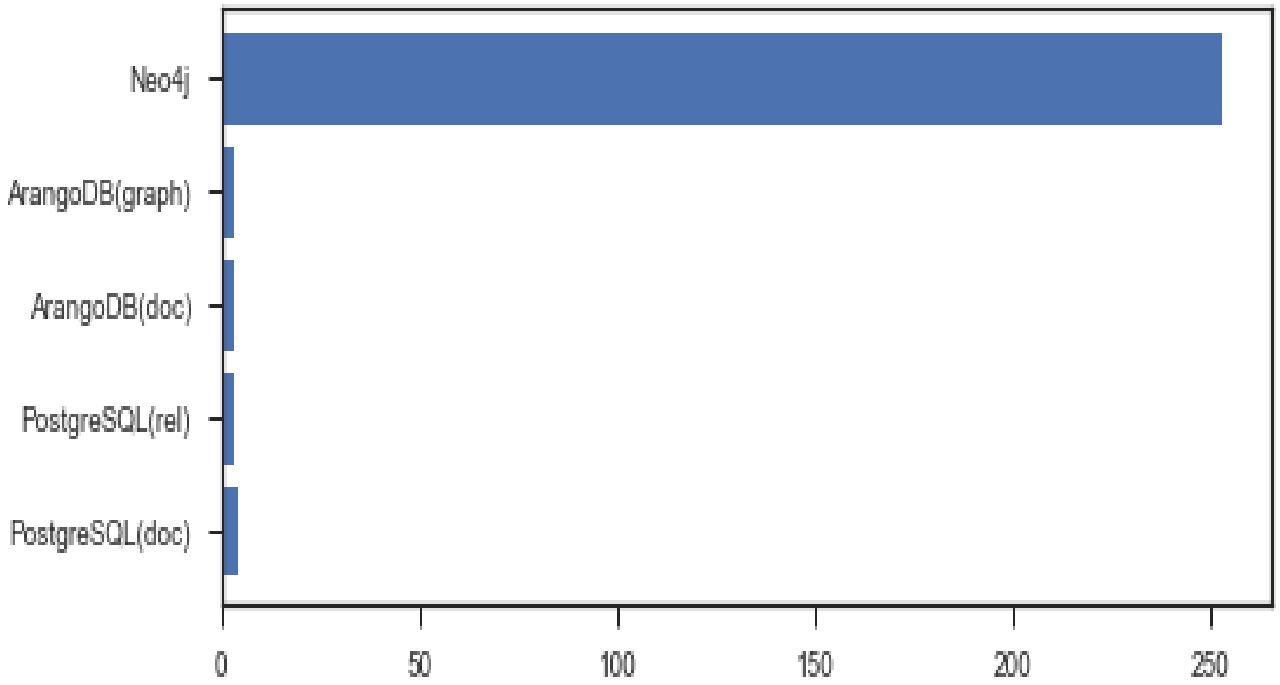
- Использование чистого реляционного подхода или документно-ориентированного хранилища, встроенного в реляционную СУБД.



