для решения задач детерминированными методами, основанными на методе ветвей и границ (МВГ). Параллельная реализация ориентирована на параллельные системы с распределенной памятью и использует MPI [4].

Основной проблемой при реализации метода ветвей и границ на многопроцессорных системах является обеспечение равномерной загрузки процессоров. Для этого применяются методы балансировки нагрузки, которые перераспределяют эту нагрузку в процессе расчетов. В предлагаемом программном комплексе балансировщик выделен в отдельный модуль, а для описания управления балансировкой применяется формализм конечных автоматов. Это позволяет не только отделить детали реализации численного метода от логики распределения вычислительной нагрузки, но и открывает широкие перспективы по исследованию алгоритмов балансировки методами имитационного моделирования.

В рамках проекта по созданию открытой программной архитектуры для оптимизации на высокопроизводительных вычислительных системах был разработан симулятор параллельной системы [5], который позволяет выполнять алгоритм балансировки нагрузки, имитируя распределенную работу метода ветвей и границ, а также передачу данных между параллельными процессами. Для имитации МВГ используется случайный ветвящийся процесс с вероятностью ветвления, уменьшающейся пропорционально удалению от корня дерева. В основу моделирования положена концепция логических часов. Вместо реального решения подзадачи производится сдвиг логического таймера на заданную величину. Передача данных моделируется аналогично, при этом, время на принимающем процессе полагается равным временной метке полученного сообщения. Для облегчения изучения методов балансировки нагрузки разработана графическая среда [6], показывающая загрузку процессоров в разные моменты времени, визуализирующая обмены между ними. С помощью разработанной среды можно изучать как трассы реального выполнения, так и трассы, собранные при помощи симулятора.

Литература

1. Евтушенко Ю. Г. Численный метод поиска глобального экстремума функций (перебор на неравномерной сетке) //Журнал вычислительной математики и математической физики. – 1971. – Т. 11. – №. 6. – С. 1390-1403.

2. Стронгин Р. Г. и др. Параллельные вычисления в задачах глобальной оптимизации–М //М.: Издательство Московского университета. – 2013.

3. Евтушенко Ю. Г., Посыпкин М. А. Применение метода неравномерных покрытий для глобальной оптимизации частично целочисленных нелинейных задач //Журнал вычислительной математики и математической физики. – 2011. – Т. 51. – №. 8. – С. 1376-1389.

4. Snir M. MPI--the Complete Reference: The MPI core. – MIT press, 1998. – Т. 1.

5. Фомин А. Л. A software simulation tool for the parallel branch and bound method implementation //International Journal of Open Information Technologies. – 2015. – Т. 3. – №. 11. – С. 10-15.

6. Орлов Ю. В. Среда комплексного анализа производительности алгоритмов балансировки в параллельном методе ветвей и границ //International Journal of Open Information Technologies. – 2015. – Т. 3. – №. 9.

И.Д. Семенов, Е.И. Чигарина

СРАВНИТЕЛЬНЫЙ АНАЛИЗ ТЕХНОЛОГИЙ ДОСТУПА К ДАННЫМ В СИСТЕМАХ БАЗ ДАННЫХ

(Самарский национальный исследовательский университет
имени академика С.П. Королёва)

Во всех современных автоматизированных информационных системах используются различные средства для долговременного хранения данных и функции доступа к ним. В настоящее время повышаются требования к скорости обработки данных, и большинство операций нецелесообразно выполнять вручную. Для того, чтобы наиболее полно использовать возможностям того или иного сервера баз данных, необходимо работать с ним напрямую. Это означает полную зависимость приложения от используемого сервера и сложность перехода на другую платформу, так как возникнет необходимость переписывать большое количество кода. Данный вопрос призваны решить различные технологии доступа к данным, являющиеся прослойкой между конкретным сервером и приложением пользователя, предоставляя программисту простой унифицированный механизм работы с данными.

Технологией доступа к данным называется система интерфейсов, обеспечивающая взаимодействие между приложением и базой данных. Во многих системах управления базами данных имеются библиотеки, содержащие интерфейсы прикладного программирования (application programming interface — API), представляющие собой функции, при помощи которых можно выполнять с данными те или иные действия.

На сегодняшний день существует множество технологий доступа к данным, таких как BDE, OLE, ODBC, ADO, и до сих пор разрабатываются новые, более надежные, удобные в работе и более быстродействующие технологии.

На выбор технологии доступа влияет выбор средства разработки приложения и средства реализации базы данных.

На данный момент одними из наиболее популярных и используемых языков программирования являются объектно-ориентированные языки Java и C#. Основным преимуществом языка Java является кроссплатформенность. Язык C# был разработан как язык разработки приложений платформы Microsoft .NET Framework. Платформа .NET основана на использовании общей среды выполнения кода CLR (Common Language Environment), подобно виртуальной Java-

машине, что делает ее удобным инструментом для работы со многими языками программирования.

