УДК 004.722+004.415.23

А.С. Колосов, Ю.А. Богоявленский

ПАРАЛЛЕЛЬНЫЙ АЛГОРИТМ ПОСТРОЕНИЯ ГРАФА ИКТ-ИНФРАСТРУКТУРЫ ИНТЕРНЕТ-ПРОВАЙДЕРА

A.S. Kolosov, Yu.A. Bogoyavlensky

A PARALLEL ALGORITHM FOR CONSTRUCTING A GRAPH OF A LOCAL INTERNET SERVICE PROVIDER'S ICT-INFRASTRUCTURE

Граф ИКТ-инфраструктуры (Сеть) необходим для решения большинства задач управления Сетью. При наличии в Сети большого числа современных мобильных устройств необходимо постоянное обновление графа Сети для поддержания его в актуальном состоянии, что требует повышения быстродействия существующих подходов построения таких графов.

Представлен параллельный алгоритм построения графа Сети на основе данных из MIB маршрутизаторов, получаемых по протоколу SNMP, и его реализация. Описанные эксперименты подтверждают трехкратное ускорение построения графа Сети ПетрГУ при использовании параллельного алгоритма.

АНАЛИЗ СЕТЕЙ. ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОЕКТИРОВАНИЕ. ГРАФ СЕТИ. РАСПАРАЛЛЕЛИВАНИЕ. СЕТЕВОЕ УПРАВЛЕНИЕ.

An ICT-infrastructure (Network) graph is a necessary tool for solving of most network management problems. While the number of modern mobile devices in a Network grow up, permanent update of the Network graph is required to keep it actual and concistent. Therefore, it's required to develop more efficient and fast methods of building such graphs.

In this paper we present parallel algorithm of Network graph discovery using routers' MIB data obtained with SNMP and its implementation. Described experiments confirm three-fold speed-up of the PetrSU Network graph building when using parallel algorithm.

NETWORK ANALYSIS. OBJECT-ORIENTED MODELLING. NETWORK TOPOLOGY. CONCURRENCY. NETWORK MANAGEMENT.

Данные об аппаратных элементах ИКТ-инфраструктуры и их взаимосвязях (далее – граф Сети) являются базовым инструментом систем сетевого управления локальных поставщиков сетевых услуг (лПСУ).

Разрабатываемая в Петрозаводском государственном университете (ПетрГУ) экспериментальная платформа Nest [1, 2] предоставляет исследователю моделей и методов сетевого управления средства автоматизированного построения графа Сети, его визуализации, выполнения запросов. Граф Сети определяется как подграф объектного графа модели архитектуры лПСУ SON [2]. Подсистема NesTopo [1] обеспечивает автоматизированное построение и хранение графа Сети (сетевой уровень) в БД с помощью последовательного алгоритма.

Экспериментальная эксплуатация подсистемы NesTopo показала ее приемлемую производительность, однако, принимая во внимание рост количества мобильных устройств, способных быстро менять свое положение в графе, необходимость оперативного отслеживания подключений и потенциальный рост количества элементов и связей в графах Сетей, следует признать актуальной задачу фиксации быстрых изменений графа, для решения которой необходимо увеличить быстродействие алгоритма [1].

В данной статье предлагается парал-

лельный алгоритм построения и хранения в объектной БД графа Сети. Описанные в статье результаты экспериментов свидетельствуют о примерно трехкратном увеличении быстродействия процедуры построения графа Сети ПетрГУ по сравнению с традиционным подходом.

Параллельный алгоритм построения графа Сети

Распространенным способом определения возможности распараллеливания алгоритма [3] является выявление в его ярусно-параллельной форме (ЯПФ) независимых вершин одного яруса, каждая из которых может выполняться параллельно.

На рис. 1 представлена ЯПФ графа алгоритма [1] для Сети, содержащей один маршрутизатор с двумя канальными интерфейсами, один из которых подключен к двум IP-подсетям. Каждая вершина графа алгоритма представляет набор операций построения вершины графа Сети, выполняющих получение данных и запись в БД для маршрутизатора (R), канального интерфейса (LI), сетевого интерфейса (NI), IP-подсети (N) и устройства (D). Здесь вершины одного яруса являются независимыми, т. е. могут выполняться параллельно.

