Методическое пособие по выполнению практического задания по курсу "Суперкомпьютерные технологии"

для студентов 5 курса ВМК
март-апрель 2010 г.

Авторы: В.Ю. Воронов, О.В. Джосан, Н.Н. Попова под руководством Н.Н. Поповой.

Тема: Исследование эффективности параллельных алгоритмов решения систем линейных уравнений на суперкомпьютерных вычислительных системах.

Общая формулировка задания

Исследование алгоритмов решения систем линейных алгебраических уравнений $Ax = b$, где A – вещественная матрица размерности $n \times n$, b– вектор вещественных значений размерности n.

Задание состоит из 2 частей. Первая часть вариантная, вторая часть задания одинаковая для всех.

Часть 1. a) На основе предлагаемой MPI-программы решения системы линейных уравнений реализовать гибридную MPI/OpenMP программу.

b) Провести исследование MPI-программы для систем Regatta p690 и Blue Gene/P.

c) Провести исследование эффективности(ускорения) разработанной гибридной программы на Blue Gene/P, сравнить эффективность MPI-программы с полученной гибридной реализацией.

Метод решения определяется вариантом задания.

Часть 2. Провести исследование эффективности параллельной программы решения системы линейных уравнений, использующей параллельную математическую библиотеку PETSc.

Для выполнения задания предоставляются программы генерации матрицы системы линейных уравнений (см. раздел "Указания к выполнению задания"). Полные описания рассматриваемых алгоритмов можно найти в [3, 1, 2]. Материалы лекции по технологиям параллельного программирования MPI и OpenMP Владимира Александровича Бахтина доступны в [4].

Форма сдачи задания – отчет о вычислительных экспериментах в соответствии с требованиями и разработанная программная реализация. Срок сдачи задания – 5 апреля 2010 г.

Вариант 1. Метод сопряженных градиентов решения систем линейных уравнений

На основе заданной MPI-реализации метода сопряженных градиентов реализовать гибридную MPI-OpenMP реализацию метода. Исходные коды и необходимая информация доступны на платформе pSeries Regatta в папке

`/home/basrav/supercomp2010/assignment_1.tar.gz`

Рассматривается задача решения системы линейных алгебраических уравнений над вещественными числами $Ax = b$, где матрица A размерности n - симметричная положительно определенная. Схема метода сопряженных градиентов представлена на Алгоритме (1).

Реализация метода сопряженных градиентов находится в файле архива `cg.c`. Генератор матрицы системы уравнений находится в файле архива `generator_spd.cpp`.

Алгоритм 1 Схема метода сопряженных градиентов

Require: система уравнений $Ax = b$, порог точности решения τ, максимальное число итераций M, начальное приближение решения x_0

$k = 0$; $r_0 = 0$;

for all пока $||r_k|| = ||b - Ax_k|| > \tau$ и $k < M$ **do**

$k \leftarrow k + 1$

if $k = 1$ **then**

$p_1 \leftarrow r_0$

else

$\beta_k \leftarrow \frac{r_{k-1} r_{k-1}}{r_{k-2} r_{k-2}}$

$p_k \leftarrow r_{k-1} + \beta_k p_{k-1}$

end if

$s_k = Ap_k$

$\alpha_k = \frac{r_{k-1} r_{k-1}}{p_k s_k}$

$x_k \leftarrow x_{k-1} + \alpha_k p_k$

$r_k \leftarrow r_{k-1} - \alpha_k s_k$

end for

return x_k

Вариант 2. Метод сопряженных градиентов с предобусловливателем Якоби

Для заданной MPI-программы метода сопряженных градиентов без предобусловливания реализовать гибридную MPI/OpenMP программу с *диагональным предобусловливанием*. Исходные коды и необходимая информация доступны на платформе pSeries Regatta в папке

