Инструменты математического сервиса MathPartner для выполнения параллельных вычислений на кластере

Е.А. Ильченко <ilchenkoea@gmail.com>
Тамбовский государственный университет имени Г.Р. Державина,
392 000, г.Тамбов, ул. Интернациональная, 33

Аннотация. Во многих прикладных областях необходимо выполнять символьно-численные расчеты с данными большого объема. Примерами таких областей являются робототехника, распознавание речи, распознавание графической информации, автоматизация производства и другие. Системы символьных вычислений, их так же называют системами компьютерной алгебры, активно развиваются с конца восьмидесятых годов. Хорошо известными системами являются Mathematica, Maple, Reduce и многие другие. Почти все эти системы не были ориентированы изначально ни на масштабные математические объекты, ни на многопроцессорные кластеры. Система Form является единственным исключением. Эта система была изначально задумана для оперирования объектами, превышающими по размеру оперативную память. Такие объекты размещаются на жестком диске. В статье дается описание алгоритмов для тех инструментов системы компьютерной алгебры MathPartner, которые предназначены для взаимодействия с вычислительным кластером. Приводится описание алгоритма работы сокетного сервера, являющегося связующим звеном между MathPartner и некоторой супер ЭВМ, который обеспечивает исполнение параллельных программ на кластере. Подробно объясняется механизм, который позволяет абстрагироваться от конкретной супер ЭВМ и установленной на нее PBS, работая исключительно с веб-интерфейсом MathPartner. Кроме запуска готовых программ, описываемый сокетный сервер дает возможность запускать пользовательские программы, отправляемые на вычислительный кластер в виде zip-архива через веб-интерфейс. В статье даются примеры использования уже реализованных параллельных алгоритмов, которые входят в состав веб сервиса MathPartner. Некоторые из параллельных программ MathPartner реализованы с помощью парадигмы «DDP» (dynamic decentralized parallelization) – подхода, позволяющего написать эффективную параллельную программу для работы с неоднородными данными, такими как разреженные матрицы. В статье показаны примеры использования DDP-программ, интегрированных в MathPartner.

Ключевые слова: параллельный алгоритм; облачная математика; MathPartner; веб-интерфейс; сокетный сервер.

DOI: 10.15514/ISPRAS-2016-28(3)-11

Для цитирования: Ильченко Е.А. Инструменты математического сервиса MathPartner для выполнения параллельных вычислений на кластере. Труды ИСП РАН, том 28, вып. 3, 2016 г., стр. 173-188. DOI: 10.15514/ISPRAS-2016-28(3)-11

1. Введение

При разработке системы компьютерной алгебры MathPartner ставилась задача обеспечения вычислений с масштабными символьно-численными математическими объектами и проведения вычислений на многопроцессорном вычислительном кластере. Сегодня этот облачный математический сервис свободно доступен на платформе проекта «Университетский кластер» на сайте http://mathpar.cloud.unihub.ru.

Сервис MathPartner содержит библиотеку символьно-численных алгоритмов, написанную на Java, а также пакеты параллельных программ, которые выполняются на вычислительном кластере [5]-[7]. Кроме того, MathPartner предоставляет возможность загружать и исполнять на кластере пользовательские программы, которые были предварительно загружены на него с помощью web-интерфейса.

В данном сообщении описываются алгоритмы взаимодействия web-сервиса MathPartner и вычислительного кластера, который управляется системой PBS.

2. Взаимодействие MathPartner с PBS

Portable Batch System (PBS) – система управления распределенными вычислениями. Основная функция PBS - запуск задач в вычислительной среде. Эта система имеет консольный интерфейс. Как правило, для взаимодействия с ней достаточно двух команд - установка программы в очередь для последующего выполнения и проверка ее состояния, чтобы выяснить, когда программа закончила работу. Команда **`qsub`** используется для установки программы в очередь на выполнение, где **`config`** – это имя файла с настройками запуска. К настраиваемым параметрам относится количество требуемых процессоров, число требуемых узлов, путь к файлам для сохранения потока вывода и потока ошибок и путь к запускаемой программе.

Для взаимодействия с PBS в MathPartner был разработан комплекс программных средств. Управляющий узел кластера с пакетом PBS и сервер, на котором установлен веб-сервис MathPartner, – это разные устройства. Они связываются по сети Интернет. Программа, которая обеспечивает их связь, устанавливается на управляющий узел кластера. С веб-сервисом MathPartner она взаимодействует через сокетное соединение.

Для взаимодействия с PBS Java-программа использует класс **`Runtime`**, в частности его метод **`exec(String[] command)`**. Это программный аналог обычной отправки команды в терминал. Программа «PBSbridge» выполняет следующие функции:

- Запуск уже готовых программ, которые входят в библиотеку MathPartner. Для этого необходимо, чтобы на управляющем узле кластера располагалась копия Java-классов MathPartner, находящихся на веб-сервере.
- Обеспечение возможности запуска пользовательских параллельных программ, написанных на Java, с использованием MPI. При этом должно осуществляться копирование скомпилированных Java-классов на управляющий узел кластера и последующий их запуск. Для этого создан специальный интерфейс, который позволяет производить копирование файлов, запуска программы на выполнение, отслеживание состояния запущенной программы и отображение результатов выполнения.
- Разграничение пользователей друг от друга и запрет доступа к файловой системе кластера. Выполнение задач от разных пользователей осуществляется в разных потоках.

