УДК 004.75

В. В. Гайдамако,
Кыргызско-Российский (Славянский) университет имени Б.Ельцина, Бишкек
Институт машиноведения, автоматики и геомеханики Национальной академии наук, Бишкек

УЧЕТ ЭНЕРГОЗАТРАТ В SIMGRID-МОДЕЛИ ОБЛАЧНОЙ СИСТЕМЫ

В статье рассматриваются способы учета энергозатрат для вычислительных и сетевых ресурсов при моделировании распределенных и облачных систем, в частности, для модели облачной системы, создаваемой с использованием фреймворка моделирования распределенных и облачных систем Simgrid. Описаны категории моделей энергопотребления, способы снижения энергозатрат, особенности учета энергозатрат в сетях передачи данных Дано описание моделирования энергопотребления при отсутствии в модели сущностей, представляющих сетевые устройства.

Ключевые слова: облачные вычисления, Центр Обработки Данных (ЦОД), энергозатраты, моделирование, Simgrid.

Введение

Облачные технологии представляют по требованию пользователя не только программы (Software-as-a-Service), но и платформу (Platform-as-a-Service), то есть аппаратное и соответствующее программное обеспечение (ПО), и полноценную инфраструктуру (Infrastructure-as-a-Service) – сетевое, вычислительное и другое оборудование, линии связи, системное и прикладное ПО, работающее на этом оборудовании, а также информационную безопасность и надежность. Организации могут перевестм свою инфраструктуру или ее часть в облако, чтобы сократить затраты на создание и содержание собственных вычислительных мощностей. Некоторые задачи, требующие высокопроизводительных вычислений, могут быть непосильными для вычислительной мощности одной организации, но их можно решать, используя ресурсы облака. Однако, прежде чем переводить часть или всю работу в облака, требуется оценить эффективность такого решения. Натурный эксперимент слишком затруднен или невозможен, поэтому для оценки различных показателей работы облачной инфраструктуры проводятся с помощью имитационного (компьютерного) моделирования.

Одним из важнейших показателей производительности облачной системы является потребляемая энергия и углеродный след. Углеродный след информационных и коммуникационных технологий (ИКТ) – это весь объем выбросов парниковых газов, вызванных производством, использованием и утилизацией продуктов и услуг ИКТ [1]. Количество энергии, потребляемой облачными Центрами Обработки Данных (ЦОД), постоянно увеличивается, в 2022 году оно составляло 460TWh (тераВатт-час), или 2% мирового потребления, и ожидается, что к 2026 году это будет более 1000TWh, если не будут приняты меры по энергосбережению [2], а углеродный след ИКТ, включающих ЦОД, сети, вычисления, компьютеры, смартфоны, серверы и маршрутизаторы, сопоставим с углеродным следом авиационного сектора, и составляет 1,8–2,8% мирового углеродного следа [1]. Облачные ЦОД вносят наиболее значимый вклад и в энергопотребление, и в углеродный след, поэтому многие средства моделирования включают энергетические модели, позволяющие разрабатывать и тестировать стратегии оптимизации использования облачных ресурсов.

Средства моделирования, включающие учет энергозатрат

Первым известным симулятором с моделированием энергопотребления в сетях передачи данных является NS2 (Network Simulator 2) с моделированием передачи пакетов, для которого было также разработано расширение для моделирования энергопотребления сети на уровне потока, Он был развит до GreenCloud [3], который разработан для моделирования облаков с использованием различных архитектур: двухуровневой, трехуровневой и трехуровневой высокоскоростной. Он обеспечивает детальное моделирование и анализ энергии, потребляемой элементами сетевых серверов, маршрутизаторов и связей между ними. Кроме того, он анализирует распределение нагрузки по сети, а также коммуникации с высокой точностью (уровень пакетов TCP). С точки зрения энергии GreenCloud определяет три типа энергии: вычисление (ЦП), коммуникации и физический вычислительный центр (система охлаждения), и включает два метода снижения энергии: динамическое масштабирование напряжения для снижения напряжения коммутаторов и динамическое отключение сети, которое позволяет отключать коммутаторы, когда это возможно. Однако GreenCloud может использоваться только для моделирования облачных систем. Кроме того, поскольку он опирается на NS2, его масштабируемость ограничена с точки зрения количества моделируемых устройств и объемов обмена данными.

В качестве замены NS2 был предложен другой симулятор на уровне пакетов NS3. Он предоставляет несколько модулей, включая ECOFEN (Energy Consumption mOdel For End-to-end Networks) [4].. Но пакетная природа NS3 препятствует использованию ECOFEN для крупномасштабных сетей. Пакетные симуляторы дают точные прогнозы, но они не масштабируются. Частота событий симуляторов уровня пакетов очень быстро увеличивается на крупномасштабных платформах с реальными рабочими нагрузками.

