ISBN 978-5-901795-16-3 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2009

УДК 519.687.4

Г. И. Есин, А. А. Кузнецов, В. А. Роганов

Экспериментальная реализация отказоустойчивой системы распределенных вычислений "SkyTS" для параллельного счета ресурсоемких T++ приложений в гетерогенной распределенной вычислительной среде

Аннотация. В данной работе рассмотрен подход к реализации распределенной вычислительной системы, предназначенной для счета ресурсоемких T++ приложений. Дано описание экспериментальной реализации этого подхода в виде системы „SkyTS“, которая обладает свойствами отказоустойчивости, масштабируемости и поддержкой неоднородных вычислительных сетей.

1. Введение

OpenTS — система параллельного программирования, разработанная в ИПС РАН в 2000-2004 годах в рамках суперкомпьютерного проекта „СКИФ“ Союзного государства России и Беларуси. После успешного завершения программы „СКИФ“ разработка OpenTS была продолжена в рамках программы „СКИФ-ГРИД“ и различных академических проектов.

Система OpenTS (Open T-System) [1, 2] представляет собой современную и наиболее удачную реализацию концепции T-системы — программной среды параллельного программирования с поддержкой автоматического динамического распараллеливания приложений, которая сочетает в себе функциональную и императивную парадигмы программирования. OpenTS — это среда поддержки исполнения приложений, написанных на языке T++. Данный язык программирования является параллельным диалектом Си++ и расширяет исходный язык семью новыми ключевыми словами. В остальных известных нам системах программирования, в которых взяты за основу языки Си/Си++, набор новых ключевых слов и команд значительно шире, что осложняет процесс изучения языка и создания программ. Среда поддержки исполнения T++ приложений (T-приложений), берет на себя основную часть работы по организации параллельного

счета (синхронизация, распределение нагрузки, транспортировка сообщений). Тем самым, система OpenTS позволяет снизить затраты на разработку параллельных программ, увеличить глубину параллелизма и более полно использовать возможности аппаратной части мультипроцессора за счет распараллеливания в динамике.

Изначально система OpenTS разрабатывалась в среде Linux для функционирования на Linux-кластерах и рабочих станциях. В ходе развития системы ее исходный код и подсистема сборки были портированы на ОС Windows [3], что позволило расширить ее область применения на кластерные установки под управлением инфраструктуры „Windows HPCS 2008“ фирмы Microsoft.

Концепция T-системы хорошо подходит для Grid-компьютинга. Вычислительные задания (T-задачи), которые порождает пользовательское T-приложение, могут быть вычислены не только на любом узле сильносвязанного кластера, но и в принципе на любом компьютере в сети Интернет. Поскольку T-задачи имеют свой „вес“, а для узлов распределенной системы вводится понятие надежности, то можно организовать работу T-приложения так, что наиболее весомые (значимые) T-задачи верхнего уровня распределяются планировщиком по наиболее надежным счетным узлам вычислительной системы. Соответственно, менее значимые задачи могут быть назначены на менее надежные листовые узлы в общей иерархии узлов системы. Поэтому в перспективе систему OpenTS можно использовать не только для кластеров, но и для территориально-распределенных неоднородных установок.

Предпосылками для развития данного подхода явились проведенные в рамках программы „СКИФ“ эксперименты по использованию совместно с системой OpenTS следующих реализаций технологии MPI в качестве транспортного уровня: MPICH-G2, IMPI, PACX-MPI. Эти пробные испытания показали, что технология OpenTS может быть доведена до полноценного Grid-решения. Под Grid-технологиями мы будем понимать набор средств и технических решений, которые позволяют объединять разрозненные разнородные компьютеры и суперкомпьютеры в территориально-распределенную гетерогенную информационную и вычислительную систему [4]. При этом основной задачей будет интеграция всех вычислительных ресурсов всех компьютеров, входящих в систему для решения тяжелых и сверхтяжелых научно-прикладных задач.

Все проведенные в последние годы доработки системы OpenTS (улучшение свойств кроссплатформенности, разработка подсистем отказоустойчивости и масштабируемости) позволили системе перейти на новый уровень развития, который состоит в поддержке выполнения параллельных T-приложений в неоднородной распределенной вычислительной среде. В рамках программ Союзного государства „Триада“ и „СКИФ-ГРИД“ разработана архитектура и проведена экспериментальная реализация распределенной отказоустойчивой системы „SkyTS“ (прежнее название „SkylighTS“), которая позволяет объединить разнородные ресурсы компьютеров в сети Интернет для счета T-приложений, решающих ресурсоемкие научно-прикладные задачи.