Ширина данной ЯПФ равна максимальному количеству подключенных сетевых устройств среди всех маршрутизаторов Сети, которое в современных лПСУ может достигать нескольких тысяч. Таким образом, непосредственная реализация параллелизма, представленного в ЯПФ (рис. 1), потребует реализации такого же количества параллельных ветвей. В то же время важным требованием к параллельному алгоритму является возможность его эффективной работы на современных серверных ЭВМ среднего класса (десятки ядер), что обеспечит возможность его широкого применения. Другое важное требование к алгоритму – масштабируемость, т. е. независимость его реализации от количества ядер.

Для выполнения двух этих требований был предложен алгоритм 1 (рис. 2), описывающий процедуру построения графа Сети как взаимодействие процесса формирования вершин графа при его обходе с параллельно выполняющимися задачами получения данных от сетевых устройств. Число этих задач ограничивается размером пула рабочих потоков.

Для текущей вершины создается задача формирования данных вершины (функция graphTraverse), выполняющая их обновление по протоколу SNMP (функция update) согласно [1]. Для обнаруженных при обновлении вершин, связанных с текущей, создаются аналогичные задачи, ко-

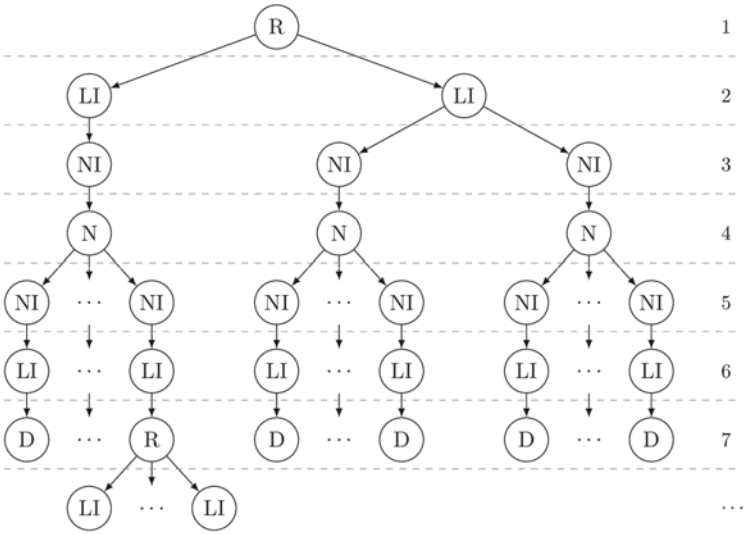

Рис. 1. Ярусно-параллельная форма алгоритма построения графа Сети

```
      Вход: threadPool, startNode, seenSet
1:    function GRAPHTRAVERSE(node)
2:    for all e in CHILDREN(node) \ seenSet do
3:          seenSet ← seenSet + e
4:          UPDATE(e)
5:          task ← {GRAPHTRAVERSE(e)}
6:          ADDTASK(threadPool, task)
7:    end for
8:    end function
9:    task ← {GRAPHTRAVERSE(startNode)}
10:   INVOKE(threadPool, task)
```

Рис. 2. Алгоритм 1: параллельный алгоритм построения графа Сети

торые добавляются в очередь пула потоков (вызов функции addTask) и выполняются по мере его освобождения. Запуск процедуры построения графа Сети начинается с инициализации и запуска в контексте пула потоков (вызов функции invoke) задачи graphTraverse для корневой вершины startNode. Предотвращение зацикливания обеспечивается за счет сопровождения множества просмотренных вершин seenSet. Отметим также, что построение всех связей вершины внутри одного потока позволяет не синхронизировать доступ к объекту данной вершины.

Реализация подсистемы NesTopo

Реализация параллельного алгоритма построения графа Сети, а также ряд архитектурных проблем последовательной версии, не позволявших расширять процедуры сбора данных о Сети независимо друг от друга, потребовали разделения подсистемы NesTopo [1] на три независимых модуля (рис. 3): управление процедурой построения графа Сети, получение данных об элементах Сети и связях между ними, построение отдельных элементов графа в соответствии с моделью SON.

С целью отделения процедур построения элементов графа в БД от процедур получения данных применена идея индуктивных графов [4], в соответствии с которой при продвижении по графу Сети для каждого устройства и связанных с ним вершин графа Сети известен контекст, предоставляющий следующую информацию: описание устройства; тип устройства (маршрутизатор или оконечный узел); таблица интерфейсов и их сетевых адресов; таблица маршрутизации; список устройств, разделяющих широковещательный домен с данным устройством.

