Использование веб-сервисов для запуска и тестирования параллельных программ в рамках учебного курса*

О.В. Сухорослов[1,2]

Институт проблем передачи информации РАН[1],
Национальный исследовательский университет Высшая школа экономики[2]

В работе описывается опыт разработки и применения веб-сервисов для поддержки учебного процесса в рамках курсов по параллельным и распределенным вычислениям. Использование универсальных сервисов для запуска программ на вычислительном кластере позволяет студентам быстро применять полученные знания, не отвлекаясь на изучение работы в командной строке кластера. Удобство запуска программ через подобные сервисы также позволяет включать в занятия больше практических заданий и демонстраций. Помимо универсальных сервисов, были реализованы сервисы для тестирования решений практических заданий. Данные сервисы позволяют автоматизировать процесс проверки решений, а также предоставить студентам оперативный и детальный отклик в процессе решения задания. Для развертывания сервисов была использована облачная платформа Everest, отличительными особенностями которой являются легкость публикации сервисов и возможность их связывания с внешними вычислительными ресурсами. Данные особенности позволяют быстро воспользоваться представленным подходом преподавателям аналогичных курсов.

Ключевые слова: суперкомпьютерное образование, параллельное программирование, распределенные вычисления, удаленный доступ к высокопроизводительным ресурсам, тестирование программ, веб-интерфейсы, веб-сервисы, облачная платформа

1. Введение

Растущая важность преподавания параллельных и распределенных вычислений (ПРВ) обусловлена распространением в последнее десятилетие многоядерных архитектур, графических процессоров, облачных сервисов и задач, связанных с обработкой больших объемов данных. Помимо теоретических основ, важной частью любого подобного курса являются практические задания, направленные на закрепление полученных знаний и выработку соответствующих навыков путем использования различных классов вычислительных систем и технологий программирования.

Однако организация практических занятий по ПРВ сопряжена с рядом проблем, связанных со сложностью используемых систем, пользовательских интерфейсов и технологий. Типичным примером является проведение лабораторных практикумов или выполнение домашних заданий на вычислительном кластере. Распространенный подход состоит в создании учетной записи на кластере для каждого студента и обучении студентов работе в командной строке кластера, включая компиляцию программ и их запуск через систему очередей. Данный подход имеет следующие недостатки. Во-первых, необходимы дополнительные ресурсы для администрирования учетных записей, обучения работе на кластере и технической поддержки. Во-вторых, студенты часто испытывают трудности при запуске программ в плохо знакомом окружении. В результате, заметные усилия расходуются как студентами, так и преподавателями на вспомогательные задачи вместо того, чтобы сконцентрироваться на усвоении базовых элементов знаний. В силу ограниченных ресурсов преподавателей традиционные подходы к организации практических занятий также плохо масштабируются

*Исследование выполнено за счет гранта Российского научного фонда (проект №16-11-10352).

на большое число студентов. Таким образом, необходимо использование новых подходов, которые бы позволили повысить эффективность обучения и охват аудитории курсов по ПРВ.

Веб-сервисы представляют собой удобную альтернативу для организации доступа к высокопроизводительным ресурсам и поддержки выполнения практических заданий на данных ресурсах. Несмотря на то, что данный подход успешно используется в рамках некоторых курсов, разработка подобного рода решений с нуля требует существенных усилий. В свою очередь, повторное использование имеющихся решений также требует значительных усилий по развертыванию и администрированию данных систем в рамках инфраструктуры вуза, а также модификации под цели конкретного курса. Существует нехватка универсальных и гибких онлайн-платформ, позволяющих с минимальными затратами, без необходимости установки и сопровождения, реализовать необходимые сервисы и использовать их в учебном процессе.

В данной работе рассматривается подход к организации практических заданий по ПРВ, основанный на применении веб-сервисов и платформы Everest [9, 10]. К показал опыт, данная платформа, изначально ориентированная на поддержку создания вычислительных веб-сервисов, также может быть эффективно использована для поддержки учебного процесса. В сравнении с известными решениями в данной области, Everest имеет ряд отличительных особенностей. Платформа реализует облачную модель Platform as a Service (PaaS) и поддерживает работу с ней множества пользователей через удаленные веб- и программные интерфейсы. Пользователи Everest могут подключать к платформе внешние вычислительные ресурсы и связывать их с создаваемыми сервисами. Это позволяет быстро развертывать на базе платформы произвольные вычислительные веб-сервисы и предоставлять доступ к ним другим пользователям.

В целях упрощения доступа студентов к вычислительным ресурсам на базе Everest было реализовано несколько универсальных сервисов для запуска различных типов параллельных и распределенных программ. Также, в целях автоматизации проверки практических заданий и предоставления студентам оперативного отклика в процессе решения задач, были реализованы проблемно-ориентированные сервисы для тестирования решений домашних заданий. Данный подход и созданные сервисы успешно используются в рамках двух курсов по ПРВ с 2014 года.

В главе 2 проводится обзор работ в данной области и сравнение предложенного подхода и существующих решений. В главе 3 приводится краткое описание платформы Everest и рассматривается использование ее для создания учебных сервисов для запуска и тестирования параллельных программ. В главе 4 описываются опыт использования предложенного подхода в рамках учебных курсов и примеры созданных сервисов для различных разделов программы. В главе 5 содержатся выводы и описываются дальнейшие планы.

2. Существующие решения

Использование веб-технологий для создания удобных интерфейсов доступа к высокопроизводительным ресурсам, в том числе в рамках учебных курсов, началось с ростом Веб в середине 90-х годов. Например, в [1] авторы описывают несколько прототипов веб-сред для параллельного программирования, включая Virtual Programming Laboratory (VPL), использовавшуюся в учебном процессе. Развитие в дальнейшем концепции грид-вычислений и технологий веб-порталов привело к появлению грид-порталов, реализующих доступ через веб-браузер к распределенным вычислительным ресурсам. В [2] описывается опыт разработки подобного портала для поддержки курса по параллельному программированию.