2. Отказоустойчивость работы T-приложений

Отказы оборудования — это одна из наиболее частых причин остановки счета в супервычислительных установках. В случае распределенных Grid-систем отказы оборудования будут происходить неизбежно и довольно часто, по сравнению с кластерными системами, поскольку узлы такой системы скорее всего будут находиться вне контроля головной организации Grid-системы. Для реализации такой распределенной системы на базе OpenTS необходимо наделить среду поддержки исполнения T-приложений возможностью восстанавливать процесс счета после аварийного сбоя (или даже штатного выключения) какого-либо из счетных узлов.

Были разработаны различные инструменты для имплементации подсистемы отказоустойчивости работы T-приложений [5, 6]:

- Улучшенная для работы по TCP/IP реализация наиболее употребимого подмножества функций MPI — DMPI, дополненная функциями автоматического мониторинга исправности вычислительной конфигурации сильно- либо слабосвязанного множества вычислительных узлов. Такая реализация позволяет в каждом конкретном случае наиболее эффективно реализовать восстановление нормального счета T-приложения путем переповтора пострадавшего фрагмента вычислений в случае аппаратного сбоя на узле или потери связи с ним.

- Доработанная реализация среды поддержки исполнения T-приложений (микроядро OpenTS). Эта среда опирается на отказоустойчивую реализацию DMPI и автоматически обеспечивает отказоустойчивость T-приложений, реализованных в функциональном стиле на языке T++, которые могут работать как в сильно-, так и в слабосвязанной вычислительной среде.

В системе OpenTS использованы следующие оригинальные методы обеспечения отказоустойчивости:

- Используемая системой OpenTS коммуникационная библиотека DMPI способна сигнализировать о сбоях, продолжая нормально функционировать и отслеживать, какие сообщения следует перепослать, а какие нет. Доработка DMPI состоит в том, что попытка запуска приложения на каждом узле производится не однократно, а многократно, то есть в случае сбоя и перезапуска узла возникает новая „реинкарнация“ вычислительного процесса, который готов продолжить работу. Также происходит корректная обработка возможных ошибок уровня сокетов TCP/IP и передаче статуса этих ошибок в функцию-обработчик сбоев.
- Реализовано сохранение пренатальных процессов (вызванных T-функций), так как они еще не получили стэка для своей работы и находятся в наиболее компактной (и даже адресно-независимой) форме. Реализовано запоминание T-функций и их аргументов, а также назначенных для их исполнения вычислительных узлов с целью обеспечения возможности их повторного вычисления на исправных узлах в случае аварии. Повторный запуск T-функций производится после реконфигурации коммуникационной подсистемы и Суперпамяти.

Безусловно, использование TCP/IP в качестве базового уровня для реализации MPI-взаимодействия может внести некоторые накладные расходы при использовании внутри тесносвязанных кластеров. Однако важнее уверенность в безукоризненной отказоустойчивости базового уровня коммуникаций. Кроме того, уже начиная с небольших метакластерных установок, накладные расходы, вносимые уровнем TCP/IP, не будут такими уж существенными хотя бы

потому, что сами кластерные установки обычно связаны именно по этому протоколу.

Кроме запуска командой mpirun/mpiexec, возможно динамическое вхождение вычислительного узла в расчетное поле по собственной инициативе. При этом, вычислительный узел может не обладать собственным IP-адресом, так как соединение с коммуникационным сервером происходит по инициативе самого узла. В этой схеме головной процесс работает дополнительно в режиме сервера, принимая запросы на соединение с узлов. После успешного установления такого соединения каждый подключенный по собственной инициативе узел взаимодействует со всеми остальными узлами по протоколу TCP/IP, обмениваясь активными сообщениями уровня OpenTS. Новая схема интегрирована с традиционной формой запуска, которая хорошо подходит для находящихся в распоряжении относительно высоконадежных узлов кластера.

3. Масштабируемость T-приложений на большое количество узлов

При разработке распределенной вычислительной системы не должно ставиться каких-либо ограничений на количество задействованных в счете узлов. В перспективе это число может достигать десятков и сотен тысяч. Поэтому в основе Grid-системы должна лежать технология, обеспечивающая эффективное масштабирование приложений, выполняющихся распределенно на большом число узлов.

Для системы OpenTS реализован набор усовершенствований, позволяющий обеспечить эффективное распараллеливание ресурсоемких T-приложений на сверхбольших кластерных и метакластерных установках (насчитывающих до миллиона вычислительных ядер), и в крупных Grid-системах. Изменения затронули подсистему Суперпамяти, встроенный планировщик и обмен информацией о ресурсах.

