

(19)



**Евразийское
патентное
ведомство**

(11) **033541**(13) **B9**

**(12) ИСПРАВЛЕННОЕ ОПИСАНИЕ ИЗОБРЕТЕНИЯ К
ЕВРАЗИЙСКОМУ ПАТЕНТУ**

(15) Информация об исправлении

Версия исправления: 1 (W1 B1)**исправления в формуле: п.7**

(48) Дата публикации исправления

2019.11.29, Бюллетень №11'2019

(45) Дата публикации и выдачи патента

2019.10.31

(21) Номер заявки

201691494

(22) Дата подачи заявки

2014.05.16(51) Int. Cl. **G06F 9/50** (2006.01)

(54) СПОСОБ ВЫЧИСЛЕНИЯ

(31) **14305102.7**(32) **2014.01.24**(33) **EP**(43) **2016.12.30**(86) **PCT/IB2014/001073**(87) **WO 2015/110856 2015.07.30**

(71)(73) Заявитель и патентовладелец:

TOTAL SA (FR)

(72) Изобретатель:

**Брюшер Матъё, Мюллер Нилс-
Александр (FR)**

(74) Представитель:

Фелицына С.Б. (RU)

(56) CYRIL BRIQUET ET AL.: "Scheduling data-intensive bags of tasks in P2P grids with bittorrent-enabled data distribution", PROCEEDINGS OF THE SECOND WORKSHOP ON USE OF P2P, GRID AND AGENTS FOR THE DEVELOPMENT OF CONTENT NETWORKS, UPGRADE '07, 26 June

2007 (2007-06-26), page 39, XP055160829, New York, USA DOI: 10.1145/1272980.1272990 ISBN: 978-1-59-593718-6, page 40, left-hand column, line 21 - line 30, page 42, right-hand column, line 57 - page 43, right-hand column, line 35 page 45, left-hand column, line 55 - line 58 figures 1, 3

Elizeu Santos-Neto ET AL.: "Exploiting Replication and Data Reuse to Efficiently Schedule Data-Intensive Applications on Grids" In: "LECTURE NOTES IN COMPUTER SCIENCE", 1 January 2005 (2005-01-01), Springer Berlin Heidelberg, Berlin, Heidelberg, XP055160411, ISSN: 0302-9743 ISBN: 978-3-54-045234-8, vol. 3277, pages 210-232, DOI:10.1007/11407522_12, page 211, line 6 - page 213, last line page 216, line 5 - page 217, line 27

YOUNG CHOON LEE ET AL.: "Practical Scheduling of Bag-of-Tasks Applications on Grids with Dynamic Resilience", IEEE TRANSACTIONS ON COMPUTERS, IEEE, US, vol. 37, no. 6, 1 June 2007 (2007-06-01), pages 815-825, XP011179954, ISSN: 0016-9340 page 818, left-hand column, line 22 - line 46

US-A1-2011060821

(57) Изобретение относится к способу вычисления задач вычислительными клиентами. Способ содержит распределение задач клиентам; определение ассоциативной связи между распределенной задачей и клиентами на основе минимизации размера объединения подмножеств данных, ассоциативно связанных с задачами, распределяемыми вычислительным клиентам по меньшей мере одной группы из множества групп клиентов. В дополнение к этому способ дополнительно содержит этапы, на которых для каждого текущего вычислительного клиента из множества вычислительных клиентов извлекают из узла хранения, относящегося к группе текущего вычислительного клиента, подмножество данных, ассоциативно связанное с задачей, распределенной упомянутому текущему вычислительному клиенту, и вычисляют задачу, распределенную упомянутому текущему вычислительному клиенту, на основе упомянутого извлеченного подмножества данных, ассоциативно связанного с задачей, распределенной упомянутому текущему вычислительному клиенту.

B9**033541****033541****B9**

Уровень техники, предшествующий изобретению

Настоящее изобретение относится к области сеточного вычисления, а более конкретно, к области распределенного вычисления для геофизических алгоритмов, таких как сейсмические алгоритмы.

Подходы, описанные в этом разделе, могли рассматриваться, но не обязательно являются подходами, которые предварительно продумывались или рассматривались. Следовательно, если здесь не указывается иное, то подходы, описанные в этом разделе, не являются прототипом для формулы изобретения в этой заявке и не признаются как являющиеся прототипом в результате включения их в состав этого раздела. Кроме того, все варианты воплощения не обязательно предназначены для решения всех или даже каких-нибудь из проблем, представленных в этом разделе.

Когда должно быть выполнено распределенное вычисление, распределяющий сервер может разделить вычисление на J-заданий. Эти J-заданий представляют собой подмножество вычисления, подлежащего выполнению. Например, если при вычислении требуется индивидуально осуществить обращение 3000 матриц, то каждое задание может представлять обращение единственной матрицы или обращение двух матриц.

Созданные задания могут, таким образом, быть распределены С "клиентам". Почти всегда целое число С не равно целому числу J. Клиенты могут представлять собой физические компьютеры, но могут также представлять собой экземпляры вычисления, созданные на суперкомпьютере. Каждый клиент должен выполнять "задание (задания)", распределенное (распределенные) ему распределяющим сервером. Для того чтобы быть способным выполнить задание, клиенту могут потребоваться внешние данные (данные из внешнего источника). Это данные могут храниться в сетевом запоминающем устройстве или на внешнем сервере хранения.

