ОРДЕНА ЛЕНИНА
ИНСТИТУТ ПРИКЛАДНОЙ МАТЕМАТИКИ
им.М.В.Келдыша РАН

Бугеря А.Б.

КОМПИЛЯЦИЯ И ЗАПУСК ПРОГРАММ НА ЯЗЫКЕ НОРМА

Москва
2001

Бугеря А.Б.
Компиляция и запуск программ на языке Норма.♦

Аннотация

Данная работа является руководством пользователя по компилятору и конфигуратору языка Норма. Рассматриваются DOS и UNIX реализации, формат запуска и состав, опции и дополнительные возможности соответствующих программ.

Непроцедурный язык Норма предназначен для автоматизации решения сеточных задач на вычислительных системах с параллельной архитекту

рой. Этот язык позволяет исключить фазу программирования, которая необходима при переходе от расчетных формул, заданных прикладным специалистом, к программе.

A.B.Bugerya.
Compiling and running the programs on the NORMA language.

Abstract

This book is the User Guide on compiler and configurator for Norma language programs. It describes the DOS and UNIX implementation, compiler and configurator structure, its running, options and other features.

The nonprocedural Norma language is a tool aimed for automated solution of the grid-oriented problems on parallel computer systems. This language eliminates the programming phase that is necessary to pass from computational formulas, derived by an application specialist, to a computer program.

♦ Работа выполнена при финансовой поддержке Российского Фонда Фундаментальных Исследований (проекты № 99-01-00842, № 01-01-06333).

Содержание.

Введение

Непроцедурный язык Норма [1] предназначен для автоматизации решения сеточных задач на вычислительных системах с параллельной архитектурой. Этот язык позволяет исключить фазу программирования, которая необходима при переходе от расчетных формул, заданных прикладным специалистом, к программе.

Процесс подготовки и запуска программ на языке Норма состоит из следующих этапов:

- Подготовка исходных текстов программных модулей на языке Норма.
- Трансляция программных модулей на языке Норма в Фортран. Это может быть Фортран 77 (последовательная программа), или один из языков – расширений Фортрана для параллельных вычислительных систем – Фортран MPI, Фортран GNS или Фортран PVM (распределенная программа).
- Конфигурация программы, объединение с модулями, написанными на Фортране.
- Трансляция и запуск полученных программ на языке Фортран.

Подготовка исходных текстов программ на языке Норма

Тексты программ на языке Норма могут быть подготовлены с использованием любого текстового редактора, позволяющего сохранять текст программы в файл “так как он есть”, без каких-либо управляющих кодов или

последовательностей. Текст программы должен быть сохранен в файл с расширением *.hop.

Трансляция программы из языка Норма в Фортран

Для трансляции программы из языка Норма в Фортран используется компилятор языка Норма. Существуют версии компилятора для DOS и для UNIX. Под Windows можно использовать DOS версию в окне MS DOS. DOS версия состоит из двух файлов – *normapc.exe* и *normapc.ovl*. Версия для UNIX представляет собой один исполняемый файл *norma* и поставляется в исходниках, которые могут быть скомпилированы в конкретной версии операционной системы с использованием компилятора *cc*. Параметры запуска, опции, требования к входным и содержание выходных файлов у версий компилятора одинаковы. Никаких дополнительных библиотек и настройки переменных окружения не требуется.

Запуск компилятора

Запуск на трансляцию осуществляется командой:
norma <имя файла>.hop <опции> - в UNIXe;
normapc <имя файла>.hop <опции> - в DOSe;

Компилятор должен выдать листинг на экран и записать его в файл <имя файла>.lst и при отсутствии ошибок создать файл с результатом трансляции. Результатом трансляции является файл <имя файла> с расширением .fmp (если выходным языком является Фортран MPI), с расширением .ft (если выходным языком является Фортран PVM), или с расширением .for (в любом другом случае). Указанные файлы создаются в текущей директории и содержат текст получившихся Фортран-программ со спецкомментариями для конфигуратора (см. ниже).

В листинг выдается и исходный текст программы, и полученный текст на Фортране. Все ошибки и предупреждения выдаются в текстовом виде. Если ошибка или предупреждение относятся к конкретной строке программы, то они помещаются в листинг непосредственно после этой строки. Если же ошибка или предупреждение относятся ко всему разделу, они помещаются в листинг непосредственно после всего этого раздела. После каждого раздела приводится общее количество ошибок и предупреждений, относящихся к этому разделу и время трансляции раздела. В конце листинга приводится общее количество ошибок и предупреждений во всех разделах транслируемой программы и общее время трансляции.

