

УДК 004

Научные вычисления в облаках

09, сентябрь 2012

Романова А.О.

*Научный руководитель д. ф.-м. наук, профессор, Карпенко А.П.
МГТУ имени Н.Э.Баумана, Москва, Россия*

МГТУ им. Н.Э. Баумана
anastasia.o.romanova@gmail.com

Введение

Облачные или рассеянные вычисления представляют собой технологию обработки данных, в которой компьютерные ресурсы и мощности предоставляются пользователям как Интернет-сервисы [1]. Пользователь не должен заботиться об инфраструктуре, операционной системе и собственно программном обеспечении, к которому предоставляется доступ. Абстракция ресурсов происходит на самом высоком уровне: оборудование объединяется с виртуализированными приложениями, позволяя ИТ-администратору предоставлять информационные технологии в качестве услуги для потребителей. Таким образом, можно создать каталог ИТ-услуг, и конечный пользователь будет выбирать то, что ему нужно, исходя из его запросов на аппаратное и программное обеспечение.

Главными преимуществами в использовании облаков в вычислениях являются:

- быстрое изменение конфигурации выделенных ресурсов;
- плата только за использование ресурсов.

По сравнению с персональным компьютером, вычислительная мощность, доступная пользователю облаков, практически ограничена лишь размером облака, то есть общим числом серверов. Пользователи могут запускать более сложные задачи, с большим количеством необходимой памяти и места для хранения данных, тогда, когда это необходимо. Иными словами, пользователи могут при желании легко и дешево поработать с суперкомпьютером без каких-либо фактических приобретений.

Следует отметить так же некоторые недостатки такой модели:

- необходимость постоянного соединения с сетью Интернет;
- риск кражи или потери данных.

В настоящее время все программные приложения, обеспечивающие вычисления в облаках, по сути, реализуют новую парадигму взаимодействия между ИТ-инфраструктурой и бизнесом и не поддерживают научные вычисления. В данной работе исследуется возможность использования облачной модели в научных вычислениях.

Постановка и методы решения задачи

Облака предоставляют широкий спектр возможностей. Но возникает вопрос – какую конфигурацию серверов и какое их число следует заказывать для решения стоящей перед пользователем задачи. Современное облачное программное обеспечение не решает данной проблемы, и решение ее в рамках облачной парадигмы в настоящее время не предполагается (исходя из ориентации этой парадигмы на бизнес и бизнес-решения).

В работе рассматриваются две взаимосвязанные задачи:

- исходя из спецификации задачи пользователя, разработать методы, алгоритмы и соответствующее программное обеспечение для выбора оптимального числа и параметров виртуальных серверов (архитектуры облака);
- разработать методы, алгоритмы и программное обеспечение для выбора оптимальных параметров алгоритма решаемой задачи для фиксированной архитектуры облака.

Решение указанных выше задач требует разработки следующих моделей и методов:

- модель архитектуры облака,
- модель алгоритма, с помощью которого решается задача пользователя,
- метод оптимального отображения алгоритма на архитектуру облака.

Модель облака. В настоящее время нет оснований полагать, что в облаках будет предоставляться доступ к специализированным вычислителям, например, типа графических процессорных устройств. Поэтому мы исходим из того, что все возможные виртуальные серверы являются универсальными (концепция гомогенного облака) и каждый из них может решить любую из задач пользователя.

Физические сервера облака находятся, как правило, в центрах обработки данных (ЦОД), связаны между собой «широкими» каналами связи и число их не слишком велико. На этом основании мы полагаем, что все виртуальные серверы облака связаны между собой напрямую, т.е. граф их связей представляет собой клику.

Виртуальный сервер P_i , $i \in [1: N]$ определяется следующими параметрами:

N_i^{CPU} – число процессоров в сервере;

p_i – производительность сервера, MFLOP/с;

m_i^{CPU} – стоимость использования одного процессора в час;

m_i^{RAM} – стоимость использования заказанного объема оперативной памяти в час;

m_i^{HDD} – цена использования заказанного объема памяти на жестком диске в час

Отметим следующее обстоятельство. Реальная производительность виртуального сервера может несколько отличаться от заданной по следующим причинам:

- неизвестна фактическая загрузка физического сервера, на котором функционирует данный виртуальный сервер;
- виртуальный сервер может находиться в состоянии миграции (задача штатного балансировщика облака перераспределения виртуальных серверов по физическим серверам).

