УДК 004.052.3

И. А. Капчинский, П. В. Ковалев, С. Н. Гриценко

ГРАФОАНАЛИТИЧЕСКИЙ МЕТОД АНАЛИЗА МУЛЬТИВЕРСИОННЫХ АРХИТЕКТУР ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Предложена методика анализа мультиверсионных архитектур программного обеспечения, позволяющая использовать алгоритмы и методы анализа сетей на этапе определения надежности отказоустойчивого программного обеспечения.

Ключевые слова: надежность, мультиверсионность, сети, методы оценки, методы анализа сетей.

Компоненты программного обеспечения (ПО), обладающие свойством отказоустойчивости, уже много лет являются неотъемлемой составляющей как коммерческих, так и специальных систем управления и обработки информации. Известны различные методы проектирования отказоустойчивого программного обеспечения [1]. Хорошие перспективы в этой области имеет метод мультиверсионного проектирования [2].

Однако, учитывая сложность мультиверсионных систем управления и обработки информации и множество параметров системы, которые могут изменяться во времени, осуществлять прогноз времени завершения задач, и оценивать надежность системы, основываясь только на статических или детерминированных моделях систем или программ, достаточно сложно, а в некоторых случаях просто невозможно. Это обстоятельство представляется нам научной проблемой, выражающейся в необходимости поиска новых подходов к анализу надежности, а также временных характеристик работы программного обеспечения, построенного на основе мультиверсионной архитектуры.

Одним из таких подходов является графоаналитический метод, основанный на использовании GERT-сетей [3]. Основное достоинство этого подхода заключается в том, что он может быть успешно применен к решению практически любой задачи сетевого анализа и дает возможность составить формальные процедуры для определения вероятностно-временных характеристик системы.

Опишем модель мультиверсионного ПО, основанного на N-версионной архитектуре [4] (рис. 1) в виде GERT-сети (рис. 2), и рассчитаем ее основные характеристики при условии, что количество мультиверсий $N = 3$, а остальные параметры представлены в таблице.

Узлы стохастической сети (см. рис. 2) могут быть интерпретированы как состояния системы, а дуги – как переходы из одного состояния в другое. Такие переходы можно рассматривать как выполнение обобщенных операций, характеризуемых плотностью распределения, или функцией массы, и вероятностью выполнения [5].

Каждый внутренний узел стохастической сети выполняет две функции, одна из которых касается входа в узел, а другая – выхода. Обычно эти функции называют входной и выходной. Входная функция определяет условие, при котором узел может быть выполнен, а выходная – совокупность условий, связанных с результатом выполнения узла. Другими словами, с помощью выходной функции указывается, должны ли выполниться все операции, которым данный узел непосредственно предшествует, или только одна из них. Отметим, что начальный узел сети выполняет только выходную функцию, в то время как конечный узел – только входную.

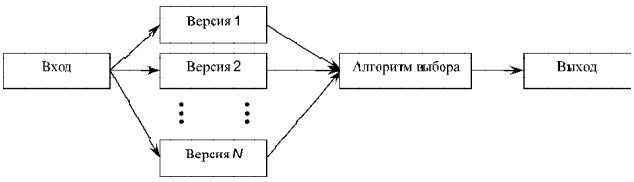

Рис. 1. Модель N-версионного программирования

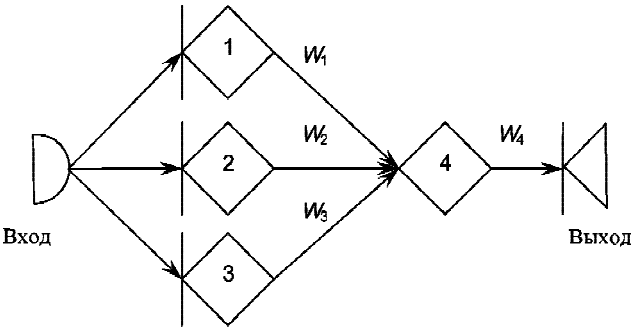

Рис. 2. Модель N-версионного программирования в виде GERT-сети

Существуют следующие виды входных функций:

– AND-функция (узел активируется, если выполнены все дуги, входящие в него);

– IOR-функция (узел активируется, если выполнена любая дуга, входящая в него);

Характеристика операций

Ветвь	Вероятность того, что операция будет выполнена p_i	Тип распределения	Параметры, мс
(1, 4)	0,85	Нормальное	$m = 0,5$ $\sigma = 0,1$
(2, 4)	0,85	Нормальное	$m = 0,5$ $\sigma = 0,1$
(3, 4)	0,85	Нормальное	$m = 0,5$ $\sigma = 0,1$
(4, выход)	0,99	Нормальное	$m = 2$ $\sigma = 0,5$

– EOR-функция (узел активируется, если выполнена любая дуга, входящая в него, при условии, что в данный момент времени может выполняться только одна дуга, входящая в данный узел).