В настоящее время широкое распространение получили распределенные системы – собрание независимых компьютеров (узлов), соединенных сетью и ПО, обеспечивающих их совместное функционирование. Наиболее популярной является трехуровневая модель, состоящая из трех слоев: слоя данных, слоя бизнес-логики и слоя клиента [1]. Важным этапом функционирования системы является взаимодействие между слоями бизнес-логики и данных. На этом этапе происходит установка соединения, формирование запросов и обработка результатов.

Ввиду широкого использования приложений, разработанных на платформе .NET и программ, написанных на языке Java, было принято решение использовать для анализа следующие технологии доступа к данным: ADO.NET и JDBC.

ADO.NET представляет собой набор библиотек, входящих в Microsoft .NET Framework и предназначенных для взаимодействия с различными хранилищами данных из .NET-приложений. Библиотеки ADO.NET включают все необходимые классы для подключения к источникам данных практически произвольного формата, выполнения запросов к этим источникам и получения результата. ADO.NET можно использовать в качестве средства создания масштабируемых приложений, ориентированных на Web-технологии [2].

Взаимодействие Java-программы с внешним сервером баз данных осуществляется посредством специализированного протокола, отвечающего за совместимость Java с базами данных (Java Database Connectivity, JDBC). Он построен на принципах интерфейса ODBC и применяется для стандартизации Java-кода при организации доступа к различным СУБД.

В данной работе проводится сравнительный анализа выбранных технологий доступа к данным на примере работы с основными объектами баз данных для выявления преимуществ и недостатков их использования при работе с различными СУБД. В качестве средств реализации баз данных выбраны следующие СУБД: MySQL, MS SQL Server, PostgreSQL и Oracle, которые занимают ведущие позиции среди серверных СУБД и широко используются при разработке распределенных систем. Для сравнения технологий доступа к данным выполняется изучение особенностей средств реализации соединения с СУБД, средств подготовки запросов с их последующим выполнением, а также предоставляемые каждой из технологий доступа возможности по работе с полученными данными.

В качестве сравнительных характеристик рассматриваемых технологий доступа были выбраны следующие параметры:

- время, требуемое для соединения с каждой из СУБД;
- время выполнения запросов, работающих со сложными структурами данных;
- время выполнения операций ведения данных;
- время работы с хранимыми процедурами, функциями, видами.

Время исполнения запросов оценивается с учетом количества записей, количества соединений между таблицами, особенностей типов данных.

Литература

1. Трехзвенная архитектура с сервером приложений Электронный ресурс]. – https://moodle.vsu.ru/mod/book/view.php?id=12520&chapterid=237

2. Использование баз данных в приложениях ASP.NET [Электронный ресурс]. – http:// http://www.intuit.ru/studies/courses/1139/250/lecture/6440?page=1.

Е.Г. Скорюпина, С.В. Востокин

АНАЛИЗ РЕАЛИЗАЦИЙ АКТОРНОЙ МОДЕЛИ НА ПЛАТФОРМЕ JAVA

(Самарский национальный исследовательский университет
имени академика С.П. Королёва)

Выполнение задач управления и распределенного взаимодействия в сетях динамических систем подразумевает распределение пакета заданий между несколькими вычислительными потоками (устройствами), что позволяет эффективно использовать многоядерные системы. В основе модели программирования, принятой в языках, аналогичных Java, лежат потоки. Помимо потоковой модели существуют другие подходы к параллелизму. Одним из таких подходов, который приобретает все большую популярность среди разработчиков на Java, является акторная модель.

В основе акторной модели вместо параллельных потоков, взаимодействующих при помощи общей памяти и блокировок, лежат так называемые «акторы», которые обмениваются асинхронными сообщениями при помощи специальных почтовых ящиков. В ответ на полученные сообщения, актор может принимать локальные решения, создавать акторов, посылать сообщения, а также устанавливать, как следует реагировать на последующие сообщения [1].

Почтовые ящики не предоставляют процессам доступа к общей памяти. Акторы выступают в роли изолированных и независимых друг от друга объектов, не использующих разделяемую память при взаимодействии. Данная модель не подразумевает синхронизирующие блокировки, что устраняет связанные с ними потенциальные проблемы, такие как взаимные блокировки или состояние состязания при попытке одновременного чтения/записи данных разными потоками. Таким образом, эта модель значительно безопаснее в силу отсутствия ошибок синхронизации, характерных для многопоточного программирования [1-3]

Акторная модель напрямую не поддерживается платформой Java, однако существует ряд библиотек, реализующих эту модель, в частности Kilim, Actors Guild, ActorFoundry, Akka. Работа библиотек основана на преобразовании байт-кода во время компиляции или во время выполнения программы.