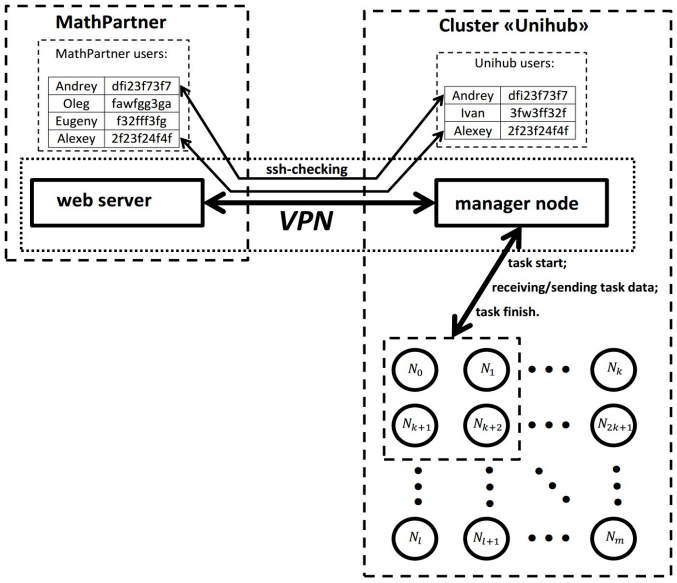

Рис. 1. Общая схема взаимодействия web-сервиса MathPartner и кластера Unihub.

Fig. 1. The general scheme of interaction of MathPartner and Unihub.

Поскольку необходимо разграничить всех пользователей, необходимо обеспечить их регистрацию в системе. Для этого в MathPartner добавлена процедура аутентификации пользователя. Для хранения данных используется внешняя СУБД. После того, как пользователь вошел в систему, он может использовать команды MathPartner для работы с PBSbridge. Вся система работает по принципу запрос-ответ. Пользователь вводит в интерфейс MathPartner команду, после чего соответствующий ей запрос отправляется по сокетному соединению программе PBSbridge. Выполняются действия, соответствующие этому запросу, и результат возвращается по тому же соединению обратно.

Опишем общую схему работы программы PBSbridge. Класс, содержащий **`main`**-метод для запуска, имеет имя **`Server`**. На рис.2 представлен алгоритм метода **`main`**, с которого начинается выполнение программы.

В начале метода **`main`** закрываются потоки ввода-вывода, чтобы предупредить возможность существования чужих открытых потоков. После этого создаются необходимые директории и файл для записи лог-файла. При этом все настройки берутся из констант, которые определены в классе **`AlgorithmsConfig`**. Метод **`writeLog`** служит для записи лог-файла с отчетом. После этого запускается бесконечный цикл, в котором происходит прослушивание порта, указанного в настройках сокета. Если происходит соединение, то для обработки запросов, приходящих по этому соединению, создается новый поток выполнения.

```
//Закрытие потоков ввода-вывода
System.in.close();
System.out.close();
//создание файла, в который будет записываться отчет работы программы
logFile=new FileWriter(AlgorithmsConfig.CNF_DATA_PATH+"/log.txt", false);
//создание серверного сокета
ServerSocket server = new ServerSocket(AlgorithmsConfig.CNF_SERV_PORT);
//создание структуры для хранения состояний задач
taskStates=new TreeMap<Integer,TreeMap<Integer,Integer>>();
writeLog("Server successfully started");
while true do
 | new Server(server.accept());
end
```

*Рис. 2. Алгоритм **`main`**-метода класса **`Server`**.*

*Fig. 2. **`Main`**-method algorithm of class **`Server`**.*

Файлы пользователей хранятся на сервере. Для каждого пользователя создается своя папка, в качестве имени берется идентификатор пользователя (**`id`**) из базы данных. Программа PBSbridge не имеет доступа к базе, поэтому идентификатор пользователя **`id`** приходит вместе со всеми запросами на сервер.

Обеспечивается возможность осуществления одного из двух режимов. Можно либо запускать свои собственные задачи, предварительно загруженные на сервер, либо запускать задачи, являющиеся частью MathPartner.

Для каждой задачи создается своя папка, которая находится внутри папки данного пользователя. Например, они могут располагаться вот так: **`../userX/taskY`**. Перед запуском задачи в папке будут находиться 2 файла: файл с настройками запуска параллельной программы и файл с входными данными для запускаемой программы.