Используются следующие виды выходных функций:

– детерминированная функция (все дуги, выходящие из узла, выполняются, если узел активирован);

– стохастическая функция (только одна дуга, выходящая из узла, выполняется с заданной вероятностью, если узел активирован).

Комбинируя все входные и выходные функции (рис. 3), получаем шесть различных типов узлов. Активация узла означает, что система перешла в некоторое состояние, и определяет множество возможных дальнейших действий. Одно или несколько действий начинают свое выполнение сразу после активации узла, являющегося их началом. Активация узла происходит, если его входная функция выполнена. После выполнения выходной функции активированного узла, т. е. начала выполнения соответствующей дуги, он становится неактивным [3].

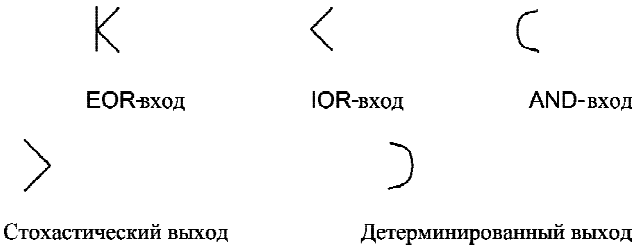

Рис. 3. Графическое обозначение входных и выходных функций GERT-сети

Необходимым и достаточным условием функционирования мультиверсионного модуля ПО является выполнение хотя бы одной мультиверсии. Поэтому узел 4 на рис. 2 является узлом с IOR-входом. Выбор данного типа узла основан на том, что этот узел активируется при выполнении любой дуги, входящей в него.

Рассмотрим методику расчета сети, содержащей IOR-вход и детерминированный выход (рис. 4) и состоящей из нескольких подсетей (рис. 5).

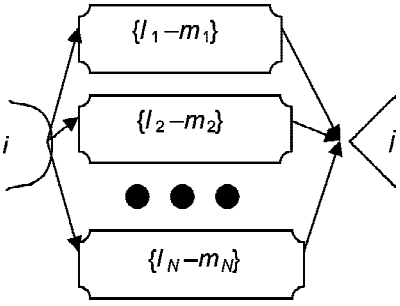

Рис. 4. IOR-вход: j – вычисляемый IOR-вход, имеющий стохастическое начало в узле i; $\{l_1 - m_1\}$, $\{l_2 - m_2\}$, …, $\{l_N - m_N\}$ – N непересекающихся подсетей

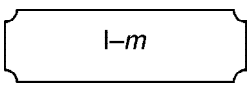

Рис. 5. Произвольная подсеть с начальным узлом l и конечным узлом m

Пусть p_{ij} – вероятность того, что операция (i, j) будет вы-полнена при условии, что узел i выполнен. Тогда по представленной выше таблице имеем:

$$p_j = p_i\,[1-(1-p_{l1,m1k})\times \\ \times(1-p_{l2,m2k})...(1-p_{l_N,m_Nk})]. \qquad (1)$$

Учитывая, что от входа до узла 4 находится три непересекающиеся подсети, используя выражение (1), в данном частном случае получим

$$p_4 = p_{\text{вх}}\,[1-(1-p_1)\,(1-p_2)\,(1-p_3)\,] = \\ = 1-\left(1-0{,}85\right)^3 = 0{,}996\,625. \qquad (2)$$

Воспользовавшись правилом Мейсона для замкнутых потоковых графов, выведем эквивалентную W-функцию [5] для этой сети:

$$W_E(s) = W_\Sigma(s)\cdot W_4(s) = \\ = 0{,}996\,6e^{0{,}5s+\frac{1}{2}0{,}1^2\cdot s^2}\cdot 0{,}99e^{2s+\frac{1}{2}0{,}5^2\cdot s^2} = \\ = 0{,}986\,6e^{2{,}5s+0{,}13s^2}. \qquad (3)$$

Легко проверить, что для рассматриваемой задачи $W_E(0) = 0{,}986\,6$. Данную величину можно интерпретировать как вероятность безотказной работы мультиверсионного ПО. В свою очередь математическое ожидание времени выполнения сети составит

$$\mu_{1E} = \left.\frac{\partial M_E(s)}{\partial s}\right|_{s=0} = \left.\frac{\partial\left(\dfrac{W_E(s)}{W_E(0)}\right)}{\partial s}\right|_{s=0} = 2{,}5 \text{ мс}, \qquad (4)$$

а дисперсия будет

$$\mu_{1E} = \left.\frac{\partial^2 M_E(s)}{\partial s^2}\right|_{s=0} = 6{,}51 \text{ мс}^2. \qquad (5)$$

Следует заметить, что при расчете сети, содержащей IOR-вход и детерминированный выход, если все подсети представляют собой мультиверсионный модули с одинаковой надежностью, то формула (1) может быть сведена к виду

$$p_j = p_i\cdot[1-(1-p_{1,m})^n]. \qquad (6)$$

Графическая зависимость между числом мультиверсионных модулей и надежностью системы приведена ниже (рис. 6). На оси X обозначено количество мультиверсий, на оси Y – надежность системы для разного количества модулей с разной надежностью каждого модуля (от 30 до 80 %).