Симуляторы уровня потока для создания эффективных моделей производительности работают на более высоком уровне абстракции, и поэтому лучше подходят для моделирования крупномасштабных сетей. Они показывают хорошие способности к прогнозированию, и могут использоваться для симуляций потребления энергии в сети.

CloudSim относится к симуляторам уровня потока и построен на основе GridSim [5]. Это известный симулятор для моделирования облачных среды с оценкой энергопотребления. Он реализует такие зеленые стратегии, как динамическое масштабирование частоты напряжения и энергосберегающие методы предоставления виртуальных машин. Но так как CloudSim разработан для облачной среды, он не может использоваться для моделирования других распределенных систем. Другой момент заключается в том, что модели CloudSim очень похожи на модели GridSim, которые, как известно, содержат отклонения модели совместного использования полосы пропускания [5].

SimGrid – также симулятор уровня потока, он очень универсален и может использоваться для моделирования высокопроизводительных вычислений, Grid, Peer-to-peer, Fog и т. д. Он хорошо масштабируем и расширения позволяют моделировать энергопотребление [6],

Модели энергопотребления

Понимание связи между использованием ресурсов и потреблением энергии необходимо и для пользователей, и для персонала, ответственного за работу ЦОД, но особенно для разработчиков алгоритмов эффективного распределения ресурсов, так как оптимизация энергопотребления в облаке должна проводиться автоматически, Например,

энергоэффективность современного оборудования сильно снижается с понижением нагрузки, поэтому нужно стремиться к полной загрузке какого-то количества физических машин, в то время как другие могут быть отключены. Существует множество моделей энергопотребления, позволяющих проводить автоматическую оптимизацию, в [7] определены общие требования к этим моделям:

- Точность: модель должна быть достаточно точной, чтобы обеспечить желаемую экономию энергии, например, в среднем ошибка не должна превышать 10%, хотя в некоторых случаях (для некоторых алгоритмов оптимизации распределения ресурсов), может потребоваться и более высокая точность.
- Скорость: модель должна генерировать прогнозы достаточно быстро, чтобы ее можно было использовать в политиках энергосбережения; например, одна выборка в секунду достаточно быстра.
- Универсальность и переносимость: структура моделирования должна подходить для как можно большего числа систем, охватывая мобильные, настольные и серверные системы; различные семейства процессоров и технологии памяти и хранения; а также различные потребляемые мощности и динамические диапазоны. Она также должна допускать различное распределение потребляемой мощности между компонентами, а не предполагать доминирования каких-либо компонент в системе. Кроме того, создание моделей для новой системы не должно требовать тщательного исследования пространства проектирования или настройки.
- Низкая стоимость: создание и использование модели не должно требовать дорогостоящей или навязчивой инфраструктуры.
- Простота: при прочих равных условиях простые модели должны быть предпочтительнее сложных как с точки зрения количества входов, так и сложности взаимосвязи между входами и выходами.

Определены две категории моделей энергопотребления: комплексные (comprehensive) модели и высокоуровневые модели, или модели черного ящика. Комплексные модели центрального процессора (ЦП) были предложены в [8] и [9]. Эти модели достигают высокой точности с точки зрения энергопотребления ЦП и опираются на конкретные детали микроархитектуры ЦП. Высокоуровневые модели являются более общими и игнорируют конкретные низкоуровневые детали работы и характеристики моделируемой инфраструктуры. Но они повышают переносимость модели и скорость моделирования. Эти простые модели жертвуют точностью вычислений, но не требуют детального знания архитектуры используемых процессоров.

Высокоуровневый подход чаще применяется в моделировании, а именно линейная модель мощности, используемая при фиксированной частоте:

$$P_{TOT} = (1 - \alpha)P_{CPUIdle} + \alpha P_{CPUFull}$$

где α— загрузка ЦП, а $P_{CPUIdle}$ и $P_{CPUFull}$ — потребляемая ЦП мощность при 0% и 100% использовании соответственно.

Средства экономии энергии

Экономить энергию можно, минимизируя ее использование на каждой машине ЦОД, а также при передаче по сети. В [10] описаны три распространенных инструмента

экономии потребляемой энергии, причем необходимо, чтобы все рассматриваемые хосты использовали технологию виртуализации.

