Архитектура распределенной системы устойчивой к произвольным отказам

А.Н. Фирсов

Пермский государственный университет
a_firsov@mail.ru

Аннотация

В статье рассматривается архитектура и некоторые вопросы реализации статистически оптимизируемой отказоустойчивой системы, которая позволяется добиться большей, чем существующие решения, эффективности при работе в гетерогенных сетях, благодаря выбору на основе статистических данных наилучших для данной ситуации алгоритмов и их параметров.

1. Введение

В настоящее время все острее встают вопросы обеспечения отказоустойчивости информационных систем (ИС). Этому есть несколько причин. Во-первых, системы с каждым годом усложняются, включают в себя все большее количество компонент, а, следовательно, вероятность того, что в одной из компонент системы произойдет сбой, увеличивается. Это может привести к отказу всей системы, если она не спроектирована с учетом такой возможности. Во-вторых, информационные системы проникают во все области жизнедеятельности, люди все больше полагаются на ИС, все больше зависят от них, поэтому последствия отказа ИС становятся все более и более критичными, если не катастрофическими.

Византийская модель сбоев [1] (или модель произвольных сбоев) является наиболее общей, включающей в себя все остальные, моделью. Она не накладывает никаких ограничений на действия сбойных процессоров и, благодаря этому, позволяет моделировать такие типы сбоев, как отказы оборудования, потеря связи, неисправность оборудования, программные сбои, ошибки операторов и действия злоумышленников.

Особый интерес представляют асинхронные распределенные системы, т.к. они не накладывают никаких временных ограничений, и, благодаря этому, идеально подходят для работы в сети интернет. Наиболее универсальной моделью вычислительного узла является машина с конечным числом состояний (state machine).

В данной работе рассматриваются асинхронные распределенные системы, которые, во-первых, устойчивы к произвольным отказам, во-вторых, позволяют выполнять прикладные программы, требующие сохранять свое состояние между обращениями к ним, в-третьих, обеспечивают отказоустойчивость на программном уровне без необходимости в какой-либо специальной аппаратуре и, в-четвертых, делают все это прозрачно для прикладных программ.

В настоящее время существует несколько таких систем: Rampart [2], SecureRing [3], BFT [4] и т.д.. Все они реализуют единственный известный на сегодня программный метод защиты от произвольных сбоев – активную репликацию. Суть этого метода заключается в том, что параллельно выполняются нескольких копий (или, как их еще называют, реплик) одной и той же программы. Отличие между этими системами в основном заключается в стеке протоколов, который обеспечивает согласованную работу всех реплик.

Существующие системы обладают несколькими недостатками, корнем которых является то, что эти системы плохо приспособлены для работы в гетерогенных сетях. Под гетерогенностью сети здесь понимается различная надежность включенных в эту сеть компьютеров, различная скорость и задержка при обмене информацией между компьютерами, различная производительность. Именно такой сетью является Интер-

нет, для работы в котором и создавались приведенные выше отказоустойчивые системы, однако то, что они не учитывают неоднородность, приводит к неэффективной работе этих систем:

- Предоставляемый ими стек протоколов фиксирован, а как показано в работе [7], в разных ситуациях эффективным оказывается применение разных протоколов.
- Не предоставляется алгоритм для определения оптимального значения параметров работы системы (частота создания контрольных точек, время ожидания у ненадежных детекторов сбоев и т.д.). А следует заметить, что значения этих параметров существенно влияют на эффективность работы системы[7].
- Не дается оценка надежности получаемого решения. А именно в этом показателе наиболее заинтересован пользователь отказоустойчивых систем, т.к. абсолютная надежность не достижима.
- Не определяется вклад каждого процессора в общий вычислительный процесс. Создатели приведенных выше систем такую задачу перед собой и не ставили, но как продемонстрировано платформой BOINC[10], это может оказаться очень полезным.

Возможный подход к решению этих проблем представлен в работах [6-9]. Основная его идея заключается в том, что на основе статистических данных о сети (производительность и надежность вычислительных узлов, скорость и задержка у линий связи), на основе которой работает система, прикладной программе (частота групповых рассылок и время, требующееся на создание контрольной точки) и требований пользователя возможно однозначным образом определить наилучший для данной ситуации протокол и его оптимальные параметры, оценить надежность полученного решения и вклад каждого вычислительного узла.

