Параллельные программы библиотеки JINRLIB

Л.В. Попкова, А.П. Сапожников, Т.Ф. Сапожникова[a]

Объединенный институт ядерных исследований,
Россия, 141980, г. Дубна, ул. Жолио-Кюри, д. 6

E-mail: [a] tsap@jinr.ru

JINRLIB (http://www.jinr.ru/programs/jinrlib/) — библиотека программ, предназначенных для решения широкого круга математических и физических задач. Программы объединяются в библиотеки объектных модулей или существуют в виде самостоятельных пакетов прикладных программ. В настоящий момент насчитывается более 60 программных пакетов.

В последнее время происходит бурное развитие технологий программирования параллельных вычислений, в частности, MPI (Message Passing Interface). Эта тенденция нашла свое отражение и в библиотеке JINRLIB. Была сформулирована следующая стратегия распараллеливания: программа, подготовленная для работы в среде MPI, должна успешно работать при любом количестве NP параллельных процессов, вовлекаемых в решение задачи, в том числе и при NP, равном 1. Таким образом, возникает единый исходный текст программы, равно пригодный к эксплуатации как на традиционных последовательных вычислительных системах, так и на современных кластерах, состоящих из большого числа процессоров. Эта идея была успешно реализована при распараллеливании программ, описанных ниже.

MINUIT — параллельная версия программы минимизации функций многих переменных. На примере MINUIT обсуждаются проблемы распараллеливания больших вычислительных программ.

PFUMILI — модификация известной программы FUMILI, допускающая ее эффективную эксплуатацию на современных вычислительных кластерах, объединяющих сотни однотипных процессоров.

CLEBSCH2 — вычисление простейшей формы коэффициентов Клебша-Гордана. Программа свободна от типичных при вычислении факториалов "в лоб" случаев переполнения при умножении.

PRIMUS — программа, реализующая классический алгоритм так называемого решета Эратосфена для генерации простых чисел. Авторский интерфейс был модифицирован для упрощения возможности использования нескольких процессоров в рамках технологии MPI.

PROFILE — программный инструмент для исследования производительности программ в определяемых пользователем интервалах. Программа пригодна для использования в традиционных (последовательных) фортранных программах, так и в распараллеленных с использованием технологии MPI.

IntroMPI — подборка учебных программ-примеров по технологии параллельного программирования MPI с описанием, где в доступной форме излагаются основы MPI и даются некоторые рекомендации для работы.

Ключевые слова: библиотека программ, параллельные вычисления, MPI (Message Passing Interface)

1. Введение

JINRLIB — библиотека программ, предназначенных для решения широкого круга математических и физических задач [Попкова и др., 2008]. Пополнение библиотеки происходит программами и программными пакетами, создаваемыми сотрудниками ОИЯИ и их коллаборантами. Программы объединяются в библиотеки объектных модулей или существуют в виде самостоятельных пакетов прикладных программ (Рис.1). Каждая программа идентифицируется уникальным индексом или именем.

Информация о программах размещается на сайте библиотеки — http://www.jinr.ru/programs/jinrlib, где можно найти каталог, исходные тексты, описания программ и программных пакетов, библиотеки объектных модулей математических программ. Ведется каталог вновь поступивших программ и программных пакетов. Добавлен раздел для параллельных программ. В настоящий момент насчитывается более 60 программных пакетов, большинство которых решает задачи автоматизации обработки экспериментальных данных и вычислительной математики.

