# YarcData uRiKA – эврика в семантическом анализе

*Текст* Александр Фролов

**В настоящее время мир стоит на пороге радикального изменения информационных систем, связанного с масштабным внедрением технологий, основанных на семантическом анализе данных. Весной 2012 года Google объявил о запуске Google Knowledge Graph – технологии поиска с использованием семантической базы данных, а производитель суперкомпьютеров Cray – о создании YarcData uRiKA, первой высокопроизводительной системы, ориентированной на анализ семантических баз данных, в том числе и в реальном времени.**

## Cray и YarcData

В феврале 2012 года компания Cray объявила о создании нового подразделения YarcData, целью которого является внедрение и развитие продукта под названием uRiKA (произносится [jur'i:ka]), ориентированного на рынок систем анализа семантических баз данных. Потенциальными областями применения таких систем являются бизнес-аналитика, биоинформатика, медицина, телекоммуникации, логистика, анализ социальных сетей, поисковые системы. В целом рынок семантических баз данных в США оценивается в 40 млрд долларов.

## Семантические (графовые) базы данных

В отличие от реляционных баз данных, где объекты и отношения между ними задаются в виде таблиц, в семантических (или графовых) базах данных объекты представляются в виде вершин с некоторыми атрибутами, а отношения – в виде дуг, соединяющих вершины графа. Такой подход к хранению данных является естественным для отображения отношений между объектами. Соответственно, анализ отношений в графовых базах данных реализуется значительно проще, а следовательно, эффективнее, чем в реляционных.

## Модель RDF

Одним из стандартов, разработанных консорциумом W3C для реализации концепции Web 3.0 (его еще называют семантическим Web`ом), является RDF (Resource Description Framework). Модель RDF позволяет строить ориентированный граф, состоящий из объектов и отношений между ними. Для обработки данных, представленных в RDF, разработан специальный язык запросов SPARQL, также стандартизованный W3C.

## Информативность графа (реальные графы)

Существует гипотеза о том, что информативность семантического графа прямо пропорциональна квадрату количества его вершин. Другими словами, чем больше размер графа, тем более высока степень полезности извлекаемой из него информации. Таким образом, задача графового анализа может быть охарактеризована как проблема класса Big Data, или «Больших Данных».

## Особенности параллельного анализа графов

Параллельный анализ графов имеет ряд особенностей. Во-первых, граф крайне сложно, а в общем случае невозможно равномерно распределить между узлами параллельной вычислительной системы. Как следствие, во время выполнения анализа графа возникает интенсивный обмен сообщениями между узлами, характеризующийся коммуникационным паттерном «все-всем». Во-вторых, связи между вершинами графа нерегулярны, что не позволяет использовать преднакачку данных из памяти (как локальной, так и удаленной). В-третьих, графы могут динамически изменять свою структуру, при этом эти изменения невозможно предсказать заранее, вследствие чего необходимо обеспечение высокоскоростного ввода-вывода данных ко всем вычислительным узлам системы, что особенно критично для анализа графовых баз данных в режиме реального времени. В итоге традиционные подходы к построению систем для анализа «Больших Данных» приводят к низкой производительности на графовых задачах из-за неадекватности архитектуры вышеперечисленным проблемам.

## Общая память, Shmem, MPI

Идеальным способом программирования при решении графовых задач является использование общей памяти с единым глобальным адресным пространством (в стиле OpenMP) – в этом случае не приходится решать задачу распределения графа между вычислительными узлами. Менее удобным является использование программной модели PGAS (распределенное глобальное адресное пространство) с односторонними коммуникациями и ее реализаций, таких как Shmem или UPC. Ну, и самое неудобное – это библиотека MPI с двусторонним обменом сообщениями. Поэтому наиболее распространенной платформой для графовых приложений являются системы с общей памятью типа SMP. Недостатками SMP-систем являются ограниченная масштабируемость и высокая стоимость.

## YarcData uRiKA Graph Applience

YarcData uRiKA – высокопроизводительный аналитический программно-аппаратный комплекс, ориентированный на выявление зависимостей в семантических базах данных сверхбольшого объема. Расшифровывается uRiKA как «universal RDF integration Knowledge Applience», из чего следует, что основным способом предоставления данных является RDF-формат. Уникальность uRiKA заключается в применении в качестве аппаратной платформы мультитредового суперкомпьютера с общей памятью Cray XMT2.


## Cray XMT2: где установлены системы

На сегодняшний день Cray XMT2 является последним из серии массово-мультитредовых суперкомпьютеров, разработанных фирмой Cray: Tera MTA, MTA-2 и XMT. В настоящий момент суперкомпьютеры Cray XMT2 были установлены в Швейцарском Национальном Суперкомпьютерном Центре (CSCS), Центре Прикладных Высокопроизводительных Вычислений (САНРС) в Денвиле в США, медицинском центре Mayo Clinic, а также в одной из государственных структур, относящихся к правительству США.

