А.Д. Журавлев[1], О.С. Заикин[2]

ПОВЫШЕНИЕ ЭФФЕКТИВНОСТИ ПРОЕКТОВ ДОБРОВОЛЬНЫХ РАСПРЕДЕЛЕННЫХ ВЫЧИСЛЕНИЙ ПРИ ПОМОЩИ ВАРЬИРОВАНИЯ КРАЙНЕГО СРОКА ВЫПОЛНЕНИЯ ЗАДАНИЙ*

([1]Интернет-портал BOINC.ru, [2]Институт динамики систем и теории управления им. В.М. Матросова СО РАН)

Мы живем в информационную эпоху, и с каждым днем в мире становиться все больше разнообразных вычислительных систем (ПК, ноутбуки, планшетные компьютеры, смартфоны). На основе этих устройств создается все больше грид-систем, основанных на концепции добровольных распределенных вычислений. Основной принцип этого вида распределенных вычислений состоит в привлечении вычислительных ресурсов частных лиц со всего мира для решения масштабных научных задач. Вычисления организуются в рамках проектов добровольных вычислений, направленных, как правило, на решение задач из некоторой предметной области (астрономия, медицина, криптография и т.д.). Подавляющее большинство проектов основано на платформе BOINC [1], разрабатываемой с 1999 года в Калифорнийском университете в Беркли.

У каждого проекта добровольных вычислений есть отдельный сервер, на котором в рамках конкретного эксперимента генерируются и отправляются на хосты пользователей вычислительные задания (англ. workunit). Под хостом далее понимается вычислительная система (обычно это ПК), которую доброволец подключил к проекту. Результаты вычислений отправляются с хостов на сервер, после чего осуществляется их обработка и анализ. За проведенные вычисления добровольцы получают от проекта кредиты, которые не являются реальной материальной ценностью, но при этом стимулируют соревнования между добровольцами и командами добровольцев в рамках этого проекта.

Хосты пользователей могут работать нестабильно, т.к. не являются специализированным вычислительными устройствами с коррекцией аппаратных и программных ошибок. Поэтому осуществляемые на хостах расчеты могут быть некорректными. Для решения этой проблемы каждое **вычислительное задание** создается в нескольких экземплярах (англ. task), которые рассылаются разным добровольцам. Если результаты некоторого количества экземпляров задания совпадают (это количество называется кворум, обычно он равен 2), то эти результаты принимаются как корректные. Еще одной особенностью добровольных вычислений является то, что после получения экземпляра некоторого задания хост может по каким-либо причинам больше никогда не связаться с сервером проекта. Поэтому для каждого экземпляра задания сервер ожидает возвращения результата определенное заданное время, называемое крайним сроком

*Работа была поддержана РФФИ (гранты № 14-07-00403-а и 15-07-07891-а) и Советом по грантам Президента РФ (грант для поддержки ведущих научных школ НШ-5007.2014.9, стипендия СП-1184.2015.5).

выполнения задания (далее используется термин дедлайн, от англ. deadline). Если результат не был получен за этот срок, сервер отсылает еще один экземпляр данного задания на другой хост другого пользователя. На Рис. 1 представлена схема работа сервера проекта при дедлайне равном 10 дням.

Рис. 1. Схема работы с дедлайном в проектах добровольных вычислений

Значение дедлайна оказывает значительное влияние на эффективность проекта. Маленькое значение дедлайна может привести к тому, что хосты добровольцев не будут успевать обрабатывать все присылаемые экземпляры заданий (хосты, как правило, подключены к нескольких проектам, между которыми ресурсы хоста делятся поровну). Если значение дедлайна велико, то при превышении дедлайна по экземпляру задания, в котором находится решение всего эксперимента, будет произведено большое количество лишних вычислений. Кроме этого, большое значение дедлайна может привести к увеличению срока выдачи кредитов добровольцам за вычисления в проекте. Это связано с тем, что кредиты за обработанный экземпляр задания начисляются хосту только в том случае, если второй экземпляр тоже был обработан, и результаты обработки этих экземпляров совпали. Если по второму экземпляру превышается дедлайн, то, как минимум все это время один из добровольцев недополучит кредиты за это задание. Сказанное означает, что проблема выбора значения дедлайна для проектов добровольных вычислений является актуальной. В настоящей статье исследуется возможность повышения эффективности проектов добровольных вычислений при помощи изменения значения дедлайна. Отметим, что в открытой литературе подобные исследования нами найдены не были.