Содержимое файла с настройками запуска описывается спецификацией, которая используется PBS системой. В файле с входными данными хранится массив входных данных в виде сериализованных объектов. После того как программа завершит работу, в этой папке появятся еще 3 файла, которые являются результатами вычислений. Это будет массив сериализованных объектов, содержащих результаты вычислений, файл, содержащий стандартный поток вывода, и файл, содержащий сообщения об ошибках. Содержимое каждого из этих файлов может быть получено с помощью соответствующих команд интерфейса MathPartner.

Поскольку для каждой задачи создается несколько файлов, необходимо каким-то образом позаботиться об их удалении. Это можно сделать следующим образом: для каждой задачи будем запоминать время последнего обращения к ее файлам. Если разница между текущим временем на сервере и пометкой какой-либо задачи больше, чем, например, двое суток, то эти файлы можно считать устаревшими, и они подлежат удалению.

Этот механизм реализован в виде еще одного потока, который раз в сутки перебирает все папки, проверяя даты последнего обращения к ним. Также необходимо удалять пользовательские загруженные файлы. Для них можно использовать механизм, описанный выше, но требуется увеличить продолжительность хранения этих файлов, например, до 30 дней.

Взаимодействие с программой PBSbridge осуществляется с помощью запросов, которые могут идти либо с веб-части, либо от того узла кластера, с которого началось выполнение на кластере текущей задачи.

Необходимо иметь механизм получения входных данных на ведущем узле кластера, которые передаются с управляющего узла, и механизм для отправки результата вычислений на управляющий узел. Для этого служат соответствующие запросы к программе PBSbridge. Формат запросов следующий: сначала идет целое число, являющееся идентификатором запроса, а потом данные. Программа PBSbridge считывает это число, являющееся идентификатором, и, в зависимости от его значения, принимает решение о получении дополнительных данных. Количество этих данных для каждого запроса может быть различно.

Обработка всех запросов происходит в цикле, работа которого завершится только тогда, когда сокетное соединение будет разорвано. На рис. 3 приведен алгоритм такого цикла, где показана реализация обработки запроса

`QS_ADD_TASK`, который служит для запуска программ. Он универсален, потому что используется для запуска как готовых MathPartner-алгоритмов, так и пользовательских загруженных программ. Настройки запуска передаются в виде сериализованного объекта класса **`TaskConfig`**. В случае успешного запуска пользователю возвращается номер, присвоенный текущей задаче при запуске, он будет использоваться для отслеживания состояния задачи и для получения ее результатов.

Непосредственное создание файла с настройками запуска для PBS осуществляется классом **`Launcher`**. Он создает обычный текстовый файл, учитывая все настройки, которые были указаны в запросе от web-части, устанавливает права на исполнение для этого файла и отправляет соответствующую команду для запуска. На рис. 4 приведен алгоритм такого метода.

Параллельная программа может иметь 4 состояния: она может находиться в очереди на выполнение, в процессе выполнения, вычисления программы может быть успешно завершены, работа программы может быть завершена с ошибками.

Далее рассмотрим механизм отслеживания состояния задачи. После того как завершится выполнение метода **`CreateAndRunPBSfile`**, задаче присваивается состояние «в очереди». Для запуска задачи на счет должны быть получены входные данные с управляющего узла кластера. Это происходит по запросу **`QS_GET_DATA_FOR_CALC`**. Задаче присваивается состояние «в процессе выполнения».

```
//инициализация потоков ввода-вывода:
InputStream is = socket.getInputStream();
ObjectInputStream inp=new ObjectInputStream(is);
OutputStream os = socket.getOutputStream();
ObjectOutputStream oos=new ObjectOutputStream(os);
//создание класса, с помощью которого мы будем запускать задачи:
Launcher launcher =new Launcher();
while true do
    //получение идентификатора входящего запроса:
    Integer qType=(Integer)inp.readObject();
    Integer userID;
    //выполнение действий для соответствующего запроса
    switch qType do
        case AlgorithmsConfig.QS_ADD_TASK
            //получение id пользователя
            userID=(Integer)inp.readObject();
            //получение данных и настроек для алгоритма
            Object []data=(Object[])inp.readObject();
            TaskConfig taskConf=(TaskConfig)inp.readObject();
            //номер задачи, который присваивается данной задаче:
            Integer taskNumb=addTaskInMap(userID);
            //запуск задачи
            Integer launchResult=launcher.launch(userID, taskNumb, data, taskConf);
            //если по каким-то причинам задачу не удалось запустить, удаляем записи о ней:
            if launchResult!=AlgorithmsConfig.RES_SUCCESS then
                removeTaskFromMap(userID, taskNumb);
            end
            //возвращение результата запуска задачи и присвоенного ей номера:
            oos.writeObject(launchResult);
            oos.writeObject(taskNumb);
        end
        case AlgorithmsConfig.QS_GET_DATA_FOR_CALC
            //далее действия для обработки этого запроса
            ...
        end
        //далее идет обработка прочих запросов
        ...
    endsw
end
```

Рис. 3. Алгоритм обработки поступающих запросов.

Fig. 3. Algorithm for processing incoming requests.