## Cray XMT2: архитектура, Бартон Смит

Архитектура Cray XMT была разработана еще в 1990-х годах выдающимся ученым и инженером Бартоном Смитом. Первая реализация появилась в 1998 году. Cray XMT2, как и предшествующие ей системы, является специализированной системой, ориентированной на эффективное выполнение задач, характеризующихся интенсивным нерегулярным доступом к памяти. Частным примером таких задач являются задачи графового анализа.

## Cray XMT2: мультитредовый процессор ThreadStorm

В основе Cray XMT2 лежит микропроцессор ThreadStorm4, поддерживающий одновременное выполнение 128 аппаратных потоков (тредов). Таким образом, процессор может выполнять одновременно до 128 программных потоков. Для того чтобы быстро переключать контекст, каждый аппаратный тред имеет свой собственный набор регистров, как управляющих, так и общего назначения. Кроме того, каждое тредовое устройство может выдать до восьми одновременно выполняющихся команд обращений к памяти. Такой подход позволяет обеспечить толерантность к задержкам выполнения команд обращений к памяти за счет большого количества одновременно выполняющихся обращений к памяти и лучшего использования пропускной способности памяти и сети.

## Cray XMT2: глобальная общая память

Другой важной особенностью Cray XMT2 является поддержка общей памяти с глобальным адресным пространством, что позволяет вычислительному узлу обратиться к памяти любого другого вычислительного узла, выполнив обычную команду чтения или записи. В

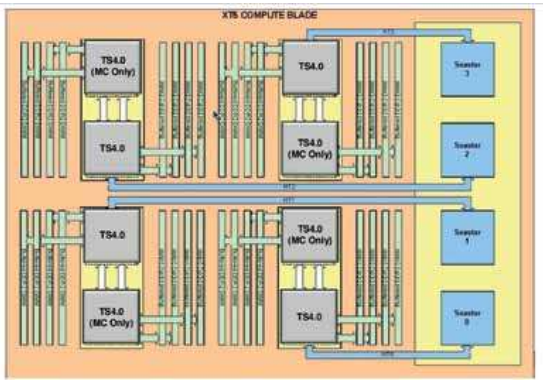


Рис. 2. Вычислительный модуль Cray XMT2

максимальной конфигурации (8192 узла) размер доступной памяти составляет 512 ТБайт. Кроме того, виртуальное адресное пространство скремблируется, то есть адреса равномерно распределяются по вычислительным узлам, использующимся задачей. Такой прием позволяет работать с моделью памяти с псевдоравномерным доступом (UMA).

## Cray XMT2: синхронизация, f/e-биты

Важнейшим элементом в архитектуре Cray XMT2 является поддержка синхронизации с использованием теговых битов в ячейках памяти. Каждая ячейка памяти, представленная 64-разрядным словом, имеет два дополнительных бита – f/e-бит и ext-бит. В зависимости от режима обращения к ячейке памяти (Normal, Sync и Future) и значения f/e-бита (full или empty) контроллер памяти выполняет команду обращения к памяти или отклоняет ее, вызывая тем самым повторную выдачу обращения. При превышении заданного предела переповторов происходит вызов исключительной ситуации, снятие программного треда с тредового устройства и передача управления тредовым устройством другому программному треду.
Данная возможность позволяет осуществить эффективную синхронизацию программных тредов, избегая активного ожидания.
В системе команд Cray XMT поддержаны атомарные операции обращения к памяти, которые также позволяют эффективно реализовать основные синхронизационные примитивы.

## Cray XMT2: инфраструктура XT5 и сеть SeaStar2, сравнение SS2 и Gemini

В качестве сокета процессор ThreadStorm4 использует AMD Socket F, поэтому ThreadStorm4 может быть установлен в вычислительные модули Cray XT4 и Cray XT5. В качестве сети используется Cray SeaStar2+ с топологией 3D-тор.

## Cray XMT2: вычислительный модуль (TS4 и TS4 (MC only))

Вычислительный модуль Cray XMT2 состоит из 4 узлов, каждый из которых состоит из двух микросхем: процессора ThreadStorm4 и внешнего контроллера памяти (который является тем же процессором, только с отключенным ядром), соединенного с этим процессором через линк NPL (Node Pair Link). В качестве контроллера памяти используется DDR2 с шириной канала 16 байт и частотой 300 MHz, что дает пропускную способность памяти в 9,6 ГБ/с на один процессор (троекратное увеличение пропускной способности относительно Cray XMT).