Веб-интерфейсы также применялись для поддержки отправки и автоматической оценки решений практических заданий по ПРВ. В [3] авторы описывают программную платформу, позволяющую создавать веб-порталы для автоматического тестирования программ

студентов в рамках курсов по распределенному программированию. В недавней работе [4] описывается веб-приложение для автоматического тестирования и оценки параллельных программ. В [5] предлагается похожее решение для запуска и валидации параллельных программ, написанных с использованием различных парадигм.

Кроме того, появилось несколько веб-сред, ориентированных на поддержку преподавания новых тем, таких как распределенная обработка данных и программирование на графических процессорах. В [6] описана среда WebMapReduce, реализующая упрощенный веб-интерфейс к платформе Hadoop для поддержки обучения программированию в модели MapReduce. В рамках онлайн-курса Heterogeneous Parallel Programming [7] для поддержки выполнения практических заданий была разработана среда WebGPU.

В отличие от рассмотренных решений, разработанных с нуля для целей обучения, предлагаемый в данной статье подход основан на использовании веб-платформы общего назначения, предназначенной для создания вычислительных веб-сервисов. Возможность быстро создавать произвольные сервисы и связывать их с вычислительными ресурсами позволяет значительно сократить время разработки. Гибкость сервис-ориентированного подхода допускает разработку различных типов сервисов для отдельных случаев и областей применения. Наконец, облачная модель PaaS обеспечивает возможность повторного использования данного подхода другими преподавателями без необходимости установки платформы. Данные возможности могут быть особенно востребованы в случае, когда преподавателям требуется легкое в использовании и гибкое решение при отсутствии ресурсов, необходимых для развертывания и поддержки данного решения в рамках инфраструктуры вуза.

3. Технические аспекты

3.1. Платформа Everest

Everest [8, 9] - облачная программная платформа, поддерживающая публикацию, выполнение и композицию вычислительных приложений в распределенной среде. Публично доступный экземпляр платформы функционирует по адресу [10]. В данном разделе приведено краткое описание платформы.

На рис. 1 изображены архитектура и базовые сущности Everest. В отличие от традиционных платформ распределенных вычислений, Everest реализует облачную модель Platform as a Service (PaaS) путем предоставления своей функциональности через удаленные веб- и программный интерфейсы. Один экземпляр платформы может применяться одновременно многими пользователями для создания, запуска и разделения приложений. Каждое созданное в рамках платформы приложение автоматически публикуется как веб-форма и веб-сервис. Последний обеспечивает программный доступ к приложениям, интеграцию с внешними системами и композицию приложений в рамках прикладных сценариев (workflow). Другой отличительной чертой Everest является поддержка подключения пользователями внешних вычислительных ресурсов и гибкого связывания ресурсов с приложениями.

Серверная часть платформы состоит из трех основных уровней: программный интерфейс (REST API), приложения и вычислительное ядро. Клиентская часть платформы включает пользовательский веб-интерфейс (Web UI) и клиентские библиотеки.

REST API реализует удаленный программный интерфейс для доступа к функциональности платформы. Данный интерфейс, реализованный как набор REST-сервисов [11], является единой точкой входа для всех клиентов платформы, включая веб-интерфейс и клиентские библиотеки. Спецификация API открыта, что позволяет создавать сторонние клиенты.

Уровень приложений соответствует среде размещения приложений, созданных пользователями платформы. Приложения являются главными сущностями платформы, представляющими собой вычислительные блоки с четко описанными интерфейсами. Приложение имеет набор входов, которые образуют запрос на запуск приложения, и набор выходов, которые составляют результат запуска. Каждое размещенное в Everest приложение автома-

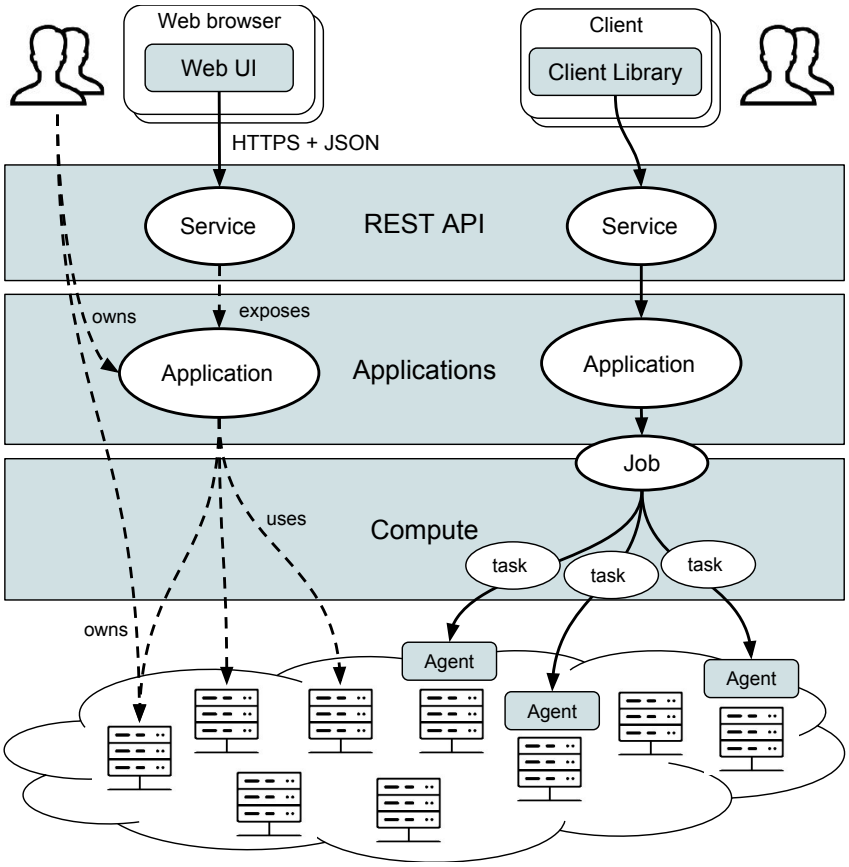

Рис. 1. Архитектура платформы Everest

тически публикуется как веб-сервис через REST API, что позволяет обращаться к приложению удаленно как через Web UI, так и через клиентские библиотеки. Владелец приложения может настраивать список пользователей, которым доступен запуск приложения.