Модель алгоритма. Полагаем, что задача сводится к циклическому решению совокупности подзадач T_j , $j \in [1: n]$ (рисунок 1). Вычислительная сложность $q_j = q_j(t)$ (число операции, необходимых для решения подзадачи T_j на итерации t) полагается априори неизвестной, различной на различных итерациях и многократно превышающей вычислительную сложность подзадач T_0, T_Σ . В таком виде можно, например, представить модели популяционных алгоритмов решения задачи глобальной оптимизации таких, как алгоритм роя частиц, колонии пчел и т.д.

Метод оптимального отображения. Отображение подзадач T_j , $j \in [1: n]$ на архитектуру облака задаем отображающей $(N \times n)$ -булевой матрицей $X = X(t)$, в которой $x_{i,j} = 1$ означает, что подзадача T_j должна быть выполнена на виртуальном сервере P_i , а $x_{i,j} = 0$ – что указанная подзадача не должна выполняться на сервере P_i ; $i \in [1: N]$

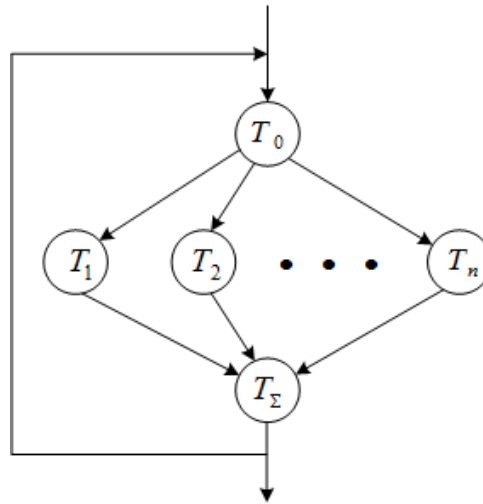


Рис. 1 Модель задачи

Полагаем, что после решения задачи T_j на том из процессоров виртуального сервера, которому она назначена на выполнение на итерации t , становится известной оценка $q_j(t)$ текущей вычислительной сложности этой задачи. Полагаем также, что эта вычислительная сложность «не слишком быстро» меняется в процессе итераций.

В качестве критерия оптимальности отображения используем суммарную стоимость решения задачи

$$M(\mathbf{X}) = \sum_{t=1}^{\hat{t}} m(X(t)) = \sum_{t=1}^{\hat{t}} \sum_{i=1}^N (N_i^{CPU} m_i^{CPU} + m_i^{RAM} + m_i^{HDD}) h(X(t)),$$

где $\mathbf{X} = (X(1), X(2), \dots, X(\hat{t}))$ - множество всех отображающих матриц; $m(X(t))$, $h(X(t))$ - стоимость и время решения задачи на итерации t соответственно; $\hat{t}$ - общее число итераций. Время $h(X(t))$ определяется с помощью таймера программного обеспечения облака.

На каждой из итераций $t \in [1:\hat{t}]$ задачу оптимального отображения ставим как задачу балансировки загрузки

$$\max_{i \in [1:N]} \sum_{j=1}^n p_i q_j(t-1) x_{i,j}(t) \rightarrow \min_{X(t)}, \quad (1)$$

т.е. как задачу отыскания такой отображающей матрицы $X^*(t)$, которая на данной итерации максимально равномерно загружает все виртуальные сервера. Заметим, что сумма в формуле (1) представляет собой оценку времени решения виртуальным сервером P_i всех распределенных ему подзадач.

Задачу балансировки загрузки (1) решаем жадным алгоритмом упаковки в контейнеры [2], где контейнеры формализуют производительности виртуальных серверов, а объекты упаковки - вычислительные сложности подзадач.