3.1. Подсистема Суперпамяти OpenTS

Подсистема Суперпамяти является ключевым элементом реализации системы OpenTS, поэтому масштабирование структур данных и алгоритмов работы Суперпамяти явилось ключевым этапом доработки.

Для реализации Суперпамяти в OpenTS используется модель общей памяти. В модели программирования с общей памятью все процессы совместно используют общее адресное пространство, к которому они асинхронно обращаются с запросами на чтение и запись. В таких моделях для управления доступом к общей памяти используются всевозможные механизмы синхронизации типа семафоров и блокировок процессов. Преимущество этой модели с точки зрения программирования состоит в том, что понятие собственности данных (монопольного владения данными) отсутствует, следовательно, не нужно явно задавать обмен данными между производителями и потребителями. При использовании системы OpenTS программист освобожден от необходимости явно специфицировать общие данные и упорядочивать доступ к ним с помощью средств синхронизации, поскольку все эти функции автоматически выполняет система.

Как известно, существуют различные схемы организации общей памяти в распределенных системах. Некоторые из них напрямую отображают виртуальное адресное пространство на области памяти локальных узлов. Наряду с простотой, такие схемы обладают определенными недостатками. Прежде всего, единицей работы с памятью в такой схеме является аппаратная страница, что замедляет работу с совокупностями небольших по размеру объектов. На 32-разрядной архитектуре пределом совокупного объема данных оказывается 4 Гб, что по современным меркам выглядит достаточно серьезным ограничением, особенно в суперкомпьютерных применениях. Этих недостатков лишены схемы так называемой объектно-ориентированной общей памяти, где единицей адресации является не аппаратная страница, а объект (ячейка). Перекладывая часть работы на программное обеспечение, удается достичь снятия указанных ограничений. Дополнительно, такая схема организации памяти может быть легко интегрирована с различными аспектами объектно-ориентированных технологий, такими как автоматическая сборка мусора.

В OpenTS Суперпамять организована в виде матрицы; при этом общий размер матрицы равен *N*M*sizeof(TCell)*, где N и M — максимальное количество узлов установки и максимальное количество супер-ячеек на каждом узле соответственно.

В случае использования ОС Linux Суперпамять размещается в виртуальном адресном пространстве. Это позволяет зарезервировать большой объем виртуального адресного пространства, расходуя физическую память по мере необходимости. Однако в случае других ОС,

а также и под ОС Linux в некоторых случаях разумнее не выходить в захвате виртуального адресного пространства за некие разумные рамки. А поскольку в случае больших установок произведение N*M может быть очень большим, то данный подход неприемлем без доработок. В случае использования 32-разрядных ОС ограничений, накладываемых системой, становится еще больше.

Путь решения этого вопроса классический; он используется во многих аппаратных MMU (Memory Management Unit) и состоит в том, что делается несколько уровней/директорий. Нижний уровень содержит непосредственно супер-ячейки, предыдущие содержат массивы ссылок на супер-ячейки и так далее. Фактически это означает, что вместо сплошной суперматрицы (матрицы ячеек суперпамяти в OpenTS) хранится разреженная суперматрица.

3.2. Требования к ресурсам

Хорошая масштабируемость предполагает не только эффективность работы на больших установках, но и экономное (по потребности) расходование системных ресурсов. Этот момент очень важен для разработчиков T-приложений, поскольку при отладке и тюнинге T-приложения часто запускают на небольших установках или просто на персональном компьютере в так называемом „режиме эмуляции большого кластера“. При этом запускается значительно больше системных процессов, чем имеется вычислительных ядер.

Новая схема реализации суперсегментов в совокупности с оптимизацией некоторых системных констант позволили снизить минимальные требования к объему необходимой T-приложению памяти приблизительно в восемь раз.

Измерения проводились под ОС Linux Fedora 9 x86_64 на двухъядерном ПК с 2-мя гигабайтами оперативной памяти. До доработки удавалось имитировать кластер с 8-ю узлами, после доработки — с 64-мя.

3.3. Оптимизация обменов информацией о ресурсах

Семантика Суперпамяти в OpenTS различна для разных сегментов, но в первой ее реализации каждый узел обменивается с каждым информацией о наличии у него подзадач и свободных мощностей. Разумеется, это эффективно либо на узком классе задач, либо для кластеров небольшой размерности. В случае общей задачи на большой установке количество обменов может вырасти квадратично.

По этой причине обмены информацией о ресурсах целесообразно также сделать многоуровневыми: то есть поступать так же, как поступают при реализации функции *broadcast()*: вместо рассылки информации всем узлам сразу посылать нескольким (скажем, четырем) с просьбой распространить эту информацию далее.