За объект исследования был выбран проект добровольных вычислений SAT@home [2, 3], разработанный совместно ИДСТУ им. В.М. Матросова СО РАН и ИППИ им. А.А. Харкевича РАН для решения трудных задач, эффективно сводимых к задаче о булевой выполнимости (SAT). В настоящий момент в нем насчитывается около 3.5 тысяч активных хостов добровольцев со всего мира, что обеспечивает среднюю производительность около 10 терафлопс. Проект занимается несколькими комбинаторными задачами, для каждой из которых существует свое приложение с различными версиями.

Нами была рассмотрена конфигурация организации и хранения данных об участниках, хостах, заданиях и экземплярах заданий в проектах, основанных на BOINC. Данная информация храниться на сервере проекта в базе данных (БД) MySQL. Для проведения дальнейшего исследования была снята копия с реальной БД проекта SAT@home. Данная копия был запущена на виртуальной машине, дальнейшие исследования по значению дедлайна проводились именно с ней. Первым шагом стал анализ структуры хранения данных в БД. Были определены таблицы, в которых хранятся данные о заданиях и экземплярах заданий. В частности, анализу подверглись данные о моменте отправки экземпляров заданий и моменте возвращения результатов. Для принятия решения о необходимом значении дедлайна было использовано распределение времени, необходимого для выполнения заданий (далее CPU_Time). В итоге была создана процедура для БД MySQL, выполняющая подсчет распределения и выбора дедлайна по заданному проценту корректных возвращенных задач и занесение полученного значения в таблицу, далее называемую исторической таблицей. Для автоматизации данного процесса был создан набор shell-скриптов, предназначенных для запуска на сервере проектов добровольных вычислений. В таблице № 1 представлено распределение количества возвращенных задач от интервала в днях, полученное на основе БД SAT@home.

Таблица № 1. Распределение количества возвращенных задач по дням в проекте SAT@home

Интервал в днях	Количество возвращенных задач	Интервал в днях	Количество возвращенных задач	Интервал в днях	Количество возвращенных задач
0-1	121437	10-11	1286	20-21	5
1-2	29927	11-12	204	21-22	34
2-3	15055	12-13	55	22-23	2
3-4	7728	13-14	92	23-24	1
4-5	3844	14-15	74	24-25	5
5-6	2165	15-16	13	25-26	1
6-7	2067	16-17	10	26-27	0
7-8	1489	17-18	15	27-28	0
8-9	1176	18-19	20	28-29	0
9-10	1021	19-20	23	29-30	0

Анализу подвергались сразу все данные из таблицы с экземплярами заданий, и этим был обусловлен существенный недостаток – данные были очень сильно усреднёнными (за 30 дней, примерно столько времени хранятся результаты в БД). При этом в процессе эксперимента в проекте SAT@home был момент, когда сложность обработки экземпляра заданий уменьшилась примерно в 2 раза. Следовательно, при неизменных остальных параметрах проекта, время, за которое хост должен был вернуть свое задание, должно было сократиться в 2 раза. Следовательно, и дедлайн должен был сократиться в 2 раза. Для учета таких изменений в длительности обработки экземпляров заданий потребовалось дополнительное решение. В БД проекта SAT@home, как и в БД любого BOINC-проекта в таблице экземпляров заданий содержится время обработки этих экземпляров на хосте в секундах. Это значение необходимо для назначения кредитов. Процедура была переписана в соответствии со следующим алго-

ритмом, на вход которому подается процент экземпляров заданий, обработка которых должна успевать закончиться за искомый дедлайн.

Шаг 1. Анализируем распределение экземпляров заданий.

Шаг 2. Вычисляем значение дедлайна в днях, ориентируясь на требуемое значение процентов.

Шаг 3. Подсчитываем среднее время выполнения 1 экземпляра задания за текущие сутки, сравниваем его

с усреднённым значением за предыдущие сутки. Если текущее среднее время больше (меньше) среднего времени за прошлые сутки, то назначаем коэффициент больше (меньше) 1.

Шаг 4. Пересчитываем дедлайн с помощью умножения на соответствующий коэффициент.