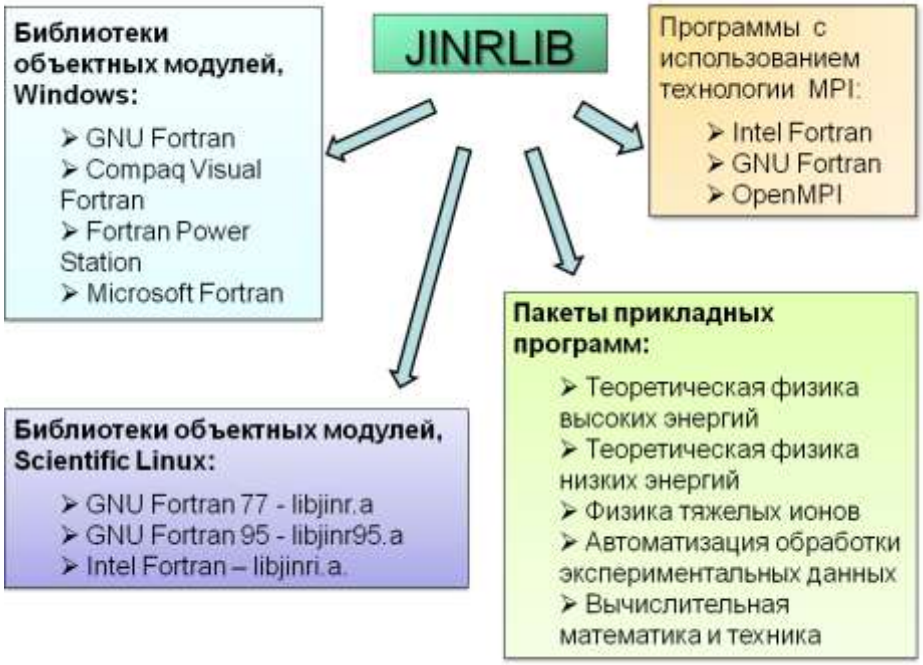

Рис.1. Библиотека программ JINRLIB

2. Поддержка программирования в среде MPI

В последнее время происходит бурное развитие технологий программирования параллельных вычислений, в частности, MPI (Message Passing Interface). Эта тенденция нашла свое отражение и в библиотеке JINRLIB. Была сформулирована следующая стратегия распараллеливания: программа, подготовленная для работы в среде MPI, должна успешно работать при любом количестве NP параллельных процессов, вовлекаемых в решение задачи, в том числе и при NP=1. Таким образом, возникает единый исходный текст программы, пригодный к эксплуатации как на традиционных последовательных вычислительных системах, так и на современных кластерах, состоящих из большого числа процессоров. Необходимым условием для обеспечения такой унификации является наличие программ-заглушек пакета MPI, что позволяет использовать библиотечные программы даже на тех машинах, где никакого MPI нет. Базовые операции MPI сконструированы таким образом, что в качестве заглушек достаточно иметь пустые

подпрограммы, лишь бы системный загрузчик смог реализовать все внешние ссылки. Эта идея была успешно реализована при распараллеливании программ, описанных ниже.

3. Параллельные программы библиотеки JINRLIB

MINUIT — параллельная версия программы минимизации функций многих переменных.

Классическим примером больших вычислительных программ является MINUIT — программа минимизации функции многих переменных, написанная в начале 70-х годов прошлого века Ф. Джеймсом (ЦЕРН) и весьма популярная до сих пор, что и побудило выбрать ее как объект распараллеливания прежде других [Сапожников, 2003]. В ходе распараллеливания MINUIT наметился ряд достаточно типичных для любой большой вычислительной программы этапов.

1. Организация обрамления программы. Добавлены подпрограммы MN_Start и MN_Finish для захвата и освобождения MPI-ресурсов, каковыми являются временная группа процессов и вспомогательные типы данных.

2. Подготовка программ-заглушек пакета MPI — для обеспечения возможности эксплуатации этой версии MINUIT на тех машинах, где никакого MPI нет вообще.

3. Реорганизация операций ввода и вывода. Весь вывод, естественно, должен выполнять только один из NP процессов, участвующих в совместной работе, и, очевидно, это должен быть процесс 0. Поэтому все операторы вывода должны предваряться проверкой номера процесса:

```
If (MyProc.eq.0) Write(*,*) Data
```

Что же касается ввода, то тут чуть сложнее. Ясно, что файлы с входной информацией должны принадлежать кому-то одному, стало быть, нулевому процессу. Прочитав из файла, он должен поделиться прочитанным со своими партнерами:

```
If (MyProc.eq.0) read(1,*) (data(i),i=1,count)
call MN_Bcast(data,count) !Propagate for all processes
```

MN_Bcast практически идентична MPI_Bcast, но маскирует от внешнего окружения как используемый коммуникатор, так и номер координирующего процесса.

Параллельная версия MINUIT была протестирована как на системе с общей памятью, так и на PC-Linux кластере. В обоих случаях работа происходила под управлением ОС типа UNIX.

PFUMILI — параллельная версия программы минимизации FUMILI [Сапожников, 2009].

Прежде всего, нужно было найти наиболее времяёмкие места распараллеливаемой программы. Для этого использовалась следующая техника:

- в исходном тексте программы выделялось некоторое количество интервалов;
- в начале и конце каждого такого N-го интервала помещались засечки времени;
- в самом конце тестовой программы делалась печать накопленной статистики.

Для этого использовалась описанная ниже программа Profile из библиотеки JINRLIB, которая подсчитывает количество посещений заданного интервала и суммарное время, затраченное процессом на эти посещения. При запуске программы FUMILI с достаточно типичным набором данных была получена следующая статистика:

```
Proc: 0 of  1. Time  1 =     0.02 sec. Ncall=    45
Proc: 0 of  1. Time  3 =     0.25 sec. Ncall=     3
Proc: 0 of  1. Time  4 =   21.14 sec. Ncall=    45
Astime:    21.48  CPU_time:    21.48  for process  0
```

Легко видеть, что наиболее времяёмким местом является интервал номер 4 (подпрограмма SGZ, вычисляющая производные минимизируемой функции по всем ее параметрам), занимающий львиную долю (98.5%) общего времени прогона теста.