`/home/basrav/supercomp2010/assignment_2.tar.gz`

Диагональное предобусловливание заключается в решении методом сопряженных градиентов модифицированной системы уравнений $P^{-1}Ax = P^{-1}b$, где P – диагональная матрица со значениями

$$p_{ij}^{-1} = \begin{cases} \frac{1}{a_{ij}} & \text{если } i = j \\ 0 & \text{если } i \neq j \end{cases} \quad (1)$$

Реализация метода сопряженных градиентов без предобусловливания находится в файле архива `cg.c`. Генератор матрицы системы уравнений находится в файле архива `generator_spd.cpp`. В качестве простейшей реализации выполнить преобразование исходной системы уравнений до рассылки данных по процессорам, т.е. в последовательном варианте. Параллельная реализация предобусловливателя приветствуется.

Вариант 3. Метод Якоби для решения системы линейных уравнений

По заданной MPI-реализации метода Якоби решения системы линейных уравнений разработать гибридную MPI/OpenMP реализацию метода. Исходные коды и необходимая информация доступны на платформе pSeries Regatta в папке

`/home/basrav/supercomp2010/assignment_3.tar.gz`

Краткая схема метода Якоби представлена на Алгоритме (2).

Реализация метода Якоби находится в файле архива `jacobi.c`. Генератор матрицы системы уравнений находится в файле архива `generator_relax.cpp`.

Алгоритм 2 Схема метода Якоби

Require: система уравнений $Ax = b$, порог точности решения τ, максимальное число итераций M, начальное приближение решения x_0, $A = D + R$, где D – матрица из элементов, стоящих на главной диагонали; R –матрица внедиагональных элементов.

$k = 0$; $r_0 = 0$;

for all пока $||r_k|| = ||b - Ax_k|| > \tau$ и $k < M$ **do**

  $k \leftarrow k + 1$

  if $k = 1$ **then**

    $p_1 \leftarrow r_0$

  else

    $x_k \leftarrow D^{-1}(b - Rx_{k-1})$

  end if

end for

return x_k

Вариант 4. Метод Гаусса-Зейделя для решения системы линейных уравнений

По заданной MPI-реализации метода Якоби решения системы линейных уравнений разработать гибридную MPI/OpenMP реализацию метода Гаусса-Зейделя. Исходные коды и необходимая информация доступны на платформе pSeries Regatta в папке
`/home/basrav/supercomp2010/assignment_4.tar.gz` Схема метода Гаусса-Зейделя представлена на Алгоритме (3).

Алгоритм 3 Схема метода Гаусса-Зейделя

Require: система уравнений $Ax = b$, порог точности решения τ, максимальное число итераций M, начальное приближение решения x_0, $A = L + D + U$, где D – матрица из элементов, стоящих на главной диагонали; L –матрица элементов, стоящих под главной диагональю, U – матрица элементов, стоящая над главной диагональю.

$k = 0$; $r_0 = 0$;

for all пока $||r_k|| = ||b - Ax_k|| > \tau$ и $k < M$ **do**

  $k \leftarrow k + 1$

  if $k = 1$ **then**

    $p_1 \leftarrow r_0$

  else

    $x_k \leftarrow (L + D)^{-1}(b - Ux_{k-1})$

  end if

end for

return x_k

Реализация метода Якоби находится в файле архива `jacobi.c`. Генератор матрицы системы уравнений находится в файле архива `generator_relax.cpp`.

Вариант 5. Метод Гаусса для решения системы линейных уравнений

По заданной MPI-реализации метода Гаусса решения системы линейных уравнений разработать гибридную MPI/OpenMP реализацию метода Гаусса. Исходные коды и необходимая информация доступны на платформе pSeries Regatta в папке
`/home/basrav/supercomp2010/assignment_5.tar.gz`

Реализация метода Гаусса находится в файле архива `gauss_elimination.c`. Генератор матрицы системы уравнений находится в файле архива `generator_relax.cpp`. Поскольку метод Гаусса является прямым, требования к отчету по заданию варианта отличаются от других вариантов.

