УДК 004.724.4(045)

Кулаков Ю. А., д-р техн. наук
Кулаков А. Ю.
Желудков В.В.

СПОСОБ ПЛАНИРОВАНИЯ ЗАДАЧ В DESKTOP GRID-СИСТЕМАХ

Национальный технический университет Украины «КПИ»

Предложен способ повышения эффективности прохождения задач в DESKTOP GRID-системах на базе применения группового планировщика. Планировщик группирует ресурсы по общим характеристикам и применяет разные механизмы для планирования в каждой группе

Введение

В настоящее время стремительное развитие вычислительной мощи компьютеров и высокоскоростных сетевых технологий, а также снижение их стоимости привлекает все больше энтузиастов к распределенным вычислением и *GRID*-системам. Классические *GRID*-системы призваны объединить разнородные, крупномасштабные и относящихся к разным организациям ресурсы, чтобы обеспечить прозрачный, безопасный и координированный доступ к этим ресурсам. В то время как *DESKTOP GRID*-системы позволяют собрать воедино вычислительные ресурсы, которые простаивают у пользователей персональных компьютеров, при помощи интернет.

Планирование в *DESKTOP GRID* отличается от *GRID* следующими характеристиками: типом ресурсов, степенью участия волонтеров, характеристиками доверия и отказов, спецификой выполнения задач и т.д. В основном, планирование в *DESKTOP GRID* – это процесс назначения задач к наиболее подходящим ресурсам, выполняемый централизовано или распределенно. Большинство *DESKTOP GRID*-систем не требуют локального планировщика, поскольку цель планирования персональный компьютер, а не подразделение как в *GRID*. Планирование в *DESKTOP GRID* усложнено неоднородностью, неустойчивостью, отказами и ненадежностью ресурсов[1]. Но, самое важное требование к планировщику – это гибкость. *DESKTOP GRID*-системы должны уважать автономность волонтеров, которые могут свободно участвовать в публичных вычислениях. Поэтому, планировщик в *DESKTOP GRID* обязан использовать доступные или свободные ресурсы так быстро как это возможно. Таким образом, возникает необходимость разработки нового планировщика, который сможет приспособиться к этим особым требованиям.

Обзор существующих решений

Привлекательность и интерес к *GRID*-системам стремительно увеличивается. В подтверждение этому можно привести такие удачные проекты как *GIMPS* [2], *distributed.net* [3], и *SETI@Home* [4] которые насчитывают огромное количество участников-энтузиастов по всему миру. В области *DESKTOP GRID* были сделаны некоторые исследования и разработаны следующие платформы: *BOINC* [5], *XtremWeb* [6], Entropia [7], *Bayanihan, Javelin, Computer Power Market* (*CPM*), *Charlotte, POPCORN, Web-Com, Cluster Computing On the Fly* (*CCOF*) и т.д.

Типичная среда вычислений *DESKTOP GRID* состоит из клиентов и серверов. Клиент размещает параллельную задачу. Сервер предоставляет свои свободные вычислительные ресурсы. Такие сервера часто называют волонтерами. Также в случае централизованного планирования существует управляющий сервер, который контролирует выполнение задачи.

Постановка задачи

Для адаптации к особенностям

DESKTOP GRID планировщик должен иметь метод группировки ресурсов, который гарантирует, что волонтеры с похожими характеристиками (такими как, возможность, производительность, доступность, нагрузка, репутация, изменчивость) будут сгруппированы вместе. Необходимо применять алгоритм планирования отдельно для каждой группы в зависимости от ее характеристик. Использование группировки ресурсов должно помочь более эффективно планировать и управлять задачами. Очень важно как группировать волонтеров по группам и по каким критериям, потому-что планирование и управление ресурсами производиться в зависимости от базовых характеристик групп.

Внедрение метода группировки ресурсов в планирование дает следующие преимущества.

1. Планировщик может применять различные алгоритмы репликации, сертификации результатов и отказоустойчивости для каждой однородной группы.

2. Возможность применения репутации в планировании.

3. Повышается надежность и производительность.