Результат работы подпрограммы SGZ есть сумма по всем заданным экспериментальным точкам (NED). Основная идея распараллеливания — поделить всю работу (NED экспериментальных точек) между всеми mpi_size процессами приблизительно поровну:

```
nn=ned/mpi_size              ! points for 1 process
n1=1+mpi_rank*nn             ! 1-st point
```

```
n2=n1+nn                    ! last point
if(mpi_rank.eq.mpi_size-1) n2=ned
```

Здесь mpi_rank = 0,1,…mpi_size-1 — внутренний номер процесса, а последний оператор корректирует номер последней точки для последнего процесса, если NED не делится нацело на mpi_size.

После чего цикл DO L1=1,NED по всем экспериментальным точкам заменяется на более короткий цикл DO L1=n1,n2. Поскольку этот цикл является самым внешним циклом, распределение его работы между mpi_size процессами-исполнителями следует признать хорошим распараллеливанием.

Также процессы обязаны сделать всеобщим достоянием полученные ими частные результаты: надо совершать межпроцессные обмены информацией. Результат запуска того же теста на исполнение коллективом из NP=2 процессов на двухпроцессорном компьютере:

```
Proc: 0 of  2. Time  1 =     0.02 sec. Ncall=    45      Proc: 1 of  2. Time  1 =     0.02 sec. Ncall=    45
Proc: 0 of  2. Time  3 =     0.33 sec. Ncall=     3      Proc: 1 of  2. Time  3 =     0.33 sec. Ncall=     3
Proc: 0 of  2. Time  4 =   13.16 sec. Ncall=    45      Proc: 1 of  2. Time  4 =   13.12 sec. Ncall=    45
Astime:  13.70  CPU_time:   13.55  for process  0        Astime:  13.59  CPU_time:   13.50  for process  1
```

Здесь астрономического времени AsTime затрачено в 1.7 раз меньше, чем при NP=1, поскольку 2 процесса работали одновременно, а ускорение "не дотянуло" до идеальных 2 именно из-за необходимости совершать межпроцессные обмены данными.

CLEBSCH2 - вычисление простейшей формы коэффициентов Клебша-Гордана:

Программа CLEBSCH2 [Сапожников, 2011] свободна от типичных при вычислении факториалов "в лоб" случаев переполнения при умножении. В программе реализован вариант — разумно чередовать операции умножения и деления.

Присмотревшись к выражению коэффициента, можно увидеть, что он получается равным произведению первых K членов ряда 1,2,3,…,N, деленному на произведение K последних членов этого ряда. Поэтому цикл работы программы можно сократить с N до K и учесть симметрию *C(k,n)=C(n-k,n)*. В результате имеем простейший цикл:

```
c=1.
do i=1, Min(K,N-K)
   c=c*float(i)/float(N-i+1)
enddo
Clebsch=c
```

Программа делает всего не более N/2 умножений и N/2 делений, является коллективной операцией пакета MPI и прямым потомком операции MPI_AllReduce.

PRIMUS — программа, реализующая классический алгоритм так называемого решета Эратосфена для генерации простых чисел [www.jinr.ru/programs/jinrlib/primus/].

В работе [Alexandrov et al., 2002] опубликована программа Eratosthenes, реализующая классический генератор простых чисел, то есть целых чисел, нацело делящихся только на 1 и самих себя.

В параллельной программе авторский интерфейс был модифицирован: вместо одного параметра N введен диапазон [N0,N], что позволило легко использовать ее при параллельной работе нескольких процессов в рамках единой задачи. Была написана программа Primus, которая по существу является надстройкой над Eratosthenes, позволяющая задействовать заданное количество NP процессов. Распараллеливание в ней свелось к простому делению отрезка [2,N] числовой оси между процессами приблизительно поровну с последующим объединением полученных результатов в памяти главного процесса.

PROFILE — программный инструмент для исследования производительности программ в определяемых пользователем интервалах [www.jinr.ru/programs/jinrlib/profile/]. Программа при-

годна для использования в традиционных (последовательных) фортранных программах, так и в распараллеленных с использованием технологии MPI.

Измерение времени работы процессора при исполнении исследуемой программы является чрезвычайно полезным мероприятием, которое позволяет:

- оценить производительность программы в целом, определяя ее конкурентоспособность в ряду аналогичных программ;

- выявить наиболее времяёмкие места изучаемой программы, чтобы в дальнейшем сосредоточить свои усилия именно на них.

