Задание управления во фрагментированных программах сетями Петри и проверка корректности управления

В.В. Ларин
Институт вычислительной математики и математической геофизики СО РАН, Новосибирск

Представлены средства для задания во фрагментированной программе управления и проверки корректности этого управления при помощи сетей Петри. Рассматривается проблема отображения вычислительного алгоритма на ресурсы вычислительной системы и поиск следующих ошибок управления: дедлок, недостижимость нужного результата, недетерминированность результата, бесконечные циклы.

Фрагментированная программа (ФП) [1] представляется в виде набора фрагментов, каждый из которых является совокупностью вычислительной процедуры, входных и выходных переменных. Программистом задается порядок исполнения фрагментов.

Если есть алгоритм, вычисляющий некоторую функцию, то для задания ФП, реализующей алгоритм, необходимо описать множество переменных, множество вычислительных фрагментов, реализующих соответствующие операции, управление, определяющее порядок исполнения фрагментов, и задать отображение переменных и фрагментов на вычислительные ресурсы компьютера.

Отдельного рассмотрения заслуживает проблема задания управления в ФП. Она тесно связана с задачей отображения переменных и фрагментов на вычислительные ресурсы. На средства задания управления предъявляются следующие требования: удобство и возможность контроля корректности управления.

В работе исследуется применение сетей Петри (СП) для задания управления. Пусть переходы СП соответствуют фрагментам вычислений, а места СП – переменным. Наличие фишки в месте говорит о готовности переменной. Это позволяет задать необходимое потоковое управление. Ограничения в ресурсах могут требовать задания прямого управления, ужесточающего потоковое. Так, если при вычислении выражения $x = a \cdot b + c \cdot d$ ресурсы памяти не ограничены, то программа для вычисления выражения выглядит, как представлено на рис. 1, а её управляющая сеть представлена на рис. 2. Если же компьютер имеет

только ячейки памяти *x,m1,m2,* то программа изменится (рис. 3). Изменится также и управляющая сеть (рис. 4).

m1=a
m2=b
m3=c
m4=d
*m5=m1*m2*
*m6=m3*m4*
m7=m5+m6

Рис. 1. Программа без ограничений

m1=a
m2=b
*x=m1*m2*
m1=c
m2=d
*m1=m1*m2*
x=x+m1

Рис. 2. Программа с ограничениями

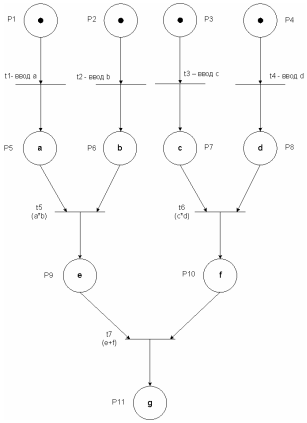


Рис. 3. Управляющая сеть Петри для программы без ограничений

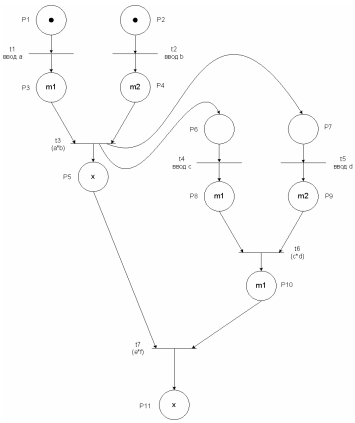


Рис. 4. Управляющая сеть Петри для программы с ограничениями

Видно, что из-за ограничения по ресурсам управляющая сеть перестраивается, и некоторые места сети уже не соответствуют переменным алгоритма. Так, места *P*6 и *P*7 отвечают за освобождение ячеек памяти *m*1 и *m*2. СП для программы без ограничений задает потоковое управление, СП для программы с ограничением по ресурсам задает более строгое потоковое управление. С точки зрения пользователя, программа выполняет один и тот же алгоритм, и значит обе управляющие сети должны вести себя одинаковым образом, т.е. если в первой сети переход A срабатывает после перехода B, то и во второй сети переход A должен сработать после B. Соблюдение этого условия можно автоматически проверять.

Для задания программы нужно описать отображение переменных и вычислений на ресурсы машины. Для программы без ограничений оно представлено в табл. 1, а для программы с ограничениями – в табл. 2.