В работах [6-9] рассматривается теоретическая часть предлагаемого подхода. В данной же статье описывается архитектура предлагаемой системы SFT (Statistical Fault-Tolerance) и некоторые вопросы ее реализации.

2. Требования к архитектуре

В предыдущем разделе уже упоминалось, что в разных ситуациях оптимальными могут оказаться разные протоколы, поэтому первое требование, которое предъявляется к статистически оптимизируемой отказоустойчивой системе, это возможность динамического изменения используемых протоколов.

В существующих системах можно выделить три уровня протоколов:

1. Протокол членства в группе – обеспечивает исключение отказавших вычислительных узлов из группы и добавление взамен их новых.
2. Протокол надежной групповой рассылки – обеспечивает, что все корректно работающие вычислительные узлы приходят к соглашению о принятом сообщении, несмотря на то, что рассылающий процесс может быть сбойным и посылать разные сообщения разным вычислительным узлам. Это обеспечивает нераспространение сбоя на корректно работающие процессы.
3. Протокол атомарной групповой рассылки – гарантирует, что все вычислительные узлы обрабатывают сообщения в одном и том же порядке.

Взаимодействие между уровнями не стандартизировано. Каждая система его организует по-своему. Так в системе BFT [4] протокол членства в группе в явном виде даже не присутствует.

Для обеспечения возможности динамической замены протоколов, прежде всего, необходима стандартизация интерфейсов их взаимодействия между собой и с прикладными задачами. Это позволит системе гибко подстраиваться под среду, в которой она работает.

Второе требование, предъявляемое к разрабатываемой системе, относится к собираемым статистическим данным. Данные о некотором вычислительном узле (надежность, производительность, скорость и задержки при обмене сообщениями с другими вычислительными узлами) не могут находиться на этом же самом узле, т.к. при отказе этого узла они будут потеряны или искажены. Вообще ни один из вычислительных узлов не должен содержать какие-либо уникальные данные, т.к. каждый узел может выйти из строя. Необходимо использовать некий механизм дублирования этих данных, чтобы защитить их от сбоев. С другой стороны, хранение всех статистических данных на каждом из узлов является неэффективным. Логичнее всего использовать тот самый механизм дублирования, который эта система и реализует. Работать с задачей хранения статистических данных, как с любой прикладной задачей, отказоустойчивость которой требуется обеспечить. А поступающие со всех вычислительных узлов новые статистические обрабатывать как клиентские запросы к прикладной программе.

Так как в системе одновременно должны будут работать несколько задач, то потребуется некоторая служба, которая будет объединять доступные физические процессоры в группы, на основе которых будут работать отказоустойчивые программы. По аналогии со сбором статистики, такая служба балансировки должна быть отказоустойчивой и проще всего обеспечить эту отказоустойчивость теми же способами, что и отказоустойчивость всех остальных прикладных задач.

На первый взгляд, может показаться, что образуется некоторый замкнутый круг: отказоустойчивая система в своем функционировании опирается на службы статистики и балансировки. А они в свою очередь опираются на отказоустойчивую систему. В действительности же, сбор статистики лишь повышает эффективность работы системы, а не является для нее жизненно важным. Поэтому, первое время отказоустойчивая система может при своей работе использовать не самые оптимальные в конкретной ситуации алгоритмы. Собрать же и назначить группы для выполнения системных служб при первом запуске системы должен будет администратор.

Еще одной проблемой, которую должна решать структура отказоустойчивой системы, является «миграция задач». Она заключается в том, что вследствие отказов вычислительных узлов, они исключаются из групп, на основе которых выполняются прикладные задачи, а на их место назначаются новые вычислительные узлы. Поэтому по истечению достаточно большого промежутка времени составы групп могут целиком измениться. Если клиентский компьютер часто обращается к прикладной задаче, то он будет получать информацию об изменении группы, от тех вычислительных узлов, которые остались в группе с момента его последнего обращения. Однако при длительном отсутствии коммуникаций группа может целиком измениться и тогда клиент потеряет с ней связь.