Предполагается, что пользователь хочет измерить время работы своей программы на N<200 интервалах ее исполнения. Тогда при N=-1 единожды совершается первоначальная засечка времени с помощью стандартной системной программы Seconds. Далее, в начале и конце каждого N-го интервала жизни исследуемой программы необходимо вызвать Profile(N). Засечки делаются с использованием стандартной программы CPU_Time. Общее количество временных засечек должно быть четным.

В конце программы следует вызвать Profile(0) для вывода всех сделанных засечек.

IntroMPI — подборка учебных программ-примеров по технологии параллельного программирования MPI с описанием, где в доступной форме излагаются основы MPI и даются некоторые рекомендации для работы [www.jinr.ru/programs/jinrlib/intrompi/].

4. Заключение

Программы подготовлены для использования на многопроцессорных кластерах HybriLIT и ЦИВК Многофункционального информационно-вычислительного комплекса Лаборатории информационных технологий Объединенного института ядерных исследований (Дубна). Результаты тестирования демонстрируют реально достигнутый параллелизм.

Список литературы

Попкова Л. В., Сапожников А. П., Сапожникова Т. Ф., Федорова Р. Н. Библиотека программ JINRLIB // Электронные библиотеки: перспективные методы и технологии, электронные коллекции. Труды Десятой Всероссийской научной конференции, Дубна, 7 - 11 октября 2008 года, стр.284-289.

Popkova L. V.,. Sapozhnikov A. P., Sapozhnikova T. F. Program library JINRLIB // Digital Libraries: Advanced Methods and Technologies, Digital Collections. The Tenth Anniversary of All-Russian Research Conference. Dubna, Russia, October 7 - 11. — 2008. — P. 284-289 (in Russian).

Сапожников А. П. Второй опыт распараллеливания больших вычислительных программ. Параллельная версия программы FUMILI // Scientific report 2008-2009 years. — LIT, JINR, Dubna. — 2009.

Sapozhnikov A. P The second experience of large computational programs parallelization. Parallel version of FUMILI Program // Scientific report 2008-2009 years. — LIT, JINR, Dubna. — 2009 (in Russian).

Сапожников А. П. Опыт распараллеливания больших вычислительных программ. Параллельная версия программы MINUIT // Дубна, ОИЯИ, Р11-2003-216. — 2003.

Sapozhnikov A. P Parallel version of MINUIT program - minimization of a multi-parameter function // Dubna, JINR, P11-2003-216. — 2003 (in Russian).

Сапожников А. П. Программа для вычисления простейшей формы коэффициентов Клебша-Гордана. Параллельная версия // Scientific report 2010-2011 years. — LIT, JINR, Dubna. — 2011.

Sapozhnikov A. P Calculation of simple kind for Clebsch-Gordan coefficient. Parallel version // Scientific report 2010-2011 years. — LIT, JINR, Dubna. — 2011 (in Russian).

Alexandrov L., Baranov D. B., Yotov P. Polynomial splines interpolating prime series // JINR-P5-2002-228. — 2002.

Parallel programs in JINRLIB Library

L.V. Popkova, A.P. Sapozhnikov, T.F. Sapozhnikova[a]

Joint Institute for Nuclear Research,
6 Joliot-Curie st., Dubna, 141980, Russia

E-mail: [a] tsap@jinr.ru

JINRLIB (http://www.jinr.ru/programs/jinrlib/) is a software library intended for solving a wide range of mathematical and physical problems. The programs are grouped into libraries of object module or exist as separate software packages. At the moment there are more than 60 software packages.

For the past few years, the technologies of programming parallel computing, in particular MPI (Message Passing Interface) have been intensively developed. This trend was reflected in the library JINRLIB. Formulated was the following strategy for parallelization: the program prepared to work in the MPI environment, needs to work successfully with any number of NP parallel processes involved in the solution of the problem, including when NP equals 1. Thus, there appears a unified source text of the program which can be exploited both on the traditional sequential computing systems and the modern clusters consisting of a large number of processors. This idea has been successfully implemented when paralleling the programs described below.

MINUIT — parallel version of well-known software package performing minimization of a multi-parameter function. The problems of paralleling large computational programs are discussed on this example.

PFUMILI — modification of the well-known FUMILI program may be effectively exploited on modern large computational clusters, containing hundreds processor units.

CLEBSCH2 — calculation of a simple kind for Clebsch-Gordan coefficients. Function is free of typical overflow errors while direct calculation of factorials.

PRIMUS — a program for prime numbers generation is described. It uses a classical algorithm of Eratosthenes sieve. The author's interface was modified to simplify the usage of several processors under MPI-technology.

PROFILE — a program tool for studying the productivity in a program being investigated by user on the set of intervals of interest is described. Usage of MPI-technology is possible.

IntroMPI — a selection of training programs-examples on parallel programming technologies MPI with a description in a simple form of the basics of MPI and some recommendations for work.

Keywords: program library, parallel computing, MPI (Message Passing Interface)