Секция «Вычислительная математика и кибернетика»

Применение генетических алгоритмов для тестирования модулей ядра ОС Linux

Туманян Ани Гагиковна
Студент
РАУ-Российско-Армянский(Славянский) Университет , Факультет Прикладной Математики и Информатики, Ереван-Yerevan, Армения
E-mail: ani.tumanyan92@gmail.com

Операционные системы, основанные на ядре Linux, становятся все более популярными, особенно на рынках суперкомпьютеров, серверов и мобильных устройств. Ядро Linux распространяется как свободное ПО на условиях GNU General Public License. Из-за перечисленных факторов верификация как самого ядра, так и модулей Linux является актуальной задачей[4]. Решение этой задачи связано с определенными трудностями, обусловленными тем, что режимы работы модулей и драйверов ОС зависят от большого числа факторов, связанных с состоянием самого ядра ОС. Следовательно, тестовый набор, наиболее полно проверяющий работу модуля, должен проверять взаимодействие этих факторов между собой и их влияние на поведение модуля. Для построения такого набора перебор всех комбинаций значений факторов требует слишком больших затрат, поэтому в данной работе используется генетический алгоритм (ГА), который имеет широкое применение для решения задач оптимизации, обладающих высокой вычислительной сложностью[3].

Существуют различные критерии, оценивающие качество тестирования - полноту тестового набора. Критериями полноты тестового набора нами выбраны: выполнение всех строк и покрытие ветвей в графе потока управления[1]. Следовательно элементами покрытия являются как ветви управления (ребра графа), так и строки кода (вершины графа).

Тест представляет собой вызов интерфейсных функций модуля ядра с соответствующими аргументами при определенных состояниях различных факторов, влияющих на его работу. Хромосома в терминах ГА — представление теста, с которым и будет работать ГА. В ней закодированы: вызов определенных функций как строка из 0 и 1 (вектор вызова), аргументы этих функций, закодированные с помощью веществennознаковых или двоичных векторов, а также состояния влияющих факторов. Множество хромосом — популяция. Оператор кроссовера выполняет обмен частей между векторами вызова или между компонентами, соответствующих аргументам некоторой вызываемой функции. Мутация - инвертирование гена в векторе вызова или допустимое изменение значения элемента хромосомы, соответствующей аргументу некоторой функции. При создании новых хромосом проверяется допустимость соответствующих им тестов для модуля. Функция приспособленности хромосомы для вершины графа определяется как минимальное расстояние от покрытых соответствующим ей тестом элементов до данного элемента с учетом метрик покрытия. Покрытие вычисляется программой gcov, входящей в GCC[5]. Для ребер графа функция приспособленности определяется как вероятность прохождения по нему[2]. Искомый тестовый набор формируется по ходу эволюции из тестов, покрывающих ранее непокрытые элементы тестового покрытия.

С помощью данного подхода реализовано построение тестовых наборов для драйверов символьных устройств. Планируется его применение для тестирования драйверов файловых систем ОС Linux и сравнение с другими методами тестирования по затратности ресурсов и по качеству покрытия.

Литература

1. Владимиров М.А. Критерии полноты тестового покрытия в генетических алгоритмах генерации тестов //Труды ИСП РАН, 2006. No 9. С 57-66.

2. Paolo Tonella. Evolutionary testing of classes. // ISSTA '04: Proceedings of the 2004 ACM SIGSOFT international symposium on Software testing and analysis, pp 119-128, New York, NY, USA, 2004.

3. S.N. Sivanandam, S.N. Deepa Introduction to Genetic Algorithms. Spinger-Verlag,Berlin,2008.

4. Сайт "Linux Documentation Project " http://tldp.org/LDP/intro-linux/html/chap_01.html.

5. Документация по Gcov: http://gcc.gnu.org/onlinedocs/gcc/Gcov.html