Для упрощения публикации приложений Everest предоставляет универсальный каркас для приложений с интерфейсом командной строки, который позволяет избежать программирования при размещении приложения на платформе. Пользователю необходимо описать входные и выходные параметры приложения, параметризованный шаблон команды запуска приложения, а также соответствия между параметрами приложения и файлами, читаемыми и создаваемыми приложением.

Вычислительное ядро управляет выполнением приложений на вычислительных ресурсах. При вызове приложения через REST API формируется задание (job), состоящее из одной или нескольких вычислительных задач (tasks). Ядро реализует все действия, связанные с передачей данных, запуском и мониторингом задач на удаленных ресурсах. Также ядро осуществляет мониторинг состояния ресурсов и использует данную информацию при планировании задач.

Everest не предоставляет вычислительную инфраструктуру для выполнения приложений и вместо этого использует внешние ресурсы, подключенные пользователями. Платформа реализует интеграцию с одиночными машинами и вычислительными кластерами при помощи специально разработанной программы, т.н. *агента*. Агент выполняется на стороне ресурса и играет роль посредника между платформой и ресурсом, обеспечивая передачу данных и управление выполнением задач на ресурсе. Платформа также поддерживает интеграцию с ресурсами грид-инфраструктуры EGI [12].

Пользовательский веб-интерфейс (Web UI) представляет собой графический интерфейс для взаимодействия пользователей с платформой и размещенными приложениями.

Данный интерфейс реализован в виде приложения на языке JavaScript, которое может выполняться в любом современном веб-браузере без необходимости установки дополнительного ПО.

Клиентские библиотеки предназначены для упрощения программного доступа к платформе через REST API. Они позволяют пользователями создавать программы, которые вызывают размещенные на платформе приложения и комбинируют их в прикладные сценарии. В настоящий момент реализована подобная библиотека для языка Python.

3.2. Универсальные сервисы для запуска параллельных программ

Студенты, изучающие ПРВ, часто сталкиваются со сложностями при доступе к высокопроизводительным ресурсам и запуске на данных ресурсах своих программ. Окружение командной строки, используемое на данных ресурсах, является незнакомым и низкоуровневым для студентов без опыта работы с Linux. Платформа Everest может быть использована для устранения этих технических барьеров путем создания веб-сервисов для запуска параллельных программ. Данные сервисы оформляются в виде приложений на Everest и связываются с используемыми в рамках учебного курса ресурсами. Поскольку различные модели и технологии программирования используют различные языки программирования и параметры выполнения, то необходимо предусмотреть отдельные сервисы для каждой из рассматриваемых в курсе технологий.

Рассмотрим основные шаги, из которых состоит процесс создания подобных сервисов. Детальное описание данных шагов с техническими подробностями можно найти в руководстве пользователя на сайте Everest [10].

Для создания приложения преподавателю необходимо ввести через веб-интерфейс Everest метаданные приложения, описать входные и выходные параметры, шаблон команды, соответствия между параметрами и файлами, и т.п. Основной частью приложения является вспомогательный скрипт, который запускается на ресурсе при вызове приложения, принимает входные параметры и управляет выполнением переданной программы. Данный скрипт может быть реализован на любом языке программирования, поскольку Everest выполняет его запуск через интерфейс командной строки. Вспомогательный скрипт, как правило, выполняет компиляцию переданной программы, подготовку среды выполнения, постановку задания в очередь кластера и т.п. Написание данного скрипта представляет собой наиболее сложный этап при создании приложения. Однако, после того как подобный скрипт создан, его части могут быть повторно использованы в других приложениях.

Для того, чтобы связать созданное приложение с используемым в рамках учебного курса кластером, преподавателю необходимо подключить ресурс к платформе путем установки и запуска на нем агента. Данный этап обычно не занимает много времени, поскольку агент легок в установке и поддерживает интеграцию с распространенными системами очередей, не требуя прав суперпользователя и открытых для доступа извне сетевых портов. Для выполнения агента и программ студентов, отправляемых через Everest, можно создать на кластере отдельную учетную запись. При этом не требуется создание учетных записей для каждого из студентов.

Как только приложение создано, протестировано и готово к использованию студентами, преподавателю необходимо настроить доступ к приложению путем указания имен и групп пользователей, которым разрешен его запуск. Everest поддерживает создание произвольных групп пользователей. Для поддержки учебного курса удобно создать две группы для студентов и преподавателей соответственно, после чего разрешить обоим группам запуск приложения. Для студенческой группы может быть настроена самостоятельная регистрация путем ввода секретного кода, что позволяет избежать ручной регистрации студентов. После этого достаточно попросить студентов зарегистрироваться на Everest, добавить себя в группу и открыть список приложений, в котором должно быть доступно созданное приложение.