Когда задание закончено, клиент обращается к распределяющему серверу, и этот сервер назначает этому клиенту новое задание (если оно имеется).

Если размер данных, подлежащих загрузке каждым клиентом на сервере хранения, для выполнения задания является значительным, канал(ы) связи между сервером хранения и клиентами могут быстро стать перегруженными. В дополнение к этому ограничивающей операцией может также быть скорость считывания данных на сервере хранения.

Например, при выполнении интенсивных вычислений в сейсмической области (например, вычисления Стека общей глубинной точки), считывание данных может представлять до 80% всего времени вычисления.

Соответственно существует, таким образом, потребность в архитектуре вычисления, которая может повысить скорость распределенного вычисления, особенно в случае, когда ограничивающим является доступ к данным для вычисления.

Сущность изобретения

Изобретение относится к способу для вычисления множества вычислительных задач, причем каждая вычислительная задача ассоциативно связана с некоторым подмножеством данных, подмножество данных, ассоциативно связанное с текущей вычислительной задачей, привлекается во время вычисления упомянутой текущей задачи.

Множество вычислительных клиентов разделено на множество клиентских групп, каждая текущая группа в упомянутом множестве групп клиентов содержит некоторый текущий узел хранения.

Способ содержит этапы, на которых

a) для каждого текущего клиента в этом множестве вычислительных клиентов определяют ассоциативную связь между задачей в множестве вычислительных задач и упомянутым текущим клиентом;

b) распределяют упомянутое множество вычислительных задач в соответствии с этой определенной ассоциативной связью;

c) для каждой текущей группы в упомянутом множестве групп клиентов извлекают посредством узла хранения, относящегося к упомянутой текущей группе, объединение подмножеств данных, ассоциативно связанных с задачами, распределенными вычислительным клиентам упомянутой текущей группы;

d) для каждого текущего вычислительного клиента во множестве вычислительных клиентов:

d1) извлекают из узла хранения, относящегося к группе текущего вычислительного клиента, подмножество данных, ассоциативно связанное с задачей, распределенной упомянутому текущему вычислительному клиенту, и

d2) вычисляют задачу, распределенную упомянутому текущему вычислительному клиенту, основываясь на упомянутом извлеченном подмножестве данных, ассоциативно связанном с задачей, распределенной упомянутому текущему вычислительному клиенту;

В дополнение к этому определение ассоциативной связи между задачей и клиентами основано на минимизации размера объединения подмножеств данных, ассоциативно связанных с задачами, распределенными вычислительным клиентам по меньшей мере одной группы в множестве групп клиентов.

Вычислительная задача представляет собой индивидуальную проблему/вопрос/вычисление, которое может быть решено одним компьютером или вычислительными блоками, например, в несколько минут или часов.

Каждая задача ассоциативно связана с подмножеством данных, которое необходимо локально для

вычисления этой задачи. Почти всегда эти подмножества данных хранятся на внешнем сервере хранения (например, SAN-сервере (сервере сети хранения данных)), специально предназначенном для этого хранения.

Перед вычислением клиент может запросить доступ к своему подмножеству данных. Узел хранения также известен как "узел" или "I/O-узел (сокращение от "узел ввода/вывода)".

На этапе а), ассоциативная связь может представлять собой "взаимно однозначную" ассоциативную связь, но ассоциативная связь может также одновременно ассоциативно связывать несколько задач с единственным клиентом.

Распределение задач на этапе b) может содержать этап, на котором отправляют сообщения, которые включают в себя идентификационную информацию ассоциативно связываемой задачи (ассоциативно связываемых задач) (и, если требуется, идентификационную информацию ассоциативно связываемого подмножества данных, необходимых для вычисления), отправляемую клиентам.

Узел ввода/вывода, основываясь на этих сообщениях, отправленных клиентам его группы, может извлекать с сервера хранения объединение подмножеств данных. Это извлечение может быть выполнено за один этап (то есть вычисляют идентификационную информацию этого объединения и отправляют его на сервер хранения, для того, чтобы извлечь это объединение одним сообщением) или за множество этапов (например, отправляют некоторое первое сообщение для извлечения первого подмножества, затем отправляют второе сообщение для того, чтобы извлечь данные второго подмножества, которое еще не было отправлено при первом извлечении, и так далее).

Для улучшения размера объединения, извлекаемого узлом ввода/вывода, важным является определение задач, распределяемых клиентам одной той же группы. Таким образом, имеется возможность определить задачи таким образом, чтобы минимизировать размер объединения подмножества данных, ассоциативно связанных с задачами этой группы. Эта минимизация может быть выполнена множеством способов:

- создают сотни случайных распределений задач для каждого клиента группы и определяют то случайное распределение, которое минимизирует размер объединения;

- если для задач может быть определено расстояние (например географическое расстояние центральных точек в модели), то выбирают случайным образом (или основываясь на некотором предварительно заданном месте расположения в модели, например, в верхнем левом угле) некоторую первую задачу, выбирают некоторую вторую задачу, соответствующую самым близко расположенным другим задачам, выбирают некоторую третью задачу, для которой минимизирована сумма расстояний между третьей задачей - второй задачей и между третьей задачей - первой задачей, и так далее;

- определяют случайным образом (или основываясь на некотором предварительно заданном месте расположения в модели, например, в верхнем левом угле) некоторую первую задачу, используют некоторый алгоритм модельной "закалки" для того, чтобы определить самые близкие задачи, подлежащие ассоциативному связыванию с клиентами этой же самой группы,

- и так далее.

