электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

УДК 004.052.42

Вероятностный метод выбора тестируемых путей выполнения функции

***Медников А.В.**, магистр*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Программное обеспечение ЭВМ и информационные технологии»

Научный руководитель: Рудаков И. В., к.т.н., доцент
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана
irudakov@bmstu.ru

Введение

С целью контроля качества в процессе разработки ПО производится тестирование. Одним из видов тестирования является модульное тестирование, при котором выполняется изолированная проверка функций и модулей программы. При проведении модульного тестирования программист реализует набор тестов, которые проверяют функцию или модуль на соответствие предъявляемым требованиям. Для выполнения теста функции необходимо задать некоторые входные данные, которые, как правило, задаются вручную.

Одним из подходов к генерации тестовых данных является символьное выполнение. Рассмотрим символьное выполнение на примере функций.

В процессе выполнения программы каждой переменной в каждый момент времени соответствует некоторое значение. *Символьным выполнением*[2, 3] называется подход к анализу программы, при котором производится интерпретация инструкций программы с заменой переменных *символьными переменным*. Символьная переменная хранит не фиксированное значение, а *выражение, зависимое от других символьных переменных или констант*. Символьное выполнение не является выполнением программы в привычном смысле, когда программа выполняется на заданных входных данных. Множество символьных переменных и их значений образуют состояние символьного выполнения.

Одним из недостатков символьного выполнения является возможность экспоненциального увеличения числа анализируемых путей выполнения при невозможности определения базового блока, которому должно быть передано управления после условного переход[3]. Данная проблема вызвана тем, что символьное выполнение является статическим подходом к анализу программ и в процессе работы недоступна информация времени выполнения программы

Существующие подходы к решению данной проблемы[3, 4] не ставят цель выбора путей, в которых возможно возникновения ошибок определенного типа. Например, в [4] предлагается уменьшить количество рассматриваемых путей, отбросив те пути, у которых наблюдаются те же побочные эффекты, что и у уже рассмотренных путей.

В данной работе предлагается подход к выбору тестируемых путей выполнения на основе графа потока управления функции для обнаружения ошибок использования указателей.

Определение пути выполнения функции через граф потока управления

Одним из используемых представлений функции является граф потока управления (ГПУ)[4]. ГПУ является ориентированным графом, вершинами которого являются базовые блоки, а ребра указывают порядок следования базовых блоков в потоке управления. Таким образом, ГПУ позволяет отобразить передачу потока управления при выполнении функции. Пример исходного кода и графа потока управления функции представлены в листинге 1 и на рис. 1 соответственно.

Листинг 1. Функция CountSymbols

```
unsigned int CountSymbols(char * szPath, char chSymbol)
{
	FILE * fd = fopen(szPath,"r");
	unsigned int nCount = 0;
	if (fd)
	{
		char c;
		do
		{
			if ((c = fgetc(fd)) == chSymbol)
			{
				nCount++;
			}
		}
		while (c != EOF);
		fclose(fd);
	}
	return nCount;
}
```

Выполнению функции на заданных входных данных соответствует некоторый путь в ГПУ. Определим *путь выполнения как путь в ГПУ функции, начинающийся входным*

базовым блоком и завершающийся выходным базовым блоком.

В качестве ограничения не рассматриваются функции, содержащие бесконечные циклы, поскольку в данном случае количество базовых блоков в пути выполнения может неограниченно увеличиваться, что увеличивает вероятность выбора нереализуемого пути.

Выбор пути выполнения

Путь P состоит из последовательности базовых блоков $\{BB_{i_0}, BB_{i_1}, \dots, BB_{i_n}\}$, в которой $BB_{i_0} = BB_{ENTER}$ - входной базовый блок, $BB_{i_0} = BB_{EXIT}$ - выходной базовый блок, причем $\forall\, BB_{i_j}$, $BB_{i_{j+1}}$ существует ребро $(BB_{i_j}, BB_{i_{j+1}})$ в ГПУ. Выбор пути P может выполняться последовательным формированием списка базовых блоков в результате обхода ГПУ.