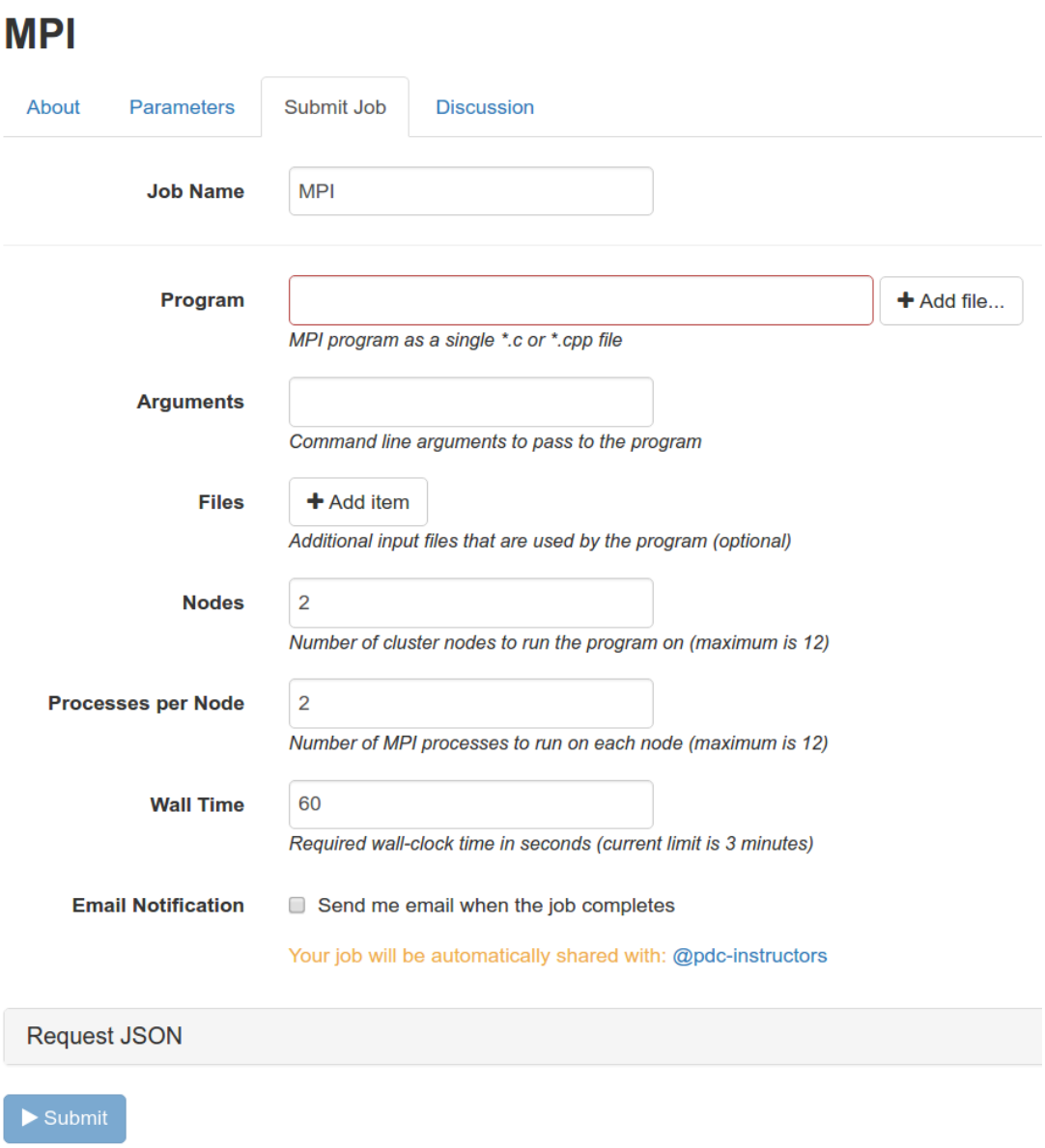

Рис. 2. Веб-форма сервиса для запуска MPI-программ

На рис. 2 приведен пример созданного на Everest универсального сервиса для запуска MPI-программ. Изображенная на рисунке веб-форма содержит входные параметры, которые необходимо заполнить студенту для запуска задания. Также можно ввести имя для задания и включить уведомление по электронной почте о завершении задания, что может быть удобно для долго выполняющихся или ожидающих в очереди заданий. Отметим, что данный пример позволяет передать только один файл с программой, что зачастую достаточно для учебных целей. Однако при необходимости можно легко модифицировать данный сервис для поддержки случаев, требующих передачи нескольких файлов с исходным кодом.

После запуска задания студент перенаправляется на страницу с динамически обновляемой информацией о состоянии задания. На рис. 3 приведен пример данной страницы для выполнившегося задания сервиса MPI. Открытая секция Outputs предоставляет доступ к значениям выходных параметров, полученных в результате выполнения задания. Страница задания также включает секции с общей информацией о задании и значениях входных параметров, указанных студентом, включая переданную программу. По умолчанию, задание в Everest доступно только запустившему его пользователю. Для учебных целей реализо-

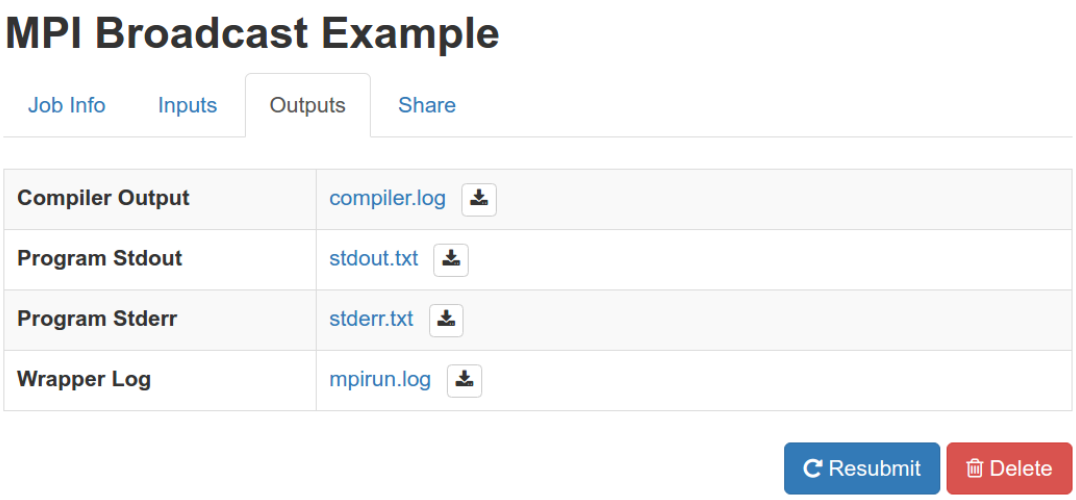

Рис. 3. Результаты запуска MPI-программы через сервис

вана возможность настройки доступа ко всем заданиям сервиса указанных пользователей, например, группы преподавателей. Таким образом, при сдаче решения или возникновении технических проблем, студенту достаточно отправить преподавателю ссылку на соответствующее задание.