В дополнение к этому минимизация размера объединения может быть выполнена посредством максимизации размера пересечения между этими подмножествами (размер зоны, которая является пересечением пересечениями дважды, может быть учтена в размере пересечения дважды).

В некотором возможном варианте воплощения изобретения минимизация может быть выполнена для всех групп.

Например, определение ассоциативной связи между задачами и клиентами может быть основано на минимизации суммы размера объединения подмножеств данных, ассоциативно связанных с задачами, распределенными вычислительным клиентам каждой группы в этом множестве групп клиентов.

Следовательно, оптимизация может быть выполнена для всех групп, и можно искать глобальную оптимизацию.

Если определено расстояние между каждой задачей, то определение ассоциативной связи может содержать, для текущей группы во множестве групп клиентов, следующие этапы, на которых:

- выбирают некоторую первую задачу;

- выбирают вторые задачи, каждое расстояние между каждой второй задачей и первой задачей меньше, чем некоторое предварительно заданное расстояние, при этом предварительно заданное расстояние увеличивают, если количество выбранных вторых задач меньше, чем количество задач, которые могут быть распределены клиентам текущей группы.

Иначе говоря, задачи, распределенные группе, могут быть в одной и той же области (в некотором "кругу" вокруг первой задачи) и, таким образом, могут быть очень близки друг к другу. Если количество задач, имеющихся для этого предварительно заданного расстояния, является слишком низким и не соответствует количеству клиентов в группе (или количеству задач, которые могут быть распределены клиентам группы), то это предварительно заданное расстояние может быть увеличено. В некотором возможном варианте воплощения изобретения это предварительно заданное расстояние может начинаться с 0.

Расстояние представляет собой математическое расстояние, и, таким образом, "круг" может быть представлен как квадрат или прямоугольник.

Полезно, чтобы способ мог дополнительно содержать этапы, на которых, когда текущий клиент заканчивает вычисление предшествующей задачи, распределенной упомянутому текущему клиенту:

определяют новую ассоциативную связь между некоторой задачей во множестве вычислительных задач и упомянутым текущим клиентом;

распределяют упомянутую новую задачу текущему клиенту.

Определение этой новой ассоциативной связи может быть основано на минимизации размера объединения подмножества данных, ассоциативно связанного с предшествующей задачей, и подмножества данных, ассоциативно связанного с новой задачей.

Следовательно, может быть использован локальный кэш в вычислительном клиенте, и может быть ограничен запрос узлу ввода/вывода.

В дополнение к этому способ может дополнительно содержать этапы, на которых, когда текущий клиент заканчивает вычисление предшествующей задачи, распределенной упомянутому текущему клиенту:

определяют новую ассоциативную связь между задачей во множестве вычислительных задач и упомянутым текущим клиентом;

распределяют упомянутую новую задачу текущему клиенту.

Определение этой новой ассоциативной связи может быть основано на минимизации размера объединения подмножеств данных, ассоциативно связанных с задачами, вычисляемыми вычислительными клиентами этой группы, и подмножества данных, ассоциативно связанного с новой задачей.

Следовательно, может быть использован локальный кэш в узле ввода/вывода, и может быть ограничен запрос серверу хранения.

В некотором возможном варианте воплощения изобретения на этапе а) по меньшей мере с одним текущим клиентом может быть ассоциативно связано множество вычислительных задач.

В таком случае клиент может иметь множество задач для вычисления и может избежать отправки сообщения на распределяющий сервер всякий раз, когда он заканчивает вычисление каждой задачи.

Это множество вычислительных задач может быть распределено клиенту на этапе b).

На самом деле, доступ на сервер хранения содержит полезную нагрузку, и доступ к очень малому количеству данных подобен доступу к большему количеству данных. Таким образом, выгодно группировать малые запросы и снабжать клиента множеством малых задач, подлежащих вычислению.

В некотором возможном варианте воплощения изобретения этап d2) может содержать этапы, на которых

определяют порядок для вычисления распределенных задач таким образом, чтобы максимизировать пересечение двух подмножеств данных, ассоциативно связанных с двумя последовательно вычисляемыми задачами в этом определенном порядке задач,

вычисляют распределенные задачи в соответствии с этим определенным порядком задач.

Этот порядок может повысить использование локального кэша в вычислительных клиентах.

Этот порядок может быть вычислен локально или распределяющим сервером.

Этап d2) может содержать этапы, на которых

определяют порядок для вычисления распределенных задач, основываясь на кривой Пеано,

вычисляют распределенные задачи в соответствии с этим определенным порядком задач.

Кривая Пеано или кривая Гильберта-Пеано делает возможной максимизацию пересечения двух подмножеств данных, ассоциативно связанных с двумя последовательно вычисляемыми задачами в этом определенном порядке задач.

Второй аспект относится к компьютерному программному продукту, содержащему машиночитаемый носитель информации, имеющий на себе компьютерную программу, содержащую команды программы. Эта компьютерная программа способна быть загружена в блок обработки данных и приспособлена для того, чтобы, когда эта компьютерная программа выполняется блоком обработки данных, представлять блок обработки данных осуществлять способ, описанный выше.