В случае возникновения какой-либо внутренней ошибки транслятора на экран выдается сообщение вида:

RULEX ABNORMAL STOP <код ошибки> AT STEP <номер шага>
REFAL-M ERROR: <описание ошибки>
In function <имя функции>
With Entry: <аргументы функции>

В этом случае сообщите разработчикам транслятора о возникшей ошибке с приложением текста исходной программы на языке Норма и вышеописанного сообщения.

Опции компилятора

Опции задаются в командной строке после имени файла.

[f-77 | mpi | gns | pvm | pvm-p] [eng | rus] [cache | nocache]
[reliability] [max=<число>] [length=<число>] [bufsize=<число>]

Опции выходного языка:

- f-77 – генерировать последовательную программу на Фортране-77 (используется по умолчанию);
- mpi – генерировать программу на Фортране MPI;
- gns – генерировать программу на Фортране GNS;
- pvm – генерировать программу на Фортране PVM;
- pvm-p – генерировать программу на Фортране PVM для Parsytec.

Опции языка сообщений и комментариев:

- eng – использовать английский язык (используется по умолчанию);
- rus – использовать русский язык.

Опции порядка генерации циклов для многомерных массивов (переменных на многомерных областях):

- cache – внутренние (вложенные) циклы идут по первым индексам массивов (используется по умолчанию);
- nocache – внутренние (вложенные) циклы идут по последним индексам массивов.

Комментарий. Чтобы понять смысл опций порядка генерации циклов, необходимо отметить 3 следующих момента:

1. Пусть у нас имеется величина A(I, J, K), которой надо присвоить значение величины B(I, J, K). Организовать циклы на языке Фортран мы можем как

do I = …		do K = …
do J = …	так и	do J = …

do K = …	do I = …
A(I, J, K) = B(I, J, K)	A(I, J, K) = B(I, J, K)

В первом случае внутренний цикл идет по последнему индексу (K), во втором – по первому (I).

2. Реализации Фортрана по традиции размещают массивы в памяти так, что соседними являются элементы со сдвигом по первому индексу (A(I, J, K) и A(I+1, J, K)).
3. Процессор имеет кэш оперативной памяти, который обменивается с оперативной памятью блоками.

Поэтому при организации циклов в обратном порядке, когда внутренние циклы идут по первым индексам массивов, вероятность того, что после обращения к B(I, J, K) при последующем обращении к B(I+1, J, K) величина будет взята из кэша, очень велика. При наличии значительного количества таких обращений в программе, что типично для вычислительных задач, время выполнения существенно уменьшается (по экспериментальным данным до 2-х раз).

Опция для контрольных точек:

- reliability – задает режим повышенной надежности. Контрольные точки дублируются в дополнительный файл, что исключает потерю данных при сбое во время записи основного файла. По умолчанию выключена.

Опция максимального числа процессоров по 2-й координате (только GNS):

- max=<число> - задает коэффициент для виртуального номера процессора в функции NEWTASK. Должно совпадать со значением аналогичной опции конфигуратора (см. ниже). По умолчанию равно 32.

Опция максимальной длины идентификатора:

- length=<число> - должно совпадать со значением аналогичной опции конфигуратора (см. ниже). По умолчанию равно 6.

Опция максимального размера пересылок между процессорами:

- bufsize=<число> - в килобайтах. Пересылки, превышающие указанный размер, будут разбиваться на несколько. По умолчанию равно 1000 Kb.

Конфигуратор

После того, как все разделы программы на языке Норма транслированы, запускается программа-конфигуратор *normacnf.exe* (DOS) или *normacnf* (UNIX). Конфигуратор проверит наличие всех необходимых модулей программы и соберет все модули, составляющие исполняемую задачу, в один файл. В случае использования Фортрана GNS будет построено описание

конфигурации (или сразу выполнено, это зависит от заданных опций). В случае Фортрана MPI производится настройка операторов вызовов разделов.

Затем конфигуратор выполнит командный файл конфигуратора, если он имеется. DOS версия для выполнения командного файла вызывает внешний интерпретатор *ncn.exe*, UNIX версия программы выполняет файл сама.

Запуск конфигуратора

При запуске в качестве параметров указываются опции и имена файлов с получившимися в результате трансляции Фортран-программами. Например, если исходный текст Норма-программы был в файлах file1.hop и file2.hop, то запуск конфигуратора в случае MPI выглядит следующим образом:

normacnf.exe /mpi [опции] file1.fmp file2.fmp

Опции конфигуратора описаны ниже. В процессе анализа спецкомментариев конфигуратор строит дерево вызовов разделов и проверяет полноту программы, т.е. наличие всех оттранслированных модулей. Кроме того, в случае Фортрана MPI все модули собираются в один файл с расширением ‘.for’. В случае Фортрана PVM или GNS все модули, относящиеся к одной задаче, собираются в один файл с именем распределенного или главного раздела, породившего эту задачу и с расширениями ‘.f’ и '.gns' соответственно. Из этих файлов после компилирования получается файл с исполняемым кодом.

