Технологии разработки паралелльных программ с использованием параллельных математических библиотек

Василий Воронов

Московский Государственный Университет
факультет Вычислительной математики и кибернетики

27 октября 2009 г.

План курса

1. Математическая библиотека PETSc
   - Обзор функциональности PETSc
   - Установка
2. Матрично-векторные операции в PETSc
3. Case study: Решение систем линейных уравнений
4. Отладка и профилирование параллельных программ
5. Решение обыкновенных дифференциальных уравнений
   - Case study: решение ОДУ
   - Case study: эффективное управление памятью
6. Решение нелинейных уравнений
   - Case study: решение уравнения Брату
7. Пакет SLEPc: решение задач на собственные значения
   - Case study: решение обобщенной задачи на собственные значения
   - Case study: частичное сингулярное разложение

О чем этот курс?

- На типичных примерах продемонстрировать жизненный цикл программы, разработанной на основе функциональности параллельных математических библиотек PETSc, SLEPc
- Указать особенности, не вошедшие либо слабо освещенные в официальной документации
- Целевая платформа – IBM BlueGene/P
- Формат занятий: лекция + демонстрация работы с программами на массивно-параллельной платформе

План курса

1. Математическая библиотека PETSc
   - Обзор функциональности PETSc
   - Установка
2. Матрично-векторные операции в PETSc
3. Case study: Решение систем линейных уравнений
4. Отладка и профилирование параллельных программ
5. Решение обыкновенных дифференциальных уравнений
   - Case study: решение ОДУ
   - Case study: эффективное управление памятью
6. Решение нелинейных уравнений
   - Case study: решение уравнения Брату
7. Пакет SLEPc: решение задач на собственные значения
   - Case study: решение обобщенной задачи на собственные значения
   - Case study: частичное сингулярное разложение

Общие сведения о пакете PETSc

PETSc [4]– пакет для параллельного решения задач линейной алгебры, дифференциальных уравнений, нелинейных уравнений.
Документация:

- Официальный manual
  `http://www.mcs.anl.gov/petsc/petsc-as/snapshots/petsc-current/docs/manu`
- Tutorials
  `http://www.mcs.anl.gov/petsc/petsc-as/documentation/tutorials/index.html`
  (обратите внимание на слайды 2008 и 2009 годов.)
- Онлайн-документация по функциональности с примерами программ `http://www.mcs.anl.gov/petsc/petsc-as/documentation/index.html`

Структура пакета PETSc

Подробнее о компонентах PETSc

Таблица:

Модуль	Назначение
Mat	Структуры данных различных типов матриц, матрично-векторные операции
Vec	Структуры данных различных типов вектора, векторные операции
DM	Компонента управления данными, поддерживает распределенные структуры данных, обобщенные индексы и механизмы соответствия локальной и глобальной нумерации данных
KSP	Итерационный решатель СЛАУ
PC	Переобусловливатель СЛАУ
TS	Решатель ОДУ
SNES	Решатель нелинейного уравнения
LIBPETSC	Общие компоненты PETSc, организующие ввод-вывод данных, профилирование приложений

Прикладные пакеты, разработанные на основе PETSc

- Пакет Toolkit for Advanced Optimization (TAO)
- Scalable Library for Eigenvalue Problems (SLEPC)
- Prometheus - scalable unstructured finite element solver
- freeCFD - general purpose CFD solver
- OpenFVM - finite volume based CFD solver
- OOFEM - object oriented finite element library
- libMesh - adaptive finite element library
- DEAL.II - sophisticated C++ based finite element simulation package

Пакеты доступны по ссылкеhttp://www.mcs.anl.gov/petsc/petsc-as/miscellaneous/external.html

Установка PETSc

Версии PETSc доступны по ссылке `http://www.mcs.anl.gov/petsc/petsc-as/download/index.html`, последняя версия `http://ftp.mcs.anl.gov/pub/petsc/release-snapshots/petsc-3.0.0-p8.tar.gz`.
На linux-like платформах PETSc входит в дистрибутивы, пример установки для Debian:

```
>sudo apt-get install libpetsc2.3.3 libpetsc2.3.3-dbg
 petsc2.3.3-doc
```