Описанный подход был использован для реализации универсальных сервисов для запуска различных типов параллельных программ, описанных в главе 4.

3.3. Сервисы для тестирования решений практических заданий

Проверка решений практических заданий в рамках курсов по ПРВ требует значительных усилий и является одним из основных факторов, ограничивающих масштабируемость курсов по числу студентов. Рассмотренные выше универсальные сервисы могут использоваться для быстрых демонстраций на занятиях, простых упражнений или сложных проектов. Однако, в силу своей универсальности, данные сервисы не позволяют получить детальную информацию, необходимую для проверки решений конкретных заданий. Например, произвела ли программа корректный результат или какие показатели эффективности у данной программы. Наличие мгновенного и детального отклика важно для студентов, поскольку помогает избежать трудоемкой ручной валидации и сконцентрироваться на решаемой задаче. Данный отклик также важен для преподавателей, так как позволяет уменьшить время и усилия, необходимые для оценки решения.

Для автоматизации проверки параллельных программ требуется разработка отдельных проблемно-ориентированных сервисов, которые производят запуск программы студента на специально подготовленном наборе тестов. Данные сервисы могут быть реализованы на базе платформы Everest, используя аналогичный подход. В отличие от универсальных сервисов, в данном случае вместо вспомогательного скрипта используется набор тестов для соответствующей задачи. Данные тесты, как правило, выполняют многократные запуски программы с различными входными данными и параметрами выполнения, проводят проверку результатов запусков и измерение различных показателей. Выходы сервисов тестирования могут содержать отчеты отдельных тестов, значения измеренных показателей, набранные решением баллы и т.п.

В главе 4 приведены примеры подобных сервисов для тестирования решений заданий из различных разделов ПРВ.

4. Опыт использования

Описанный подход и платформа Everest успешно используются с 2014 года для поддержки двух курсов по ПРВ для студентов различных уровней.

Курс «Параллельные и распределенные вычисления» в Школе анализа данных Яндекса (ШАД) читается автором с 2008 года для студентов-магистров и охватывает многопоточное программирование, параллельные вычисления и распределенную обработку данных. Оценка за курс определяется на основе сдачи домашний заданий, которые подразумевают написание параллельных и распределенных программ, запускаемых на учебном кластере.

Применение Everest в курсе ШАД началось в 2014 году с разработки сервисов для тестирования решений домашних заданий. Данные сервисы развиваются и активно используются в данном курсе по настоящее время. До 2016 года в рамках курса не использовались универсальные сервисы, и студенты работали с учебным кластером напрямую в командной строке через отдельные учетные записи.

В 2015 году описанный подход был применен также в рамках курса «Высокопроизводительные вычисления» для студентов-бакалавров факультета компьютерных наук НИУ ВШЭ. С учетом меньшего опыта данных студентов, для упрощения работы с учебным кластером на базе Everest были реализованы и использованы в учебном процессе универсальные сервисы для запуска на кластере различных типов программ. Данные сервисы также позволили включить больше практических демонстраций и упражнений в ходе занятий. Студенты быстро освоили работу с предоставленными сервисами через веб-браузер и успешно применяли их для запуска программ как во время занятий, так и во время решения домашних заданий.

В 2016 году подход с использованием сервисов обоих типов был применен при преподавании курса в ШАД. В этот раз студенты могли выполнять все практические занятия через Everest, без необходимости прямого доступа к командной строке кластера. По желанию студенты могли запросить прямой доступ к кластеру для того, чтобы получить дополнительный практический опыт работы. Однако только несколько студентов воспользовались данной возможностью в течение курса, при этом никто из них не отказался полностью от использования сервисов. Уменьшение нагрузки по администрированию кластера и повышение степени автоматизации процессов запуска и тестирования программ помогли масштабировать курс на большее число студентов (118 по сравнению с 48 в 2014 году).

По имеющемуся опыту для создания нового сервиса требуется от часа до нескольких дней в зависимости от конкретного случая. Основную часть этого времени занимают реализация и отладка вспомогательного скрипта или набора тестов, которые обычно пишутся на языках Python или Bash. Реализация наборов тестов и соответствующих сервисов для проверки решений обычно требует больше времени, чем создание универсальных сервисов, где имеется больше возможностей для повторного использования ранее написанного кода.

Далее приведено краткое описание сервисов, разработанных и использованных в рамках упомянутых курсов, для различных разделов ПРВ.

4.1. Многопоточное программирование

В данном разделе рассматриваются понятие «одновременности» (concurrency) и основы многопоточного программирования. Студенты знакомятся с типовыми проблемами, возникающими при реализации одновременных программ, учатся их избегать и грамотно реализовывать координацию между потоками. Для написания программ используется язык C++ и стандартная поддержка потоков, появившаяся в C++11.

Поскольку студенты обычно не испытывают затруднений с компиляцией и запуском программ на C++ на своих машинах, необходимости создавать сервисы для запуска многопоточных программ на кластере не было. Однако, создание сервисов для тестирования решений заданий по данной теме оказалось очень полезным.