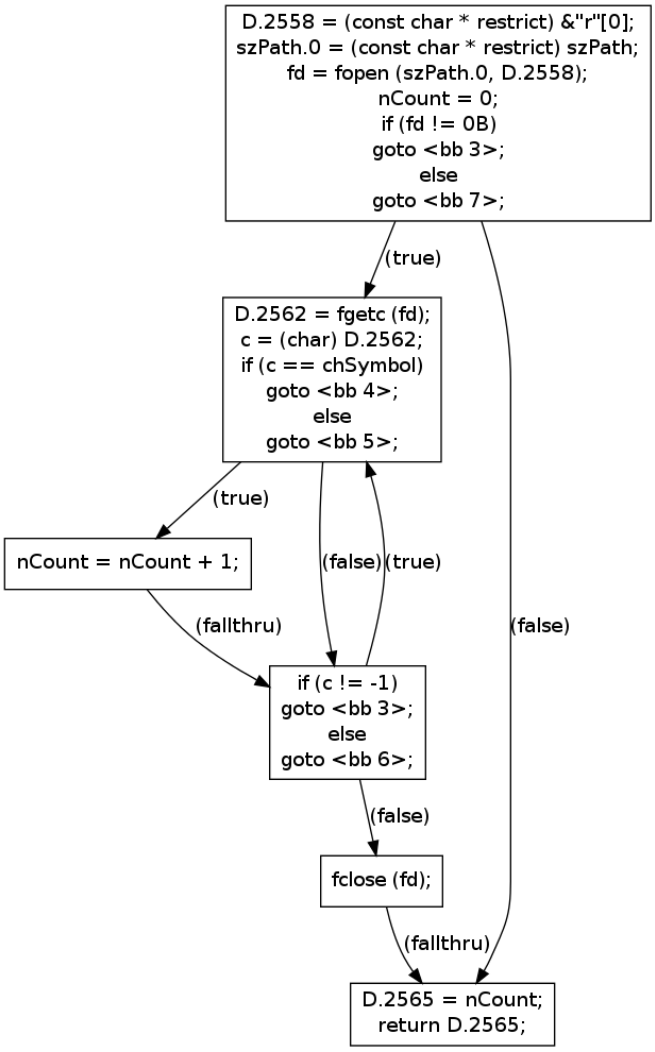

Рис. 1. Граф потока управления функции CountSymbols

Учитывая требование $BB_{i_0} = BB_{ENTER}$, следует формировать список базовых блоков пути, начиная с входного базового блока, выполняя обход графа методом поиска в

глубину до достижения выходного базового блока. Пройденные в результате данного обхода вершины составят выбранный путь.

Учитывая структуру ГПУ необходимо определить:

1. Способ предотвращения зацикливания работы алгоритма при наличии контура в ГПУ.
2. Правило выбора базового блока, которому передается управление после условного перехода.

Рассмотрим связь между циклами в программе и структурой ГПУ. Для каждого цикла в функции существует контур в ГПУ этой функции. Поскольку функции с бесконечными циклами не рассматриваются, то для любого цикла будет существовать базовый блок, из которого существуют две альтернативы передачи управления — завершение цикла или переход к очередной итерации цикла. Поскольку длина тестируемого пути ограничена, то в путь будет включено фиксированное количество повторений контура, соответствующего итерации цикла, после чего будет включен базовый блок, на который выполняется передача управление при завершении цикла.

Рассмотрим подробнее выбор следующего базового блока в ГПУ при условной передаче управления.

Выбор базового блока следующего после условного перехода в потоке управления

Выбор базового блока, на который будет передано управление, зависит от значения логического условия в условном переходе (рис. 2). В данном случае необходимо выбрать один из двух базовых блоков, который будет включен в список базовых блоков пути.