Вычислительный эксперимент

Был проведен эксперимент по решению в облаке задачи многокритериальной оптимизации методом нейронечеткой аппроксимации функции предпочтений лица, принимающего решения [3]. Исследовалась зависимость достигнутого ускорения вычислений от числа виртуальных серверов и вычислительная сложность целевой функции. Некоторые результаты эксперимента иллюстрирует рисунок 2. Вычислительный эксперимент выполнен в облаке, разработанном компанией *Parallels* [4]. В качестве коммуникационной библиотеки использована известная библиотека *MPI*.

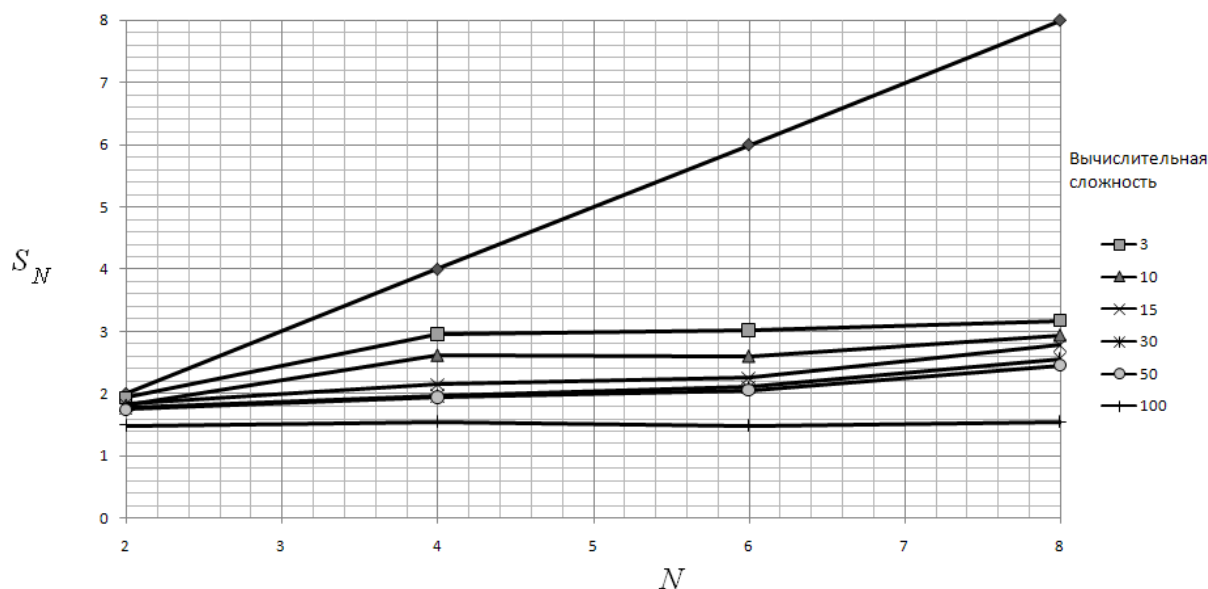


Рис. 2 Ускорение S_N в функции числа виртуальных серверов

Под вычислительной сложностью понимаем число стартовых точек используемого метода мултистарта.

Заключение

Результаты работы показывают, что

- облачная среда может быть адаптирована для ресурсоемких научных вычислений,
- приемлемое ускорение может быть получено только при высокой вычислительной сложности подзадач,
- для оптимизации научных вычислений в облаках необходима разработка специальных методов, алгоритмов и программного обеспечения,
- предложенный в работе подход может быть положен в основу разработки указанного программного обеспечения.

Литература

1. Mell P., Grance T. [The NIST Definition of Cloud Computing](#). Recommendations of the National Institute of Standards and Technology. [NIST](#) (20 October 2011).
2. [Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Жадные алгоритмы](#) // Алгоритмы: построение и анализ / Под ред. И. В. Красикова - 2-е изд.- М.: Вильямс, 2005.- 1296 с.
3. Карпенко А.П., Мухлисуллина Д.Т., Овчинников В.А. Нейросетевая аппроксимация функции предпочтений лица, принимающего решения, в задаче многокритериальной оптимизации // Информационные технологии, 2010, №10, с.2-9.
4. Parallels Automation for Cloud Infrastructure (PACI)