Возможны 2 варианта: либо задача успешно завершается и результаты пересылаются в PBSbridge с помощью запроса **QS_RECV_RESULT_FOR_TASK_CLUSTER**, либо задача аварийно завершает свою работу и в этом случае она не возвращает результат.


```
//файл, в который мы запишем настройки для PBS:
FileWriter runFile;
String run_file_path = folderPath+"/run";
File dir = new File(run_file_path);
runFile = new FileWriter(dir);
//запись строки настроек в файл запуска:
runFile.append("Здесь строка запуска, учитывающая все необходимые настройки");
//команда для изменение прав доступа к запускающему файлу:
String []chMod="chmod", "777", run_file_path;
Process chmod = Runtime.getRuntime().exec(chMod);
//команда запуска команды qsub:
String []command="/opt/pbs/bin/qsub", run_file_path;
Process qsub = Runtime.getRuntime().exec(command);
```

Рис. 4. Алгоритм создания файла с настройками для PBS и его последующего исполнения системой запуска.

Fig. 4. File creation algorithm with settings for PBS and its execution of the launch system.

Если задача аварийно завершит работу, то результат вычислений не будет отсылаться на сервер, и ее состояние будет по-прежнему «в процессе выполнения». Но в файл, содержащий поток ошибок, будет записан протокол появления исключительной ситуации (Exception), которая привела к остановке выполнения программы. Если задача завершила работу успешно, то файл с сообщением об ошибках будет пустым.

Поэтому при каждом запросе с веб-части о состоянии задачи производится проверка этого файла, и в случае, если он не пуст, то пользователю возвращается соответствующее сообщение, а задаче присваивается статус «аварийное завершение».

3. Параллельные алгоритмы в MathPartner

MathPartner содержит параллельные программы, которые должны выполняться на кластере с использованием web-интерфейса. Часть алгоритмов реализована с использованием парадигмы **DDP** – децентрализованного динамического управления параллельным вычислительным процессом. Эта схема управления позволяет эффективно использовать узлы кластера, даже когда входные данные для алгоритма имеют неоднородную структуру (разреженные матрицы). В табл. 1 представлен перечень имеющихся на данный момент параллельных программ в составе MathPartner.

Для любых параллельных алгоритмов можно производить настройку запуска на кластере. Описание этих настроек приведено в таблице 2. Все переменные здесь могут принимать только целые положительные значения.

На рис. 6 показан пример выполнения оператора **\matMultPar1x8**, который выполняет параллельное блочно-рекурсивное умножение двух матриц. Сначала выполняются настройки всех необходимых переменных для запуска,

затем инициализируются матрицы-сомножители A и B, после этого запускается задача умножения матриц на кластере. Сообщение «`Task ID is 1`» означает, что задача успешно помещена в очередь выполнения на кластере и ее идентификатор равняется единице. Информацию о состоянии задачи можно получить с помощью оператора **`\getStatus`**, передав ему в качестве аргумента полученный **`Task ID`**. Сообщение «`Task is finished`» означает, что задача успешно завершила свою работу. Результат выполнения программы можно получить с помощью команды **`\getCalcResult(taskID)`**.

В том случае, когда входные данные имеют большой объем, в MathPartner предусмотрен файловый импорт и экспорт математических объектов (матрицы, векторы, функции, полиномы). На рис. 7 показан пример матричного умножения, когда входные данные вводятся из файлов пользователя, с последующим сохранением результата в файле. Файлы *`matrixA.txt`* и *`matrixB.txt`* содержат текст «`[[1,2],[3,4]]`» и «`[[4,5],[6,7]]`», соответственно, файл *`matrixC.txt`* содержит результат «`[[16,19],[36,43]]`». Перед тем, как выполнять операциями с файлами их необходимо сначала загрузить на веб-сервер, как показано на рис. 5.

Табл. 1. Параллельные программы в составе MathPartner.

Название алгоритма	Оператор mathpar	Реализовано с помощью **DDP**
Матрицы и полиномы		
Матричное умножение [13]	**`\matMultPar1x8(A,B)`**	да
Вычисление присоединенной матрицы	**`\adjointPar(A)`**	нет
Умножение полиномов	**`\polMultPar(A,B)`**	да
Тропические вычисления		
Решение однородного уравнения Беллмана $Ax=x$ [16]	**`\BellmanEquationPar(A)`**	да
Решение однородного уравнения Беллмана $Ax+b=x$ [16]	**`\BellmanEquationPar(A,b)`**	да
Решение однородного неравенства Беллмана $Ax \leq x$ [16]	**`\BellmanInequalityPar(A)`**	да
Решение однородного неравенства Беллмана $Ax+b \leq x$ [16]	**`\BellmanInequalityPar(A,b)`**	да