Требования к содержанию отчета

Отчет о выполнении задания должен содержать краткое описание проделанной работы по созданию гибридного кода, краткое описание того, как формулируется задача (какое выбрано решение, начальное приближение, порог точности решения, максимальное число итераций) и включать следующие результаты расчетов:

0) На платформе Regatta: необходимо построить график ускорения MPI-программы для числа процессоров `np = 1, 2, 4, 8` и СЛАУ размерности `N = 1024, 2048, 4096`.

а) По первой части задания (гибридная реализация):

1. Построить таблицу, содержащую время решения, число итераций при решении системы линейных уравнений размерности `N = 1024, 2048, 4096` для числа узлов `np = 1, 2, 4, 8, 16, 32, 64, 128` в режиме `mode = SMP` с 3 нитями. **Для прямого метода Гаусса** в таблице указать вместо числа итераций получаемую степень точности решения.

Таблица 1: Пример таблицы из задания а).1

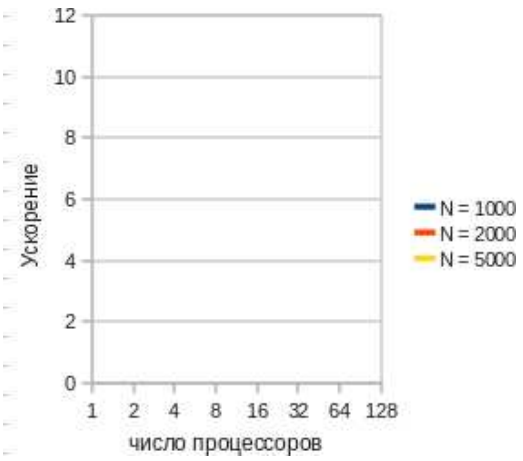
Число узлов n	Размерность СЛАУ N	Время решения T	Ускорение S	Кол-во итераций I
1	1024			
2	1024			
4	1024			
8	1024			
16	1024			
32	1024			
64	1024			
128	1024			
1	2048			
2	2048			
4	2048			
8	2048			
16	2048			
32	2048			
64	2048			
128	2048			
1	4096			
2	4096			
4	4096			
8	4096			
16	4096			
32	4096			
64	4096			
128	4096			

2. Графики ускорения программы для числа узлов `np = 1, 2, 4, 8, 16, 32, 64, 128` для режима выполнения задач `mode = SMP` с тремя OpenMP-нитями для размерности системы уравнений `N = 1024, 2048, 4096`. Ускорение вычисляется по формуле:

$$S(n) = \frac{T(\text{последовательная программа})}{T(n)},$$

где $T(n)$ – время счета программы на заданном числе процессоров.

Рис. 1: Пример графика для задания а).2



3. График значений нормы невязки $||r_n||$ в зависимости от номера итерации для числа узлов `np = 128` и размера системы уравнений `N = 1024, 2048, 4096` для режима выполнения задачи `mode = SMP` с 3 нитями. **Для прямого метода Гаусса** данный график в отчете не требуется.

б) По второй части задания (изучение эффективности алгоритмов из пакета PETSc):

1. Построить графики ускорения метода сопряженных градиентов (без предобусловливания, с диагональным предобусловливателем Якоби, итого 2 графика) для аналогичных размеров задачи `N = 1024, 2048, 4096` и требований к точности решения τ для числа узлов `np = 1, 2, 4, 8, 16, 32, 64, 128` в режиме `mode = VN`. Поскольку рассматриваемые системы уравнений генерируются псевдо-случайно, сходимость методов для заданной точности решения не всегда достигается. В случае отсутствия сходимости построить графики ускорения для одинакового значения максимального числа итераций (например, 50 итераций).