Рассматриваемый подход реализован в модуле получения данных в базовом классе ProviderContext и его наследниках SnmpContext (получение данных по SNMP) и IsolatedContext (получение данных об оконечных устройствах).

Процедуры создания и связывания вершин графа Сети на основе данных, получаемых от конкретного контекста, реализованы в т. н. классах-поставщиках (базовый класс – NElementProvider) модуля построения элементов графа. Экземпляры классов-поставщиков реагируют на события обращений к свойствам обслуживаемых вершин графа, реализуя паттерн «Издатель-Подписчик» [5].

Интерфейс для настройки и запуска процедуры построения графа Сети предоставляется классом Collector. Последовательный алгоритм [1] построения реализован в классе BasicCollector и используется для тестирования. Параллельный алгоритм реализован в классе ConcurrentCollector с помощью программного каркаса Fork/Join из состава пакета java.util.concurrent JDK7.

Тестирование и эксперименты

Процесс тестирования был разделен на три части: 1) тестирование процедур построения узлов графа Сети (классы-поставщики) с помощью специальной тестовой реализации контекста, которая предсказуемо отве-

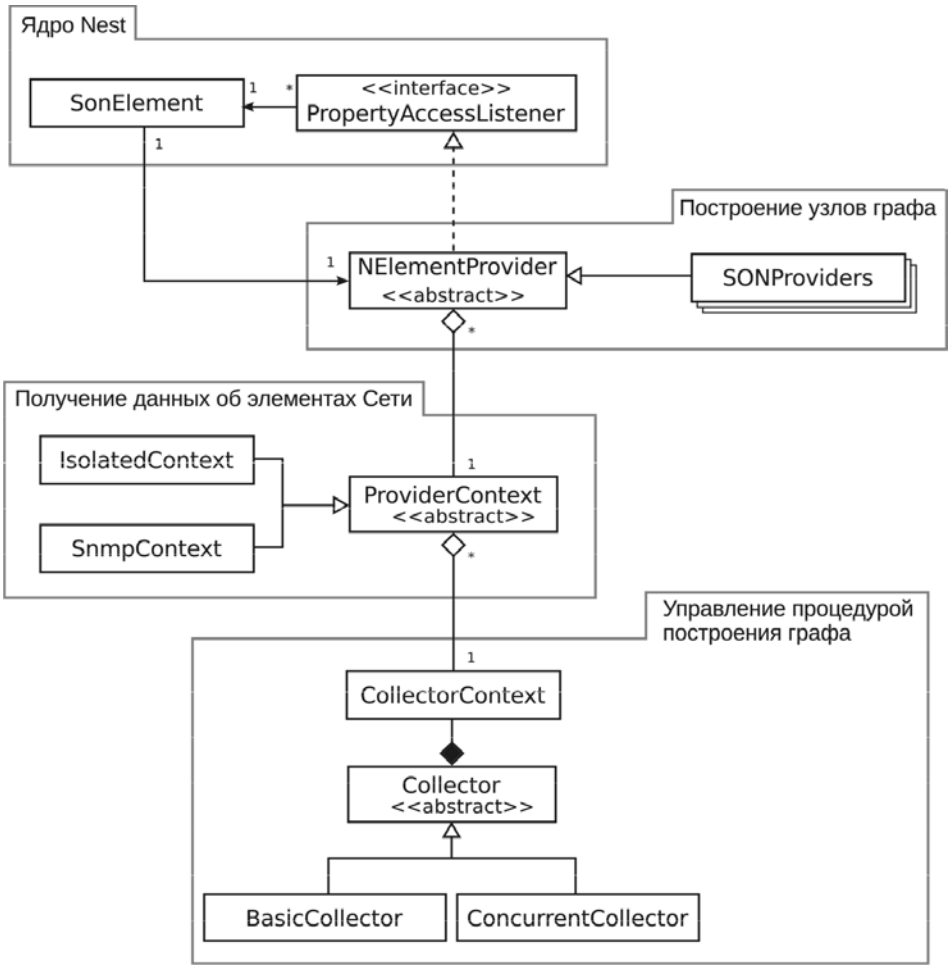

Рис. 3. UML-диаграмма классов подсистемы NesTopo

Результаты экспериментов

Участок сети	Количество ядер	Размер пула	Время построения (параллельный алгоритм), с	Время построения (последовательный алгоритм), с
1	1	4	4,70	7,12
2	1	2	15,49	22,30
3	1	9	21,53	41,17
1	2	9	4,62	7,12
2	2	16	18,10	22,30
3	2	15	21,54	41,17
1	4	14	4,59	7,12
2	4	15	14,62	22,30
3	4	15	14,42	41,17
1	8	11	4,56	7,12
2	8	16	8,90	22,30
3	8	16	12,14	41,17