Другие признаки и преимущества раскрываемых здесь способа и аппарата станут очевидны из нижеприведенного описания неограничивающих вариантов воплощения изобретения при обращении к прилагаемым чертежам.

Краткое описание чертежей

Настоящее изобретение проиллюстрировано в порядке примера, а не в порядке ограничения, на фигурах прилагаемых чертежей, на которых одинаковые ссылочные позиции относятся к аналогичным элементам и на которых

фиг. 1 представляет собой графическое представление возможной схемы вычисления;

фиг. 2 является представлением подмножеств данных, ассоциативно связанных с двумя различными задачами в модели;

фиг. 3 является представлением задач, перегруппированных в три узла ввода/вывода;

на фиг. 4а описано определение новой задачи, подлежащей ассоциативному связыванию с клиентом узла ввода/вывода;

на фиг. 4b описано определение новой задачи, подлежащей ассоциативному связыванию с клиен-

том, на основе кривой Пеано;

фиг. 5 представляет собой блок-схему алгоритма, на которой описан некоторый возможный вариант воплощения настоящего изобретения;

фиг. 6 представляет собой некоторый возможный вариант воплощения для устройства, которое делает возможным настоящее изобретение.

Осуществление изобретения

Фиг. 1 представляет собой графическое представление возможной схемы вычисления.

При вычислении распараллеливаемой проблемы имеется возможность разделить эту проблему на множество отдельных задач (или "вычислительных задач") и вычислять эти задачи посредством множества, сотен тысяч вычислительных блоков (или "вычислительных клиентов" (102-109)).

Следовательно, центральный распределяющий сервер (101) управляет вычислением и централизует результаты. Этот распределяющий сервер (101) разбивает проблему на множество задач, подлежащих решению.

Каждая вычислительная задача ассоциативно связана с подмножеством данных, которое привлекается во время вычисления этой задачи, например для вычисления локального среднего значения для пиксельного изображения, присоединяемое подмножество данных представляет собой значения пикселей на некотором заданном расстоянии от текущего пикселя. Локальное среднее значение для этого текущего пикселя представляет собой, в таком случае, средние значений пикселей на этом заданном расстоянии.

Одна или более задач могут быть распределены всем имеющимся клиентам (102-109). Почти всегда они не имеют в своей локальной памяти (например, оперативном запоминающем устройстве (RAM), флэш-памяти или на жестком магнитном диске) подмножество данных, ассоциативно связанное с распределенными задачами, и они запрашивают доступ к этим данным, отправляя сообщение на сервер (113) хранения, который обрабатывает статические данные для каждой задачи. Таким образом, если имеются тысячи вычислительных клиентов, то сервер хранения должен быть в состоянии принимать тысячи запросов и отправлять назад клиентам тысячи сообщений, содержащих запрашиваемое подмножество данных.

Вследствие ограничения доступа к памяти (памяти, содержащей почти всегда для ограничения издержек стандартные платы магнитной памяти) обрабатывать тысячи запросов может быть трудно и долго.

Следовательно, может быть реализован один (или больше) промежуточный уровень: уровень узлов ввода/вывода. Узел ввода/вывода может представлять собой специализированный вычислительный блок, но может также представлять собой вычислительные клиенты (102-109), которые также исполняют роль ввода/вывода.

Узлы (110-112) ввода-вывода представляют собой устройства, которые спроектированы таким образом, чтобы служить интерфейсами с сервером хранения и ограничивать доступы к этому серверу хранения.

Все вычислительные клиенты могут быть ассоциативно связаны с узлом ввода/вывода, например вычислительные клиенты (102-104) ассоциативно связаны с узлом (220) ввода/вывода, вычислительные клиенты (105-106) ассоциативно связаны с узлом (111) ввода-вывода, а вычислительные клиенты (107-109) ассоциативно связаны с узлом (112) ввода/вывода.

Таким образом, сообщения, запрашивающие доступ к подмножеству данных, сначала отправляются на узел ввода/вывода, ассоциативно связанный с этим клиентом. Узел ввода/вывода способен, таким образом, определить объединение всех запрашиваемых подмножеств и отправить на сервер хранения только один запрос. В дополнение к этому, если запрашиваемые подмножества данных сходны (например, если они содержат значительное пересечение), то объединение запрашиваемых подмножеств данных может быть значительным образом уменьшено.

В дополнение к этому узел ввода/вывода может использоваться в качестве кэш-узла: при реализации кэша (например, кольцевого кэша) узел ввода/вывода может избежать обращения к серверу хранения, если предшествующий запрос аналогичен, и могут быть повторно использованы ранее присланные данные.

В таком случае выгодно определять задачи, подлежащие ассоциативному связыванию с различными клиентами (102-109) таким образом, чтобы максимизировать пересечение для подмножеств данных в группе одного и того же узла (то есть 102-104, 105-106, 107-109) (или минимизировать объединение запрашиваемых подмножеств) и/или максимизировать повторное использование данных для двух последовательно вычисляемых задач.

Фиг. 2 является представлением подмножеств данных, ассоциативно связанных с двумя различными вычислительными задачами в модели (200).

Как подробно говорилось в отношении фиг. 1, задачи, распределяемые клиентам одной и той же группы (то есть клиентам, ассоциативно связанным с одним и тем же узлом), могли бы быть определены разумным образом.