Указания к выполнению задания

1. Для исследования эффективности программы предоставлен генератор псевдо-случайных симметричных положительно определенных матриц (для метода сопряженных градиентов) и матриц особого вида для остальных методов. Файлы с исходными текстами генераторов `generator_spd.c` и `generator_relax.c` соотвественно находится в архиве с заданием. Распаковать архив можно следующими командами:

```
>gunzip assignment_N.tar.gz
>tar -xvf assignment_N.tar
```

Использование генератора:

```
>xlC generator_spd.cpp -o generator_spd # для первого генератора
>xlC generator_relax.cpp -o generator_relax # для второго генератора
>./generator_spd N output_file
>./generator_relax N output_file
```

где `N` – размерность генерируемой матрицы, `output_file` – файл, в который будет сохранена матрица. **Обратите внимание**, что в MPI-реализациях задания предполагается, что размерность матрицы должна нацело делиться на число процессоров при запуске программы.

2. Создание вектора правой части b предлагается выполнять из известного решения x путем умножения $b = Ax$. В предлагаемых MPI-реализациях функция `double* GenerateSolution(int size)` создает решение x, по которому вычисляется вектор правой части. При необходимости изменить решение системы уравнений изменения следует вносить в данную функцию. **Предполагается, что задание будет выполнено для решения, отличного от реализованного в функции по умолчанию.**

3. Глобальные константы `EPSILON`,`MAX_ITERATION` задают требуемый порог нормы невязки для останова итерационного процесса и максимальное число итераций соответственно. **Предполагается, что задание будет выполнено для параметров, отличных от значений по умолчанию.**

4. Начальное приближение решения x_0 задается в функции `double* GenerateInitialSolution(int size)`, все изменения следует вносить в данную функцию. **Предполагается, что задание будет выполнено для начального приближения, отличного от заданного в функции по умолчанию.**

5. В силу псевдо-случайной генерации матрицы системы уравнений, сходимость итерационных методов с заданной степенью точности может не достигаться. Для получения результатов и формирования отчета предлагается пользоваться двумя основными приемами. Во-первых, скорректировать начальное приближение x^0 к решению x так, чтобы оба вектора находились достаточно близко в смысле нормы. Это может уменьшить необходимое число итераций. Во-вторых, в случае если сходимость не достигается или не улучшается, предлагается зафиксировать в вычислитеьных экспериментах максимальное число итераций разумным значением и строить графики ускорения решения на основе полученного результата.

Указания по работе с программой решения систем уравнений на основе пакета PETSc

Для решения систем линейных уравнений на основе пакета PETSc доступна простая реализация. В архиве с заданием находится `make_petsc` и исполняемый код `petsc_linsol.c`.

Для работы с программой необходимо выполнить несколько шагов.

```
>export PETSC_DIR=/home/basrav/petsc-2.3.3-p15
>export PETSC_ARCH=bgp-production
```

После этого можно компилировать код

```
> make -f make_petsc petsc_linsol
```

Код работает с файлами матрицы системы уравнений, которые получены с помощью описанного выше генератора.

Выбор методов решения системы уравнений в программе может выполняться с использованием параметров командной строки. Пример параметров командной строки (параметры команды **mpisubmit.bg** для краткости не указаны) для метода сопряженных градиентов без предобусловливания, порог точности решения $\tau = 1.0e-05$, максимальное число итераций метода 100

```
mpisubmit.bg ... petsc_linsol -- -f matrix -ksp_type cg
-pc_type none -atol 1.0e-05 -rtol 0.0 -ksp_max_iter 100
```

Пример параметров командной строки для метода сопряженных градиентов с предобусловливателем Якоби, порог точности решения $\tau = 1.0e-05$, максимальное число итераций метода 100

```
mpisubmit.bg ... petsc_linsol -- -f matrix -ksp_type cg
-pc_type jacobi -atol 1.0e-05 -rtol 0.0 -ksp_max_iter 100
```

Для вывода нормы невязки на каждой итерации решения использовать ключ `-ksp_monitor`.
Для вывода информации о производительности программы использовать ключ `-log_summary`.