Первое решение – метод включения-выключения(ON/OFF), отключает хосты, используемые недостаточно (по сравнению с пороговым значением) и включает их снова при необходимости. Все процессы, запущенные на таком хосте, переносятся на другие хосты, а затем хост отключается. И наоборот, когда все хосты интенсивно используются и поступают новые запросы на обслуживание, один или несколько хостов снова включаются. Этот процесс имеет только одну цель – сократить количество включенных хостов в заданное время.

Второе решение – миграция, перемещение процессов между [11] хостами. Собственно, переносятся не просто процессы, а виртуальные сущности – машины или контейнеры. Миграция позволяет переносить работающую виртуальную среду с одного хоста на другой. Миграция затратна и по времени, и по энергии, когда данные передаются между источником и местом назначения. Также важно учитывать общую стоимость таких действий. Эта техника позволяет освободить некоторые хосты и деактивировать их, в то же время оставшиеся работающие хосты могут использоваться на максимальном уровне (консолидация).

И третье решение – DVFS (Dynamic Voltage and Frequency Scaling, динамическая настройка напряжения и частоты) [12] может динамически изменять напряжение и частоту ЦП хоста в соответствии с его нагрузкой. В ядре Linux DVFS может быть активирован в пяти различных режимах: Performance, PowerSave, UserSpace, Conservative и OnDemand. Каждый режим имеет регулятор, который решает, следует ли изменять частоту (увеличивать или уменьшать) или нет. Три из этих пяти режимов используют фиксированную частоту: Performance использует ЦП на его самой высокой частоте, PowerSave использует самую низкую частоту, а режим UserSpace позволяет пользователю выбрать одну из доступных частот для ЦП.

Два последних режима, Conservative и OnDemand, являются динамическими. Это означает, что частота ЦП может меняться со временем в зависимости от нагрузки ЦП. Регуляторы этих двух режимов работают с пороговыми значениями (одним или двумя) и периодически проверяют, находится ли загрузка ЦП ниже (или выше) этих пороговых значений, после чего принимается решение об изменении частоты ЦП.

Консервативный регулятор работает с верхним порогом up_threshold и и нижним down_threshold. Когда загрузка ЦП превышает up_threshold, частота увеличивается, а когда загрузка ЦП ниже down_threshold, частота ЦП уменьшается. Этот режим является прогрессивным, и каждое изменение частоты ЦП выполняется с шагом один с использованием всех доступных частот.

Режим OnDemand (по запросу) использует только один порог и способствует производительности системы, напрямую устанавливая самую быструю частоту, когда загрузка ЦП превышает порог. Уменьшение частоты ЦП, выполняемое с шагом один, как в консервативном режиме, выполняется, если загрузка ЦП остается ниже порога в течение предопределенного периода времени. Более низкая частота, которая подразумевает более слабое напряжение, снижает энергопотребление ЦП, но также замедляет вычислительную мощность ЦП.

DVFS в Simgrid. Модуль учета энергопотребления Host Energy

Для учета потребления энергии в Simgrid используется подключаемый модуль (плагин) Host Energy. Этот плагин позволяет вести учет энергии и во время вычислений, и

во время простоя, и даже рассеивание энергии. Плагин должен быть активирован перед загрузкой платформы (sg_host_energy_plugin_init()), для получения данных о потреблении данной ФМ используется sg_host_get_consumed_energy(). При включенной ФМ потребление энергии зависит от текущей загрузки ЦП и от профиля энергопотребления хоста. Simgrid считает, что потребление линейно зависит от количества ядер на полной скорости, с аномалией, когда все ядра простаивают (потребление не уменьшается до минимума) [6]. Энергетическая модель принимает 4 параметра:

- Idle – потребление при бездействии (т. е. мгновенное потребление в ваттах), когда хост запущен и работает, но ему нечего делать, у него нет нагрузки;
- Epsilon − потребление, когда все ядра находятся в состоянии 0 или Epsilon %, но не в состоянии бездействия;
- AllCores – потребление при полной загрузке всех ядер (100 %);
- Off – потребление, когда хост выключен.