Основная идея оптимизации этих обменов состоит в том, что OpenTS не совершает обращений к ячейкам суперпамяти, которые лежат вне пределов некоторого графа, как раз соответствующего иерархии узлов, используемой доработанным планировщиком (см. ниже). При этом „лишним“ обменам в большинстве случаев попросту неоткуда взяться. Тем не менее, они не запрещены, поэтому приложение не ограничено в своей свободе устраивать и свои собственные схемы обменов данными.

Отдельно изменяется схема обмена информацией о вычислительной мощности каждого вычислительного узла в самом начале счета.

3.4. Доработка планировщика

Новый планировщик, который распределяет подзадачи в больших системах, имеет иерархическую структуру. Каждый вычислительный узел принадлежит тому или иному уровню иерархии. На практике иерархия может возникать естественным путем (например, метакластер, состоящий из нескольких кластеров), но в некоторых случаях дополнительные уровни целесообразно создавать искусственно (например, когда в кластере нескольких сотен узлов).

В текущей экспериментальной версии используется синтетическая иерархия, построенная на основе одинаковых по размеру гиперкубов, соединенных в некоторых местах в древоподобную структуру.

При поиске задач каждый свободный от работы узел обращается к вышестоящим, публикуя свои свободные ресурсы. При поиске помощников для вычисления подзадач каждый перегруженный (имеющий более одной задачи) узел обращается к свободным нижестоящим. Кроме этого, узлы-члены каждого уровня иерархии связаны в гиперкуб и также обмениваются по горизонтали. На основе полученной информации происходит обмен задачами.

Роль „досок объявлений“ как раз играют суперсегменты с публикуемыми задачами и свободными ресурсами узлов. Собственно, изначально OpenTS и была ориентирована на использование иерархии „досок объявлений“, через которые бы шел обмен задачами. Однако реально использовалась лишь одна доска объявлений на весь кластер.

4. SkyTS — Grid-система для счета T++ приложений

С учетом всех выполненных доработок среды поддержки исполнения T-приложений стало возможным разработать систему распределенных вычислений для счета T-приложений. На основе разработанных ранее механизмов запуска T-приложений в отказоустойчивом режиме, а также подсистемы масштабируемости, была разработана экспериментальная версия Grid-системы „SkyTS“, которая позволяет осуществить интеграцию вычислительных ресурсов разрозненных и разнородных компьютеров в территориально-распределенную гетерогенную информационно-вычислительную систему. Она обладает следующими свойствами:

- Высокая масштабируемость: число параллельных процессов T-приложения может достигать сотен тысяч; также возможно подключение к системе любого количества серверных (управляющих) и рабочих (вычислительных) узлов.
- Отказоустойчивость: в системе задействованы механизмы отказоустойчивой работы T-приложений, что позволяет системе функционировать в территориально-распределенной среде.
- Кроссплатформенность: серверные и счетные модули реализованы на интерпретируемом языке программирования, что делает их код переносимым.
- Поддержка эмуляции: T-приложение, созданное для какой-либо ОС, способно работать на других ОС за счет использования инструмента эмуляции Wine.
- Поддержка виртуализации: серверные и счетные модули способны работать в среде виртуальной машины в гостевой ОС Linux.

Система состоит из следующих компонентов:

- Web-интерфейс пользователя, который служит для получения вычислительных задач от пользователей и обратной связи с ними;
- база данных для хранения информации о заданиях, а также репозитории для хранения исполняемых файлов заданий;
- серверный (управляющий) модуль, который распределяет T-задания между счетными модулями;
- счетный модуль, который осуществляет запуск заданий на счет на одиночных компьютерах в сети Интернет.

4.1. Web-интерфейс пользователя системы SkyTS

Для управления подачей и запуском заданий служит Web-интерфейс пользователя системы SkyTS. Интерфейс системы является важной компонентой, так как обеспечивает обратную связь между конечным пользователем и самой системой. После авторизации через Web-интерфейс доступны следующие операции:

- загрузка исполняемого кода T-приложений;
- параметризация T-приложений входными данными;
- постановка заданий на счет;
- доставка результатов счета;
- управление пользователями и их правами.

Эта компонента SkyTS представляет собой Web-приложение, и работа с ним производится посредством использования штатного браузера. Такой подход был выбран потому, что Web-приложения не имеют ограничений на использование какой-либо операционной системы, соответственно они являются кросс-платформенными, что делает их универсальным и удобным инструментом для работы в любых условиях.