После скачивания необходимо найти и модифицировать конфигурационный файл, примеры файлов в папке $PETSC_DIR/config.
Установка:

```
>export PETSC_DIR=./petsc-2.3.3;
>export PETSC_ARCH=production;
>export BOPT=O_++;
>$PETSC_DIR/config/bluegenep.py
>$PETSC_DIR/make all tests -j 2
```

Простейшая программа на PETSc

```
>cat hello.c
#include "petsc.h"
int main( int argc, char *argv[] ) {
    int rank;
    PetscInitialize( &argc, &argv, 0, 0 );
    MPI_Comm_rank( PETSC_COMM_WORLD, &rank );
    PetscPrintf("Hello World from rank %d\n", rank );
    PetscSynchronizedFlush( PETSC_COMM_WORLD );
    PetscFinalize();
    return 0;
}
```

Простейшая программа на PETSc-II

Компиляция и запуск с помощью Makefile

```
>cat Makefile
SHELL = /bin/bash
EXAMPLE = hello
include $(PETSC_DIR)/bmake/common/base
EXECS = hello
hello: hello.o chkopts
$(CLINKER) -o hello hello.o $(PETSC_LIB)
${RM} -r hello.o
>make hello
```

Дополнительные возможности

- Интерфейсы к функциональности PETSc доступны из C/C++, Fortran, Python
- Средства экспорта-импорта данных с Matlab, конвертеры матриц в форматы MatrixMarket, Harwell-Boeing
- Средства визуализации данных (вывод портретов матриц, построение графиков и гистограм a la упрощенный gnuplot)
- Механизм создания собственного метода

План курса

1. Математическая библиотека PETSc
   - Обзор функциональности PETSc
   - Установка
2. Матрично-векторные операции в PETSc
3. Case study: Решение систем линейных уравнений
4. Отладка и профилирование параллельных программ
5. Решение обыкновенных дифференциальных уравнений
   - Case study: решение ОДУ
   - Case study: эффективное управление памятью
6. Решение нелинейных уравнений
   - Case study: решение уравнения Брату
7. Пакет SLEPc: решение задач на собственные значения
   - Case study: решение обобщенной задачи на собственные значения
   - Case study: частичное сингулярное разложение

Обзор функциональности

Операции над матрицами и векторами с помощью функциональности компонент Mat и Vec–основа программ PETSc. Типы поддерживаемых матриц (последовательных и параллельных):

- `MATSEQAIJ/MATMPIAIJ`–разреженная матрица общего вида
- `MATSEQBAIJ/MATMPIBAIJ`–разреженная матрица блочной структуры
- `MATSEQDENSE/MATMPIDENSE`–плотная матрица

Матрица и вектор в PETSc определяются интерфейсом Основные функции:

- `Mat*Create()`, `Mat*Destroy()`, `MatSetFromOptions()`–создание, удаление, настройка типа матрицы
- (MatSetValues(), MatInsertValues(), MatAssemblyBegin(), MatAssemblyEnd())–добавление/модификация значений матриц
- `MatMultAdd()`, `MatTranspose()`, `...`–матрично-векторные операции разных типов
- `VecLoad()`, `MatLoad()`, `MatView()`, `VecView()`–ввод-вывод матрицы и вектора на диск

Пример: умножение матрицы на вектор $y = Ax$

```
MatCreate(PETSC_COMM_WORLD, A);CHKERRQ(ierr);
VecCreate(PETSC_COMM_WORLD, x);CHKERRQ(ierr);
VecCreate(PETSC_COMM_WORLD, y);CHKERRQ(ierr);
...
ierr = MatMult(A, x, y);CHKERRQ(ierr);
```

План курса

1. Математическая библиотека PETSc
   - Обзор функциональности PETSc
   - Установка
2. Матрично-векторные операции в PETSc
3. Case study: Решение систем линейных уравнений
4. Отладка и профилирование параллельных программ
5. Решение обыкновенных дифференциальных уравнений
   - Case study: решение ОДУ
   - Case study: эффективное управление памятью
6. Решение нелинейных уравнений
   - Case study: решение уравнения Брату
7. Пакет SLEPc: решение задач на собственные значения
   - Case study: решение обобщенной задачи на собственные значения
   - Case study: частичное сингулярное разложение