3. Архитектура статистически оптимизируемой отказоустойчивой системы

Предлагаемая архитектура статистически оптимизируемой системы, устойчивой к произвольным отказам, приведена на рисунке 1. Ключевым элементом предлагаемой архитектуры является понятие **виртуального процессора** (или **виртуального вычислительного узла**). Оно позволяет абстрагироваться от используемых протоколов членства в группе, надежной групповой и атомарной рассылки. С точки зрения прикладной программы, отказоустойчивость которой требуется обеспечить, или системной службы совершенно все равно какие протоколы используются. На текст программы это никоим образом не влияет.

Так же происходит абстрагирование от физических процессоров. В системах Rampart, SecureRing и BFT надежность получаемого решения определяется максимальным количеством вычислительных узлов, которые одновременно могут выйти из строя. В неявном виде это подразумевает, что все вычислительные узлы, входящие в систему

имеют одинаковую надежность. В действительности же это далеко не так. В предлагаемой системе задается один параметр – вероятность отказа получаемого решения. Если такую или меньшую вероятность отказа может обеспечить один единственный вычислительный узел, то он и будет использоваться в одиночку. Прикладной задаче без разницы количество одновременно исполняемых реплик.

На основе виртуального процессора работают служба сбора и хранения статистики. Она принимает клиентские запросы от подсистемы сбора статистики каждого вычислительного узла, агрегирует и сохраняет их. К службе сбора статистики при своей работе обращается служба балансировки, так же работающая на основе некоторого виртуального процессора.

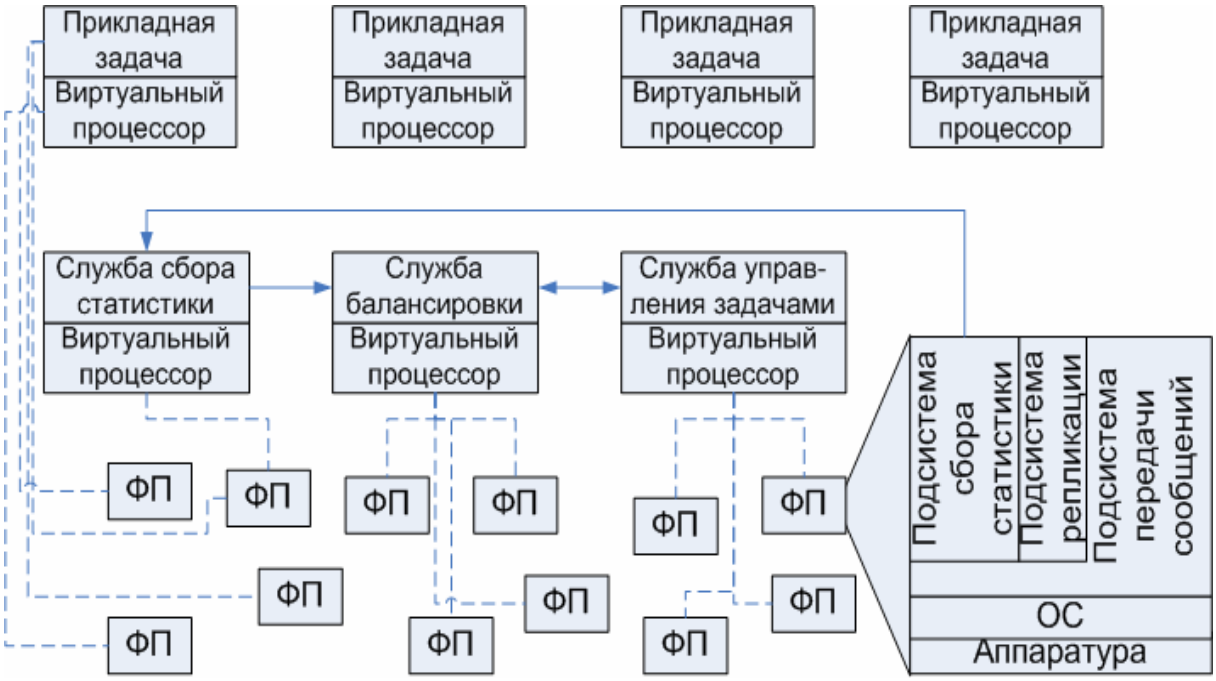

Рисунок 1. Архитектура статистически оптимизируемой отказоустойчивой системы.

