ИНФОРМАТИКА И ЕЁ ПРИМЕНЕНИЯ, 2013. Т. 7. Вып. 2. С. 50–54

СТАТИСТИЧЕСКАЯ ПРОВЕРКА НЕИСПОЛНЯЕМОСТИ ФРАГМЕНТОВ КОДА ПОСЛЕДОВАТЕЛЬНОЙ ПРОГРАММЫ*

В. Ю. Королев[1], Р. Л. Смелянский[2], Т. Р. Смелянский[3], А. В. Шалимов[4]

Аннотация: Рассматривается задача статистической проверки неисполняемости фрагментов кода последовательной программы. Рассматриваются методы, основанные на минимизации как априорных, так и апостериорных вероятностей ошибок.

Ключевые слова: проверка статистических гипотез; геометрическое распределение; вероятность ошибки первого рода; вероятность ошибки второго рода; лемма Неймана–Пирсона; апостериорная вероятность ошибки

1 Введение

В данной статье рассматривается задача статистической проверки неисполняемости фрагментов кода последовательной программы. Такая задача возникает в некоторых приложениях, связанных с необходимостью оптимизации размера программы. Эта проблема весьма актуальна при создании программного обеспечения миниатюрных устройств, в частности таких, как кардиостимуляторы или кардиомониторы, а также при создании бортового программного обеспечения специального назначения, когда небольшой физический объем носителя программы вынуждает оптимизировать объем самой программы. При этом, естественно, потери функциональности программы должны быть незначительными.

В данной статье используется строгое определение частоты выполнения фрагмента кода последовательной программы, предложенное и проанализированное в работе [1]. Под фрагментом кода понимается линейный участок программы, то есть такой участок кода, где не нарушается естественный порядок выполнения.

Рассматриваемая задача имеет тесную формальную связь с областью профилировки программ [2, 3], где возникает задача определения частоты выполнения инструкций программы. Существующие решения в этой области сводятся к применению различных эмпирических данных или к предположению, что все входные данные программы равномерно распределены на области значений, что в общем случае не всегда верно. Данная статья продолжает исследования методов оценивания частоты выполнения фрагментов кода последовательных программ, начатые в работах [1, 4].

2 Описание задачи

Пусть $\Pi(x_1, \ldots, x_p) = \{V, E\}$ — исходная последовательная программа с p входными параметрами $x_1, \ldots, x_p$. Программа представлена в виде графа потока управления с вершинами $V = \{b_j\}$, где b_j — линейные участки программы, $j = 1, \ldots, m$, и дугами $E = \{(b_{j_1}, b_{j_2})\}$, где (b_{j_1}, b_{j_2}) — переход управления от j_1-го линейного участка к j_2-му линейному участку.

Конкретное значение параметра x_i будем обозначать $\hat{x}_i$, $i = 1, \ldots, p$. Множество $\mathcal{X} = \{(\hat{x}_1, \ldots \ldots, \hat{x}_p)\}$ всех допустимых входных параметров программы будем считать метрическим пространством, что позволит формально рассмотреть борелевскую σ-алгебру его подмножеств и определить на ней распределение вероятностей P, которое позволяет интерпретировать вектор $(\hat{x}_1, \ldots, \hat{x}_p)$ как значение случайного элемента, принимающего значения в $\mathcal{X}$ и имеющего известное распределение P. Заметим, что, вообще говоря, в силу дискретного характера представления информации в компьютере для

*Работа поддержана Российским фондом фундаментальных исследований (проекты 11-01-00515а, 11-07-00112а, 11-01-12026-офи-м, 12-07-00109а).

[1]Факультет вычислительной математики и кибернетики Московского государственного университета им. М. В. Ломоносова; Институт проблем информатики Российской академии наук, victoryukorolev@yandex.ru

[2]Факультет вычислительной математики и кибернетики Московского государственного университета им. М. В. Ломоносова, smel@sc.msu.ru

[3]Факультет вычислительной математики и кибернетики Московского государственного университета им. М. В. Ломоносова, smelyanskiy.t@bk.ru

[4]Факультет вычислительной математики и кибернетики Московского государственного университета им. М. В. Ломоносова, ashalimov@lvk.cs.msu.su

каждого входного параметра $x_1, \ldots, x_p$ множество допустимых значений является конечным, так что упомянутое выше множество $\mathcal{X}$ конечно, а распределение входных значений дискретно. Однако для удобства будем рассматривать более общую модель, поскольку для многих реальных программ мощность $|\mathcal{X}|$ множества $\mathcal{X}$ может иметь порядок 10^{100} [5], что, например, превосходит число атомов водорода (наиболее распространенного химического элемента) в видимой части Вселенной, которое по современным оценкам составляет примерно 10^{48} [6]. Поэтому методы теории вероятностей и математической статистики, ориентированные на конечные модели, в рассматриваемом случае неэффективны.