Запуск задач на Blue Gene/P

Детальная информация по компиляции, запуску задач, отладке и анализу производительности находится на сайте `http://hpc.cmc.msu.ru`. Приведем схему основных шагов, которые позволят быстро начать работу.

1. Компиляция Си-программы, содержащей только MPI-вызовы

   ```
   >mpixlc filename.c -o outfile
   ```

2. Компиляция Си-программы с MPI/OpenMP

   ```
   >mpixlc_r -qsmp=omp filename.c -o outfile
   ```

   При реализации на языке C++ MPI или MPI/OpenMP кода следует использовать компилятор `mpixlC` или `mpixlC_r` соответственно.

3. Копировать полученный исполняемый файл на параллельную файловую систему (запуск заданий выполняется только с нее)

   ```
   >cp outfile /gpfs/data/username/userfolder
   >cd /gpfs/data/username/userfolder
   ```

4. Постановка задачи в очередь с помощью скрипта `mpisubmit.bg`. Для выполнения заданий практикума зарезервированы окна времени, в течение которых поставленные в очередь задачи будут работать. Чтобы узнать информацию об имеющихся окнах нужно выполнить команду `llqres`, пример:

   ```
   >llqres
   ID                      Owner      ST Start Time  Duration #Nodes #BG C-nodes
   ----------------------- ---------- -- ----------- -------- ------ -----------
   fen1.23.r               loadl      W   3/15 17:55      220      0         512
   fen1.22.r               loadl      W   3/14 20:55      190      0         512
   fen1.21.r               loadl      W   3/15 12:10      105      0         512
   ```

В данном примере указано, что на платформе доступно 3 окна времени (указаны время начала каждого окна, длительность, число процессорных узлов). В первой колонке указан идентификатор окна, который необходимо использовать с командой `mpisubmit.bg` для того, чтобы задача выполнялась в зарезервированном окне. Для этого нужно устанавливать при запуске задачи переменную окружения `LD_RES_ID` значение идентификатора окна времени. Более подробную информацию по выделенным окнам времени см. по ссылке `http://hpc.cmc.msu.ru/bgp/jobs/reservations`. **Внимание** Если не использовать параметр окна времени, то задачи никогда не будут выполняться.

Пример запуска MPI-задачи `outfile` на 64 узлах в режиме выполнения `VN` и максимальным лимитом времени выполнения 30 минут, параметры коммандной строки программы `parameter1 parameter2`:

```
>LL_RES_ID=fen1.23.r mpisubmit.bg -n 64 -m vn
-w 00:30:00 outfile -- parameter1 parameter2
```

Пример запуска MPI/OpenMP задачи `outfile` на 16 узлах в режиме выполнения `SMP` и запуском для каждого MPI-процесса трех OpenMP-нитей, максимальным лимитом времени выполнения 30 минут, параметры коммандной строки программы `parameter1 parameter2`:

```
>LL_RES_ID=fen1.23.r mpisubmit.bg -n 16 -m smp -e "OMP_NUM_THREADS=2"
-w 00:30:00 outfile  -- parameter1 parameter2
```

5. Просмотр статуса задачи в очереди:

```
>llq
```

6. После завершения вычислительных экспериментов необходимо удалять исполняемый файл из раздела `/gpfs/data/username`, удалять из него входные данные и все выходные файлы переносить в домашнюю папку. Раздел `/gpfs/data/username` должен служить только для запуска задач и необходимых для этого файлов с входными данными.

Список литературы

[1] G.H. Golub and C.F. Van Loan. *Matrix computations*. Johns Hopkins Univ Pr, 1996.

[2] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial Mathematics, 2003.

[3] Н. С. Бахвалов, Н. П. Жидков, and Г. М. Кобельков. *Численные методы*. М: Наука, 1987.

[4] В. А. Бахтин. Курс лекций по программированию на mpi/openmp. 2010. URL `ftp://ftp.keldysh.ru/K_student/MSU2010/MSU2010_MPI_OpenMP.pdf`.