Решение задачи

Критерии группировки ресурсов

Во время классификации волонтеров важны следующие факторы: производительность *CPU,* объем оперативной и постоянной памяти и пропускная способность сети. Однако, самые важные факторы – это расположение, время участия, степень отказов, доступность и надежность волонтера (рис. 1)

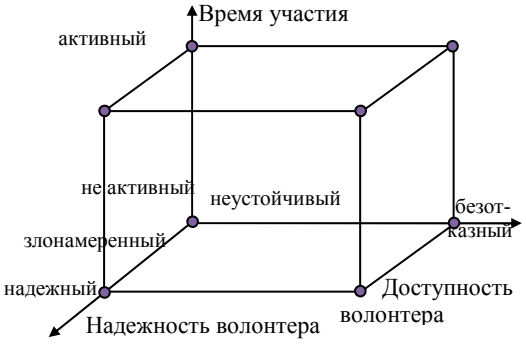

Рис. 1. Критерии группировки

Неустойчивость и доступность волонтера

Неустойчивость или непостоянство означает, что волонтер может покинуть публичное выполнения задачи не завершив свою часть. Также волонтеры подвержены сбоям и отказам линий Internet. Неустойчивость и отказы вызывают задержку или блокировку выполнения задачи и могут спровоцировать частичную или полную потерю вычислений.

Доступность волонтера (α_v) – это вероятность того что волонтер будет работоспособный и сможет участвовать в публичном выполнении.

$$\alpha_v = \frac{t_{VP}}{t_{VP} + t_{Boco}}, \qquad (1)$$

где t_{VP} – среднее время устойчивой работы, t_{Boco} – среднее время восстановления. Доступность волонтера отражает степень автономности волонтера, в то время как, традиционная доступность в распределенных системах в основном относится к аварийным отказам.

Доверительные отношения и надежность волонтера

Доверительные отношения подразумевают, что волонтер выполняет задачу корректно и возвращает правильный результат. Злонамеренные волонтеры могут вмешаться в вычисление.

Надежность волонтера – это вероятность того что результат произведенный волонтером правильный.

$$C_v = \frac{ПР}{ОР + ПР + НР}, \qquad (2)$$

где *ПР* отображает количество правильных результатов, *ОР* – ошибочных результатов, *НР* – незавершенный результат. *ОР+ПР+НР* отображает общие количество задач которые выполнил волонтер.

Время участия волонтера

Когда волонтер предоставляет ресурсы для публичных вычислений он, также, может выполнять свои приватные вычисления.

Время участия волонтера – период во

время, которого предполагается, что волонтер будет предоставлять ресурсы.

$$T = T_a + T_н, \qquad (3)$$

где T_a – активное участие волонтера когда, в основном, выполняются публичные вычисления, а $T_н$ – не активное, когда преобладают приватные вычисления.

Время эффективного участия волонтера – период во время которого волонтер участвует в публичных вычислениях.

$$\tau = a_v \times T. \qquad (4)$$

В процедуре планирования τ более подходящий параметр чем T, поскольку он учитывает автономность волонтера. Поэтому группы строятся в зависимости от τ.

Построение групп

Вначале волонтеры разбиваются на домашних и региональных волонтеров. Домашние волонтеры характеризуются как доноры ресурсов дома. В то время как региональные волонтеры относятся к всевозможным организациям. Региональные волонтеры подсоединены к локальной сети или интернет, тогда как домашние волонтеры подсоединены к интернет.

Затем разделим волонтеров на четыре класса в зависимости от α_v и τ (рис. 2). Если у волонтера высокое значение α_v и $\tau \geq \Delta$ ему присваивается класс A. Если у волонтера высокое α_v, но $\tau < \Delta$, то ему присваивается класс B. Если у волонтера низкое α_v то он, в зависимости от τ, попадает, либо в класс *C*, либо в класс *D*.