Решение СЛАУ

- Решение СЛАУ является вычислительным ядром широкого класса научных и инженерных алгоритмов.
- Классическая постановка задачи $Ax = b, A \in R^{m,n}; x \in R^{n,1}, b \in R^{m,1}$.
- В PETSc доступен следующий набор алгоритмов решения СЛАУ: `http://www.mcs.anl.gov/petsc/petsc-as/documentation/linearsolvertable.ht`
- Параметры командной строки позволяют выбирать тип итерационного решателя (`-ksp_type <тип>`) и предобусловливателя (`-pc_type <тип>`), а также модифицировать их параметры

Демонстрация примера. Пример–решение системы уравнений с трехдиагональной матрицей.

План курса

1. Математическая библиотека PETSc
   - Обзор функциональности PETSc
   - Установка
2. Матрично-векторные операции в PETSc
3. Case study: Решение систем линейных уравнений
4. Отладка и профилирование параллельных программ
5. Решение обыкновенных дифференциальных уравнений
   - Case study: решение ОДУ
   - Case study: эффективное управление памятью
6. Решение нелинейных уравнений
   - Case study: решение уравнения Брату
7. Пакет SLEPc: решение задач на собственные значения
   - Case study: решение обобщенной задачи на собственные значения
   - Case study: частичное сингулярное разложение

Отладка программ

В PETSc встроены мощные средства отладки программ.

- Практически все функции PETSc поддерживают синтаксис `ierr = Function(...);CHKERRQ(ierr);`
- В случае ошибки выполнения выводится стек вызовов функций и типичные причины
- Возможность подключить отладчик к отдельному процессу/процессам (vargrind)
- Использовать параметры командной строки `-start_in_debugger`, `-on_error_attach_debugger`

Демонстрация примера.

Профилирование программы

- Параметр командной строки `-log_summary` используется для получения детальной информации о производительности
- Для разбивки информации для различных регионов программы использовать функции `PreLoadBegin()`, `PreLoadStage()`, `PreLoadStage()`,..., `PreLoadEnd()`
- Функциональность для отдельных событий `PetscLog()`...

Демонстрация примера.

План курса

1. Математическая библиотека PETSc
   - Обзор функциональности PETSc
   - Установка
2. Матрично-векторные операции в PETSc
3. Case study: Решение систем линейных уравнений
4. Отладка и профилирование параллельных программ
5. Решение обыкновенных дифференциальных уравнений
   - Case study: решение ОДУ
   - Case study: эффективное управление памятью
6. Решение нелинейных уравнений
   - Case study: решение уравнения Брату
7. Пакет SLEPc: решение задач на собственные значения
   - Case study: решение обобщенной задачи на собственные значения
   - Case study: частичное сингулярное разложение

Постановка задачи

Модуль TS предоставляет функциональность для решения дифференциального уравнения

$$\frac{du}{dt} = F(u, t)$$

где функция F может быть общего вида, а также двух специальных видов

$$\frac{du}{dt} = Au,$$
$$\frac{du}{dt} = A(t)u.$$

Функциональность TS:

- `TSCreate()`, `TSDestroy()`, `TSSetFromOptions()`–создать/уничтожить объект TS
- `TSSetProblemType()`, `TSSetMatrices()`–установить тип решаемой проблемы, установить матрицы ОДУ
- `TSSetUp()`, `TSSetInitialTimeStep()`, `TSSetDuration()`, `TSStep()`–установка параметров решателя и запуск
- `TSGetTimeStep()`, `TSSetTimeStep()`–функции для реализации псевдо-шаговых методов