При генерации описания конфигурации и при компоновке Фортран-программ конфигуратор пишет результат во временный файл, если создаваемый файл уже существует. Затем производится сравнение существующего и получившегося файла, и, если изменений нет, временный файл просто уничтожается. Это позволяет не перетранслировать не изменившиеся задачи и/или описание конфигурации.

Опции конфигуратора

Опции задаются в командной строке.

[/mpi | /pvm] [/eng | /rus] [/s=<секция>] [/lq=< 2 | 3 >] [/full]
[/proc=n] [/d=<идентификатор>] [/p=<файл с проектом>] [/n=<имя>]

Опции используемого языка (может быть задана только одна или не задано ни одной):

- /pvm – используется Фортран PVM;
- /mpi – используется Фортран MPI;

Если ничего не задано, то используется Фортран GNS.

Опции для описания конфигурации (параметры конфигурации, только GNS):

- /s=<секция> - задает секцию, на которой будет выполняться программа;
- /lq=< 2 | 3 > - задает значение LinkQuota;
- /full - задает FullHardNet;
- /proc=n - максимальное число процессоров по линейке j, 32 по умолчанию. Должно быть таким же, как и в соответствующей опции компилятора языка Норма.

Опции для компонуемых Фортран-программ:

- /l=<число> - задает максимальную длину идентификатора, должно соответствовать заданному при трансляции Норма-программы. 6 по умолчанию.

/d=<идентификатор> - определить идентификатор. В зависимости от того, определен идентификатор 'program' или нет, конфигуратор строит программу с описанием конфигурации или сразу ее исполняет. Определение идентификаторов может использоваться в командном файле конфигуратора (см. ниже).

/p=<файл с проектом> - определят, что программу описывает проект - это текстовый файл, в котором перечислены каждый с новой строки все файлы с модулями, составляющими программу.

/n=<имя> - задает имя файлу с описанием конфигурации или исполняемой задаче в случае MPI. Если эта опция не задана, то именем берется имя проекта. Если и проект не задан - имя первого указанного в командной строке файла.

Опции используемого языка сообщений:
/rus - сообщения на русском языке;
/eng - сообщения на английском языке.
По умолчанию используется язык модуля с главным разделом.

Командный файл конфигуратора

Так как конфигуратор обладает полной информацией о составе всей программы, он может вызвать компилятор Фортрана и скомпилировать все полученные задачи. Таким образом, пользователь, пишущий на Норме может быть избавлен от большого количества рутинной работы - он должен только оттранслировать все модули программы с Нормы в Фортран и запустить конфигуратор.

Все вызовы, которые конфигуратор должен сделать, записаны в специальном виде в командном файле конфигуратора. Это позволяет легко менять местоположение на диске компиляторов и библиотек, параметры и опции компиляторов и т.п. Файл должен называться normacnf.cnf и находиться в текущей директории при запуске конфигуратора. После генерации файлов с текстами задач конфигуратор, если присутствует командный файл, начинает его исполнять. Команды, указанные в файле, исполняются последовательно. В командах можно использовать следующие переменные:

- *%s_btl%* - имя файла (без расширения) с описанием конфигурации и с исполняемым модулем в случае MPI.
- *%s_gns%* - имена файлов (без расширения) с исполняемыми модулями в случае GNS.

Если в команде использована переменная, то перед выполнением команды она получает значение, соответствующее данной задаче. Команда с переменной *%s_gns%* выполняется столько раз, сколько имеется исполняемых модулей, каждый раз с новым именем.

Формат командного файла следующий:

<команда>
 ...
<команда>

<команда>::=
[<условие выполнения>]
 ...
[<условие выполнения>]
<заголовок>
<командная строка>
[<ошибка>]
 ...
[<ошибка>]
"[end command]"

<заголовок> - строка, выдаваемая на экран перед выполнением команды. В ней также возможно использование переменных.
<командная строка> - собственно выполняемая команда.

<ошибка>::=
<сообщение об ошибке>
<код возврата команды>

<свой код возврата>

После выполнения команды все <ошибки> просматриваются по порядку. Если полученный код возврата больше или равен указанному <коду возврата команды>, то на экран выдается <сообщение об ошибке>, в котором также можно использовать переменные, и программа-конфигуратор прекращает выполнение командного файла с кодом возврата <свой код возврата>.

С помощью <условия выполнения> можно определить, в каких ситуациях выполнять ту или иную команду. <Условий выполнения> может быть несколько. Команда выполняется, если истинны все <условия выполнения>.