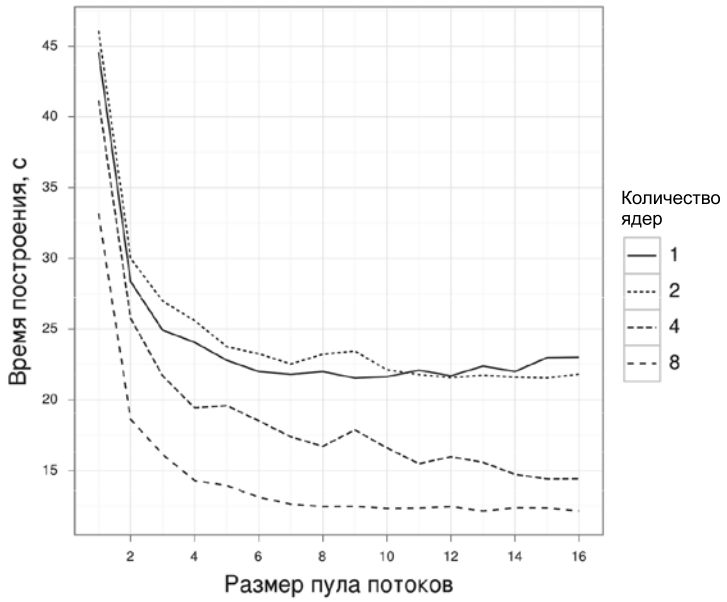

Рис. 4. Зависимость времени построения графа Сети ПетрГУ от размера пула потоков при работе на разном количестве ядер

чает на запросы классов-поставщиков, позволяя оценить корректность построения узлов графа путем сравнения ожидаемых результатов с реальными; 2) тестирование процедур получения данных из MIB сетевых устройств по протоколу SNMP с использованием «поддельного» (*mock*) объекта [6], имитирующего поведение SNMP-сессии; 3) сверка двух графов Сети ПетрГУ, построенных с помощью последовательного и параллельного алгоритмов, в ходе которой различий обнаружено не было.

Всего было разработано 55 тестов, из них 21 тест для классов-поставщиков и 13 тестов для SNMP-контекста. Обнаружено и исправлено 16 ошибок. Разработка и выполнение тестов производились в среде JUnit.

В ходе экспериментов выполнялось построение графа для различных участков Сети ПетрГУ: 1) ЛВС кафедры информатики и математического обеспечения (один опрашиваемый маршрутизатор, 50 устройств); 2) ЛВС учебных корпусов ПетрГУ (два маршрутизатора, 1 500 устройств); 3) вся вычислительная сеть ПетрГУ (четыре маршрутизатора, 1 700 устройств). Эксперименты проводились на ЭВМ с процессором Intel Xeon 2,5 ГГц (8 ядер), ОЗУ 1 ГБ, ОС Linux 3.1.0 в среде выполнения Java JRE 1.7.0_09.

Оценки времени производились для каждой комбинации параметров: номер участка сети, количество выделенных для задачи ядер процессора, размер пула потоков. Каждый эксперимент выполнялся три раза, в качестве окончательных оценок взяты средние значения полученных характеристик. В таблице приведены результаты для комбинаций с теми размерами пула, при которых получено минимальное время построения графа.

По сравнению с результатами эксперимента, проводимого на идентичной аппаратной конфигурации с использованием последовательного алгоритма [1], время построения графа Сети ПетрГУ без учета сетевого взаимодействия и работы с БД сократилось с 41 до 12 с.

Из таблицы и графика (рис. 4) видно, что увеличение размера пула потоков позволяет получить выигрыш в производительности при работе даже на одном ядре процессора.