Пример описания хоста:

```
<host id="HostA" speed="100.0Mf" core="4">
     <prop id="wattage_per_state" value="100.0:120.0:200.0" />
     <prop id="wattage_off" значение="10" />
</host>
```

Если заданы только два значения, для отсутствующего значения Idle используется Epsilon. В этом примере даны следующие параметры: Off = 10 Вт; холостой ход (Idle) 100 Вт; Эпсилон – 120 Вт, а AllCores – 200 Вт. Этого достаточно, чтобы рассчитать потребление в зависимости от количества загруженных ядер (таблица 1):

Таблица 1 – Потребление в зависимости от количества загруженных ядер

Загрузка ядер	Вт	Описание
0 (простой)	100	Значение Idle
0 (но нет простоя)	120	Значение Epsilon
1	140	Линейная экстраполяция между Epsilon и AllCores
2	160	Линейная экстраполяция между Epsilon и AllCores
3	180	Линейная экстраполяция между Epsilon и AllCores
4	200	Линейная экстраполяция между Epsilon и AllCores

Если у хоста несколько уровней потребления (pstates) DVFS (Dynamic voltage and frequency scaling) [12], например, у машин с графическими процессорами), указывается энергетический профиль каждого уровня потребления энергии (pstate):

```
<host id="HostC" speed="100.0Mf,50.0Mf,20.0Mf" core="4">
    <prop id="wattage_per_state"
       value="95.0:120.0:200.0, 93.0:115.0:170.0, 90.0:110.0:150.0" />
    <prop id="wattage_off" value="10" />
</host>
```

Получаемые значения приведены в таблице 2.

Таблица 2 – Значения потребления для различных режимов работы процессора

Уровень потребления (pstate)	Производитель-ность	Idle	Epsilon	AllCores
0	100 Mflop/s	95 Вт	120 Вт	200 Вт
1	50 Mflop/s	93 Вт	115 Вт	170 Вт
2	20 Mflop/s	90 Вт	110 Вт	150 Вт

Энергопотребление каналов связи напрямую зависит от текущей загруженности трафиком. В файле платформы это потребление может быть указано следующим образом:

```
<link id="SWITCH1" bandwidth="125Mbps" latency="5us"
sharing_policy="SHARED" >
     <prop id="waitage_range" value="100.0:200.0" />
     <prop id="wattage_off" value="10" />
</link>
```

Первое свойство означает, что когда линия связи включена, но бездействует, она будет потреблять 100 Вт. При полной загрузке будет рассеиваться 200 Вт, при 50% нагрузки − 150 Вт. Второе свойство означает, при выключенной линии будет рассеиваться 10 Вт. Для запуска плагина используется функция sg_link_energy_plugin_init(), перед загрузкой платформы, для получения потребления линии связи – функция sg_link_get_consumed_energy() [6].

Особенности энергопотребления в сетях связи

Распределенные вычисления широко используются с момента появления Интернета, их роль и появление новых технологий (облачные системы, Интернет вещей. Fog-, Mist, Edge-системы) требуют повышения производительности сетей передачи данных, и их энергопотребление становится все более важным показателем для оптимизации. Именно в сетях связи наблюдается самый высокий рост потребления энергии – 10%, тогда как для пользовательских устройств он составляет 5%, для ЦОД – 4% [2]. Для сдерживания этой тенденции требуются энергоэффективные решения, для поиска которых и создаются модели. Модели энергопотребления в сетях, основанные на передаче пакетов, учитывающие каждый передаваемый пакет, точны, но с ростом количества устройств становятся малопригодными, слишком сложными для описания, и слишком сложными в интерпретации результатов.

Симуляторы уровня потока при работе не моделируют каждый пакет, тем самым экономя память и вычисления, практически представляя собой хороший компромисс между абстракцией сети и масштабируемостью с точки зрения использования памяти и времени вычислений. В [13] на примере симуляторов NS3 и Simgrid показано, что моделирование потребления энергии сети в симуляторе уровня потока приводит к прогнозам, которые соответствуют прогнозам симуляторов уровня пакета, а также, что простых линейных моделей достаточно для получения точных значений.

Классическая модель

Особенностью сетевых устройств является то, что даже если через них не проходит сетевой трафик, они должны оставаться включенными, чтобы в любой момент быть готовыми к принятию и передаче пакета. Эта часть энергопотребления в простое называется также статической. В некоторых случаях стоимость энергии простоя сетевого

устройства может составлять 85% от его пиковой мощности [14]. Классически энергопотребление оценивается с использованием определения энергии

$$E(t) = \int_0^t P(t)dt$$

Потребление энергии для сетевого оборудования меняется со временем в соответствии с сетевым трафиком. Поскольку его мощность состоит из двух частей: статической и динамической, мы можем определить ее уравнением 1:

$$P_{netdev} = \underbrace{P_{idle}}_{\text{Static}} + \underbrace{\sum_{i \in S} Np_i \times Pp_i + Nb_i \times Pb_i}_{\text{Dynamic}} \qquad (1)$$

где S — набор портов на данном сетевом устройстве, N_{pi} и N_{bi} — соответственно количество пакетов и количество байтов, проходящих через порт i, а P pi и P bi – стоимость энергии на пакет и на байт для порта [13].