## DDR2 и DDR3

Выбор DDR2 был также обусловлен тем, что DDR2 имеет вдвое меньшую длину вектора (или burst-a), выкачиваемого за одно обращение к памяти из модулей DRAM, по сравнению с DDR3. Поскольку основным режимом работы с памятью является однословный доступ по случайным адресам, то уменьшение длины выкачиваемого из памяти вектора увеличивает долю полезных данных, что повышает производительность задач.

## uRiKA

Программное обеспечение uRiKA состоит из клиентской и серверной частей. Клиентская часть предоставляет пользователю набор инструментальных средств для построения SPARQL-запросов и отображения результатов. В основе клиентской части uRiKA лежит программное обеспечение с открытым исходным кодом, в том числе используются такие пакеты, как Apache Tomcat, Apache Jena-Fuseki, а также средства визуализации WS02, Google Gadgets, Relfinder.

## Cray Query Engine (CQE)

На вычислительных узлах анализ графов выполняется Cray Query Engine (CQE). Граф в CQE представлен в виде списковых структур, работа с которыми аппаратно поддержана в Cray XMT. Ядро CQE является параллельной мультитредовой библиотекой работы с графами, реализующей такие функции, как поиск вширь, кластеризация графа, проверка изоморфизма графов и т. п. При необходимости CQE можно использовать совместно с другими графовыми библиотеками, такими как MTGL или PBGL, для чего предусмотрен специальный интерфейс.

## Совместимость uRiKA с другими СУБД

Также предполагается, что uRiKA может быть использована совместно с другими СУБД, как реляционными, так и NoSQL-типа, например Hadoop. Такая возможность позволит пользователям создавать сложные неоднородные системы обработки и анализа данных с возможностью адаптации к конкретным нагрузкам.

## uRiKA 1.0

В настоящий момент выпущена uRiKA с версией 0.9, выход uRiKA 1.0 запланирован на сентябрь этого года. В uRiKA 1.0 планируется добавить поддержку спецификации SPARQL 1.1, что позволит работать с графовой базой данных не только в режиме read-only, но и выполнять запросы по добавлению и удалению данных. Кроме того, планируется расширить набор форматов импортируемых данных, а также улучшить интерфейс администратора базы данных.

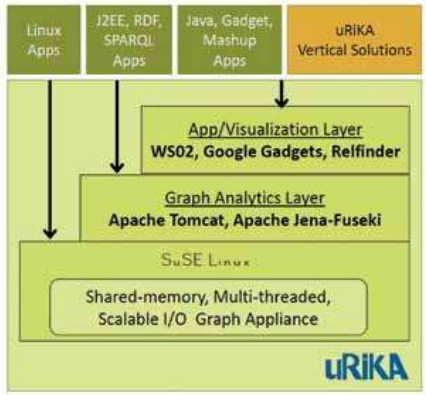


Рис. 3. Программный стек uRiKA

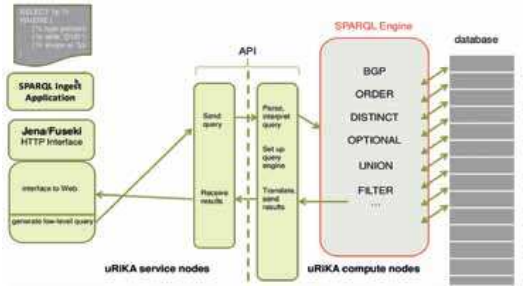


Рис. 4. Взаимодействие вычислительных и сервисных узлов в uRiKA

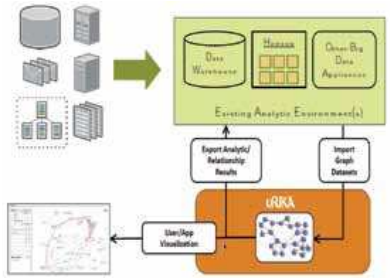


Рис. 5. Взаимодействие uRiKA с другими СУБД

## Перспективы

Задачи с интенсивной нерегулярной работой с памятью, в частности графовые задачи, становятся все более востребованы. Наиболее ярким примером графовых приложений являются семантические базы данных. Развитие технологий высокопроизводительного семантического анализа связано с созданием новых перспективных архитектур, учитывающих специфику графовых задач. Пример компании Cray показывает, каким образом аппаратная поддержка общей памяти и массово-мультитредовой вычислительной модели позволяет получить качественно новый результат, не имеющий аналогов в своем сегменте.

Весьма интересен вопрос, сохранит ли Cray разработку собственных специализированных процессоров, особенно в свете недавней продажи подразделения по разработке высокоскоростных интерконнектов. Представители Cray не спешат раскрывать свои планы по реализации XMT3.