Летняя международная XXIX молодежная Школа-конференция по параллельному программированию
3-14 июля 2017 г.
ИВМиМГ СО РАН, Новосибирск
НГУ, Новосибирск

Применение теории структурного синтеза параллельных программ на вычислительных моделях для конструирования фрагментированных программ

Софронов Иван Викторович

Руководитель Малышкин Виктор Эммануилович
д.т.н., проф., ИВМиМГ СО РАН, зав. лабораторией

12.07.17

Сложность написания параллельных программ (ПП)

Помимо реализации алгоритма решения задачи требуется:

- дополнительные усилия для обеспечения вычислений: организация коммуникаций, взаимоисключений и др.
  - требует времени
  - требуется навык
- обеспечить переносимость между вычислителями
  - поддержка различных конфигураций и архитектур

В итоге: сложная поддержка и отладка

Снижение сложности создания ПП

Создаются системы параллельного программирования (СПП), в которых ПП генерируется автоматически на основе высокоуровневого описания.

СПП производит контроль исполнения сгенерированной программы.

Усилия пользователя тратятся на описание алгоритма решения задачи, а его реализацией занимается система.

Существуют и другие подходы к снижению сложности создания ПП, но в рамках работы рассматривается только такой подход.

Проблема выбора реализации ПП на основе высокоуровневого описания

На основе высокоуровневого описания можно конструировать ПП разными способами, получая при этом реализации с разными свойствами.

Возникает необходимость выбирать такую реализацию, чтобы программа была оптимальной относительно заданного критерия (например, времени работы результирующей программы)

Обзор подходов к решению проблемы выбора

Существующие решения аналогичной проблемы выбора одного из наборов оптимизаций компилятора для получения оптимальной программы:

- фиксированные наборы преобразований
- итеративный поиск и генетический алгоритм
- онтологии в связке с алгоритмами машинного обучения

Цель работы

Разработка алгоритмов и структур данных для обеспечения выбора реализации высокоуровневого описания в процессе синтеза ПП. Создание подсистемы для управления синтезом ПП, на примере системы фрагментированного программирования LuNA.

Задачи:

1) Создать представление для хранения и систематизации возможных вариантов реализации высокоуровневого описания;
2) Разработать и реализовать алгоритмы для работы с этим представлением;
3) Провести тестирование подсистемы на реальной задаче численного моделирования.

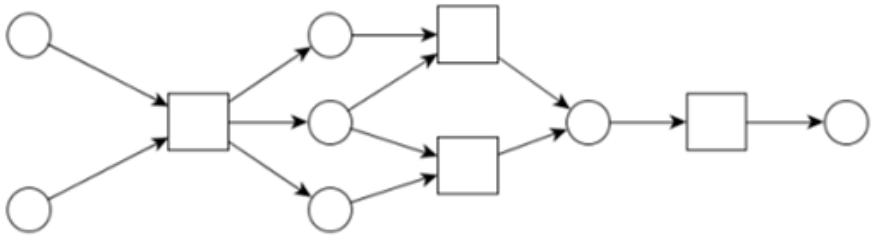
Требования к реализации:

1) автоматический выбор: участие пользователя в процессе ограничено указанием критериев и информации о вычислителе;
2) наличие средств для модификации базы вариантов реализации;
3) обеспечить переносимость создаваемой подсистемы между различными системами параллельного программирования.

Предлагаемое решение

Использовать вычислительные модели (ВМ), описанные в теории синтеза ПП[1], для хранения вариантов реализации высокоуровневого описания.

Среди особенностей предлагаемого решения можно отметить: в ВМ можно компактно описывать варианты; алгоритмы, для работы с ВМ не зависят от её объёма; ВМ изначально создавались как подход к автоматизации процессов, в частности синтеза ПП.



Графически ВМ представляют в виде двудольного ориентированный графа, в котором одно подмножество вершин называется “переменные”(круги), другое - “операции” (квадраты)

1. Вальковский В.А., Малышкин В.Э. - Синтез параллельных программ и систем на вычислительных моделях, 1988.

Система фрагментированного программирования (СФП) LuNA

- разрабатывается в ИВМиМГ СО РАН;
- реализует парадигму фрагментированного программирования;
- предназначена для поддержки параллельной реализации больших численных моделей на суперкомпьютерах;

Описание алгоритма решения задачи, состоит из фрагментов, для реализации которых необходимо выбирать один из доступных модулей.