4.2. База данных системы

Центральным элементом системы SkyTS является база данных. Она создана на основе СУБД MySQL версии 5.0. В ней хранятся все сведения о сущностях системы: приложения, зарегистрированные пользователи, серверы, и т.д. В качестве хранилищ загруженных приложений используются SVN-репозитории. Данный подход является стандартным при проектировании эффективных информационных систем, поскольку в этом случае происходит разделение информационной части системы (база данных) от файловой, управление которой поручается файловому серверу. Это ведет к уменьшению нагрузки как на СУБД, так и на вычислительные узлы и, соответственно, повышению быстродействия системы в целом.

4.3. Репозитории приложений

Данный компонент необходим для хранения загруженных приложений. В данный момент используется репозиторий, основанный на системе Subversion (SVN). Репозиториев может быть несколько. Это необходимо для распределения нагрузки на сетевые каналы и ускорения доставки приложений к вычислителям. Соответственно,

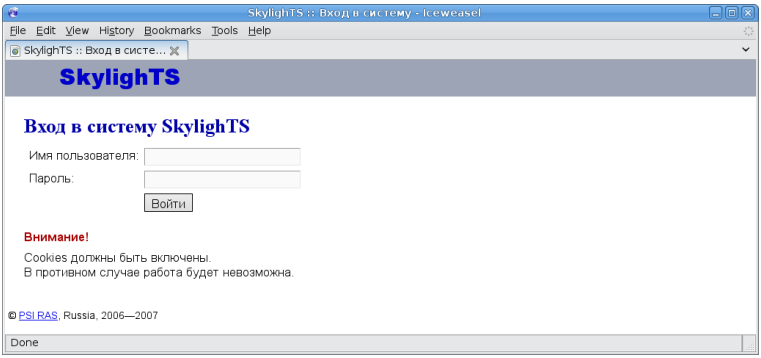

Рис. 1. Форма авторизации

производится синхронизация хранилищ с некоторой периодичностью, которая задается администратором системы SkyTS.

4.4. Роли пользователей системы

В целях обеспечения безопасности, в системе SkyTS существуют 3 роли пользователей:

(1) Администраторы; пользователи с такой ролью имеют доступ ко всем функциям интерфейса и могут просматривать любую информацию, содержащуюся в базе данных.
(2) Модераторы; пользователи с такой ролью имеют ограничение на использование функций интерфейса и доступ к информации.
(3) Непривилегированные (обычные) пользователи.

4.5. Авторизация

Авторизация производится с помощью Web-формы, содержащей 2 атрибута: имя пользователя и его пароль (см. рис. 1).

4.6. Управление приложениями

4.6.1. *Просмотр доступных приложений*

После авторизации, отображается форма, содержащая список доступных приложений (см. рис. 2). Среди них могут быть такие, о которых можно только просмотреть сведения, а есть такие, которые

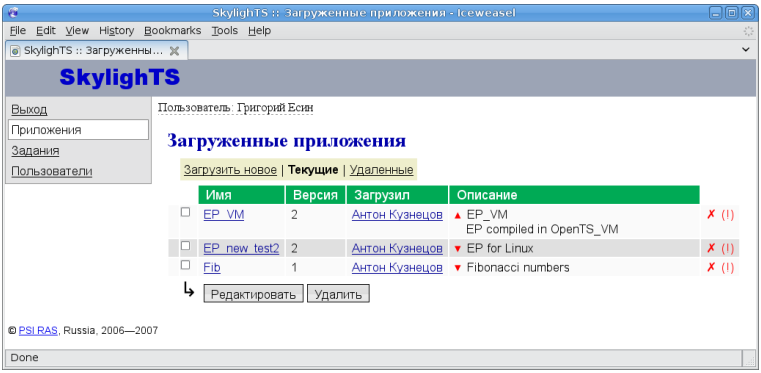

Рис. 2. Список доступных приложений

можно редактировать и удалять. Доступность этих функций определяется ролью пользователя и его участием в разработке приложения.

Также справа от таблицы указаны один или более флагов, сообщающих является ли программа проверенной, находится ли она в данный момент в очереди и если да, то какой у нее статус счета.

4.6.2. *Загрузка нового приложения*

Загрузка нового приложения производится из Web-формы (см. рис. 3). В ней содержатся поля для составления описания приложения. Загружаемое приложение оформляется в виде файлового архива. После загрузки приложения его архив распаковывается и сохраняется в SVN-репозитории. Сведения о приложении сохраняются в базу данных, в числе прочего отмечается кто загрузил и указывается список разработчиков приложения. Этот список необходим как для общего информирования, так и для того, чтобы обозначить, кто может изменять сведения о данном приложении или обновлять его файлы в репозитории.