Данный алгоритм позволил более динамично вычислять дедлайн. Было принято решение запустить периодический вызов SQL-процедуры, реализующей этот алгоритм, на реальном сервере проекта SAT@home. После интеграции за несколько недель были накоплены первые результаты с реального сервера, по которым стало очевидно, что большинство пользователей успевают выполнить свои задачи значительно быстрее установленных на тот момент 10 дней. Основываясь на полученных данных администратором проекта было принято решение выбирать дедлайн по значению в 98%. По границе в 98% новое значение дедлайна равнялось 8 дням, что на 2 дня меньше, чем действующее на тот момент. Был инициализирован опрос пользователей проекта SAT@home с вопросом: «Считаете ли вы корректным сокращение дедлайна с 10 до 8 дней?» Большинство пользователей поддержало инициативу сократить дедлайн до 8 дней, указывая на то, что их хосты обычно возвращают значения значительно раньше. Было установлено новое значение дедлайна, равное 8 дням. В итоге это привело к уменьшению времени ожидания выполнения заданий, т.е. эффективность проекта увеличилась. Кроме того, как было сказано выше, это привело к ускорению начислению кредитов пользователям проекта, что положительно сказалось на его популярности.

В проектах добровольных вычислений периодически появляются новые расчетные приложения, а также новые версии уже имеющихся расчетных приложений. В идеале для каждой версии каждого приложения нужно рассчитывать дедлайн отдельно, но это может привести к большой нагрузке на сервер проекта. Поэтому было принято решение анализировать данные только за последние сутки, как уже делалось с параметром CPU_Time. Была создана версия процедуры, учитывающая данную методику и позволяющую снимать оптимальное значение дедлайна в секундах, а не днях, тем самым увеличивая точность. При этом выполняется всего 2 прохода по таблице для каждой версии каждого приложения. Из-за изменившегося алгоритма пришлось отказаться от анализа CPU_Time. В ходе тестирования были сняты зависимости распределения по значению 97 процентов. На Рис. 2 изображен результат применения данной процедуры.

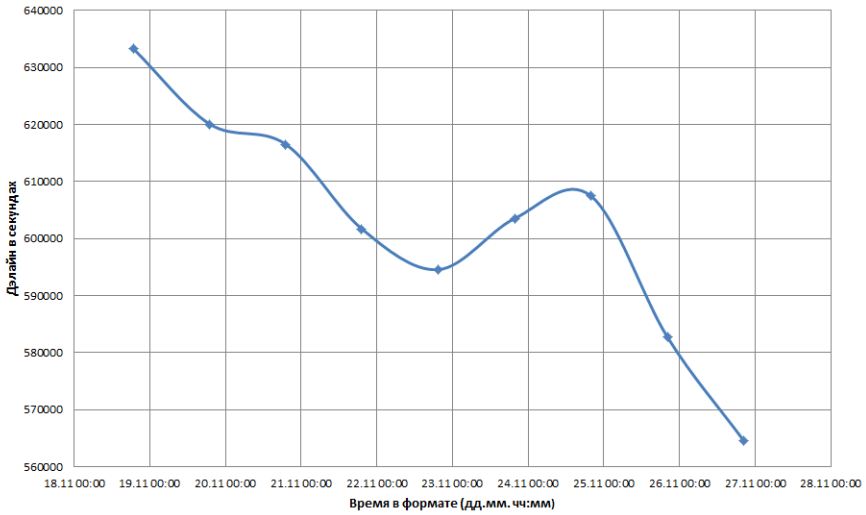

Рис. 2. Дедлайн по распределению с анализом одного дня

По графику видно, что есть всплески, вызванные разными причинами, предположительно это выходные или праздничные дни. Также стало видно еще одну проблему - устанавливаемое значение дедлайна являлось оптимальным на момент запуска процедуры, то есть для уже выполненных заданий, а ведь для получения максимальной эффективности необходимо делать предсказания на сутки вперед. Был использован метод наименьших квадратов, служащий для линейной аппроксимации функций. В экономике полученная данным методом линия называется линией тренда, поэтому для дальнейшего именования было взято именно данное название. Была написана процедура, использующая данные методы и на их основе делающая предсказания на следующий день. Также в неё был перенесен прежний механизм анализа CPU_Time. В итоге тестирования были получены следующие графики (см. Рис. 3).