Существующие энергетические модели в литературе основаны на этом уравнении и, следовательно, они несовместимы напрямую с симуляторами уровня потока. Действительно, симуляторы уровня потока не предоставляют никакого понятия порта в своих моделях или пакетах. Энергетические модели должны быть адаптированы к сетевому представлению потока.

Представление сетевого соединения в Simgrid

Симулятор SimGrid не содержит контроллеров сетевых интерфейсов и маршрутизаторов. Вместо этого пути между двумя хостами представлены маршрутами (Routes). Каждый маршрут состоит из нескольких сущностей, называемых связями (Link), которые представляют как саму линию связи, так и сетевой адаптер на каждом конце. Маршрут между двумя хостами может состоять из нескольких связей, но не обязательно из нескольких хостов. В симуляторах уровня потока нет понятия порта, но потребление энергии портами может быть включено в потребление энергии самими связями. Затем можно определить мощность всей сетевой платформы с помощью уравнения

$$P_{total} = \sum_{i \in L} LinkPower_i \times LinkUsage_i$$

где L представляет набор связей на моделируемой сетевой платформе.

Однако в каждой сети каждая связь подключена к двум разным портам, возможно с разными схемами потребления энергии. Таким образом, абстракция симуляторов уровня потока создает трудности для транспонирования в них моделей энергии на уровне пакетов. Чтобы решить эту проблему, в [13] определяются две модели энергии, одна для однородной энергетической платформы (одинаковое потребление энергии для портов на данном соединении) и одну для неоднородной энергетической платформы. Это различие на самом деле можно упростить.

Однородная (гомогенная) модель для связи

Для линий связи (Link) нужно учесть потребление энергии каждым из портов и потребление при использовании полосы пропускания каждым из участников. Для достижения этой цели в [13] предлагается использовать одно сплит- или раздельно-дуплексное соединение (split-duplex) между конечными узлами, то есть полнодуплексная связь в модели представляется двумя односторонними линиями, которые обозначаются как восходящая и нисходящая, или исходящая и входящая, или UP и DOWN. Так как в SimGrid нет портов, для учета двух портов (по одному на каждом конце соединения), динамическое потребление энергии каждым соединением удваивается. Поскольку

раздельно-дуплексные каналы приводят к созданию двух соединений, потребление простоя уже будет удвоено, и нужно удвоить только динамическую часть потребления. Энергия, потребляемая портами, описывается линейной моделью с минимальным значением, равным потреблению энергии в режиме простоя, и максимальным значением, определяемым как:

$$max = idle + (BW \times ByteCons + {}^{PktCons}/_{MTU}) \times 2 \qquad (2)$$

В этом уравнении BW представляет скорость порта в бит/с, ByteCons – потребление энергии на байт в джоулях, P ktCons – потребление энергии сетевым устройством в джоулях, необходимое для обработки пакета. Наконец, MTU, максимальная единица передачи, используется в качестве приближения размера пакета [13]. При рассмотрении односторонней связи канал потребляет энергию для передающего (Tx) и принимающего (Rx) портов одновременно и наоборот. Если есть несколько потоков, разделение полосы пропускания происходит на восходящих или нисходящих каналах в соответствии с направлениями потоков.

Эта энергетическая модель имеет несколько преимуществ. Во-первых, ее легко реализовать – на канал приходится только одно значение расхода энергии. что упрощает реализацию на симуляторах уровня потока. Эти симуляторы всегда используют каналы для связи (но не обязательно порты). Эта модель обеспечивает низкие вычислительные издержки, однако во многих реальных топологиях используются неоднородные сетевые устройства и потребление энергии от одного сетевого устройства к другому варьируется в зависимости от его спецификации. Такая неоднородность не может быть явно представлена с помощью однородной модели.

Неоднородная (гетерогенная) модель для связи

В SimGrid маршруты от одного хоста к другому могут состоять из нескольких ссылок. Поэтому можно воспользоваться этой функцией для построения гетерогенной модели.

Для моделирования двух портов с разным потреблением энергии на одном канале в [13] вводятся маршруты из трех связей. Первая связь моделирует потребление энергии первого порта, вторая связь управляет совместным использованием полосы пропускания и третья связь моделирует потребление энергии второго порта. Так как в модели используются разделенные дуплексные связи, SimGrid создаст две связи, и, следовательно, мощность простоя будет умножена на два. Чтобы решить эту проблему, мощность простоя каждого энергетического канала нужно разделить на два. Таким образом, эта энергетическая модель следует линейному поведению с минимальным значением, равным idle/2, и максимальным значением, определяемым как

$$max = {}^{idle}/_{2} + (BW \times ByteCons + {}^{PktCons}/_{MTU}) \qquad (3)$$

Итак, в Simgrid-модели возможно осуществлять учет потребления энергии и для вычислительных устройств, и для сетей передачи данных. Однако, модели энергопотребления, перед тем как они будут включены в рабочую «большую» модель, должны пройти «обкатку» на небольших тестовых моделях, пройти валидацию [15]. Поэтому первым шагом в эксперименте будет создание «микромоделей», интерпретация наблюдений которых достаточно проста, результаты можно проверить теоретическими вычислениями или на доступных реальных системах.