Термины и определения

- Задача на ВМ - найти подграф, связывающий два непересекающихся подмножества вершин-переменных. При этом каждый такой подграф не содержит повторяющихся вершин-операций;
- Планирование – процесс выполнения задачи на ВМ
- План – результат планирования;
- Атрибут операции ВМ - свойство операции, существенное с т.з. критериев, используется при выборе оптимального плана

Расширение исходной модели

Для решения поставленных в работе задач исходное представление ВМ было дополнено:

- С каждой операцией связан набор атрибутов, а также набор команд - вызовов системных скриптов для осуществления реализации варианта;
- Каждая задача, помимо множеств исходных и искомых переменных, может содержать описание вычислителя и критерий.
- При описании вычислителя указывается количество задействованных процессов на узле, общее число доступных процессов на узле и количество задействованных узлов

Использование атрибутов для формирования оценки плана

- анализируя атрибуты операции, можно определить её вес в плане относительно заданного критерия;
- атрибут может удовлетворять критерию, не удовлетворять и не учитываться им, в соответствии с этим проставляется числовая оценка
- сложив все оценки атрибутов получим оценку операции
- сложив все оценки операций в плане получим общую оценку плана
- план с наибольшей оценкой считается наиболее подходящим под указанные требования

Описание ВМ

Для описания ВМ был создан язык LCMD (Language of Computational Model Description).

Позволяет описывать множества операций и переменных ВМ, а также задачи.

Для одной ВМ может быть описано несколько задач.

Описание ВМ: пример

```
model simple {
variable input, output;
< attr1, attr2>
 { input } -> simple_op1 -> { output } :
            command 'Run command1' run 'some_prog1' with 'arg1', 'arg2’ mod default;
};

task simple {
   given input;
   required output;
   using 2 of 4 in 3 nodes;
   criteria time_improvement;
};
```

ВМ конструирования ПП

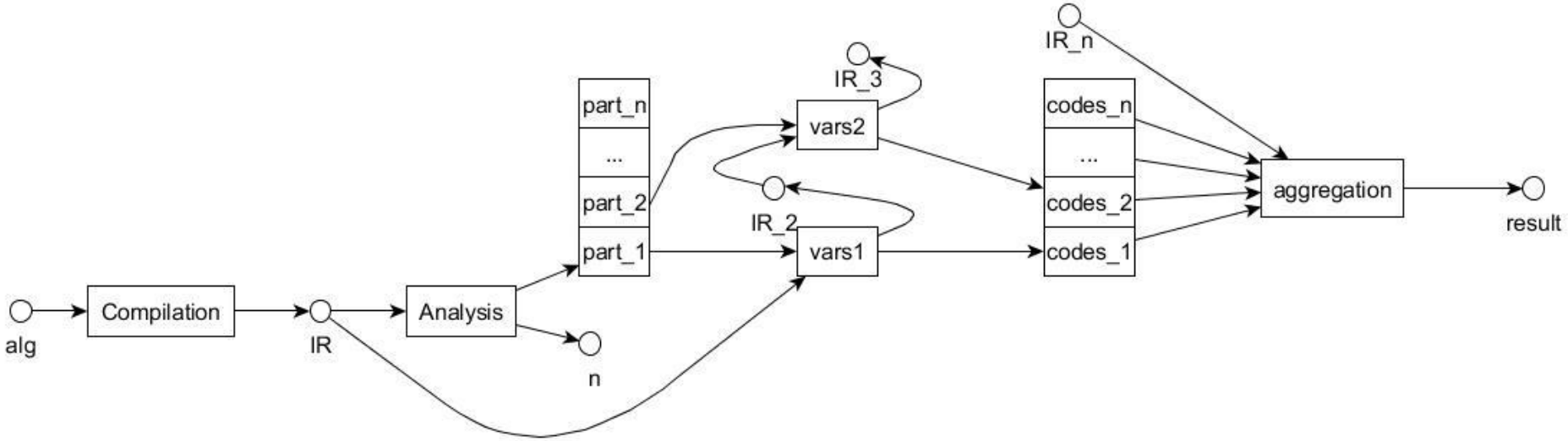
Содержит три стадии:

- анализ;
- выбор реализации;
- агрегация.