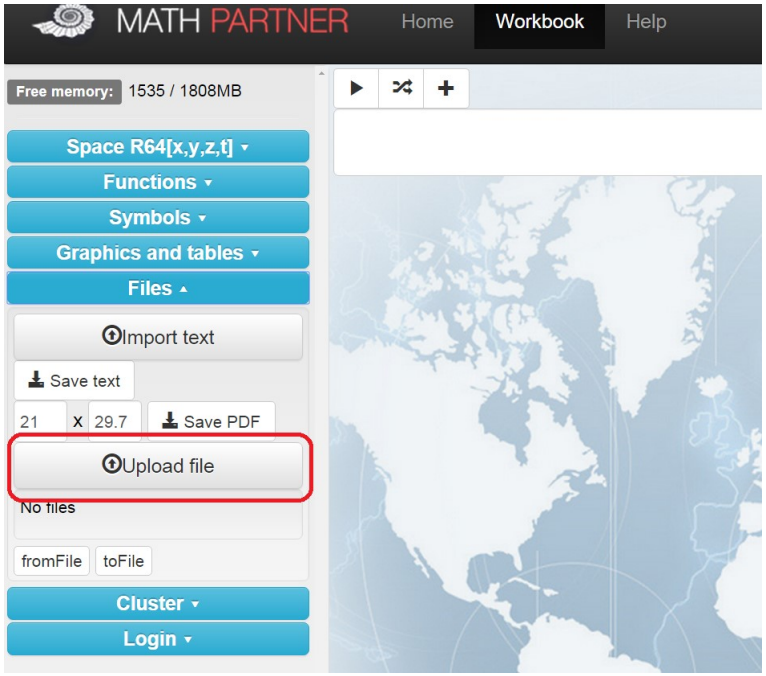

Рис. 5. Загрузка пользовательского файла на сайт.

Fig. 5. Downloading the user file to the site.

Табл. 2. Описание переменных для настроек запуска параллельных программ.

Имя переменной	Назначение
`\TOTALNODES`	Количество узлов кластера, на которых будет запущена программа
`\PROCPERNODES`	Количество программ, которые будут запущены на одном узле кластера (количество MPI-процессов для одного узла)
`\CLUSTERTIME`	Максимальное время работы программы, превысив которое она будет аварийно завершена
`\MAXCLUSTERMEMORY`	Количество памяти, доступное для JVM для одного MPI-процесса

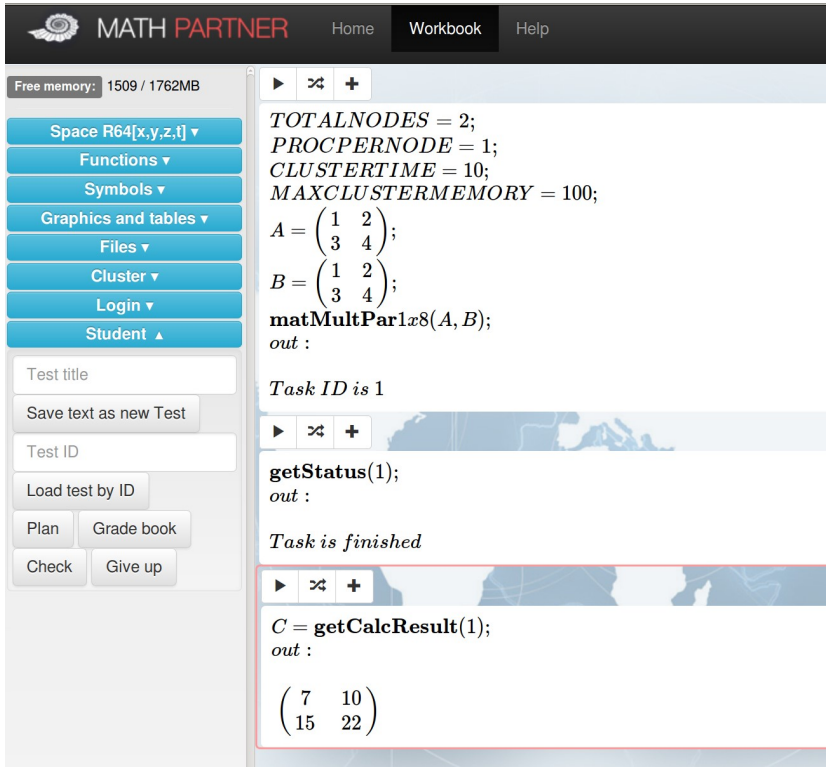

Рис. 6. Пример запуска параллельного матричного умножения с настройками параметров запуска.

Fig. 6. Example of running a parallel matrix multiplication with setting startup options.