Примеры определения количества потребленной энергии

Для учета энергопотребления ЦП Simgrid предоставляет плагины HostEnergy, HostDVFS, для связей – плагины LinkEnergy, WifiEnergy [6].

Плагин HostEnergy содержит функции, позволяющие получить количество потребленной хостом энергии (sg_host_get_consumed_energy), количество энергии, потребляемой хостом в состоянии простоя (sg_host_get_idle_consumption), минимальное и максимальное потребление в указанном состоянии (sg_host_get_wattmin_at, sg_host_get_wattmax_at).

Пример 1. Потребление хоста в разных состояниях. В файле платформы для хоста указывается потребление DVFS [6]:

```
<!-- Атрибут 'pstate' описывает начальное состояние, наименьшее значение pstate соответствует максимальной скорости процессора) -->
```

Первый хост, 4 ядра:

```
<host core="4" id="Host1" pstate="0" speed="100.0Mf,50.0Mf,20.0Mf">
<!--Список Idle:Epsilon:AllCores (Вт) соответствует потреблению во время простоя, при загрузке epsilon у всех ядер, при полной загрузке всех ядер-->
<!—Список должен содержать энергетический профиль для каждого описанного состояния pstate, здесь 100.0Mf,50.0Mf,20.0Mf -->
<prop     id="wattage_per_state"     value="100.0:93.33333333333333:200.0,     93.0:90.0:170.0, 90.0:90.0:150.0" />
```

При отключении питания

```
<prop id="wattage_off" value="10" />
</host>
```

Второй хост, одно ядро, работает при полной загрузке или простаивает (Epsilon=AllCores):

```
<host core="1" id="Host2" pstate="0" speed="100.0Mf,50.0Mf,20.0Mf">
<prop id="wattage_per_state" value="100.0:200.0:200.0, 93.0:170.0:170.0, 90.0:150.0:150.0" />
<prop id="wattage_off" value="10" />
</host>
```

Третий хост, одно ядро

```
<host core="1" id="Host3" pstate="0" speed="100.0Mf,50.0Mf,20.0Mf">
<prop id="wattage_per_state" value="100.0:200.0:200.0, 93.0:170.0:170.0, 90.0:150.0:150.0" />
<prop id="wattage_off" value="10" />
</host>
```

Описание коммуникации:

```
<link bandwidth="100kBps" id="bus" latency="0" sharing_policy="SHARED">
<!—диапазон потребления, близкий к реальному <prop id="wattage_range" value="10.3581:10.7479" /> -->
<!--  для теста -->
<prop id="wattage_range" value="1:3" />
</link>
```

Маршруты:

```
    <route dst="Host2" src="Host1">
      <link_ctn id="bus" />
    </route>
    <route dst="Host3" src="Host1">
      <link_ctn id="bus" />
    </route>
    <route dst="Host3" src="Host2">
      <link_ctn id="bus" />
```

```
    </route>
  </zone>
</platform>
```

Эта простая платформа из трех хостов, все хосты соединены между собой одной линией связи, используется далее во всех примерах.

При запуске моделирования включаются все три хоста. Затем Host1 и Host2 выполняют действия, состоящие из периодов ожидания, процессор потребляет минимальное количество энергии, и вычислений 1E+08 flops (операций с плавающей запятой).

В таблице 1 показаны затраты энергии на каждом этапе выполнения для работающего хоста Host1.

Таблица 1 – Результат выполнения для Host1, профиль 100.0:93.33333333333333:200.0, 93.0:90.0:170.0, 90.0:90.0:150.0

Действие	Pstate	Текущая пиковая скорость flop/s	Время, с	Потребление Вт	Затраты энергии Дж
Sleeping	1	1E+08	10	0	1000
Вычисления	1	1E+08	1	93-200	1120
Вычисления	2	2E+07	5		1645
Sleeping	2	2E+07	10		2545

В таблице 2 Показаны затраты и энергии на каждом этапе выполнения для хоста Host2.