Рассмотрим в качестве примера хорошо известную задачу об обедающих философах. Студентам необходимо реализовать программу, удовлетворяющую базовым требованиям безопасности и живучести, при этом обеспечивая также справедливость, производительность и масштабируемость решения. В качестве начального приближения и шаблона для реализации студентам предлагается решение, не удовлетворяющее требованию безопасности (соседи могут есть одновременно). Для оценки решений был разработан набор тестов, осуществляющий проверку всех требований путем запусков программы с различными параметрами. Помимо проверки безопасности и живучести, тесты определяют и оценивают показатели, характеризующие справедливость, производительность и масштабируемость решения (соотношение между минимальным и максимальным количествами приемов пищи, среднее время ожидания приема пищи). Масштабируемость решения оценивается путем увеличения числа философов с 5 до 5000. Набор тестов выводит результаты каждого теста и общую сводку, включающую баллы за каждое требование и совокупную оценку решения.

Реализованный набор тестов был предоставлен студентам в виде сервиса на Everest, принимающего на вход программу студента и выполняющего запуски тестов на учебном кластере. Для обеспечения надежных и воспроизводимых результатов, каждый запуск тестов использует полностью один узел кластера.

4.2. Параллельное программирование

В данном разделе студенты знакомятся с OpenMP и MPI, наиболее широко используемыми технологиями параллельного программирования для систем с общей и распределенной памятью. Для обоих типов параллельных программ были созданы универсальные сервисы, поддерживающие запуск студентами программ на учебном кластере.

В то время как студенты не испытывали сложностей с компиляцией и отладкой OpenMP-программ на своих машинах, сервис OpenMP предоставил им возможность запуска программы на мощном сервере с 12 ядрами, что позволяет оценить эффективность выполнения программы на большем числе потоков. Входами сервиса являются файл с программой, аргументы командной строки, дополнительные файлы и используемое число потоков. На выходе сервис предоставляет файлы с выводами компилятора и программы.

Аналогичный сервис для MPI позволил студентам компилировать и запускать на учебном кластере MPI-программы с различными конфигурациями числа узлов и числа процессов на узел. Интерфейс данного сервиса уже был представлен в главе 3.

В состав домашних заданий входят задачи на написание параллельных программ с использованием каждой из технологий. Примером задачи на использование OpenMP является реализация параллельной версии метода k-средних. Студентам предоставляются исходная последовательная реализация метода на языке C++ и программа для генерации входных данных. Необходимо грамотно распараллелить исходную реализацию, сохранив при этом эквивалентность последовательной программе по выходным данным. Примером задачи на использование MPI является параллельная реализация игры «Жизнь». Аналогично студентам предоставляются последовательная реализация игры на языке C и программа для генерации входных данных.

Для каждой из задач на параллельное программирование были разработаны наборы тестов, имеющие схожую структуру. Тесты выполняют компиляцию программы, после чего производят несколько запусков программы с различными конфигурациями (числом потоков или процессов), входными данными и другими параметрами. Для каждого запуска проводятся измерения времени выполнения, ускорения и эффективности. Полученные результаты и показатели сравниваются с эталонными значениями. Это позволяет выполнить всестороннюю оценку решения, включая его корректность, производительность, масштабируемость и зависимость от входных данных. На рис. 4 представлен пример результата запуска сервиса тестирования MPI-реализаций игры «Жизнь».

В 2016 году, вместе с включением в программу курса ШАД программирования на гра-

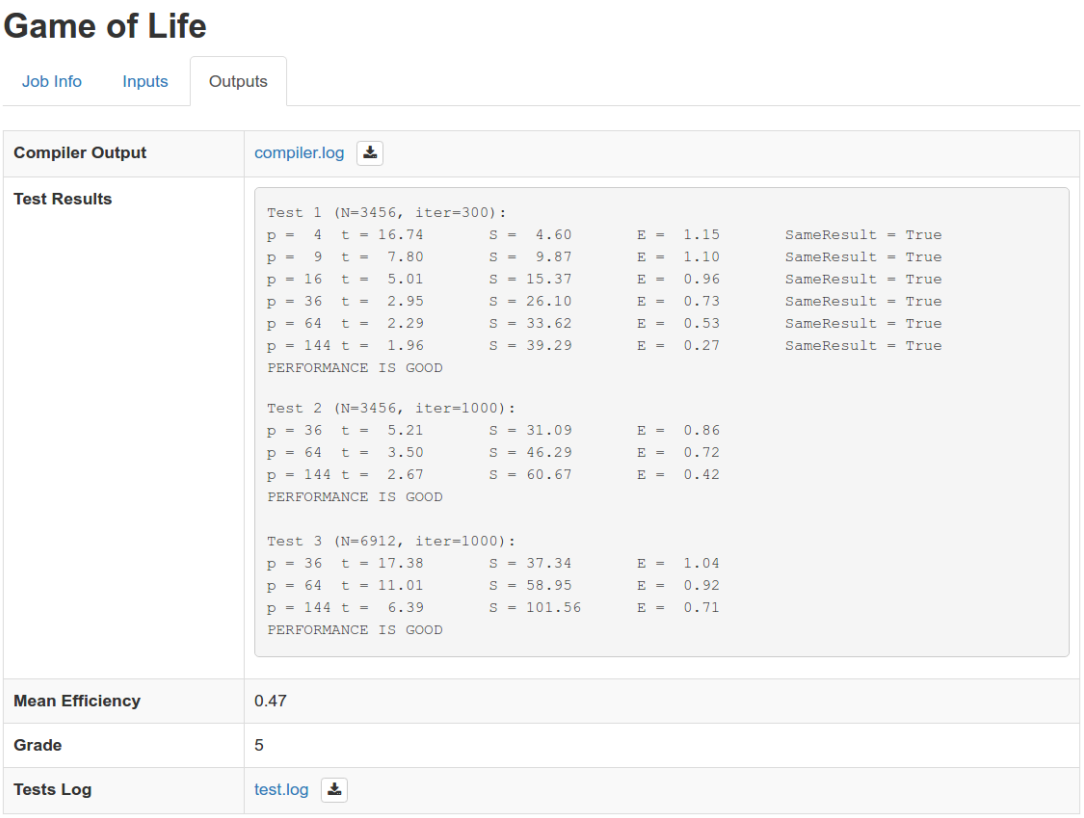

Рис. 4. Пример тестирования параллельной реализации игры «Жизнь» на MPI

фических процессорах, были также созданы сервисы для запуска программ для технологий CUDA и OpenACC на учебном сервере с GPU Nvidia Tesla K40. Особенностью данных сервисов является возможность по выбору выполнить запуск программы в профилировщике nvprof и получить файл с результатами профилирования.