Параметры командной строки:`-ts_max_steps`, `-ts_max_time`,

Пример кода программы

```
int main(int argc, char **argv) {
 PetscInt m, time_steps_max;
 PetscScalar time_total_max, steps, ftime;
 Mat A; Vec u; TS ts;
  PetscInitialize(argc, argv, 0, 0);
  ierr = MatCreate(PETSC_COMM_WORLD,&A);CHKERRQ(ierr);
  ierr = MatSetSizes(A,PETSC_DECIDE,PETSC_DECIDE,m,m);CHKERRQ(ierr); MatSetI
  # инициализировать значения матрицы
  ierr = SetupMatrix(A);CHKERRQ(ierr);
  ierr = VecCreate(PETSC_COMM_WORLD, &u);
  ierr = VecSetSize(u,PETSC_DECIDE, m); VecSetFromOptions(u);
  ierr = TSCreate(PETSC_COMM_SELF,&ts);CHKERRQ(ierr);
  # тип задачи u_t = A u с постоянной матрицей
  ierr = TSSetProblemType(ts,TS_LINEAR);CHKERRQ(ierr);
  ierr = TSSetMatrices(ts,A,PETSC_NULL,PETSC_NULL,
    PETSC_NULL,DIFFERENT_NONZERO_PATTERN,&appctx);CHKERRQ(ierr);
  ierr = TSSetInitialTimeStep(ts,0.0,dt);CHKERRQ(ierr);
  ierr = TSSetSolution(ts,u);CHKERRQ(ierr);
  ierr = TSSetDuration(ts,time_steps_max,time_total_max);CHKERRQ(ierr);
  ierr = TSSetFromOptions(ts);
  ierr = TSStep(ts,&steps,&ftime);CHKERRQ(ierr);
  MatDestroy(A);
```

Проблемы при использовании структур памяти

- Операции, связанные с управлением динамической памятью в PETSC, приводят к параличу производительности
- Пример: сложение разреженных матриц $A + B$. Необходимо избегать операций реаллокации.
- Идея: в один проход посчитать сколько нужно в каждой строчке элементов и динамически создать нужную структуру матрицы
- Функции: `Mat***SetPreallocation()`, `Mat***SetPreallocationCSR()`

Демонстрация примера.

План курса

1. Математическая библиотека PETSc
   - Обзор функциональности PETSc
   - Установка
2. Матрично-векторные операции в PETSc
3. Case study: Решение систем линейных уравнений
4. Отладка и профилирование параллельных программ
5. Решение обыкновенных дифференциальных уравнений
   - Case study: решение ОДУ
   - Case study: эффективное управление памятью
6. Решение нелинейных уравнений
   - Case study: решение уравнения Брату
7. Пакет SLEPc: решение задач на собственные значения
   - Case study: решение обобщенной задачи на собственные значения
   - Case study: частичное сингулярное разложение

Нелинейные уравнения в PETSc

Библиотека PETSc содержит модуль SNES решателей нелинейных уравнений вида

$$F(u) = 0.$$

В общем виде решение задачи осуществляется в виде итераций

$$u^{(k+1)} = u^{(k)} + \frac{F(u^{(k)})}{F'(u^{(k)})},$$

где $F'(u)$–якобиан функции. Вычисление якобиана является вычислительно сложной задачей, для решения которой могут использоваться несколько методов.

Постановка задачи

Рассмотрим некоторую область $D' = D \cup \Gamma$, в которой определено уравнение

$$\Delta u - \lambda e^u = 0,$$
$$u\,|_\Gamma = 0.$$

Численное решение задачи осуществляется переходом к схеме

$$Au_{ij}^{n+1} = f(u_{ij}^n),$$
$$A = [D_{+x} + D_{-x} + D_{+y} + D_{-y}],$$
$$f(u_{ij}^n) = e^{u_{ij}^n}.$$

Демонстрация примера.