Если группа содержит двух клиентов, то клиентам этой группы могут быть распределены две зада-

чи.

Например, для обработки сейсмических данных обычно, что для вычисления вертикального столбца пикселей по z и для точки (201) в плоскости (x, y) , к вычислению должны быть привлечены все пиксели, которые имеют координаты (x, y) в пределах расстояния R от точки (201) (круг 202).

Аналогичным образом, для вычисления вертикального столбца пикселей по z и для точки (203) в плоскости (x, y) , к вычислению должны быть привлечены все пиксели, которые имеют координаты (x, y) в пределах расстояния R от точки (203) (круг 204)).

Таким образом, заштрихованная зона (205) может быть использована задачами, связанными с точками (201) и (203).

Следовательно, для задач, ассоциативно связанных с клиентами одной и той же группы, выгодно, чтобы размер объединения подмножеств данных, ассоциативно связанных с этими задачами, был минимизирован (то есть заштрихованная зона была максимизирована). Таким образом, количество данных, запрашиваемых узлом ввода/вывода, относящимся к этой группе, может быть уменьшено до своего минимума.

Другими словами, выгодно распределять задачи клиентам одной и той же группы таким образом, чтобы минимизировать сумму расстояний d между центральными точками (201) и (203).

Фиг. 3 является представлением задач, перегруппированных в три узла ввода/вывода.

В этом представлении (300) три задачи распределены первой группе клиентов, и подмножества данных, ассоциативно связанные с этими задачами, представлены кругами (301), (302), (303). В дополнение к этому две задачи распределены второй группе клиентов, и подмножества данных, ассоциативно связанные с этими задачами, представлены кругами (304) и (305). Наконец, три задачи распределены третьей группе клиентов, и подмножества данных, ассоциативно связанные с этими задачами, представлены кругами (306), (307) и (308).

Как было указано ранее, задачи, распределенные клиентам одной и той же группы, выбираются таким образом, чтобы пересечение подмножеств данных, ассоциативно связанных с задачами, было максимальным.

Как только распределенные задачи вычислены, имеется возможность распределить клиентам упомянутых групп новые задачи, основываясь на последних вычисленных задачах. Таким образом, данные, уже загруженные узлом ввода/вывода, относящимся к группе, и/или клиентами могут быть использованы повторно: этот процесс ограничивает количество данных, подлежащих загрузке.

Например, задачи, распределенные клиентам первой группы, могут быть выбраны таким образом, чтобы круги (301), (302) и (303) остались сгруппированными и немного переместились в направлении $R1$. Аналогичным образом, задачи, распределенные клиентам второй группы, могут быть выбраны таким образом, чтобы круги (304) и (305) остались сгруппированными и немного переместились в направлении $R2$. Наконец, задачи, распределенные клиентам третьей группы, могут быть выбраны таким образом, чтобы круги (306), (307) и (308) остались сгруппированными и немного переместились в направлении $R3$.

На фиг. 4а описано определение новой задачи, подлежащей ассоциативному связыванию с клиентом узла ввода/вывода.

Для этой фигуры предположим, что круги (401), (402) и (403) представляют подмножество данных, ассоциативно связанное с тремя задачами, ассоциативно связанное с тремя клиентами узла ввода/вывода. В дополнение к этому предположим, что задачи, ассоциативно связанные с подмножеством (401) данных, были только что вычислены и что связанный с ними клиент больше не имеет задачи, подлежащей вычислению.

Следовательно, распределяющий сервер может определить новую задачу для распределения этому клиенту. Например, новая задача, распределяемая клиенту, может быть ассоциативно связана с подмножеством, представленным кругом (404), или кругом (405), или кругом (406).

Определение новой задачи может быть основано на:

(i) минимизации размера объединения подмножеств данных, ассоциативно связанных с задачами, вычисляемыми вычислительными клиентами группы (то есть, кругов (401), (402) и (403)) и подмножеств данных, ассоциативно связанных с новой задачей, (например, круга (404), или (405), или (406)), и/или

(ii) минимизации размера объединения подмножества данных, ассоциативно связанного с предшествующей задачей, (то есть круга (401)) и подмножества данных, ассоциативно связанного с новой задачей (например, круга (404), или (405), или (406)).

В случае (i) круг (405) и круг (404), как кажется, эквивалентны, поскольку заштрихованная зона (411) и (412) (которая представляет соответственно пересечение между объединением подмножеств (401), (402) и (403) данных и подмножества (404), соответственно (405) данных). Задача, ассоциативно связанная с кругом (406), кажется неподходящей, поскольку пересечение (413) круга (406) с объединением подмножества (401), (402) и (403) очень мало (и размер объединения упомянутых подмножеств значителен).

В случае (ii) только круг (404) кажется подходящим, поскольку он представляет собой единственный круг, который пересекает круг (401) (смотри заштрихованную зону (414)). Для того чтобы учесть

как (i), так и (ii), можно взвесить полученный размер и минимизировать этот взвешенный размер.

Минимизация размера объединения может также представлять собой максимизацию размера пересечения.

На фиг. 4b описано определение новой задачи, подлежащей ассоциативному связыванию с клиентом, на основе кривой Пеано.

В некотором возможном варианте воплощения изобретения множество задач может быть распределено клиенту даже прежде, чем он вычислит одну задачу. Таким образом, имеется возможность ограничить количество взаимодействий с распределяющим сервером.