<условия выполнения>::=
? ifdef <идентификатор> |
? ifndef <идентификатор> |
? <имя файла 1> -> <имя файла 2>

Условие ifdef <идентификатор> истинно, если <идентификатор> был определен при запуске конфигуратора с помощью опции '/d=' (см. раздел опции конфигуратора). Наоборот, Условие ifndef <идентификатор> истинно, если <идентификатор> не был определен при запуске конфигуратора. С помощью определения идентификаторов можно управлять исполнением файла команд (использовать различные опции и т.п.).

Другим типом условий является условие на время модификации файлов вида:
? <имя файла 1> -> <имя файла 2>

Это условие истинно, если время модификации <файла 1> больше времени модификации <файла 2>, или <файл 2> отсутствует. В <имени файла i> можно использовать переменные. С помощью условий этого типа можно не перетранслировать не изменившиеся файлы, так как при генерации текстов Фортран-программ конфигуратор не меняет получающимся файлам время модификации, если они полностью совпадают со своими предыдущими вариантами.

Полностью фрагмент командного файла, реализующий трансляцию программ на Фортране MPI, мог бы выглядеть следующим образом:

```
? %s_btl%.f -> %s_btl%.o
Compiling %s_btl%.f
mpif77 -fast -c %s_btl%.f
See errors
1
```

```
1
[end command]
? %s_btl%.o -> mpi_go
Linking %s_btl%.o
mpif77 -o mpi_go %s_btl%.o
See errors
1
1
[end command]
Program 'mpi_go' is ready to run
echo
[end command]
```

Построение описания конфигурации для Фортрана GNS

Для построения программы описания конфигурации, что необходимо только для Фортрана GNS, должна присутствовать опция '/d=program'. Конфигуратор построит по спецкомментариям описание конфигурации file1.c, которое затем надо будет транслировать каким-либо компилятором языка C с библиотекой config.lib, содержащей код функций языка описания конфигурации, и выполнить получившийся file1.exe.

В DOS версии конфигуратора кроме варианта генерации файла с описанием конфигурации существует возможность сразу получить описание конфигурации в оттранслированном виде. Конфигуратор содержит в себе библиотеку config.lib и может не генерировать программу на C, а сразу ее исполнять. Для этого надо не задавать опцию '/d=program'. Оба варианта имеют свои преимущества и недостатки. Вариант 'сразу исполнять' быстрее, проходит в один этап и не требует компилятора C и библиотеки config.lib. Зато при перетрансляции программы он повторяется еще раз, даже если конфигурация программы не изменилась. Вариант построения описания конфигурации генерирует описание и сравнивает его с предыдущим вариантом (если он есть). Если ничего не изменилось, дальнейшие шаги можно не делать.

Компиляция и запуск программ на выполнение

Компиляция и запуск полученных программ на языке Фортран (или его расширений – Фортрана MPI, PVM или GNS) производится так же, как и любых других программ на этом языке – см. руководство по соответствующему транслятору. Никаких дополнительных библиотек не требуется. См. также использование командного файла конфигуратора.

Схема выполнения Норма-программы следующая: главный раздел выполняется на одном процессоре, каждый распределенный раздел - на

указанном в нем числе процессоров. Количество процессоров задается в строке ‘DISTRIBUTION INDEX=’ в тексте Норма-программы. В случае Фортрана MPI все распределенные разделы должны иметь одинаковое количество процессоров. Выполнение программы на Фортране MPI должно производиться на (N+1) процессоре, где N – число процессоров распределенного раздела. В случае Фортрана PVM распределенные разделы могут иметь разное количество процессоров. Выполнение программы на Фортране PVM должно производиться на (N+1) процессоре, где N – наибольшее число процессоров из требуемых в каждом распределенном разделе. В случае Фортрана GNS распределенные разделы также могут иметь разное количество процессоров. Выполнение программы на Фортране GNS должно производиться на (N+1) процессоре, где N – суммарное число процессоров, требуемых в каждом распределенном разделе.

О разработчиках

В разработке языка Норма, транслятора и конфигуратора принимали участие:

Задыхайло Игорь Борисович
Андрианов Александр Николаевич, and@a5.kiam.ru
Бугеря Александр Борисович, bug@a5.kiam.ru
Ефимкин Кирилл Николаевич, efi@a5.kiam.ru

Литература

[1] А.Н.Андрианов, А.Б.Бугеря, К.Н.Ефимкин, И.Б.Задыхайло. НОРМА. Описание языка. Рабочий стандарт / Препринт ИПМ им.М.В.Келдыша РАН. №120. 1995. 52с.

Бугеря А.Б.

КОМПИЛЯЦИЯ И ЗАПУСК ПРОГРАММ НА ЯЗЫКЕ НОРМА