Необходимо отметить, что если выбор базового блока будет выполняться детерминировано, то при каждом запуске алгоритма для одной и той же функции будет получен один и тот же путь выполнения, что сократит область поиска ошибок.

В качестве альтернативы детерминированному подходу может быть применен вероятностный подход выбора базового блока, при котором для базовых блоков $<$ *Базовый блок* $T>$ и $<$ *Базовый блок* $F>$ необходимо задать вероятности выбора блоков $SelectProb_T$ и $SelectProb_F$ соответственно. Поскольку для условного перехода существуют две альтернативы передачи управления, то для заданных вероятностей должно выполняться равенство: $SelectProb_T + SelectProb_F = 1$.

Далее необходимо сформулировать правило расчета вероятности выбора базового блока.

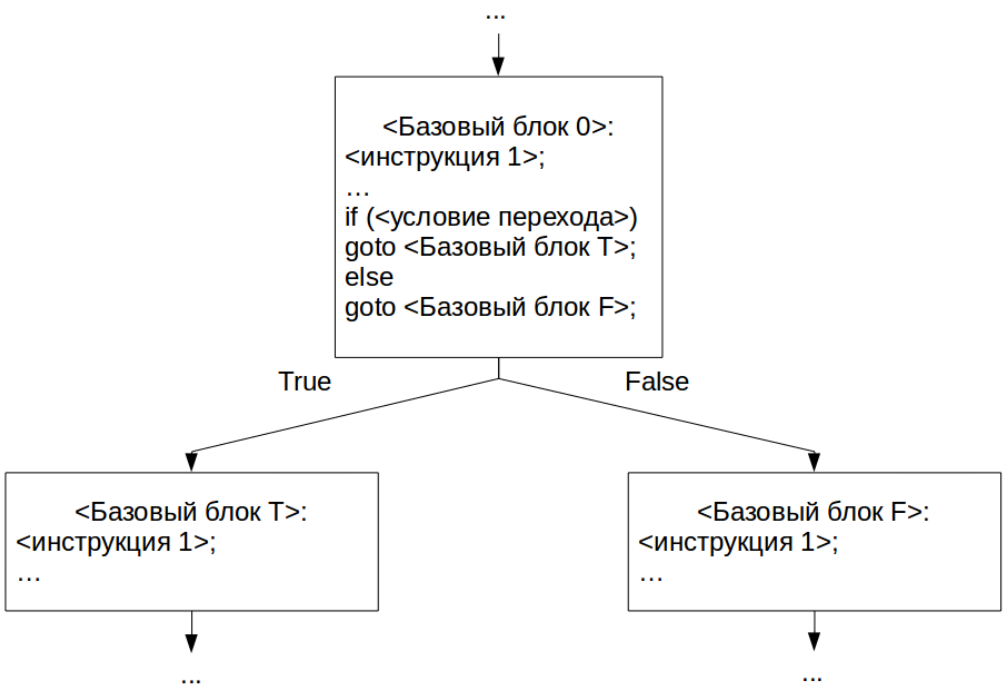

Рис. 2. Схема базового блока с условной передачей управления

Вычисление вероятности выбора базового блока при условной передаче управления

Определим требования, предъявляемые к правилу вычисления вероятностей выбора $SelectProb_T$ и $SelectProb_F$ для базовых блоков $<$ *Базовый блок* $T>$ и $<$ *Базовый блок* $F>$:

1. Для вычисленных вероятностей должно выполняться равенство: $SelectProb_T + SelectProb_F = 1$.
2. Вероятность выбора должна быть тем больше, чем больше вероятность возникновения ошибки использования указателей в данном базовом блоке.
3. Для вычисленных вероятностей должно выполняться неравенство: $0 < SelectProb_i < 1, i \in \{T, F\}$.