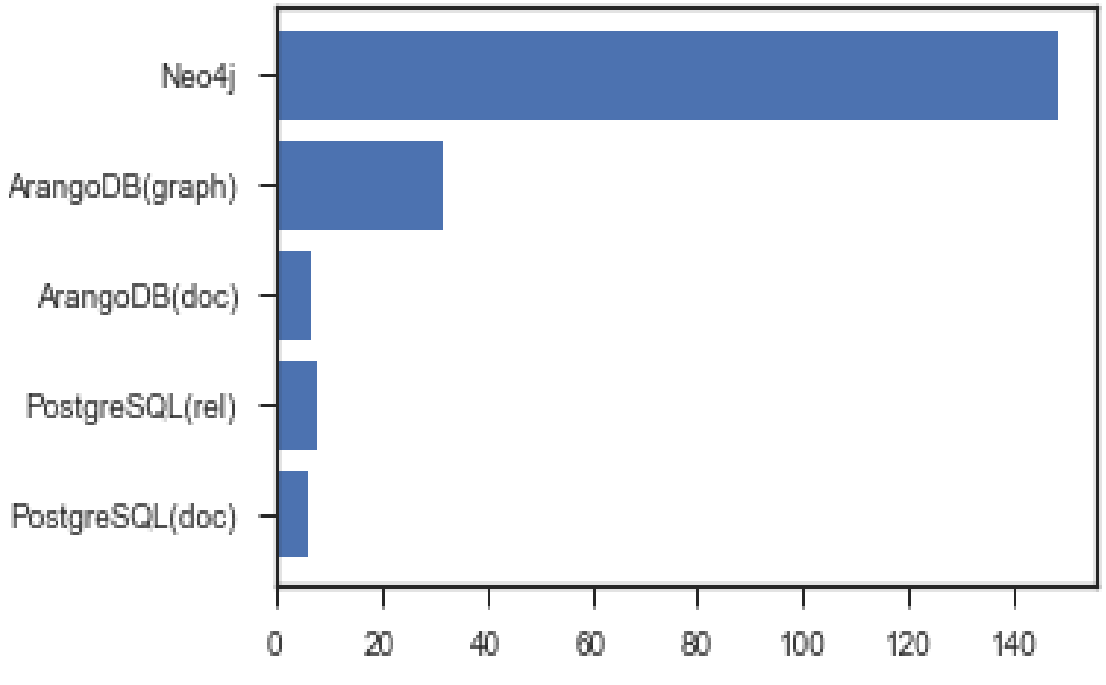
Эксперименты

- Эксперименты проводились для следующих подходов:
  - Neo4j – использование СУБД Neo4j с плоской графовой моделью;
  - ArangoDB(graph) – использование СУБД ArangoDB с плоской графовой моделью;
  - ArangoDB(doc) – использование СУБД ArangoDB с документно-ориентированной моделью;
  - PostgreSQL(rel) – использование СУБД PostgreSQL (классическая реляционная модель);
  - PostgreSQL(doc) – использование документно-ориентированного хранилища на основе СУБД PostgreSQL.
- Характеристики сервера: Intel Xeon E7-4830 2.2 GHz, 4 Gb RAM, 1 Tb SSD, OS Ubuntu 16.04.
- Время в миллисекундах, меньшее значение лучше.