Исходные положения:

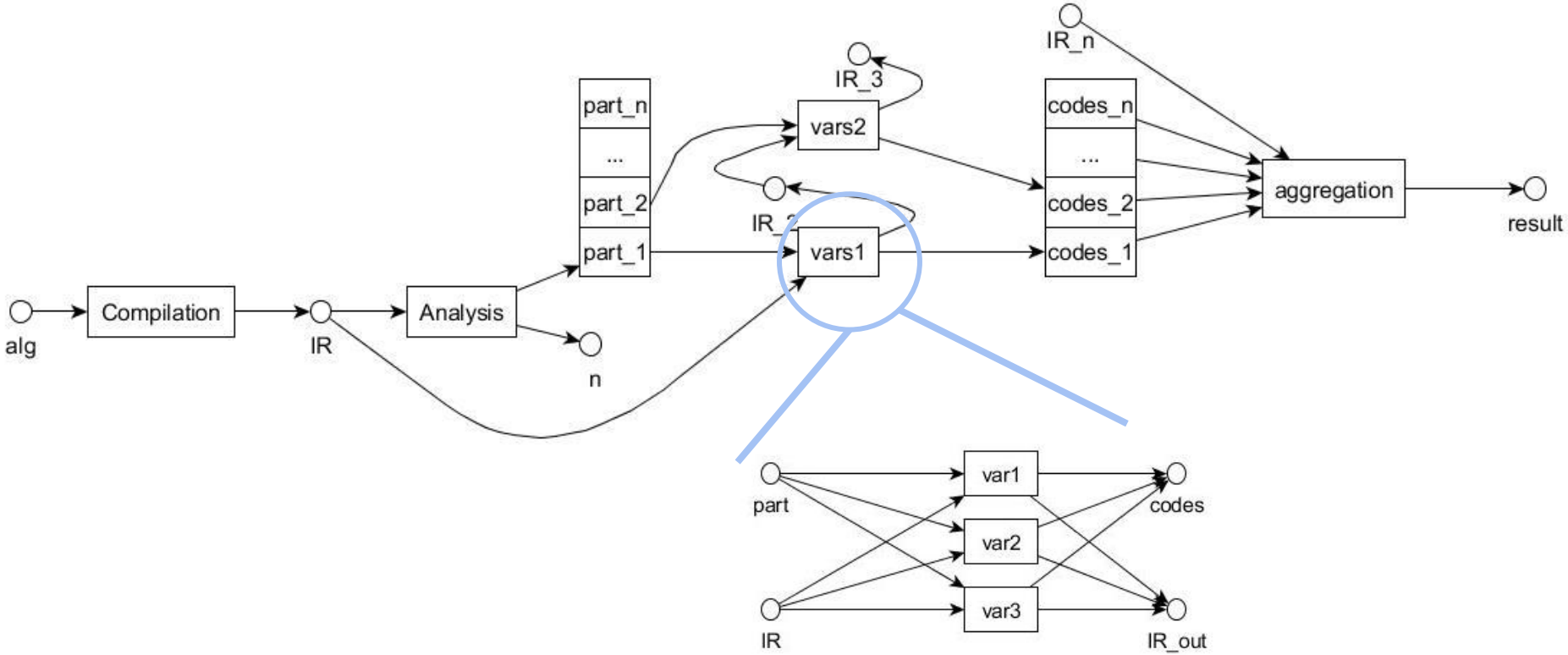
- в СФП LuNA описание алгоритма делится на части (part) явным образом;
- для каждой части имеется известный набор вариантов реализаций (vars)
- выбирается одна из реализаций на основе атрибутов (var)

ВМ конструирования ПП: графическое представление



IR - intermediate representation

ВМ конструирования ПП: графическое представление



IR - intermediate representation

Тестирование системы

Целью тестирования является демонстрация того, что, при указании конфигурации вычислителя и требований к результирующей программе, системой будет выбрана реализация описания алгоритма для этих условий.

Использовалась фрагментированная реализация решения уравнения Пуассона методом Зейделя в трёхмерной области при одномерной пространственной декомпозиции.

Для реализации основного цикла расчёта, имелись две реализации, использующие разные модели управления вычислениями:

- luna (data-flow)
- fw (event-driven)

Запуск задач производился на кластере МВС-10П (2 процессора Xeon E5-2690, 64 ГБ оперативной памяти, коммуникационная сеть на базе FDR Infiniband).