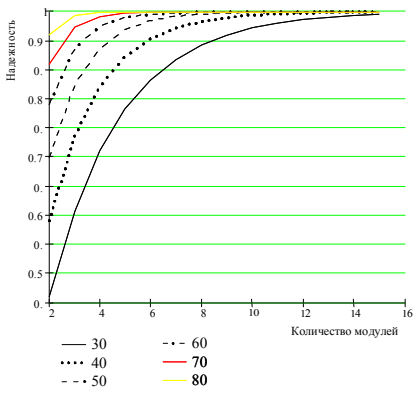

Рис. 6. Зависимость между числом мультиверсионных модулей и надежностью системы

Таким образом, использование графоаналитического метода на основе GERT-сетей для анализа мультиверсионных архитектур программного обеспечения является перспективным, так как позволяет аналитически оценить вероятностно-временные характеристики мультиверсионного ПО любой сложности без построения громоздких имитационных сред и комплексов моделирующих программ.

Библиографический список

1. Kovalev, I. System of Multi-Version Development of Spacecrafts Control Software / I. Kovalev. Sinzheim : Universitate Verlag, 2001.

2. Ковалев, И. В. Мультиверсионный метод повышения программной надежности информационно-телекоммуникационных технологий в корпоративных структурах / И. В. Ковалев, Р. В. Юнусов // Телекоммуникации и информатизация образования. 2003. № 2 (15). С. 50–56.

3. Ковалев, П. В. Определение надежности мультиверсионного программного обеспечения с использованием методов анализа сетей / П. В. Ковалев, А. Н. Лайков, С. Н. Гриценко // Вестник СибГАУ. 2009. № 1 (22). Ч. 2. С. 55–60.

4. Avizienis, A. The Methodology of *N*-Version Programming/ A. Avizienis // Software Fault Tolerance. New York : John Wiley & Sons, 1995.

5. Филлипс, Д. Методы анализа сетей / Д. Филлипс, А. Гарсиа-Диас. М. : Мир, 1984.

I. A. Kapchinsky, P. V. Kovalev, S. N. Gritzenko

GRAPH-ANALYTIC RESEARCH OF SOFTWARE BASED ON *N*-VERSION ARCHITECTURE

In this article the research technique the N-version software reliability is offered. It allows using algorithms and methods of network analysis for the reliability research of software, developed using N-version approach.

Keywords: reliability, N-version, networks, estimation methods, network analysis.

УДК 681.3

Е. А. Энгель, О. И. Завьялова

ИСПОЛЬЗОВАНИЕ ИЕРАРХИЧЕСКИХ НЕЙРОННЫХ СЕТЕЙ ДЛЯ РАСПОЗНАВАНИЯ МНОГОЭЛЕМЕНТНЫХ ЗРИТЕЛЬНЫХ СЦЕН

Описана иерархическая искусственная нейронная сеть для решения задач распознавания образов, сгруппированных в произвольную сцену. Разработана математическая модель системы распознавания связанных фрагментов образов на основе взаимодействия подсистем внимания и распознавания.

Ключевые слова: нейронная сеть, иерархия, распознавание.

Существующие в настоящее время системы распознавания образов на базе искусственных нейронных сетей (ИНС) обладают значительным количеством недостатков. В частности при разработке комплексов, решающих проблему интерпретации изображений, особое внимание уделяется узнаванию определенных групп образов. Однако поиск в доступных источниках информации о системах, способных к саморазвитию, т. е. к «расширению кругозора», приводит к крайне скудным результатам. Это объясняется высокой сложностью проектирования и реализации проектов такого класса. Ведь на самом деле реальный процесс распознавания, протекающий в самой совершенной системе анализа – человеческом мозге, состоит не только в том, чтобы проверить сходство анализируемого объекта с запомненным эталоном. Это достаточно сложное взаимодействие между различными подсистемами мозга. Ошибочно также считать, что сигналы от сетчатки до терминальных корковых центров, принимающих решение, распространяются прямолинейно. Распознавание – это итеративный процесс, в котором до принятия окончательного решения происходит не только сравнение входного образа с эталоном, но и генерация гипотез по классификации объекта. Столкновение соответствующих потоков нейронных импульсов приводит к дальнейшему уточнению характеристик образа до тех пор, пока не произойдет согласования между этими потоками.

Кроме того, в процессе классификации нередко случается так, что наш мозг не содержит точного представления эталона, однако это не мешает ему найти правильный результат.

Другим немаловажным аспектом создания систем распознования образов является выделение существенных фрагментов сцены для их интерпретации, т. е. селекция образов, расположенных на сцене. В противном слу-