Эксперименты

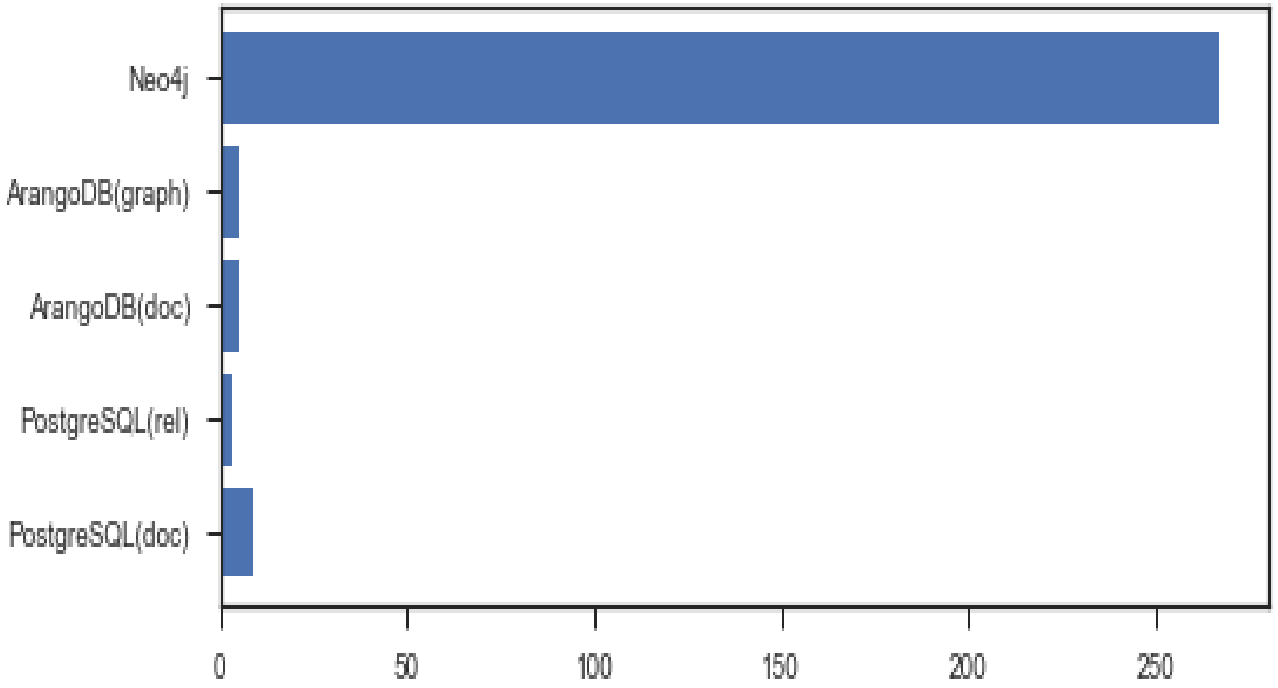


Добавление вершины в метаграф

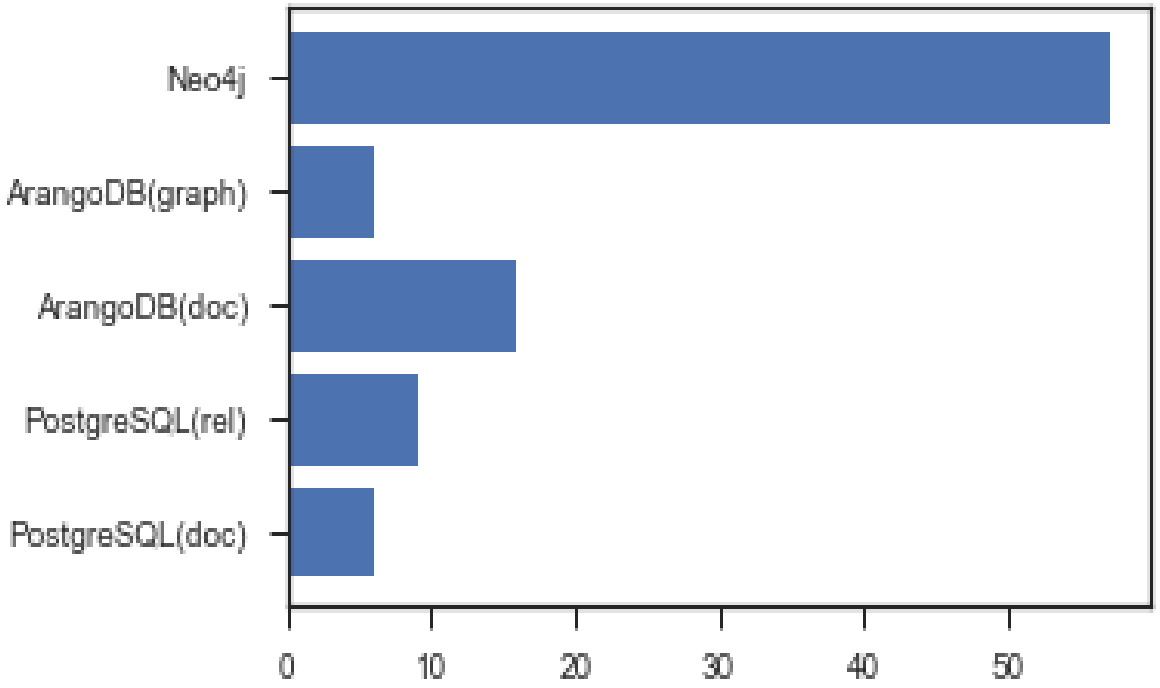


Добавление ребра в метаграф

Эксперименты

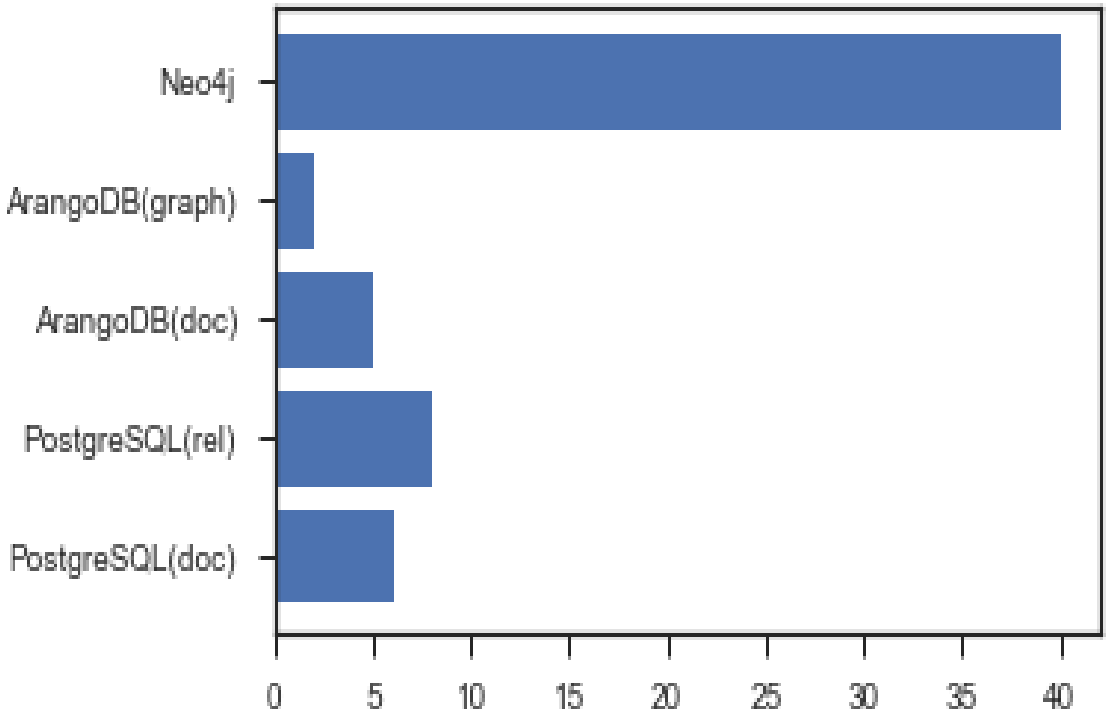


Редактирование вершины в метаграфе

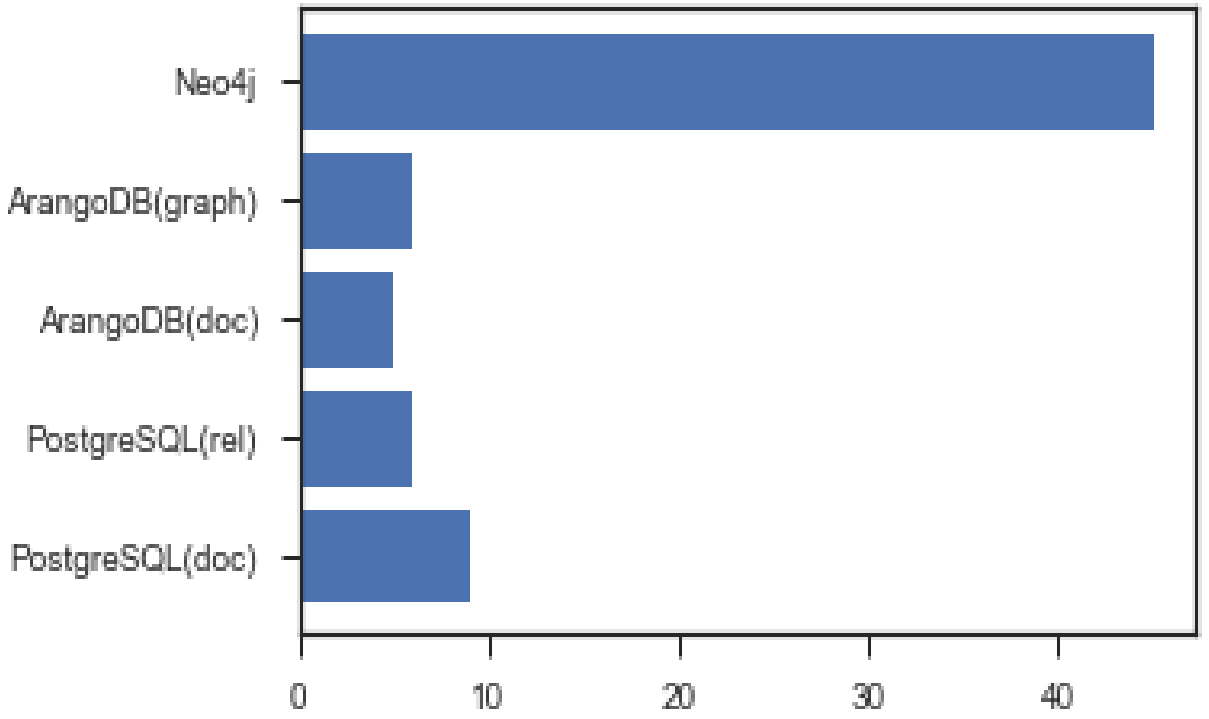


Удаление ребра из метаграфа

Эксперименты

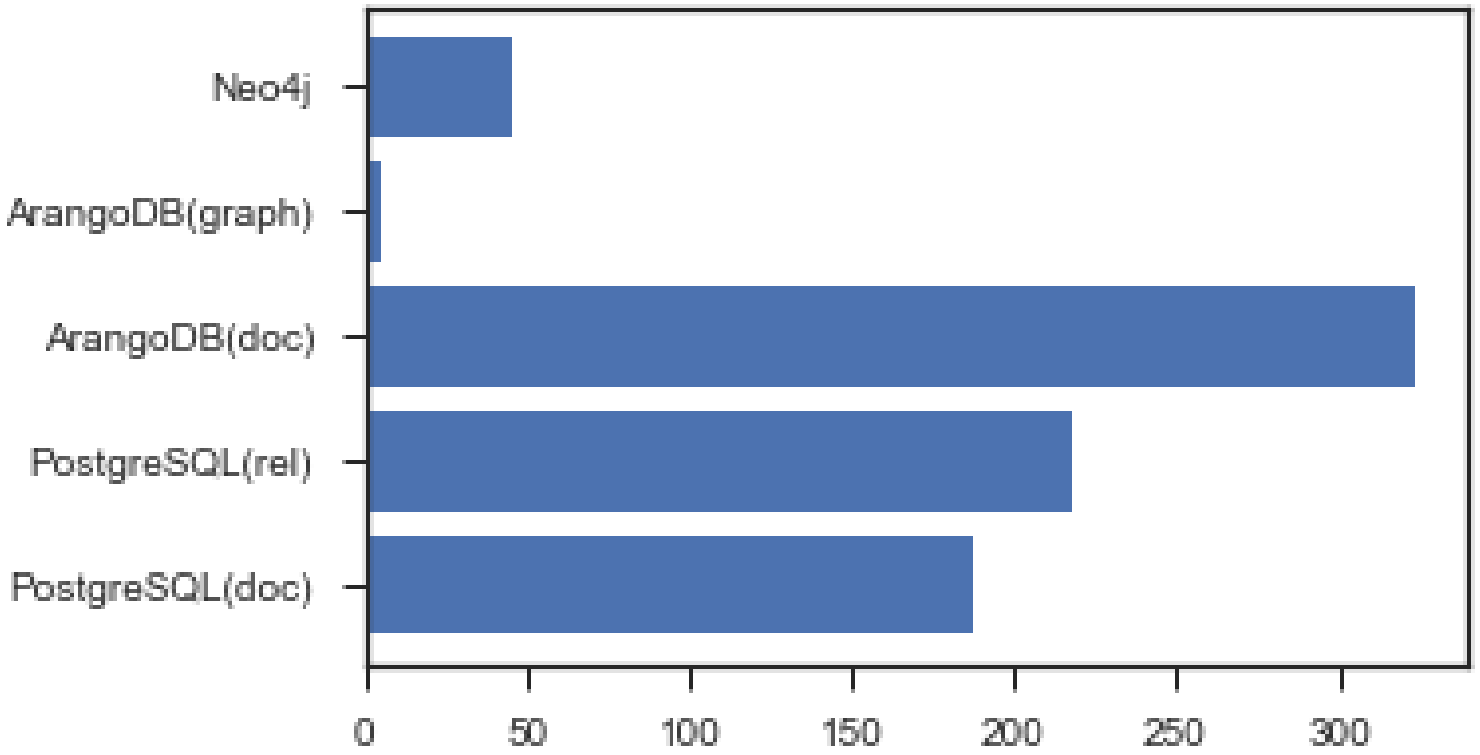


Добавление вершины в метавершину



Удаление вершины из метавершины

Эксперименты



Выборка иерархических данных из 100 связанных метавершин

Выводы (по результатам экспериментов)

- 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.

Выводы по докладу

- 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 метаграфового подхода.