4.3. Распределенная обработка данных

В данном разделе рассматриваются модели распределенных вычислений и платформы для обработки больших объемов данных, активно развивающиеся в течение последнего десятилетия. Студенты знакомятся с моделью программирования MapReduce и ее отрытой реализацией в рамках платформы Apache Hadoop, а также изучают более новую технологию распределенной обработки данных - Apache Spark.

Для запуска на учебном кластере MapReduce-программ были созданы два универсальных сервиса для программ на языках Python и Java соответственно. Оба сервиса позволяют указать при запуске файлы с программой, аргументы командной строки, пути к входным и выходным данным в распределенной файловой системе HDFS, число reduce-задач, а также дополнительные конфигурационные параметры Hadoop. Созданный вспомогательный скрипт осуществляет запуск MapReduce-задания, мониторинг состояния задания и обновление информации о задании, отображаемой в веб-интерфейсе Everest. Во время выполнения программы студенту также передается ссылка на страницу со статусом MapReduce-задания в веб-интерфейсе Hadoop. После завершения программы сервис выводит объем потребленных ресурсов в ядро-секундах и ссылку на историю выполнения задания с доступом к логам отдельных задач. Всё это дает достаточно информации для отладки и оценки эффективности программ.

Аналогичные сервисы были созданы для запуска на учебном кластере Spark-программ на языках Python и Scala/Java. В отличие от сервисов для MapReduce, данные сервисы име-

ют более сложные параметры выполнения, такие как число экзекуторов, число ядер и объем памяти на экзекутор. Также можно указать минимальную долю запущенных экзекуторов, при которой следует начать вычисления. Это позволяет студентам лучше изучить различные компромиссы, возникающие при выборе параметров выполнения. Вспомогательный скрипт для данных сервисов также устроен сложнее. Например, он реализует ограничения на максимальный объем запрашиваемых ресурсов (ядер и памяти) и число одновременно выполняющихся заданий пользователя.

В связи с большими объемами входных данных и, иногда, результатов выполнения программ, необходимо было также предоставить студентам возможность удобного просмотра данных в HDFS, без необходимости их загрузки на свою машину. Для данной цели была использована веб-среда Hue, реализующая набор веб-интерфейсов для работы с платформой Hadoop, включая удобный браузер файлов в HDFS. Отметим, что Hue также включает интерфейс для запуска программ, однако он является более сложным и низкоуровневым в сравнении с созданными сервисами.

Примерами практических задач по даной теме являются построение инвертированного индекса Википедии с помощью MapReduce и анализ графа пользователей Твиттера с использованием Spark. Для каждой из задач были созданы соответствующие сервисы тестирования решений. В отличие от предыдущих примеров, данные сервисы не запускают программы студентов, а только проверяют полученные программой результаты. Поэтому при сдаче задач студентам также требовалось предоставить ссылки на запуски своих программ через описанные выше универсальные сервисы.

5. Заключение

В статье описан практический подход к созданию веб-сервисов для поддержки учебного процесса в рамках курсов по параллельным и распределенным вычислениям. Данный подход основан на использовании платформы Everest, изначально созданной для публикации, запуска и композиции вычислительных сервисов. Опыт применения описанного подхода в рамках двух курсов по ПРВ показывает, что платформа позволяет быстро создавать как универсальные сервисы для запуска различных типов параллельных программ на высокопроизводительных системах, так и сервисы для тестирования решений практических заданий в данной области.

Использование универсальных сервисов для запуска программ на вычислительном кластере позволяет студентам быстро применять полученные знания, не отвлекаясь на изучение работы в командной строке кластера. Удобство запуска программ через подобные сервисы также позволяет включать в занятия больше практических заданий и демонстраций. Сервисы для тестирования решений заданий позволяют обеспечить надежное выполнение тестов, предоставить студентам оперативный и детальный отклик в процессе решения, а также автоматизировать процесс проверки решений.

В сравнении с другими решениями, Everest имеет ряд отличительных особенностей. Платформа реализует модель PaaS и поддерживает работу с ней множества пользователей через удаленные веб- и программный интерфейсы. Последний позволяет интегрировать платформу и размещенные сервисы с внешними системами, например LMS. Пользователи Everest могут подключать к платформе вычислительные ресурсы и связывать их с создаваемыми сервисами. Это позволяет быстро, без необходимости установки платформы, развертывать на её базе произвольные вычислительные веб-сервисы и предоставлять доступ к ним другим пользователям. Публично доступный экземпляр платформы функционирует по адресу [10].

Будучи платформой общего назначения, Everest не реализует ряд высокоуровневых функций, присутствующих в специализированных решениях по запуску и проверке программ. Например, при создании сервиса преподавателю необходимо подготовить вспомога-

тельный скрипт или набор тестов, реализующие все необходимые действия. Давая с одной стороны максимальную гибкость, данный подход требует частого написания повторяющегося кода, например, по запуску заданий на кластере или сравнению результатов программы с эталонными. В будущем планируется учесть данные проблемы путем как реализации дополнительных функций в Everest, так и публикации готовых к использованию шаблонов для быстрого развертывания подобных сервисов другими преподавателями.