Третий хост был подключен к питанию, но не выполнял никаких действий.

Таблица 2 – Результат выполнения для Host2, профиль 100.0:200.0:200.0, 93.0:170.0:170.0, 90.0:150.0:150.0

Действие	Pstate	Текущая пиковая скорость flop/s	Время, с	Потребление Вт	Затраты энергии Дж
Sleeping	1	1E+08	10		2000
Вычисления	1	1E+08	1	93-200	1200
Вычисления	2		5		1950
Sleeping	2	2E+07	4		2310

Общие затраты энергии, затраты на работающих хостах и на простаивающих хостах в Джоулях:

```
Total energy consumption: 9515.000000 Joules (used hosts: 5915.000000 Joules; unused/idle hosts: 3600.000000)
```

Затраты по хостам:

```
Energy consumption of host Host1: 3445.000000 Joules
Energy consumption of host Host2: 2470.000000 Joules
Energy consumption of host Host3: 3600.000000 Joules
```

Как видим, простаивающий хост потребляет очень много энергии, поэтому и нужны алгоритмы балансировки нагрузки и миграция ВМ, недогруженные машины потребляют много энергии, причем впустую.

Пример 2. Потребление при коммуникации (линия связи и сетевое оборудование источника и приемника). Используется та же платформа, с одной линией связи 100 Кб/с. Плагин активируется функцией sg_link_energy_plugin_init(); после этого выполняется

передача сообщения через почтовый ящик и в конце симуляции платформа выдает потребленную энергию. Передача осуществляется одним потоком (все сообщение сразу) или с помощью put_async() создается несколько потоков передачи, каждый передает сообщение, количество потоков и размер сообщения задаются аргументом командной строки.

Для сообщения 25000 байт одним потоком:

```
[10.250000] (0:maestro@) Total energy over all links: 10.750000
[10.250000] (0:maestro@) Energy consumption of link 'bus': 10.750000 Joules
```

Двумя потоками:

```
[10.541237] [link_energy/INFO] Total energy over all links: 11.623711
[10.541237] [link_energy/INFO] Energy consumption of link 'bus': 11.623711 Joules
```

Для 4 потоков:

```
[11.082474] [link_energy/INFO] Total energy over all links: 13.247423
[11.082474] [link_energy/INFO] Energy consumption of link 'bus': 13.247423 Joules
```

Для передачи 50000000 байт одним потоком:

```
[510.000000] (0:maestro@) Total energy over all links: 1510.000000
[510.000000] (0:maestro@) Energy consumption of link 'bus': 1510.000000 Joules
```

Двумя потоками:

```
[1092.474227] [link_energy/INFO] Total energy over all links: 3257.422680
[1092.474227] [link_energy/INFO] Energy consumption of link 'bus': 3257.422680 Joules
```

Для 4 потоков:

```
[2174.948454] [link_energy/INFO] Total energy over all links: 6504.845361
[2174.948454] [link_energy/INFO] Energy consumption of link 'bus': 6504.845361 Joules
```

Для относительно небольшого объема передачи увеличение количества потоков незначительно увеличивают и время передачи, и потребленную энергию. На больших объемах увеличение времени передачи и потребления очень значимо.

Пример 3. Затраты на виртуальных машинах. С хоста Host1 запускаются две ВМ на Host1 и Host2, используют по одному ядру:

```
auto* vm_host1 = host1->create_vm("vm_host1", 1);
vm_host1->start();
auto* vm_host2 = host2->create_vm("vm_host2", 1);
vm_host2->start();
```

На всех хостах запускаются по две активности:

- обе внутри ВМ на Host1;
- одна на ВМ, другая на хосте на Host2;
- обе на хосте Host3, без ВМ.

Все эти активности симулируют выполнение 1E+8 операций с плавающей точкой.

Результат:

```
[ 10.000000] (0:maestro@) Energy consumption of host MyHost1: 1120.000000 Joules
[ 10.000000] (0:maestro@) Energy consumption of host MyHost2: 1600.000000 Joules
[ 10.000000] (0:maestro@) Energy consumption of host MyHost3: 1600.000000 Joules
```

Выполнение на двух ВМ привело к меньшему потреблению энергии, то есть использование виртуальных ресурсов приводит к заметной экономии энергии.