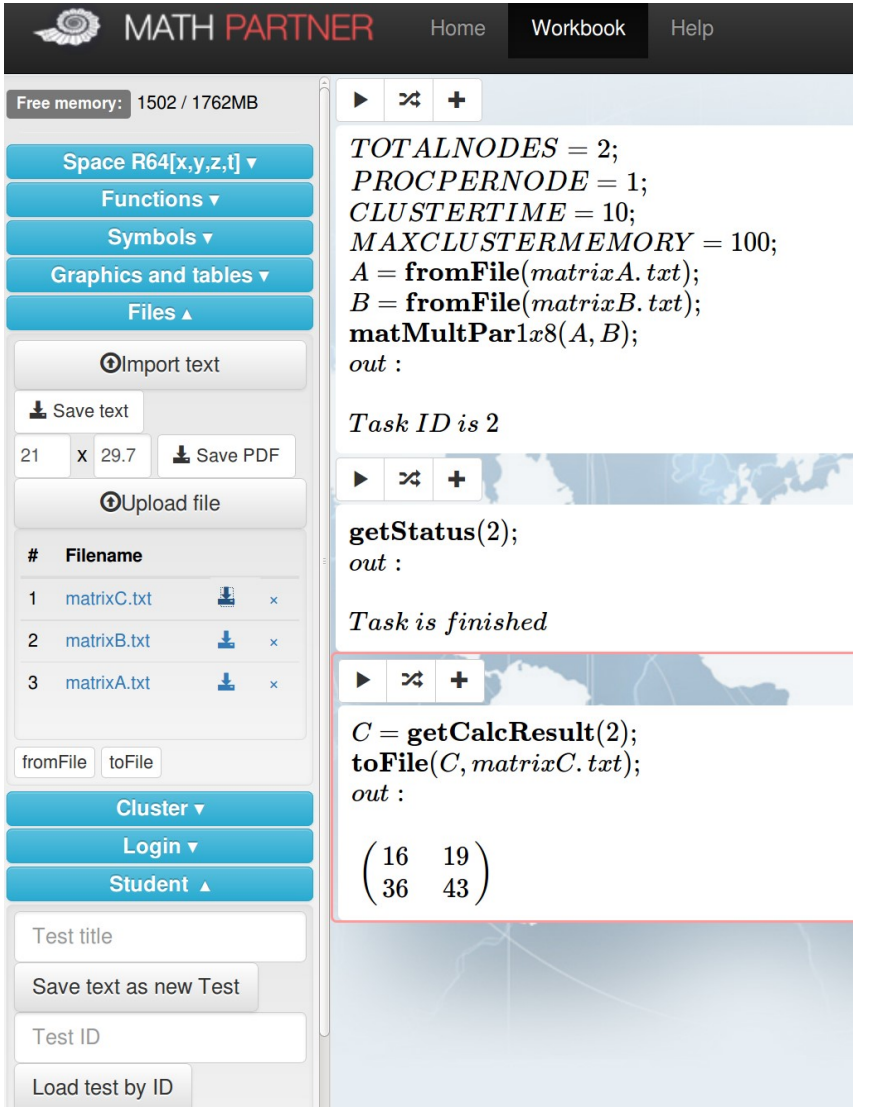

Рис. 7. Пример запуска параллельного матричного умножения с получением исходных данных из файлов.

Fig. 7. Example of running a parallel matrix multiplication where data is taken from the files.

4. Запуск параллельных программ, разработанных пользователем в проекте MathPartner

MathPartner предоставляет удобный интерфейс для запуска на кластере новых параллельных программ, разработанных пользователем в проекте MathPartner на языке Java. Процесс запуска такой параллельной программы состоит из следующих шагов:

- Необходимо создать zip-архив пакета скомпилированных java-классов.
- Выполнить вход в MathPartner.
- Загрузить архив на сайт MathPartner (рис. 5).
- Загрузить архив с программой на кластер с помощью команды **`\uploadToCluster(fileName.zip)`**.
- Поставить программу в очередь на исполнение с помощью команды **`\runUploadedClass(fileName.zip, PackageWithClass.ClassWithMainMethod, param1, param2, ...).`**

После выполнения описанных шагов пользователю будет сообщен идентификатор задачи, с помощью которого будет осуществляться дальнейшая работа. Команда **`\getStatus(taskID)`** позволяет узнать состояние задачи (в очереди, в процессе запуска, завершена, завершена аварийно). Для отображения содержимого потоков вывода и ошибок используются команды **`\getOut(programId)`** и **`\getErr(programId)`**.

5. Заключение

Были изложены алгоритмы взаимодействия веб-сервиса MathPartner с вычислительным кластером, на котором установлена система PBS.

Эти алгоритмы обеспечивают решение двух задач. Одна из них – это выполнение параллельных программ, входящих в состав MathPartner. Другая – это загрузка и выполнение на кластере параллельных программ пользователя, которые разработаны им в проекте MathPartner.

Дальнейшее развитие пакета параллельных программ в проекте MathPartner может в всецело опираться на приведенные в данной работе алгоритмы.

Приведен список параллельных прогамм, входящих в текущую версию системы MathPartner.

Описанные алгоритмы можно применить и для других веб-сервисов, когда требуется обеспечить взаимодействие с кластером, на котором установлена система BPS.

Список литературы

[1]. Strassen V. Gaussian Elimination is not optimal. Numerische Mathematik. 13, 1969, pp. 354-356.

[2]. Малашонок Г.И. Матричные методы вычислений в коммутативных кольцах. Тамбов: Изд-во Тамбовского университета, 2002. 213 с.