4.6.3. *Просмотр и редактирование информации о приложении*

На странице просмотра информации о приложении доступны все сведения о нем, включая имя, версию, дату загрузки, имя разработчика и описания. Также на ней расположены инструменты управления приложением, которые позволяют отредактировать приложение,

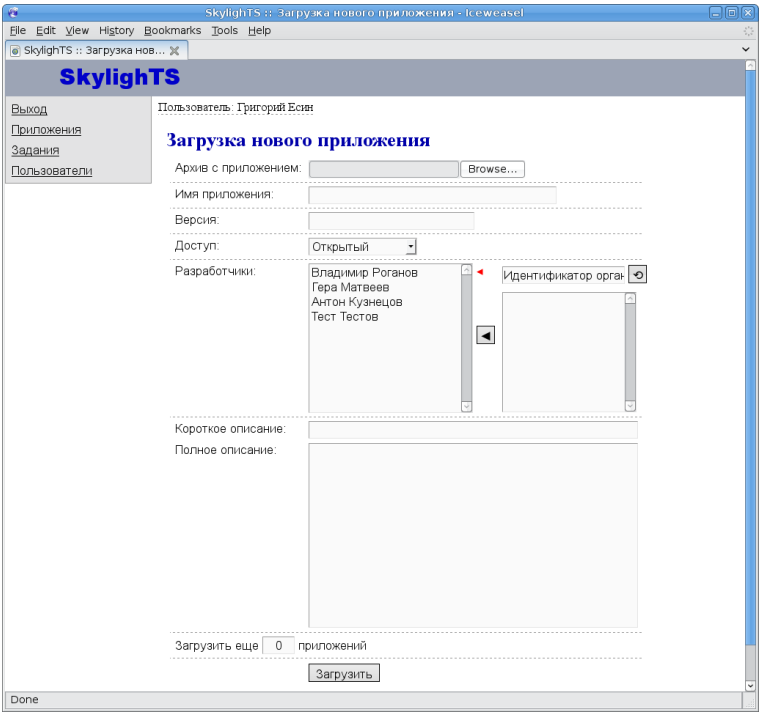

Рис. 3. Форма загрузки нового приложения

удалить его из системы или запустить на счет. Web-форма для редактирования приложения аналогична Web-форме для его загрузки (см. рис. 3) .

При удалении приложения в системе SkyTS данное приложение просто помечается как неактивное. Администратор и модератор могут изменить статус приложения с неактивного на активный и таким образом восстановить удаленное приложение.

Редактировать и удалять приложения могут только следующие категории пользователей системы:

- администраторы системы SkyTS;
- модераторы, принадлежащие к той же организации, что и владельцы приложения;
- пользователь, загрузивший приложение;

- пользователи, отмеченные как разработчики данного приложения.

4.7. Управление очередью задач

Управление очередью задач - одна из основных функций Web-интерфейса. Очередь задач была введена с целью распределения вычислительной нагрузки по сегментам вычислительной сети. Web-интерфейс системы позволяет производить следующие функции по управлению заданиями:

- создание нового задания;
- изменение его параметров (аргументов командной строки);
- запуск задания;
- приостановка задания;
- удаление из очереди;
- просмотр статуса;
- получение результатов.

4.7.1. *Постановка задачи в очередь*

При постановке задачи в очередь пользователь выбирает те приложения, которые он собирается запустить, указывает параметры командной строки, ссылки на файлы-данные и/или загружает файлы-данные, а также вносит краткое описание задания.

Кроме того, могут быть дополнительно указаны такие параметры, как количество необходимых процессов, необходимость оповещения об изменении статуса задания (задание принято/запущено/завершено) и следует ли послать по электронной почте результаты счета.

После того как указаны все необходимые параметры, заданию назначается приоритет и оно ставится в очередь. Положение задания в очереди определяется по его приоритету, который зависит от приоритета пользователя. Администратор и/или модератор могут повысить или понизить приоритет задания.

4.7.2. *Получение результатов счета*

В любой момент времени пользователь может просмотреть список заданий, находящихся в очереди. Но есть ограничения:

- задания, которые он создал сам;
- задания пользователей из той же организации;

- задания, созданные на основе приложений с открытой лицензией.

Получить результат можно через Web-интерфейс. В этом случае создается список завершенных задач. После просмотра какого-либо элемента этого списка, он (элемент) перемещается в архив заданий. Он представляет собой Web-форму, с помощью которой можно производить следующие действия:

- просмотреть какое-либо задание;
- удалить какое-либо задание из архива.