Представленный параллельный алгоритм автоматизированного построения графа Сети имеет существенно более высокую производительность, чем его последо-

вательная версия, что подтверждается экспериментами. Так, время построения графа Сети ПетрГУ при использовании пула из 16 потоков примерно в три раза меньше, чем время построения того же графа с помощью последовательного алгоритма.

СПИСОК ЛИТЕРАТУРЫ

1. **Богоявленский, Ю.А.** Организация и автоматизированная поддержка объектной базы данных графа ИКТ-инфраструктуры поставщика услуг Интернета [Текст] / Ю.А. Богоявленский, А.С. Колосов // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. —СПб.: Изд-во Политехнического ун-та, 2011. —№ 3 (126). —С. 27—36.

2. **Богоявленский, Ю.А.** Прототип экспериментальной платформы Nest для исследования моделей и методов управления ИКТ-инфраструктурами локальных поставщиков услуг Интернет [Текст] / Ю.А. Богоявленский // Программная инженерия. —2013. —№ 2. —С. 11—20.

3. **Воеводин, В.В.** Параллельные вычисления [Текст] / В.В. Воеводин, Вл.В. Воеводин. —СПб.: БВХ-Петербург, 2002. —608 с.

4. **Erwig, M.** Inductive graphs and functional graph algorithms [Text] / M. Erwig // J. of Functional Programming. —2001.

5. **Гамма, Э.** Приемы объектно-ориентированного проектирования. Паттерны проектирования [Текст] / Э. Гамма, Р. Хелм, Р. Джонсон [и др.]. —СПб.: Питер, 2001. —368 с.

6. **Бек, К.** Экстремальное программирование: разработка через тестирование [Текст] / К. Бек. —СПб.: Питер, 2003.

REFERENCES

1. **Bogoiavlenskii Iu.A., Kolosov A.S.** Organizatsiia i avtomatizirovannaia podderzhka ob"ektnoi bazy dannykh grafa IKT-infrastruktury postavshchika uslug Interneta [*Organization and automated maintenance of an ict-infrastructure graph object database of an Internet service provider*] / Nauchno-tekhnicheskie vedomosti SPbGPU. Informatika. Telekommunikatsii. Upravlenie [*St. Petersburg State Polytechnical University Journal. Computer Science. Telecommunications and Control Systems*]. St. Petersburg: Izd-vo Politekhnicheskogo un-ta, 2011. — № 3 (126). — S. 27—36. (rus)

2. **Bogoiavlenskii Iu. A.** Prototip eksperimental'noi platformy Nest dlia issledovaniia modelei i metodov upravleniia IKT-infrastrukturami lokal'nykh postavshchikov uslug Internet / Programmnaia inzheneriia. —2013. — № 2. — S. 11—20. (rus)

3. **Voevodin V.V., Voevodin Vl.V.** Parallel'nye vychisleniia. — St. Petersburg: BVKh-Petersburg, 2002. — 608 s. (rus)

4. **Erwig M.** Inductive graphs and functional graph algorithms / J. of Functional Programming. — 2001.

5. **Gamma E., Khelm R., Dzhonson R. i dr.** Priemy ob"ektno-orientirovannogo proektirovaniia. Patterny proektirovaniia. —St. Petersburg: Piter, 2001. — 368 s. (rus)

6. **Bek K.** Ekstremal'noe programmirovanie: razrabotka cherez testirovanie. — St. Petersburg: Piter, 2003. (rus)

КОЛОСОВ Александр Сергеевич — *старший преподаватель кафедры информатики и математического обеспечения Петрозаводского государственного университета.*
185910, Россия, Республика Карелия, г. Петрозаводск, пр. Ленина, д. 33.
E-mail: akolosov@cs.karelia.ru

KOLOSOV, Aleksandr S. *Peterozavodsk State University.*
185910, Lenin Str. 33, Petrozavodsk, Republic of Karelia, Russia.
E-mail: akolosov@cs.karelia.ru

БОГОЯВЛЕНСКИЙ Юрий Анатольевич — *заведующий кафедрой информатики и математического обеспечения Петрозаводского государственного университета, кандидат технических наук.*
185910, Россия, Республика Карелия, г. Петрозаводск, пр. Ленина, д. 33.

BOGOYAVLENSKIY, Yuriy A. *Peterozavodsk State University.*
185910, Lenin Str. 33, Petrozavodsk, Republic of Karelia, Russia.