[3]. Малашонок Г.И. О вычислении ядра оператора действующего в модуле. Вестник Тамбовского университета. Сер. Естественные и технические науки, 2008, том 13, вып. 1, стр. 129-131.

[4]. Gennadi Malaschonok and Evgeni Ilchenko. Decentralized control of parallel computing. International conference Polynomial Computer Algebra. St.Petersburg, PDMI RAS, 2012, pp. 57-58.

[5]. Бетин А.А. Эксперименты с параллельным алгоритмом вычисления присоединённой матрицы и параллельным умножением файловых матриц. Вестник Тамбовского университета. Сер. Естественные и технические науки, Тамбов, 2010, том 15, вып. 1, стр. 341-345.

[6]. Бетин А.А. Эксперименты с параллельным алгоритмом вычисления присоединённой матрицы. Вестник Тамбовского университета. Сер. Естественные и технические науки, 2010, том 15, вып. 6, стр. 1748-1754.

[7]. Малашонок Г.И. Компьютерная математика для вычислительной сети. Вестник Тамбовского университета. Сер. Естественные и технические науки, 2010, том 15, вып. 1, стр. 322-327.

[8]. Малашонок Г.И. Управление параллельным вычислительным процессом. Вестник Тамбовского университета. Сер. Естественные и технические науки, 2009, том 14, вып. 1, стр. 269-274.

[9]. Малашонок Г.И., Валеев Ю.Д. Организация параллельных вычислений в рекурсивных символьно-численных алгоритмах. Труды конференции ПаВТ'2008 (Санкт-Петербург). Челябинск: Изд-во ЮУрГУ, 2008, стр. 153-165.

[10]. Г.И. Малашонок, Ю.Д. Валеев. Рекурсивное распараллеливание символьно-численных алгоритмов. Вестник Тамбовского университета. Сер. Естественные и технические науки, 2006, том 11, вып. 4, стр. 536-549.

[11]. Г.И. Малашонок, Ю.Д. Валеев. О некоторых подходах к построению параллельных программ. Вестник Тамбовского университета. Сер. Естественные и технические науки, 2005, том 10, вып. 1, стр. 154-156.

[12]. Malaschonok G.I. Effective Matrix Methods in Commutative Domains. Formal Power Series and Algebraic Combinatorics. Berlin: Springer, 2000, pp. 506-517.

[13]. Е.А. Ильченко. Об эффективном методе распараллеливания блочных рекурсивных алгоритмов. Вестник Тамбовского университета. Сер. Естественные и технические науки, 2015, том 20, вып. 5, стр. 1173-1186.

[14]. О.Н. Переславцева. Параллельный алгоритм вычисления характеристического полинома и его временная сложность. Вестник Тамбовского университета. Сер. Естественные и технические науки, 2014, том 19, вып. 2, стр. 530-538.

[15]. Д.С. Ивашов. Параллельный алгоритм разложения многочленов на множители с различными наборами переменных. Вестник Тамбовского университета. Сер. Естественные и технические науки, 2014, том 19, вып. 2, стр. 558-565.

[16]. С.А. Киреев, Г.И. Малашонок. Тропические вычисления в веб-сервисе MathPartner. Вестник Тамбовского университета. Сер. Естественные и технические науки, 2014, том 19, вып. 2, стр. 539-550.

Tools of mathematical service MathPartner for parallel computations on a cluster

E.A. Ilchenko <ilchenkoea@gmail.com>
Tambov State University, Internatsionalnaya, 33, RU-392000, Tambov, Russia

Abstract. In many application areas it is necessary to perform symbolic-numerical calculations with a large volume of data. Examples of such areas are robotics, speech recognition, recognition of graphical information, automation and others. Symbolic computation systems, they also called computer algebra system, actively developed since the late eighties. Well-known systems are Mathematica, Maple, Reduce, and many others. Almost all of these systems were not originally focused any large-scale mathematical objects or on multiprocessor clusters. System FORM is a unique exception. It was conceived as a system which can operate with objects exceeding RAM. Such objects are placed on the hard drive. We give a description of such algorithms of MathPartner web services, which are designed to interact with a computing cluster. We give an algorithm to work a socket server, which is the link between MathPartner and super computers, and which provides the execution of parallel programs on a cluster. We explain in detail the mechanism which abstracts the specific features of super computers and the installed PBS package. The user can run on the cluster or program of MathPartner package, or their own programs. To run its own programs, they are able to send the compiled classes to the computing cluster in a zip-archive through the MathPartner web interface. We show examples of using parallel algorithms included in MathPartner package. Some of MathPartner parallel programs implemented with the paradigm of **DDP** (dynamic decentralized parallelization). DDP is designed as a framework that allows to write efficient parallel program for working with nonhomogeneous data such as sparse matrix. We demonstrate examples of using DDP-programs that are integrated into MathPartner.

Keywords: Parallel algorithm; Cloud mathematics; MathPartner; Web interface; the socket server.