Заключение

Показатели потребления энергии как вычислительными ресурсами, так и сетями передачи данных, являются очень важными для оценки производительности распределенных систем, включая облачные, Fog- Mist- Edge-системы и Интернет вещей, прилагается множество усилий по оптимизации и, следовательно, снижению потребления энергии всеми элементами этих систем. Для разработки и тестирования алгоритмов оптимизации используются имитационные модели распределенных и облачных систем, в статье приведены примеры симуляторов NS2, NS3, GreenCloud, CloudSim. Для дальнейших экспериментов будет использоваться платформа моделирования Simgrid, для верификации моделей могут использоваться другие платформы. Для Simgrid приведены примеры описания энергопотребления хостов в состоянии простоя (Idle), агрузке Epsilon%, нагрузке 100% (AllCores), в выключенном состоянии (Off). Для хостов, имеющих несколько уровней потребления (DVFS) указывается энергетический профиль каждого уровня. Энергопотребление при передаче данных в Simgrid моделируется через сущность Link, представляющую как саму линию связи, так и сетевые устройства, которые она связывает, причем эти устройства могут иметь как одинаковое, так и разное энергопотребление. Таким образом, в Simgrid возможно создать модель энергопотребления для всей платформы.

Литература

1. Будущее углеродного следа сектора информации и связи (ИКТ). URL: https://www.databridgemarketresearch.com/ru/whitepaper/future-of-carbon-footprint-of-information-and-communication?srsltid=AfmBOorN1KV53aAKCIHyakDtbNsZRnBPjZRYlYKISE_betTbMpPa1ZEz (дата обращения: 10.10.2024)
2. Global data center electricity use to double by 2026 - IEA report. URL: https://www.datacenterdynamics.com/en/news/global-data-center-electricity-use-to-double-by-2026-report (дата обращения: 10.10.2024)
3. Kliazovich, D., Bouvry, P., Audzevich, Y., Khan, S.U.: GreenCloud: A Packet-Level Simulator of Energy-Aware Cloud Computing Data Centers. In: IEEE GLOBECOM (2010)
4. Orgerie, A.C., et al.: Simulation Toolbox for Studying Energy Consumption in Wired Networks. In: Int. Conf. on Network and Service Management (2017)
5. CloudSim Energy-aware Simulations. URL: https://cloudsimtutorials.online/cloudsim-energy-aware-simulations/ (дата обращения: 10.10.2024).
6. Simulation of Distributed Computer Systems // URL: https://simgrid.org/ (дата обращения: 05.10.2024)
7. S. Rivoire, P. Ranganathan, C. Kozyrakis, A comparison of high-level full-system power models, in: Proceedings of the 2008 Conference on Power Aware Computing and Systems, HotPower’08, USENIX Association, Berkeley, CA, USA, 2008, p. 3.

8. R. Joseph, M. Martonosi, Run-time power estimation in high performance microprocessors, in: Proceedings of the 2001 International Symposium on Low Power Electronics and Design, ISLPED ’01, ACM, New York, NY, USA, 2001, pp. 135–140
9. C. Isci, M. Martonosi, Runtime power monitoring in high-end processors: methodology and empirical data, in: Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture, MICRO 36, IEEE Computer Society, Washington, DC, USA, 2003, p. 93.
10. Guérout, Tom and Monteil, Thierry and DaCosta, Georges and Neves Calheiros, Rodrigo and Buyya, Rajkumarand Alexandru, Mihai Energy-aware simulation with DVFS. (2013)Simulation Modelling Practice and Theory, vol. 39. pp. 76-91. ISSN1569-190X
11. C.C. Keir, C. Clark, K. Fraser, S. Hand, J.G. Hansen, E. Jul, C. Limpach, I. Pratt, A. Warfield, Live migration of virtual machines, in: Proceedings of the 2nd ACM/USENIX Symposium on Networked Systems Design and Implementation (NSDI), 2005, pp. 273–286.
12. T. Kolpe, A. Zhai, S. Sapatnekar, Enabling improved power management in multicore processors through clustered dvfs, in: Design, Automation Test in Europe Conference Exhibition (DATE), 2011, pp. 1–6
13. Loic Guegan, Betsegaw Lemma Amersho, Anne-Cécile Orgerie, Martin Quinson. A Large-Scale Wired Network Energy Model for Flow-Level Simulations. AINA 2019 - 33rd International Conference on Advanced Information Networking and Applications, Mar 2019, Matsue, Japan. pp.1047-1058, ff10.1007/978-3-030-15032-7_88ff. ffhal-02020045v2ff
14. Fiandrino, C., Kliazovich, D., Bouvry, P., Zomaya, A.Y.: Performance Metrics for Data Center Communication Systems. In: IEEE CLOUD (2015)
15. Velho, P., Schnorr, L.M., Casanova, H., Legrand, A.: On the validity of flow-level tcp network models for grid and cloud simulations. ACM Transactions on Modeling and Computer Simulation **23**(4) (2013)