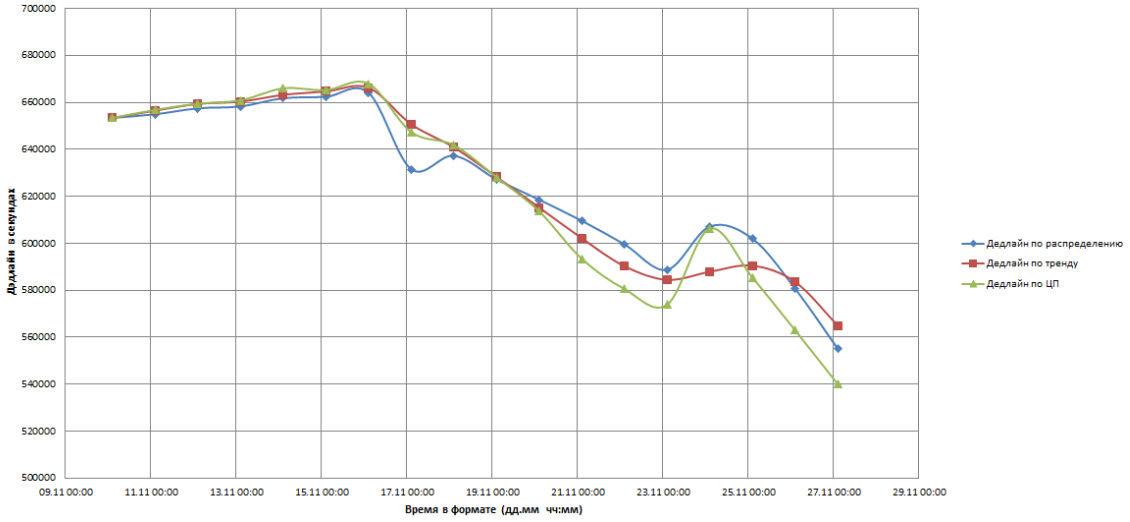

Рис. 3. Графики значения дедлайна в зависимости от метода расчета

В итоге мы произвели сравнение следующих пяти методов определения дедлайна.

1) Статический, значение заданное администратором по умолчанию.

2) Значение, полученное вследствие анализа распределения, точность в днях, с учетом CPU_Time.
3) Значение, полученное по распределению, точность в секундах, без учета CPU_Time.
4) Предсказанное значение методом наименьших квадратов, точность в секундах.
5) Предсказанное значение методом наименьших квадратов, а также дополнительный анализ CPU_Time, точность в секундах.

Результаты сравнения этих методов приведены в таблице № 2. Мы сравнивали их по отклонению от «идеального» значения дедлайна, посчитанного на фактических данных по распределению (данное значение далее называется «Измеренное значение»).

Таблица №2. Таблица сравнения различных методов установки дедлайна

Время	Измеренное значение	1 метод	2 метод	3 метод	4 метод	5 метод	Разность 1	Разность 2	Разность 3	Разность 4	Разность 5
1416263402	637306	864000	604800	631582	650751	647403	226694	32506	5724	13445	10097
1416349802	627233	864000	604800	637306	640901	641799	236767	22433	10073	13668	14566
1416436201	618599	864000	604800	627233	628297	628145	245401	13799	8634	9698	9546
1416522601	609610	864000	604800	618599	615245	613662	254390	4810	8989	5635	4052
1416609001	599587	864000	604800	609610	602158	593408	264413	5213	10023	2571	6179
1416695401	588762	864000	604800	599587	590254	580549	275238	16038	10825	1492	8213
1416781801	607163	864000	604800	588762	584451	573899	256837	2363	18401	22712	33264
1416868201	601980	864000	604800	607163	587896	606270	262020	2820	5183	14084	4290
1416954601	580728	864000	604800	601980	590494	585453	283272	24072	21252	9766	4725
1417041001	555111	864000	604800	580728	583590	562987	308889	49689	25617	28479	7876
В итоге							2613921	173743	124721	121550	102808

Исходя из данных таблицы № 2 видно, что метод № 5 дает наименьшую погрешность в секундах относительно измеренного значения. Именно этот метод будет в ближайшее время использован в проекте SAT@home, что позволит еще больше увеличить его эффективность.

Литература

1. Ивашко, Е.Е. Использование BOINC-грид в вычислительноемких научных исследованиях / Е.Е. Ивашко, Н.Н. Никитина // Вестник Новосибирского государственного университета. Серия: Информационные технологии. – 2013. – Т. 11, № 1. – С. 53-57.

2. Заикин, О.С. Опыт организации добровольных вычислений на примере проектов OPTIMA@home и SAT@home / О.С. Заикин, М.А. Посыпкин, А.А. Семенов, Н.П. Храпов // Вестник Нижегородского университета им. Н.И. Лобачевского. – 2012. – № 5-2. – С. 340-347.

3. Заикин, О.С. Процедуры построения декомпозиционных множеств для распределенного решения SAT-задач в проекте добровольных вычислений SAT@home / О.С. Заикин, А.А. Семенов, М.А. Посыпкин // Управление большими системами. М.: ИПУ РАН. – 2013. – Вып. 43. – С. 138-156.