DOI: 10.15514/ISPRAS-2016-28(3)-11

For citation: Ilchenko E.A. Tools of mathematical service MathPartner for parallel computations on a cluster. *Trudy ISP RAN / Proc. ISP RAS*, vol. 28, issue 3, 2016. pp. 173-188 (in Russian). DOI: 10.15514/ISPRAS-2016-28(3)-11.

References

[1]. Strassen V. Gaussian Elimination is not optimal. Numerische Mathematik. 13, 1969, pp. 354-356.
[2]. Malaschonok G. Matrix calculation methods in commutative domains. Tambov: Izd-vo Tambovskogo universiteta [Tambov University publishing], 2002, 213 p. (in Russian).
[3]. Malaschonok G. On computation of kernel of operator acting in a module. Vestnik Tambovskogo universiteta. Ser. Estestvennye i tekhnicheskie nauki [Tambov University Reports. Series: Natural and Technical Sciences], 2008, vol. 13, issue 1, pp. 129-131 (in Russian).
[4]. Gennadi Malaschonok and Evgeni Ilchenko. Decentralized control of parallel computing. International conference Polynomial Computer Algebra. St.Petersburg, PDMI RAS, 2012, pp. 57-58.

[5]. Betin A. Experiments with a parallel algorithm for calculation of adjoint matrix and with a parallel algorithm for multiplication of file matrices. Vestnik Tambovskogo universiteta. Ser. Estestvennye i tekhnicheskie nauki [Tambov University Reports. Series: Natural and Technical Sciences, 2010, vol. 15, issue 1, pp. 341-345 (in Russian).
[6]. Betin A. Experiments with a parallel algorithm for calculation of adjoint matrix. Vestnik Tambovskogo universiteta. Ser. Estestvennye i tekhnicheskie nauki [Tambov University Reports. Series: Natural and Technical Sciences], 2010, vol. 15, issue 6, pp. 1748-1754 (in Russian).
[7]. Malaschonok G. Computer mathematics for computational network. Vestnik Tambovskogo universiteta. Ser. Estestvennye i tekhnicheskie nauki [Tambov University Reports. Series: Natural and Technical Sciences], 2010, vol. 15, issue 1, pp. 322-327 (in Russian).
[8]. Malaschonok G. Managing of the parallel calculative process. Vestnik Tambovskogo universiteta. Ser. Estestvennye i tekhnicheskie nauki [Tambov University Reports. Series: Natural and Technical Sciences], 2009, vol. 14, issue 1, pp. 269-274 (in Russian).
[9]. Malaschonok G., Valeev U. Organization of parallel computations in recursive symbol-numerical algorithms. Trudy konferencii PaVT'2008 [Proceedings of conference PCT'2008] (St. Petersburg). Chelyabinsk: Publishing house SUSU, 2008, pp. 153-165 (in Russian).
[10]. Malaschonok G., Valeev U. Recursive disparallelizing of symbol-numerical algorithms. Vestnik Tambovskogo universiteta. Ser. Estestvennye i tekhnicheskie nauki [Tambov University Reports. Series: Natural and Technical Sciences, 2006, vol. 11, issue. 4, pp. 536-549 (in Russian).
[11]. Malaschonok G., Valeev U. On some approaches to the construction of the parallel program. Vestnik Tambovskogo universiteta. Ser. Estestvennye i tekhnicheskie nauki [Tambov University Reports. Series: Natural and Technical Sciences], 2005, vol. 10, issue 1, pp. 154-156 (in Russian).
[12]. Malaschonok G.I. Effective Matrix Methods in Commutative Domains. Formal Power Series and Algebraic Combinatorics. Berlin: Springer, 2000, pp. 506-517.
[13]. Ilchenko E. About effective methods of parallelizing block recursive algorithms. Vestnik Tambovskogo universiteta. Ser. Estestvennye i tekhnicheskie nauki [Tambov University Reports. Series: Natural and Technical Sciences], 2015, . 20, issue. 5, pp. 1173-1186 (in Russian).
[14]. Pereslavceva O. Parallel algorithm for computing the characteristic polynomials of polynomial matrices and algorithm’s computational time. Vestnik Tambovskogo universiteta. Ser. Estestvennye i tekhnicheskie nauki [Tambov University Reports. Series: Natural and Technical Sciences], 2014, vol. 19, issue 2, pp. 530-538 (in Russian).
[15]. Ivaschov D. Parallel algorithms factorizationof polinomialswith different sets of variables. Vestnik Tambovskogo universiteta. Ser. Estestvennye i tekhnicheskie nauki [Tambov University Reports. Series: Natural and Technical Sciences], 2014, vol. 19, issue 2, pp. 558-565 (in Russian).
[16]. Kireev S., Malaschonok G. Tropical calculations in web service MathPartner. Vestnik Tambovskogo universiteta. Ser. Estestvennye i tekhnicheskie nauki [Tambov University Reports. Series: Natural and Technical Sciences], 2014, vol. 19, issue 2, pp. 539-550 (in Russian).