Например, все крестики в прямоугольнике (451) могут быть распределены первому клиенту группы, а крестики в прямоугольнике (452) могут быть распределены второму клиенту группы.

Для того чтобы вычислять распределенные задачи, клиент или распределяющий сервер могут воздействовать на порядок для распределенных задач, этот порядок определяет порядок вычисления упомянутых задач клиентом.

Этот порядок может быть определен таким образом, чтобы максимизировать пересечение двух подмножеств данных, ассоциативно связанных с двумя последовательно вычисляемыми задачами, в этом определяемом порядке.

Благодаря этому имеется возможность ограничить количество обмена данными между клиентом и узлом ввода/вывода, причем клиент оперирует небольшим буфером памяти (например, кольцевым буфером) для хранения самые последних подмножеств данных.

Например, этот порядок может быть определен благодаря кривой Пеано или кривой Гильберта-Пеано (453) или (454).

Фиг. 5 представляет собой блок-схему алгоритма, на которой описан некоторый возможный вариант воплощения настоящего изобретения. Часть этой блок-схемы алгоритма может представить этапы примера компьютерной программы, которая может быть исполнена устройством, показанным на фиг. 6.

Когда должно быть вычислено множество задач (501a, 501b, 501c), можно (на этапе 502) определить множество клиентов, подлежащих использованию для этих вычислений. Отметим, что список клиентов, который может быть использован, может динамически изменяться, поскольку могут развиваться компьютерные ресурсы (например, другие приоритетные вычисления могут зарезервировать статически компьютерные ресурсы).

Основываясь на этом списке имеющихся вычислительных клиентов, можно ассоциативно связать их с множеством узлов ввода/вывода или "узлом хранения" (этап 503): клиенты, ассоциативно связанные с одним и тем же узлом ввода/вывода, образуют группу узла, или "группа клиентов".

Для каждого узла (цикл 504) и для каждого текущего клиента упомянутого узла можно определить ассоциативную связь между задачей и упомянутым текущим клиентом (этап 505), основываясь на минимизации размера объединения подмножеств данных, ассоциативно связанных с задачами, распределяемыми вычислительным клиентам упомянутого узла.

Затем распределяющий сервер может распределить (на этапе 506) упомянутые вычислительные задачи в соответствии с этой определенной ассоциативной связью.

Таким образом, каждый клиент может извлечь подмножество данных, ассоциативно связанное с распределенной задачей (этапы: 507_1, 507_2, 507_3, 507_4 и так далее), отправляя сообщение на узел ввода/вывода (как это подробно описано выше).

Узел ввода/вывода может затем извлечь объединение подмножеств данных, связанных с задачами, распределенными вычислительным клиентам упомянутой текущей группы.

Для каждого текущего вычислительного клиента текущий клиент может, таким образом, вычислить задачу, распределенную текущему клиенту, основываясь на упомянутом извлеченном подмножестве данных (этапы 508_2 и так далее). После вычисления и если не должны вычисляться никакие другие задачи, клиент может сообщить распределяющему серверу результаты (этапы 509_2 и так далее). После этого распределяющий сервер (на этапе 510_2 и так далее) может определить новую задачу (или новый набор задач), подлежащую вычислению этим клиентом, как это подробно описано выше.

Описанный здесь процесс может повторяться, пока еще имеется некоторое количество задач, подлежащих вычислению.

Фиг. 6 представляет собой некоторый возможный вариант воплощения для устройства, которое делает возможным настоящее изобретение. Например, это устройство может представлять собой распределяющий сервер (101) или вычислительный клиент (102-109) и/или узел (110-112) ввода/вывода.

В этом варианте воплощения изобретения устройство (600) содержит компьютер, этот компьютер содержит память (605) для хранения команд программы, загружаемых в некоторую схему в схему и приспособленных для того, чтобы, когда эти команды программы исполняются схемой (604), заставляя схему (604) выполнять этапы по настоящему изобретению (будь то этапы, исполняемые распределяющим сервером, клиентами или узлами).

Память (605) может также хранить данные и полезную информацию для выполнения этапов по настоящему изобретению, которые были описаны выше.

Схема (604) может представлять собой, например

процессор или блок обработки данных приспособлен для того, чтобы интерпретировать команды на машинном языке, процессор или блок обработки данных могут содержать память, содержащую команды, могут быть связаны с ней или могут быть прикреплены к ней, или

соединение процессора/блока обработки данных и памяти, причем процессор или блок обработки данных приспособлен для того, чтобы интерпретировать команды на машинном языке, память содержит упомянутые команды, или

электронную плату, на которой этапы по изобретению описаны внутри кремния, или программируемый кристалл интегральной схемы, такой как кристалл FPGA (сокращение от "Вентильная матрица с эксплуатационным программированием").

Этот компьютер содержит интерфейс (603) ввода для приема

результатов распределенных задач (для распределяющего сервера);

распределенных задач и ассоциативно связанных с ними данных (для клиентов);

запросов на подмножества данных и подмножеств данных (для узлов ввода/вывода);

и интерфейс (406) вывода для предоставления

распределенных задач (для распределяющего сервера);

результатов распределенных задач и запросов на ассоциативно связанное с ними подмножество данных (для клиентов);

подмножеств данных (для узлов ввода/вывода).