Последнее требование учитывает тот факт, что вне зависимости от базовых блоков $<$ *Базовый блок* $T>$ и $<$ *Базовый блок* $F>$ процесс выбора базового блока должен

оставаться вероятностным. Действительно, при отсутствии последнего требования, значения $SelectProb_T = 1, SelectProb_F = 0$ удовлетворяют поставленным требованиям, но при данных значениях выбор базового блока будет детерминированным — при каждом выполнении алгоритма будет выбираться базовый блок $<$ *Базовый блок* $T >$.

Каждый базовый блок содержит последовательность трехадресных инструкций $BbInstructionSeq$, выполняемую в результате передачи управления на данный базовый блок. В $BbInstructionSeq$ можно выделить множество инструкций $BbPointerInstructionSet$, выполняющих операции над указателями: чтение значения указателя, запись значения в указатель, разыменования указателя и другие. Вероятность возникновения ошибки использования указателей тем больше, чем больше мощность множества $BbPointerInstructionSet$.

В работе для расчета вероятности выбора $SelectProb_T$ базового блока $<$ *Базовый блок* $T >$ используется формула (1):

$$SelectProb_T(P_T, P_F) = \begin{cases} d + \dfrac{P_T}{P_T + P_F} \cdot (1 - 2 \cdot d), & \text{при } P_T + P_F > 0 \\ 0.5, & \text{при } P_T + P_F = 0 \end{cases} \quad (1), \text{где:}$$

$P_i = PtrInstrDensity_i = \frac{PtrInstrCount_i}{InstrCount_i}$,

$PtrInstrCount_i$ - количество трехадресных инструкций в $<$ *Базовый блок* $i >$, в которых выполняются операции над указателями,

$InstrCount_i$ - общее количество трехадресных инструкций в $<$ *Базовый блок* $i >$,

$InstrCount_i \neq 0$,

$i \in \{T, F\}$;

d - минимальное значение вероятности выбора, $0 < d < 1$;

$\forall P_T, P_F, d: SelectProb_T + SelectProb_F = 1$.

Общее количество инструкций в $<$ *Базовый блок* $T >$ равно $InstrCount_T = |BbInstructionSeq|$; количество инструкций, выполняющих операции над указателями равно $PtrInstrCount_T = |BbPointerInstructionSet|$.

Покажем, что формула (1) удовлетворяет предъявленным требованиям.

Согласно выводу, приведенному в формуле (2), формула (1) удовлетворяет требованию 1.

$$SelectProb_T + SelectProb_F = d + \frac{P_T}{P_T + P_F} \cdot (1 - 2 \cdot d) + d + \frac{P_F}{P_T + P_F} \cdot (1 - 2 \cdot d)$$

$$= 2 \cdot d + \frac{(P_F + P_T) \cdot (1 - 2 \cdot d)}{P_T + P_F} = 1 \ (2)$$

Вероятность возникновения ошибки использования указателя в базовом блоке возрастает при увеличении количества инструкций, использующих указатели. Отметим, что функция $SelectProb_T(P_T, P_F)$ является строго возрастающей относительно переменной P_T - при $v_1 > v_2$ будет выполняться неравенство $SelectProb_T(v_1, P_F) > SelectProb_T(v_2, P_F)$. Соответственно требование 2 удовлетворяется.

Для удовлетворения требованию 3 в формулу (1) введена минимальная вероятность выбора d. В таблице приведены значения функции $SelectProb_T(P_T, P_F)$ в граничных точках области определения функции, из которых видно, что при $0 < P_T + P_F < 1$ значения функции находятся в отрезке $[d, 1 - d]$.

P_T	P_F	$SelectProb_T(P_T, P_F)$
0	0	0.5
0	1	d
1	0	$1 - d$
1	1	0.5

Заданные значения вероятностей позволяют выбрать тот базовый блок, вероятность возникновения ошибки в котором больше, но не гарантируют завершения работы алгоритма – при наличии цикла, вероятность выбора блока внутри цикла может быть существенно больше вероятности выбора блока, которому передается управления после завершения цикла, что может вызвать существенное увеличение длины выбранного пути. С увеличением длины пути увеличивается вероятность того, что данный путь является недостижимым.

