XXXII Неделя науки СПбГПУ. Материалы межвузовской научно-технической конференции. Ч.V : С.95-97

УКД 004.052

Ле Куанг Минь (6 курс,. каф. АиВТ), А.С.Смирнов, д.т.н, проф.

РАЗРАБОТКА ПРОГРАММЫ ПОИСКА ВСЕХ ПУТЕЙ УСПЕШНОГО ФУНКЦИОНИРОВАНИЯ СИСТЕМЫ ДЛЯ РАСЧЁТА ПОКАЗАТЕЛЕЙ НАДЁЖНОСТИ СТРУКТУРНО-СЛОЖНЫХ СИСТЕМ

Для расчёта численных значений показателей надёжности, таких, например, как вероятность работоспособного состояния системы $P_c(t)$ и средняя наработка до отказа T_c для невосстанавливаемых и восстанавливаемых систем и в ряде других задач теории надёжности, самым важным этапом решения является этап получения функции работоспособного состояния (ФРС) структурно-сложных систем.

В работе [2] приведен алгоритм поиска путей успешного функционирования по матрице связности.

Дано: матрица связности А, матрица-столбец B_1, соответствующий стоку, номера стока и истока. Найти все пути, ведущие из истока в сток.

1) Берём столбец B_1 и проверяем, не содержит ли он дуг, начинающихся с истока. Если такие дуги есть, то выписываем их, запоминаем и заменяем в B_1 нулями. Полученный в результате такой замены столбец обозначаем B^*_1.
2) Полагаем $k = 2$.
3) Находим произведение $B_k = AB^*_{k-1}$, (1)
   где B_k – матрица-столбец, полученный при (k-1)-м умножении по формуле (1); B^*_{k-1} – матрица-столбец, полученный за счёт преобразования B_{k-1}.
4) Проверяем: a) наличие в B_k путей, начинающихся с истока, и если такие пути есть, то выписываем их, запоминаем и заменяем в B_k нулями; b) наличие путей в B_k, содержащих циклы, и если такие пути есть, то заменяем их в B_k нулями. Полученный после этих преобразований столбец обозначаем B^*_k.
5) Проверяем, не является ли столбец B^*_k нулевым: a) если $B^*_k = 0$, то переходим к шагу 7; b) если B^*_k не равен 0, то переходим к шагу 6.
6) Производим проверку условия $k \le n-1$: a) если $k < n-1$, то увеличиваем к на единицу и переходим к шагу 3; b) если $k = n-1$, то переходим к шагу 7.
7) Рассматриваем список полученных путей и производим замену дуг и вершин каждого пути элементами типа x_i, в результате чего получаем список элементарных конъюнкций. Записываем логическую сумму конъюнкций, которая будет представлять собой ФРС (при наличии в системе одного стока). Полученная функция алгебры логики (ФАЛ) будет записана в дизъюнктивной нормальной форме (ДНФ), в которой каждая элементарная конъюнкция соответствует определённому пути, имеющемуся в графе СЛС, а следовательно, и в структурной схеме системы.

Для полносвязанного графа не трудно доказать методом математической индукции, что количество путей между 2 любыми двумя вершинами графа n вершин является:

$$N(n) = A^0_{n-2} + A^1_{n-2} + A^2_{n-2} + ... + A^{n-3}_{n-2} + A^{n-2}_{n-2} \qquad (2)$$

где N – количество неповторных путей между двумя вершинами;
n – размер графа (количество вершин в графе),

$$A^i_m = \frac{m!}{(m-i)!} \qquad (3)$$

Для n = 4, очевидно, что N(n) = 1 + 2+2 = 5.

Надо отметить, что с увеличением размера графа функция N(n) резко увеличивается, так как N(n) пропорционально (n-2)!: N(4) = 5; N(5) = 15; N(6) = 65; N(7) = 326; N(8) = 1957.

Моделировать приведенный алгоритм поиска всех путей между двумя вершинами сложно, поскольку приведенный алгоритм только удобен при ручных расчетах для графов малого размера. По алгоритму надо проверять Bk, Bk* (4-й и 5-й шаги), но реализация программным путём этих этапов очень сложна, поскольку элементы матрицы Bk, Bk* являются переменными строчками. Исходя из этих сложностей, была поставлена задача разработать алгоритм и написать программу поиска всех путей между двумя вершинами графа. Эта задача сильно отличается от других известных задач по графам, так как в теории графов обычно ставиться задача поиска кратчайшего пути, или поиска оптимального пути, или поиска путей, включающих все вершины, и также поиска полных контуров (циклов). Так как надо найти все элементарные пути (маршруты) из одной вершины в другую, мы можем это сделать только способом рассмотрения всех возможностей переходов из каждой вершины в другую, которая не встречалась до этого момента в маршруте.

Алгоритм программы имеет вид «поиска по глубине», то есть из начальной вершины графа идем в другую вершину (в которую можно перейти из этой вершины), и затем дальше до конечной вершины. Если нет возможности дойти до конечной вершины, то необходимо вернуться обратно на предыдущую, от которой перейти по другим ветвлениям.

Разработанная программа написана на языке программирования C++. Реализация процедуры поиска путей использует метод рекурсивного программирования.

Используя разработанную программу, в работе получены результаты, полностью совпадающие с ожидаемыми.

<u>Пример 1:</u>

Для графа, изображенного на рис. 1

$$A1 = \begin{array}{c|cccc|} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 12 & 13 & 14 \\ 2 & 21 & 0 & 23 & 24 \\ 3 & 31 & 32 & 0 & 34 \\ 4 & 41 & 42 & 43 & 0 \\ \hline \end{array}; \qquad B1 = \begin{vmatrix} 14 \\ 24 \\ 34 \\ 0 \end{vmatrix}; \qquad B1^* = \begin{vmatrix} 0 \\ 24 \\ 34 \\ 0 \end{vmatrix}$$

В файле ввода DATA.TXT (соответствующий граф)

4	1	4	
0	1	1	1
0	0	1	1
0	1	0	1
0	0	0	0

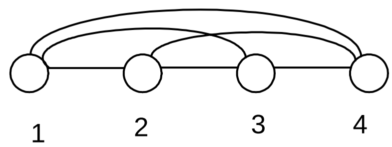

Рис 1.

В файле вывода OUT.TXT:

F(1,4) = 1 2 3 4 v 1 2 4 v 1 3 2 4 v 1 3 4 v 1 4

Полученный результат полностью совпадает с результата ручного расчета.

ЛИТЕРАТУРА:

1. Бендерская Е.Н., Колесников Д.Н. и др. Системный анализ и принятие решений. Учебное пособие. СПбГТУ, 2001.

2. Курочкин Ю.А., Смирнов А.С., Степанов В.А. Надежность и диагностирование цифровых устройств и систем. СПбГТУ, 1993.
3. Шилдт Г. Самоучитель С++, 3-е издание. BHV-СПб, 1998.
4. Вычислительные, измерительные и управляющие системы. Сборник научных трудов. Труды СПбГТУ №449. СПбГТУ, 1994.