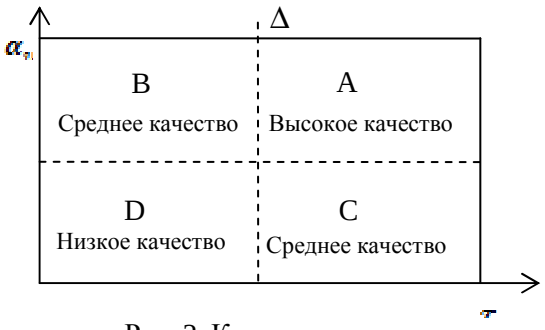

Рис. 2. Классы волонтеров

Группы волонтеров имеют следующие свойства. Волонтеры группы *A* имеют достаточно высокие значения параметров, чтобы надежно выполнять задачи. Они также размещают у себя мобильные агенты планирования. В группе *B* низкое значение τ и она не может завершить свои задачи из-за нехватки времени. В группе *C* у волонтеров достаточно времени чтобы выполнять задачи, но волонтер часто отключается, не завершив вычисление. Поэтому необходимо использовать отказоустойчивые механизмы выполнения. Среди групп волонтеров выполнением задач в основном занимаются группы *A* и *C*. Если задача перемещается во время выполнения группа *B* может использоваться для замещения. В противном случае, группа выполняет задачу в тестовом режиме для вычисления своих параметров. Группа *D* также выполняет задачи для тестирования. Если используются контрольные точки, то группы B и *D* могут выполнять не критичные ко времени приложения.

Репликация

В среде *DESKTOP GRID* отключение волонтера происходит гораздо чаще, чем аварии или сбои линий связи. Поэтому чтобы отражать автономность волонтера наш алгоритм репликации должен учитывать среднее время устойчивой работы.

Избыточность (*r*) для надежности рассчитывается при помощи неравенства:

$$\left(1 - e^{-\Delta / t'}\right)^r \leq 1 - \gamma,\ t' = \frac{\sum_{i=0}^{n} t_i}{n}, \qquad (5)$$

где t_i и t' – среднее время устойчивой работы волонтера и группы соответственно, γ – порог надежности, n – суммарное количество волонтеров в группе. В неравенстве (1) выражение $e^{-\Delta / t'}$ представляет надежность каждого волонтера в группе. $\left(1 - e^{-\Delta / t'}\right)^r$ – означает вероятность того что все реплики неисправны.

В *DESKTOP GRID* некоторые вредоносные волонтеры вмешиваются в вычис-

ление и возвращают неправильный результат. В этом случае мы должны провести сертификацию результата. С этой целью будем использовать голосование путем простого большинства. В роботе [8] для вычисления избыточности голосования используется неравенство:

$$\sum_{i=k+1}^{2k+1}(1-C_v^{'})^i(C_v^{'})^{2k+1-i} \le 1-v. \quad (6)$$

Мобильные агенты

Мобильные агенты могут адаптироваться к динамическим изменениям среды, так же как и к разнообразным свойствам волонтеров. В дополнение, поскольку агенты выполняются распределено, они могут снизить нагрузку на сервер.

Вычисление в *DESKTOP GRID* с использованием агентов состоит из следующих фаз: регистрации, предоставление работы, группировки ресурсов, размещения задач, выполнения задач, сборки результата.

Во время фазы регистрации волонтеры регистрируют свои базовые параметры такие как: мощность процессора, оббьем памяти и название ОС и дополнительные параметры такие как: время участия надежность и доступность.

Во время второй фазы предоставленная работа разделяется на множество задач, которые реализуются как мобильные агенты (мобильные агенты задачи).

Во время группировки ресурсов волонтеры разбиваются по группам в зависимости от своих параметров.

Фаза размещения не выполняется полностью на сервере. В место этого, сервер помогает мобильным агентам выполнять процедуру планирования. В мобильных агентах планирования так же реализованы алгоритмы отказоустойчивости. Сервер распространяет агенты планирования вспомогательным волонтерам в зависимости от параметров группы. Агент планирования, в свою очередь, распространяет агентов задачи внутри группы.

Во время фазы выполнения агенты задачи запускаются и взаимодействуют с агентами планирования, выполняя миграцию или репликацию, в случае возникновения отказов.

В фазе сборки результата агенты планирования собирают все результаты от агентов задачи и возвращают их серверу. После окончания работы всех агентов сервер возвращает результат клиенту.

Размещение агентов планирования

После создания групп волонтеров, сервер назначает агентов планирования в каждой группе. Тем не менее, не практично размещать агентов в каждую группу напрямую, поскольку некоторые группы не могут надежно завершать задачи. Поэтому необходимо построить новые группы планирования, объединив вместе некоторые группы волонтеров (табл. 1).

Первые две комбинации более подходящие чем последняя, поскольку задачи распространяются по обоим группам, в то время как, в последней комбинации задачи распространяются в основном в группу *AC*. Если сравнивать первые две комбинации, то первая комбинация выигрывает тем, что в ней оба параметра компенсируется. Поэтому для нашего алгоритма будем использовать комбинацию *AD & CB*.