- Метаграфовый подход является одним из вариантов описания «сетей с эмерджентностью». Эмерджентность обеспечивается за счет использования метавершин.
- В отличие от метаграфа, гиперграф не является в полной мере «сетью с эмерджентностью».
- Гиперсеть в полной мере содержит возможности описания «сетей с эмерджентностью». Эмерджентность обеспечивается за счет использования гиперсимплексов. Метаграф по сравнению с гиперсетью обеспечивает более простое и гибкое описание «сетей с эмерджентностью». Но необходимо подчеркнуть, что метаграфы и гиперсети являются лишь различными формальными описаниями одних и тех же процессов, которые происходят в «сетях с эмерджентностью», и теория гиперсетей на сегодняшний день является более развитой по сравнению с теорией метаграфов.
- С использованием метаграфовой модели возможно описание структуры как статических, так и динамических агентов для реализации ГИИС.
- Метаграфовая модель является аналогом «логической» модели СУБД и ей могут соответствовать различные «физические» модели, применяемые в различных СУБД.

Список основных публикаций

- *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.
- *Черненький В.М., Терехов В.И., Гапанюк Ю.Е.* Представление сложных сетей на основе метаграфов. XVIII Всероссийская научно-техническая конференция «Нейроинформатика-2016»: Сборник научных трудов. В 3 частях. Ч. 1. М.: НИЯУ МИФИ, 2016.
- *Федоренко Ю.С., Гапанюк Ю.Е.* Построение адаптивных моделей на основе многоуровневой нейронной сети с использованием метаграфового подхода. XVIII Всероссийская научно-техническая конференция «Нейроинформатика-2016»: Сборник научных трудов. В 3 частях. Ч. 1. М.: НИЯУ МИФИ, 2016.
- *Черненький В.М., Терехов В.И., Гапанюк Ю.Е.* Структура гибридной интеллектуальной информационной системы на основе метаграфов. Нейрокомпьютеры: разработка, применение. 2016. Выпуск №9.
- *Гапанюк Ю.Е., Ревунков Г.И., Федоренко Ю.С.* Предикатное описание метаграфовой модели данных. Информационно-измерительные и управляющие системы. 2016. Выпуск № 12.
- *Черненький В.М., Гапанюк Ю.Е., Ревунков Г.И., Терехов В.И., Каганов Ю.Т.* Метаграфовый подход для описания гибридных интеллектуальных информационных систем. Прикладная информатика. 2017. Том 12, №3.

СПАСИБО ЗА ВНИМАНИЕ.
ВОПРОСЫ?

gapyu@bmstu.ru