Для упрощения взаимодействия с компьютером могут быть предусмотрены экран (601) и клавиатура (602) и соединены с компьютерной схемой (604).

Выражения, такие как "содержать", "включать в себя", "включать в свой состав", "представлять собой" и "иметь", должны при интерпретации описания и связанной с ним формулы изобретения толковаться неисключительным образом, а именно толковаться таким образом, чтобы позволять также присутствие других элементов или компонентов, которые явно не определены. Ссылка на единственное число должна также толковаться как ссылка на множественное число и наоборот.

Специалист в данной области техники легко поймет, что различные параметры, раскрытые в описании, могут быть изменены, и что различные раскрытые варианты воплощения изобретения могут быть объединены, не выходя при этом за рамки объема изобретения.

ФОРМУЛА ИЗОБРЕТЕНИЯ

1. Способ организации вычисления множества вычислительных задач, характеризующийся тем, что каждая вычислительная задача ассоциативно связана с подмножеством данных, причем подмножество данных, ассоциативно связанное с текущей вычислительной задачей, привлекается во время вычисления упомянутой текущей задачи, множество аппаратных вычислительных клиентов разделено на множество групп клиентов, каждая текущая группа из упомянутого множества групп клиентов содержит текущий узел хранения;

при этом упомянутый способ содержит этапы, на которых:

a) для каждого текущего клиента из множества вычислительных клиентов определяют ассоциативную связь между задачей из множества вычислительных задач и упомянутым текущим клиентом;

b) распределяют упомянутое множество вычислительных задач в соответствии с определенной ассоциативной связью;

c) для каждой текущей группы из упомянутого множества групп клиентов извлекают посредством узла хранения, относящегося к упомянутой текущей группе, объединение подмножеств данных, ассоциативно связанных с задачами, распределенными вычислительным клиентам упомянутой текущей группы;

d) для каждого текущего вычислительного клиента из множества вычислительных клиентов:

d1) извлекают из узла хранения, относящегося к группе текущего вычислительного клиента, подмножество данных, ассоциативно связанное с задачей, распределенной упомянутому текущему вычислительному клиенту, и

d2) вычисляют задачу, распределенную упомянутому текущему вычислительному клиенту, на основании упомянутого извлеченного подмножества данных, ассоциативно связанного с задачей, распределенной упомянутому текущему вычислительному клиенту;

при этом определение ассоциативной связи между задачей и клиентами основано на минимизации размера объединения подмножеств данных, ассоциативно связанных с задачами, распределенными вычислительным клиентам по меньшей мере одной группы из множества групп клиентов.

2. Способ по п.1, в котором определение ассоциативной связи между задачей и клиентами основано на минимизации суммарного размера объединения подмножеств данных, ассоциативно связанных с задачами, распределенными вычислительным клиентам каждой группы из множества групп клиентов.

3. Способ по п.1 или 2, в котором определяют расстояние между каждой задачей, при этом на этапе определения ассоциативной связи для текущей группы из множества групп клиентов

выбирают первую задачу;

выбирают вторые задачи, причем каждое расстояние между каждой второй задачей и первой зада-

чей меньше заданного расстояния, причем заданное расстояние увеличивают, если количество выбранных вторых задач меньше количества задач, которые могут быть распределены клиентам текущей группы.

4. Способ по любому из пп.1-3, дополнительно содержащий этапы, на которых, когда текущий клиент заканчивает вычисление предшествующей задачи, распределенной упомянутому текущему клиенту, определяют новую ассоциативную связь между новой задачей из множества вычислительных задач и упомянутым текущим клиентом;

распределяют упомянутую новую задачу текущему клиенту;

при этом определение новой ассоциативной связи основано на минимизации размера объединения подмножества данных, ассоциативно связанного с предшествующей задачей, и подмножества данных, ассоциативно связанного с новой задачей.

5. Способ по любому из пп.1-4, дополнительно содержащий этапы, на которых, когда текущий клиент заканчивает вычисление предшествующей задачи, распределенной упомянутому текущему клиенту, определяют новую ассоциативную связь между задачей из множества вычислительных задач и упомянутым текущим клиентом;

распределяют упомянутую новую задачу текущему клиенту;

при этом определение новой ассоциативной связи основано на минимизации размера объединения подмножеств данных, ассоциативно связанных с задачами, вычисляемыми вычислительными клиентами группы, и подмножества данных, ассоциативно связанного с новой задачей.

6. Способ по любому из пп.1-5, в котором на этапе а) ассоциативно связывают множество вычислительных задач по меньшей мере с одним текущим клиентом.

7. Способ по п.6, в котором на этапе б) распределяют множество вычислительных задач клиенту из множества вычислительных клиентов.

8. Способ по п.7, в котором на этапе d2)

определяют порядок вычисления распределенных задач так, чтобы максимизировать пересечение двух подмножеств данных, ассоциативно связанных с двумя последовательно вычисляемыми задачами в определенном порядке задач,

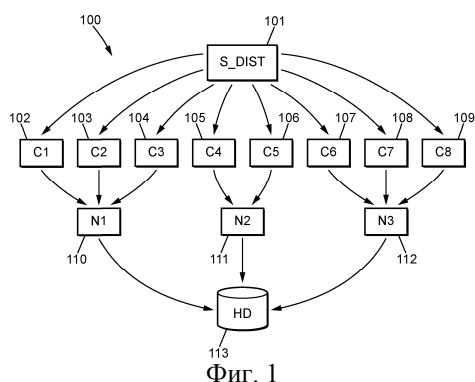
вычисляют распределенные задачи в соответствии с определенным порядком задач.