5. Серверный компонент системы SkyTS

Серверный компонент системы SkyTS действует как связующее звено между T-приложениями и ресурсами, необходимыми для счета пользовательских задач. Он способен выполнять T-приложения в отказоустойчивом режиме и взаимодействовать с вычислительными модулями и базой данных приложений. Серверный модуль написан на интерпретируемом языке TCL, что обеспечивает переносимость кода на большинство современных программно-аппаратных платформ. Язык TCL прост в освоении, активно развивается, и недавно был признан одним из 11-ти наиболее безопасных и защищенных технологий с открытым исходным кодом.

Для работы сервера необходимо наличие полноценной базы данных приложений, в которую поступают данные от Web-интерфейса. Если связь с БД отсутствует, то работа сервера невозможна. Отсутствие связи с БД еще не означает полную неработоспособность всей системы. Предусмотрено наличие нескольких территориально-распределенных серверов, которые взаимодействуют друг с другом, со счетными модулями и с общей базой данных приложений. В случае сбоя или перегруженности какого-либо сервера, другие сервера способны взять на себя дополнительную нагрузку по обслуживанию рабочих подключений.

В данном контексте, рабочее подключение — это компьютер, предоставляющий свободные ресурсы для распределенного счета T-приложений. На этом компьютере установлен, сконфигурирован и запущен счетный модуль системы SkyTS.

Через определенные промежутки времени (а также на начальном этапе работы) сервер делает запрос в БД на предмет заданий, готовых к счету. Постановка заданий в очередь на счет ведется с учетом

времени их запуска через Web-интерфейс и категорий пользователей-владельцев этих заданий. Эти данные определяют приоритет заданий и их порядок в очереди на выполнение.

В соответствии со схемой запуска Т-приложений в отказоустойчивом режиме, серверным модулем вызывается мастер-процесс. Он служит сокет-сервером, который ожидает подключения рабочих процессов по каналам TCP/IP. Рабочие процессы вызываются на тех компьютерах, на которых установлен и настроен счетный модуль системы SkyTS. Статус задания на Web-интерфейсе меняется на „работает“ в тот момент, когда происходит запуск мастер-процесса Т-приложения.

Соединение между серверным и счетными модулями защищено средствами протокола SSL. Серверный и счетный модули используют сертификаты, позволяющие шифровать все пересылки данных надежным ключом длиной в 2048 бит.

Взаимодействие по сети со счетными компонентами системы осуществляется посредством обмена сообщениями в определенном формате в соответствии с сетевым протоколом. В процессе взаимодействия со счетным модулем сервер получает от него информацию о программно-аппаратной архитектуре вычислительного узла, определяет степень его надежности, и принимает решение о том, какое из заданий очереди выделить данному рабочему подключению на счет. Если сервер принял решение об отправке задания данному счетному узлу, то он сообщает информацию о задании (название, версия, объем приложения в байтах, аргументы командной строки), а затем передает исполняемый код приложения, либо архив с приложением и всеми нужными файлами. Если это приложение уже ранее было загружено на рабочий узел, то оно не загружается снова.

6. Счетный компонент системы SkyTS

Счетный компонент системы SkyTS предназначен для инсталляции на обычные компьютеры, ресурсы которых большую часть времени простаивают. Владельцы этих компьютеров могут быть заинтересованы в утилизации аппаратных мощностей в свободное от работы время. С этой точки зрения для проекта подходят как корпоративные локальные компьютерные сети, так и одиночные компьютеры в сети Интернет. Данный компонент также написан на интерпретируемом языке TCL, что особенно важно для его переносимости.

Во время запуска счетный модуль соединяется с сервером и запрашивает задание. Если заданий нет, то программа ждет определенное время, а затем снова повторяет попытку запроса. Число таких итераций не ограничено.

После получения задания счетный модуль генерирует и исполняет командный сценарий на языке „Shell“ или „Windows command shell“ (в зависимости от ОС). Этот сценарий содержит объявления переменных окружения, нужных для подключения рабочих процессов к мастер-процессу, а также команду запуска исполняемого модуля в режиме „smp“ с указанием числа процессов, равного числу ядер процессора на рабочем узле. При успешном завершении работы приложения ведется передача серверу сообщения об этом. В случае потери связи с сервером возможны попытки повтора соединения; если они оказались неудачными, то при следующей попытке запроса задания этой рабочей машиной сервер заносит в БД запись о снижении степени надежности счетного модуля.