Таблица 1. Программа без ограничений

Переменные алгоритма	Вычислительные ресурсы
A	M1
B	M2
C	M1
D	M2
E	X
F	M1
G	X

Таблица 2. Программа с ограничениями

Переменные алгоритма	Вычислительные ресурсы
A	M1
B	M2
C	M3
D	M4
E	M5
F	M6
G	M7

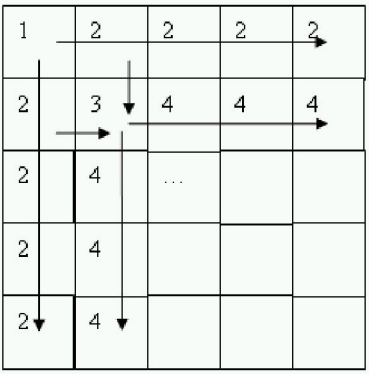


Рис. 5. Взаимодействия фрагментов в алгоритме LU-разложения

Рассмотрим задание управления на примере алгоритма фрагментированного LU-разложения [2]. На рис. 5 показана структура взаимодействия вычислительных фрагментов. Квадрат матрицы означает фрагмент вычислений.

Исходя из этих зависимостей, зададим управление СП (рис. 6). Приведенное управление статическое, но бывает, что нужно рекомендовать динамически генерируемое управление. Такая необходимость возникает, когда нельзя статически принять оптимальное решение по управлению. Для ускорения работы программы выгодно увеличение числа фрагментов, готовых к вычислениям. В этих целях фрагменты, соответствующие переходам, помеченным цифрой 2, должны исполниться быстрее помеченных цифрой 1. Однако, увеличивая число исполняющихся фрагментов, нужно помнить об ограничении ресурсов.

При записи реальных программ размер СП сильно увеличивается. Для решения этой проблемы в управляющей СП были выделены блоки повторяющихся операций.

Был разработан язык задания СП, в котором места ассоциируются ресурсными переменными, а переходы – с фрагментами вычислений.

Таким образом, был разработан язык записи ФП, который состоит из управляющей СП, таблицы отображения переменных на ресурсы и таблицы фрагментов.

При задании управления необходимо обеспечивать его корректность. Корректное управление – управление, обеспечивающее получение целевых переменных в ходе вычислений. Из-за некорректного управления при исполнении могут возникнуть следующие ошибки: дедлок, недостижимость нужного результата, недетерминированность результата, бесконечные циклы. При нахождении этих ошибок рассматривается вопрос о достижимости конечного результата.

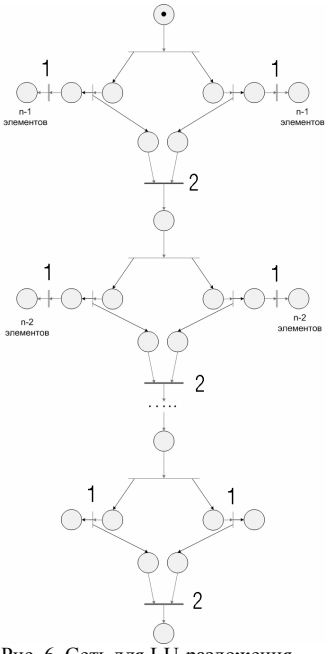


Рис. 6. Сеть для LU-разложения

Для проверки управления на корректность строится дерево достижимости, представляющее собой множество всех состояний программы. При анализе управления дерево достижимости проверяется на наличие требуемого состояния программы.

Разработана многопоточная реализация алгоритма построения дерева достижимости. Каждый поток строит свое поддерево, производится динамическая балансировка загрузки.

Заключение. Были разработаны язык для задания управления и частичного отображения ресурсов, транслятор СП, модуль проверки управления на ошибки, параллельная реализация алгоритма построения дерева достижимости.

В дальнейшем планируется разработка средства, реализующего удобное задание управления для программ, имеющих большую управляющую сеть; разработка визуальной среды для задания управления во ФП; интеграция с другими частями системы фрагментированного программирования.

Литература

1. **Arykov S., Malyshkin V.** Asynchronous Language and System of Numerical Algorithms Fragmented Programming // In proc. of the PaCT 2009, Novosibirsk, Russia, August 31 – September 4, 2009. Springer, LNCS 5698. 2009. P. 1-7.
2. **Чаюк К.Г**. Фрагментированный алгоритм LU-разложения и его параллельная реализация // Четвертая Сибирская школа-семинар по параллельным и высокопроизводительным вычислениям. Томск: Дельтаплан, 2008. С. 157-167.