9. Способ по п.7, в котором на этапе d2)

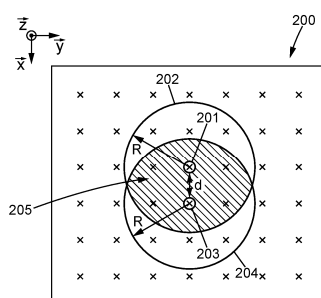
определяют порядок вычисления распределенных задач на основе кривой Пеано,

вычисляют распределенные задачи в соответствии с определенным порядком задач.

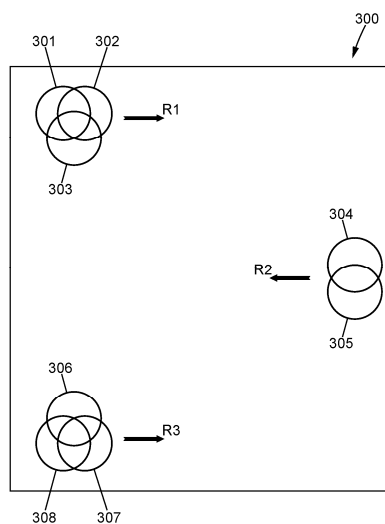
10. Энергонезависимый машиночитаемый носитель информации, имеющий хранящуюся на нем компьютерную программу, содержащую команды программы, причем компьютерная программа является загружаемой в блок обработки данных и выполнена с возможностью вызывать выполнение блоком обработки данных этапов способа по любому из пп.1-9.



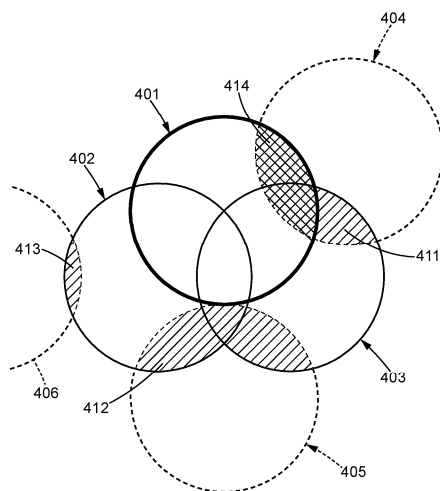
Фиг. 1



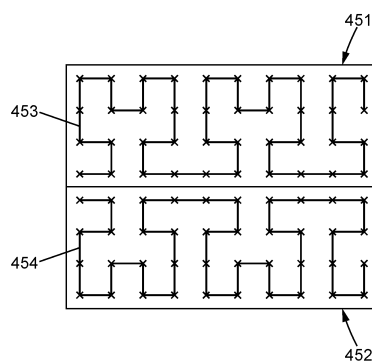
Фиг. 2



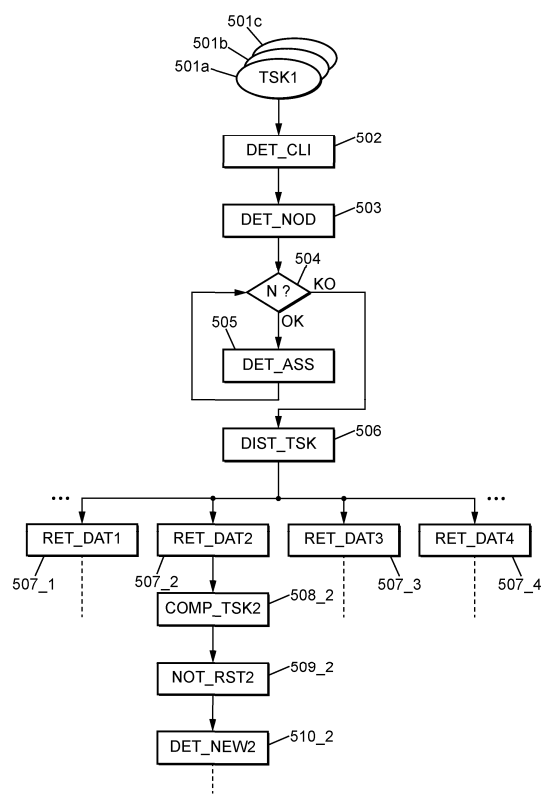
Фиг. 3



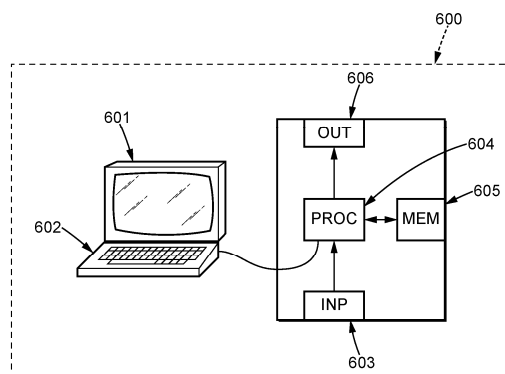
Фиг. 4а



Фиг. 4б



Фиг. 5



Фиг. 6



Евразийская патентная организация, ЕАПВ

Россия, 109012, Москва, Малый Черкасский пер., 2