Таблица 1. Комбинации групп

Комбинация	Кол-во размещаемых задач	α_v компенсация	τ компенсация
AD & CB	$AD \ge CB$	+	+
AB & CD	$AB \ge CD$	–	+
AC & BD	$AC \gg BD$	+	–

Для выполнения мобильного агента планирования выбираются волонтеры с самым высоким α_v и самыми лучшими базовыми характеристиками. Агенты выполняют разные процедуры планирования в зависимости от группы волонтеров.

Планирование группы *AD*:

1. отсортировать группу *A* вначале

по α_v, за тем по τ;

2. разместить агентов планирования на участников группы *A;*

3. если мобильный агент отказал продублировать задачу на нового волонтера из группы *A*, при помощи алгоритма репликации, либо, если разрешена миграция задач, переместить задачу на волонтера из групп *A* или *B*.

Планирование группы *CB*:

1. отсортировать группы *C* и *B* вначале по α_v, за тем по τ;

2. разместить агентов планирования на участников группы *C;*

3. при отказе реплицировать задачу на нового участника группы *C*, либо переместить задачу участнику *C* или *B* группы

В первую очередь задачи распространяются в группе *AD*, причем, самым лучшим волонтерам. Если используется репликация, то агент планирование вычисляет избыточность и выбирает реплики.

Результаты

Мы будем сравнивать наш адаптивный алгоритм с "энергичное" планированием (*eager scheduling*), которое чаще всего используется в динамической среде *GRID*. В этом способе планирования, волонтер просит у сервера новую задачу, как только завершил текущую. В результате, чем энергичнее волонтер работает, тем больше задач он выполняет.

Для моделирования будем использовать 4 варианта групп волонтеров.

Результаты моделирования (Рис 3.) показали увеличение количества выполненных задач при использовании группового планирования (ГП), по сравнения с "энергичным" планированием, на 3-22%, в зависимости от характеристик групп.

Выводы

1. Предложен способ повышения эффективности выполнения задач в DESKTOP *GRID*-системах на базе применения группового планирования, которое позволяет увеличить эффективную производительность на 3-22%.

2. Групповое планирование позволяет применять различные алгоритмы репликации, сертификации результатов и отказоустойчивости для каждой однородной группы.

3. Групповое планирование повышает эффективность работы мобильных агентов.

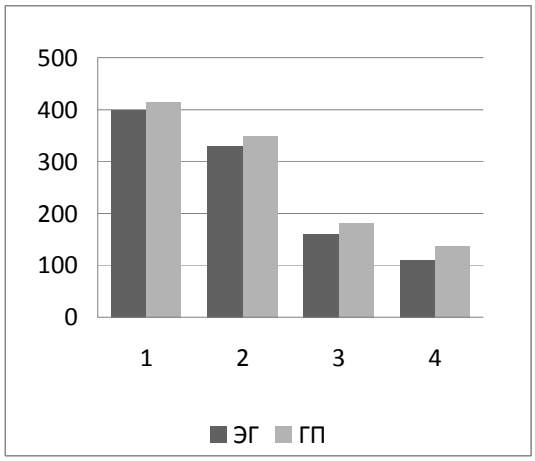

Рис 3. Результаты моделирования

Список литературы

1. *L.F.G. Sarmenta.* "Sabotage-Tolerance Mechanisms for Volunteer Computing Systems," Future Generation Computer Systems, vol. 18, issue 4, Mar. 2002. – P. 561–572.

2. The Great Internet Mersenne Prime Search (GIMPS),http://www.mersenne.org/

3. Distributed.net, http://distributed.net

4. SETI@home, http://setiathome.ssl.berkeley.edu

5. BOINC (Berkeley Open Infrastructure for Network Computing), http://boinc.berkeley.edu/

6. XtremWeb, http://www.lri.fr/ fedak/XtremWeb/

7. *G. Fedak, C. Germain, V. Neri, and F. Cappello*, "XtremWeb: A Generic Global Computing System," The 1st IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID2001): Workshop on Global Computing on Personal Devices, May 2001. – P. 582–587.

8. *Yu. A. Zuev,* "On the Estimation of Efficiency of Voting Procedures", Theory of Probability & Its Applications, vol. 42, no. 1. – 1998. – P. 73–81.