Для ограничения длины пути предлагается ограничить максимальное количество повторений каждого базового блока в пути значением параметра $BbMaxRepeat$. Если базовый блок встречается в пути $BbMaxRepeat$ раз, то выполнение алгоритма завершается, и путь не выбирается. Рассмотрим данную ситуацию подробнее.

Динамическое изменение вероятности выбора базового блока

При выборе пути в ГПУ, изображенном на рис. 3, выбор базового блока 1 после блока 4, будет выполняться чаще блока 5. При повторении блока 1 в выбираемом пути $BbMaxRepeat$ раз, процесс формирования списка базовых блоков пути прекратится, хотя вероятность возникновения ошибки использования указателей, согласно используемому критерию, в блоке 1 в $\frac{P_{41}}{P_{45}} = 19$ раз больше чем в блоке 2.

Для обработки данной ситуации предлагается динамически изменять вероятности выбора базовых блоков в процессе работы алгоритма: после включения в список базовых блоков пути блока 1 уменьшим P_{41} на некоторую величину s и увеличим P_{45} на s. В результате, при последующем выборе следующих за блоком 4 в потоке управления блоков вероятность выбора блока 5 увеличиться, что влечет завершение работы алгоритма с формированием пути. Критерий выбора значения s в данной работе не рассматривается.

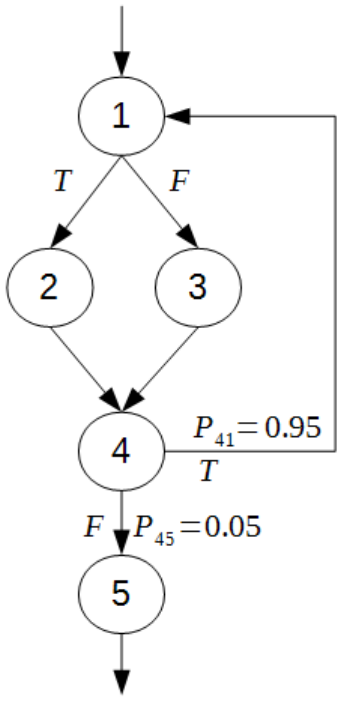

Рис. 3. Пример тестируемого алгоритма с установленными вероятностями выбора базовых блоков 1 и 5

Заключение

В работе предложен вероятностный подход к выбору путей выполнения на основе графа потока управления функции. Предложено правило вычисления вероятностей

выбора базовых блоков при условном переходе с учетом поиска ошибок использования указателей.

Работа выполнена при частичной поддержке Российского фонда фундаментальных исследований (грант № 13-07-00918).

Список литературы

1. Coward P.D. Symbolic execution systems - a review // Software Engineering Journal. 1988. No. 3(6). P. 229-239.
2. Godefroid P. Test Generation Using Symbolic Execution // IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (Hyderabad, December, 15-17, 2012 г.): proceedings. Hyderabad, 2012. P. 24-33.
3. Boonstoppel P., Cadar C., Engler D. RWset: Attacking Path Explosion in Constraint-Based Test Generation // ETAPS Conference on Tools and Algorithms for the Construction and Analysis of Systems (Budapest, March 29-April, 2008г.). Budapest, 2008. P. 351-366.
4. Ахо А.В., Лам М.С., Сети Р., Ульман Д.Д. Компиляторы. Принципы, технологии и инструментарий: пер. с англ. / под ред. И.В. Красикова. М.: ООО «И.Д. Вильямс», 2008. 1184 с. [Aho A.V., Lam M.S., Sethi R., Ullman J.D. Compilers. Principles, Techniques & Tools. 2nd ed. Addison-Wesley, 2006. 1000 p.].