План курса

1. Математическая библиотека PETSc
   - Обзор функциональности PETSc
   - Установка
2. Матрично-векторные операции в PETSc
3. Case study: Решение систем линейных уравнений
4. Отладка и профилирование параллельных программ
5. Решение обыкновенных дифференциальных уравнений
   - Case study: решение ОДУ
   - Case study: эффективное управление памятью
6. Решение нелинейных уравнений
   - Case study: решение уравнения Брату
7. Пакет SLEPc: решение задач на собственные значения
   - Case study: решение обобщенной задачи на собственные значения
   - Case study: частичное сингулярное разложение

Общие сведения о пакете

- SLEPC [6] является высокоуровневой параллельной математической библиотекой решения задач на собственные значения, спектральных преобразований, сингулярного разложения.
- SLEPC построен на использовании функциональности PETSc, соответственно внутри программы имеется возможность работать с объектами и функциями PETSc.
- Документация доступна по ссылке `http://www.grycap.upv.es/slepc/documentation/manual.htm` и включает manual, описание функций и примеров работы с библиотекой

Установка

```
>wget http://www.grycap.upv.es/slepc/download/distrib/slepc-3.0.0-p6.tgz
>tar xvzf slepc-3.0.0-p6
>export SLEPC_DIR=slepc-3.0.0-p6 # подставить нужную директорию
>./configure
> make test
```

На некоторых платформах потребуется установить полный пакет LAPACK.

Постановка задачи

Рассматривается проблема вычисления λ, x в задаче

$$Ax = \lambda Bx$$

Пример кода для задачи $Ax = \lambda x$, оператор A загружается из файла:

```
...
ierr = PetscViewerBinaryOpen(PETSC_COMM_WORLD,filename,FILE_MODE_READ,&viewe
ierr = MatLoad(viewer,MATAIJ,&A);CHKERRQ(ierr);
ierr = PetscViewerDestroy(viewer);CHKERRQ(ierr);
...
ierr = EPSCreate(PETSC_COMM_WORLD,&eps);CHKERRQ(ierr);
ierr = EPSSetOperators(eps,A,PETSC_NULL);CHKERRQ(ierr);
ierr = EPSSetFromOptions(eps);CHKERRQ(ierr);
ierr = EPSSolve(eps);CHKERRQ(ierr);
ierr = EPSGetIterationNumber(eps, &its);CHKERRQ(ierr);
ierr = PetscPrintf(PETSC_COMM_WORLD," Number of iterations of the method: %
ierr = EPSGetType(eps,&type);CHKERRQ(ierr);
ierr = PetscPrintf(PETSC_COMM_WORLD," Solution method: %s\n\n",type);CHKERR
ierr = EPSGetDimensions(eps,&nev,PETSC_NULL);CHKERRQ(ierr);
ierr = PetscPrintf(PETSC_COMM_WORLD," Number of requested eigenvalues: %d\n
ierr = EPSDestroy(eps);CHKERRQ(ierr);
ierr = MatDestroy(A);CHKERRQ(ierr);
ierr = SlepcFinalize();CHKERRQ(ierr);
```

Постановка задачи

Алгоритм сингулярного разложения оператора имеет важные приложения в ряде задач.
Проблема заключается в факторизации матрицы X в виде матриц особого вида

$$X = U\Sigma V^T.$$

U, V – ортогональные матрицы. Σ – диагональная матрица с сингулярными значениями в порядке убывания.

Для ряда практических задач требуется вычислять усеченное сингулярное разложение

$$X \approx U_{1k}\Sigma_{kk}V_{1k}^T.$$

Демонстрация примера.

Заключение

Презентация и примеры, продемонстрированные в курсе, доступны по ссылке `http://angel.cmc.msu.ru/~basrav`.

Вопросы, замечания, комментарии, предложения? Пишите `basrav@angel.cmc.msu.ru`.

Спасибо за внимание.

Литература

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

[2] М. Ю. Баландин and Э. П. Чепурина.
Методы решения СЛАУ большой размерности.
Новосибирск: Изд-во НГТУ, 2000.

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

[4] S. Balay, K. Buschelman, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang.
PETSc Web page.
See http://www. mcs. anl. gov/petsc.

[5] L. Oliker, A. Canning, J. Carter, C. Iancu, M. Lijewski, S. Kamil, J. Shalf, H. Shan, E. Strohmaier, S. Ethier, et al.
Scientific Application Performance on Candidate PetaScale Platforms.
2007.

[6] V. Hernandez, J.E. Roman, and V. Vidal.
SLEPc: A scalable and flexible toolkit for the solution of eigenvalue problems.
ACM Transactions on Mathematical Software (TOMS), 31(3):351–362, 2005.