7. Заключение

Разработана система распределенных вычислений (Grid-система) SkyTS. Система предназначена для работы T++ приложений в распределенном отказоустойчивом режиме на множестве разрозненных неоднородных компьютеров в сети Интернет. Система может состоять из нескольких серверных и неограниченного количества вычислительных модулей, устанавливаемых на компьютеры в сети Интернет. Подача и мониторинг вычислительных заданий в системе осуществляется авторизованными пользователями через специальный Web-интерфейс. По сравнению с существующими аналогами, Grid-система SkyTS имеет целый ряд преимуществ, среди которых простота реализации, безопасность за счет использования виртуальных машин, масштабируемость на неограниченное множество серверных и счетных узлов, нацеленность на решение прикладных задач в области инженерного анализа и суперкомпьютерного моделирования.

8. Благодарности

Работы, положенные в основу данной статьи, были выполнены в рамках:

- проекта „Разработка и реализация языков T++ и соответствующих им средств для эффективной поддержки высокопроизводительного параллельного счета“ по Программе фундаментальных научных исследований ОНИТ РАН „Архитектура, системные решения, программное обеспечение, стандартизация и информационная безопасность информационно-вычислительных комплексов новых поколений“;
- программы „СКИФ-ГРИД“ „Разработка и использование программно-аппаратных средств ГРИД-технологий и перспективных высокопроизводительных (суперкомпьютерных) вычислительных систем семейства „СКИФ“ (2007–2009 гг.);
- научно-технической программы Союзного государства „Развитие и внедрение в государствах-участниках Союзного государства наукоемких компьютерных технологий на базе мультипроцессорных вычислительных систем (шифр „ТРИАДА“)“ (2005–2008 гг.).

Список литературы

[1] Абрамов С. М., Адамович А. И., Инюхин А. В., Московский А. А., Роганов В. А., Шевчук Ю. В., Шевчук Е. В. *Т-система с открытой архитектурой* // Суперкомпьютерные системы и их применение SSA'2004: Труды Международной научной конференции, 26–28 октября 2004 г., Минск, ОИПИ НАН Беларуси. — Минск, 2004, с. 18–22. ↑1

[2] Официальный сайт системы программирования OpenTS: Электронный сетевой ресурс, http://www.opents.net. ↑1

[3] Абрамов С. М., Кузнецов А. А., Роганов В. А. *Кроссплатформенная версия Т-системы с открытой архитектурой* // Параллельные вычислительные технологии (ПаВТ'2007): Труды Международной научной конференции, 29 января –2 февраля 2007 г., Челябинск. — Челябинск: изд. ЮУрГУ, 2007, с. Т.1, 115–121. ↑1

[4] Абрамов С. М., Московский А. А., Роганов В. А., Велихов П. Е. Пути ученого. Е.П. Велихов: Суперкомпьютерные и GRID-технологии. — М.: РНЦ „Курчатовский институт“, 2007. — ISBN 978-5-9900996-1-6. — 314–324 с., Под общей редакцией академика РАН В.П. Смирнова. ↑1

[5] Абрамов С. М., Есин Г. И., Загоровский И. М., Матвеев Г. А., Роганов В. А. *Принципы организации отказоустойчивых параллельных вычислений для решения вычислительных задач и задач управления в Т-Системе с открытой архитектурой (OpenTS)* // Программные системы: теория и приложения (PSTA-2006): Труды Международной научной конференции, 23–28 октября 2006 г., Переславль-Залесский, ИПС РАН. — Переславль-Залесский, 2006, с. 257–264. ↑2

[6] Кузнецов А. А., Роганов В. А. *Экспериментальная реализация отказоустойчивой версии системы OpenTS для платформы Windows CCS* // Суперкомпьютерные системы и их применение (SSA'2008): Труды Второй Международной научной конференции, 27–29 октября 2008 г., Минск. — Минск: ОИПИ НАН Беларуси, 2008. — ISBN 978-985-6744-46-7, с. 65–70. ↑2

Исследовательский центр мультипроцессорных систем ИПС РАН

G. I. Esin, A. A. Kuznetsov, V. A. Roganov. *Fault-tolerant software prototype "SkyTS" for distributed computing of heavy-load T++ applications in heterogeneous distributed environment* // Proceedings of Program Systems institute scientific conference "Program systems: Theory and applications". — Pereslavl-Zalesskij, v. **1**, 2009. — p. 225–243. — ISBN 978-5-901795-16-3 (*in Russian*).

Abstract. This paper describes the design and development of the distributed computing system "SkyTS" for running heavy-load T++ parallel applications. A description of the "SkyTS" fault-tolerant scalable prototype is given that supports heterogeneous networks.