Тестирование системы

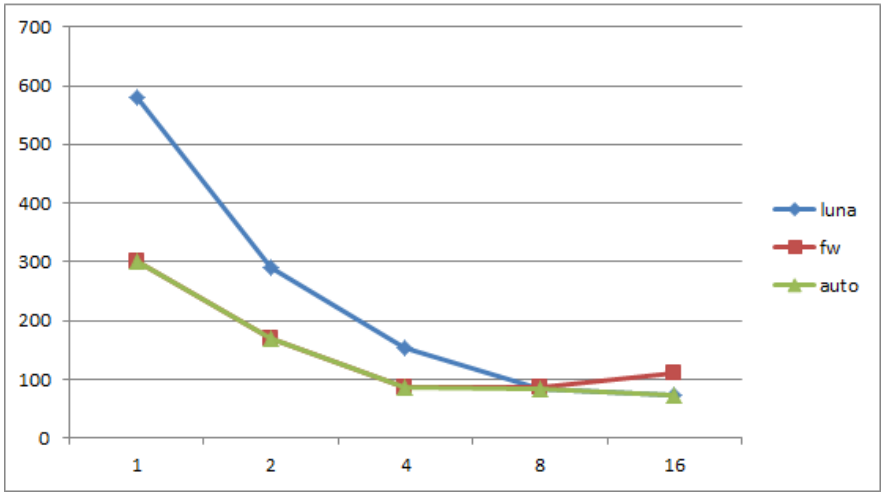
Размер области 1200x300x300 (double); 128 итераций; 40 фрагментов;

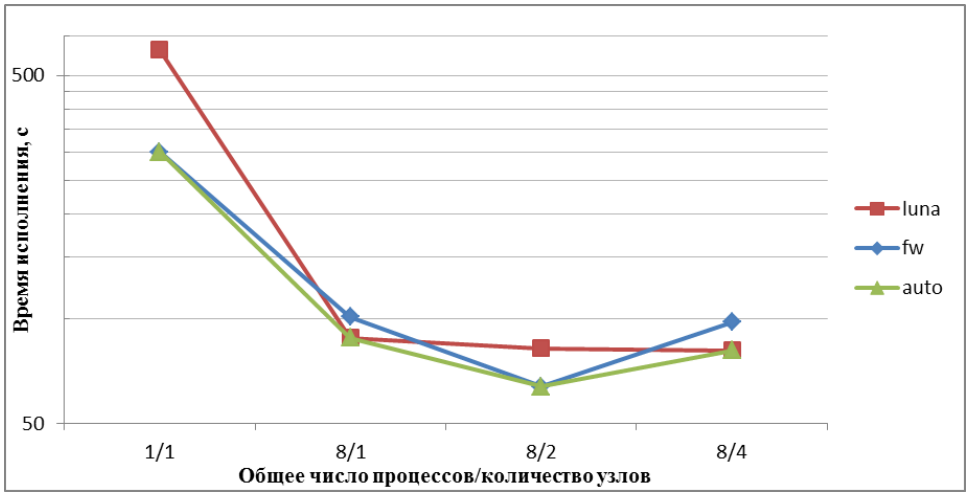
1) тестирование в общей памяти

	1	2	4	8	16
luna	581,15	289,23	152,91	**83,755**	**71,716**
fw	**301,1**	**168,87**	**85,55**	86,186	109,07

2) тестирование в распределённой памяти (кол. процессов / кол. узлов)

	1/1	8/1	8/2	8/4
luna	591,966	**88,143**	82,236	**81,101**
fw	**301,103**	101,259	**63,808**	97,749





Вывод: возможно на основе данных о вычислителе и критериях при каждом запуске получать программу под заданные условия

Заключение

- предложен подход к организации знаний по конструированию параллельных программ с использованием вычислительных моделей;
- реализована подсистема конструирования ПП, включающая в себя:
  - язык для описания ВМ и его интерпретатор;
  - вычислительные модели, описывающие процесс конструирования и алгоритмы для извлечения знаний из них;
  - модули для конструирования ПП;
- тестирование показало, что подсистема выбирает лучшую реализацию для заданных условий запуска;

Планы

Развитие предложенного подхода и созданной подсистемы:

- улучшить алгоритмы выбора вариантов реализации
- расширить класс программ, с которыми может работать подсистема.

Обеспечение возможности выбора реализации высокоуровневого описания в процессе исполнения программы, построенной на его основе

Спасибо за внимание!