Проблема миграции задач решается следующим образом. Вводится еще одна системная служба – служба управления задачами. Эта служба хранит информацию об идентификаторе задачи и соответствующей ей группе. При изменении состава группы эта информация обновляется. Если некоторый клиент после длительного отсутствия коммуникаций теряет связь с группой, вследствие ее полного изменения, он может обратиться с запросом к этой службе и получить адреса новых членов этой группы. Естественно, возникает проблема с поиском самой этой службы и ее миграцией. Для этого каждый вычислительный узел хранит идентификаторы физических процессоров, входящих в состав виртуального процессора, на котором работает служба. Если же состав группы все-таки меняется, то исключенные вычислительные узлы сохраняют информацию о новом составе группы и возвращают ее, если к ним кто-либо обратиться из-за устаревшей информации.

4. Заключение

Предлагаемая отказоустойчивая система позволяет на основе гетерогенной сети из обычных компьютеров создать высокопроизводительную вычислительную систему. Базой такой системы может служить весь компьютерный парк университета или предприятия. Так же это могут быть группы по интересам, соединенные Интернетом. Большинство персональных компьютеров в среднем используют процессор менее чем на пять процентов и не используемые ресурсы могут найти стоящее применение.

В отличие от платформы BOINC[10], организованной на таких же началах, предлагаемая система не требует наличия надежного центрального сервера, ответственного за распределение задач. Кроме, того виртуальные процессоры могут сохранять свое состояние между запросами, что позволяет создать реализацию MPI, работающую на их основе. Это даст возможность запускать MPI-приложения не только на специально выделенных кластерах, а в обычных терминальных классах, не заботясь о том, что компьютеры могут быть перезагружены или находится под управлением недоброжелателей.

5. Литература

1. Lamport L., Shostak R. and Pease M. The Byzantine Generals Problem. // ACM Transactions on Programming Languages and Systems –July 1982. –Vol. 4, N.3. –P. 382-401.
2. Reiter M. The Rampart Toolkit for Building High-Integrity Services // Selected Papers from the International Workshop on Theory and Practice in Distributed Systems – Springer-Verlag, 1994. –P. 99-110
3. Kihlstrom K. P., Moser, L. E. and Mellar-Smith, P. M. The SecureRing group communication system // ACM Transactions on Information and System Security (TISSEC), – November 2001. –Vol. 4, N. 4. –P. 371-406.
4. Castro M. Practical Byzantine Fault Tolerance // Massachusetts Institute of Technology, 2001. PhD thesis. -172 p.
5. Bracha G. and Toueg S. Asynchronous Consensus and Broadcast Protocols // Journal of the ACM. –October 1985. –Vol. 32, N.4. –P. 824-840.
6. Фирсов А.Н. Оптимизация на основе статистических данных асинхронной распределенной системы, устойчивой к произвольным отказам // Параллельные вычислительные технологии (ПаВТ'2009): Труды международной научной конференции (Нижний Новгород, 30 марта – 3 апреля 2009 г.). – Челябинск: Изд. ЮУрГУ, 2009г. – С.765-771.
7. Фирсов А.Н. Оценка эффективности некоторых оптимизаций протоколов надежной и атомарной групповой рассылки // Вестник Пермского госуниверситета. Математика. Механика. Информатика. 2009г. – С.161-168.
8. Фирсов А.Н. Статистически оптимизируемая отказоустойчивая распределенная система // III конференция-конкурс грантов аспирантов и молодых ученых механико-математического факультета Пермского государственного университета. Сборник тезисов научных докладов конференции – Пермь, 2008г. – С.74-78.
9. Фирсов А.Н. Асинхронный протокол надежной групповой рассылки с переменным числом активных реплик // Математика программных систем: Межвузовский сборник научных трудов – Пермский университет – 2008г. – С.166-177.
10. Anderson, D.P., Christensen, C. and Allen, B. Designing a runtime system for volunteer computing // Proceedings of the 2006 ACM/IEEE conference on Supercomputing –2006. –Article No. 126