Литература

1. Dincer K., Fox G.C. Design Issues in Building Web-based Parallel Programming Environments // The Sixth IEEE International Symposium on High Performance Distributed Computing. IEEE, 1997. P. 283–292.
2. Touriño J., Martín M.J., Tarrío J., Arenaz M. A Grid Portal for an Undergraduate Parallel Programming Course // IEEE Transactions on Education. 2005. Vol. 48, No. 3. P. 391–399.
3. Maggi P., Sisto R. A Grid-powered Framework to Support Courses on Distributed Programming // IEEE Transactions on Education. 2007. Vol. 50, No. 1. P. 27–33.
4. Schlarb M., Hundt C., Schmidt B. SAUCE: A Web-based Automated Assessment Tool for Teaching Parallel Programming // Euro-Par 2015: Parallel Processing Workshops. Springer, 2015. P. 54–65.
5. Nowicki M., Marchwiany M., Szpindler M., Bała P. On-line Service for Teaching Parallel Programming // Euro-Par 2015: Parallel Processing Workshops. Springer, 2015. P. 78–89.
6. Garrity P., Yates T., Brown R., Shoop E. WebMapReduce: An Accessible and Adaptable Tool for Teaching Map-Reduce Computing // Proceedings of the 42nd ACM Technical Symposium on Computer Science Education. ACM, 2011. P. 183–188.
7. Heterogeneous Parallel Programming. https://www.coursera.org/course/hetero
8. Сухорослов О.В. Интеграция вычислительных приложений и распределенных ресурсов на базе облачной программной платформы // Программные системы: теория и приложения: электрон. научн. журн. 2014. Т. 5, № 4(22). С. 171–182.
9. Sukhoroslov O., Volkov S., Afanasiev A. A Web-based Platform for Publication and Distributed Execution of Computing Applications // 14th International Symposium on Parallel and Distributed Computing (ISPDC). 2015. P. 175–184.
10. Everest. http://everest.distcomp.org/
11. Richardson L., Ruby S. RESTful Web Services. O'Reilly, 2008.
12. Сухорослов О.В. Комбинированное использование высокопроизводительных ресурсов и грид-инфраструктур в рамках облачной платформы Everest // Суперкомпьютерные дни в России: Труды международной конференции (28-29 сентября 2015 г., г. Москва). М.: Изд-во МГУ, 2015. С. 706-711.

Using web services for running and testing parallel programs within the framework of the training course*

O.V. Sukhoroslov[1,2]

Institute for Information Transmission Problems of the Russian Academy of Sciences (Kharkevich Institute)[1], National Research University Higher School of Economics[2]

The paper describes the experience of the development and use of web services to support the learning process in courses on parallel and distributed computing. The use of generic services to run programs on a computing cluster enables students to quickly apply the knowledge without having to learn the cluster command line environment. The ease of running programs through these services also allowed to include more practical exercises and demos in the class. In addition to generic services, the problem-specific services have been implemented to test solutions of practical assignments. These services helped to automate the process of testing solutions, and provide students with prompt and detailed response in the process of solving the task. The presented services were deployed on the Everest cloud platform, which has distinguishing features such as the ease of publishing services and the possibility of linking services with external computing resources. These features allow one to quickly reuse the presented approach for teaching similar courses.

Keywords: HPC education, parallel programming, distributed computing, remote access to HPC resources, program testing, web-based interfaces, web services, platform as a service

References

1. Dincer K., Fox G.C. Design Issues in Building Web-based Parallel Programming Environments // The Sixth IEEE International Symposium on High Performance Distributed Computing. IEEE, 1997. P. 283–292.

2. Touriño J., Martín M.J., Tarrío J., Arenaz M. A Grid Portal for an Undergraduate Parallel Programming Course // IEEE Transactions on Education. 2005. Vol. 48, No. 3. P. 391–399.

3. Maggi P., Sisto R. A Grid-powered Framework to Support Courses on Distributed Programming // IEEE Transactions on Education. 2007. Vol. 50, No. 1. P. 27–33.

4. Schlarb M., Hundt C., Schmidt B. SAUCE: A Web-based Automated Assessment Tool for Teaching Parallel Programming // Euro-Par 2015: Parallel Processing Workshops. Springer, 2015. P. 54–65.

5. Nowicki M., Marchwiany M., Szpindler M., Bała P. On-line Service for Teaching Parallel Programming // Euro-Par 2015: Parallel Processing Workshops. Springer, 2015. P. 78–89.

6. Garrity P., Yates T., Brown R., Shoop E. WebMapReduce: An Accessible and Adaptable Tool for Teaching Map-Reduce Computing // Proceedings of the 42nd ACM Technical Symposium on Computer Science Education. ACM, 2011. P. 183–188.

7. Heterogeneous Parallel Programming. https://www.coursera.org/course/hetero

*This work is supported by the Russian Science Foundation (project No. 16-11-10352)

8. O. Sukhoroslov. Integration of Distributed Computing Applications and Resources on the Base of Cloud Platform // Program Systems: Theory and Applications. 2014. Vol. 5, No. 4(22). P. 171–182. (In Russian)

9. Sukhoroslov O., Volkov S., Afanasiev A. A Web-based Platform for Publication and Distributed Execution of Computing Applications // 14th International Symposium on Parallel and Distributed Computing (ISPDC). 2015. P. 175–184.

10. Everest. http://everest.distcomp.org/

11. Richardson L., Ruby S. RESTful Web Services. O'Reilly, 2008.

12. Sukhoroslov O. Combined Use of HPC Resources and Grid Infrastructures with Everest Cloud Platform // Proceedings of the 1st Russian Conference on Supercomputing (Supercomputing Days 2015). 2015. P. 706–711. (In Russian)