Приведем простой пример, свидетельствующий в пользу рассмотрения в данной задаче общих (в том числе непрерывных) вероятностных моделей. Рассмотрим входной параметр, отвечающий за показания альтиметра (пилотажно-навигационного прибора, указывающего высоту полета). Погрешности показаний альтиметра, которые, безусловно, имеются и обусловлены суммарным воздействием большого числа случайных факторов, могут считаться случайными и, в силу центральной предельной теоремы теории вероятностей, нормально распределенными с дисперсией, обусловленной точностью прибора.

Рассматриваемая постановка задачи восходит к работам [7, 8], в которых впервые дана ее общая строгая формализация. Позднее с развитием техники профилировки программ появились частные случаи этой постановки [2, 3]. Будем придерживаться уточненной формализации стохастического функционирования программы, предложенной в [9] (также см., например, [10]).

Будем считать, что программа $\Pi(x_1, \ldots, x_p)$ не зацикливается на допустимых наборах входных данных, т. е. каждый линейный участок выполняется конечное число раз. Введем обозначение $\Pi_j(\hat{x}_1, \ldots, \hat{x}_p)$ для числа выполнений j-го линейного участка на векторе входных значений $\hat{x}_1, \ldots, \hat{x}_p$.

Итак, пусть $X = (X_1, \ldots, X_p)$ — случайный элемент со значениями в $\mathcal{X}$, интерпретируемый как случайный вектор входных значений программы. Пусть $Y_j = \Pi_j(X_1, \ldots, X_p)$ — соответствующее число выполнений j-го линейного участка программы. Очевидно, что Y_j — это дискретная случайная величина, принимающая значения в $\mathbb{N} \cup \{0\}$ и имеющая собственное распределение (т. е. принимающая бесконечное значение с нулевой вероятностью). Математическое ожидание, соответствующее распределению P, будет обозначаться E.

Возможно несколько подходов к вычислению вероятности активизации (выполнения) линейного участка b_j. В работах [7, 8] предложены методы построения функции распределения числа выполнений линейных участков программы по распределениям входных параметров в аналитическом виде. Фактически используются формулы свертки и при реализации требуются значительные вычислительные затраты. В работе [1] предложен подход, позволяющий не вычислять точное значение частоты выполнения, так как это сопряжено с большими вычислительными затратами, сравнимыми с прогонами программы на всех входных данных, а оценивать значение вероятности выполнения с заранее заданной точностью. Для оценки вероятности выполнения линейного участка, которая заключается в приближенном вычислении математического ожидания бернуллиевой случайной величины, в [1] предложено использовать метод статистических испытаний (метод Монте-Карло). Идея этого метода заключается в проведении многократных опытов с исследуемой случайной величиной, и на основе полученных значений делается вывод о различных характеристиках этой случайной величины.

3 Метод, основанный на лемме Неймана–Пирсона

Проведем n независимых испытаний, в результате каждого из которых наблюдается значение случайной величины Y_j. На практике это означает запуски программы $\Pi(x_1, \ldots, x_p)$ на значениях $\hat{x}_1, \ldots, \hat{x}_p$, сгенерированных независимым образом в соответствии с распределением P на множестве $\mathcal{X}$ входных параметров [1]. При этом на каждом испытании используется новый экземпляр программы $\Pi(x_1, \ldots, x_p)$. Получим выборку из n значений $Y_j^{(1)}, \ldots, Y_j^{(n)}$ — числа выполнений j-го линейного участка при запусках программы на элементах из $\mathcal{X}$, отвечающих распределению P. При этом можно считать, что моменты любых порядков случайной величины Y_j существуют [1].

Обозначим $\chi_i = \mathbb{I}(Y_j^{(i)} > 0)$, где $\mathbb{I}(A)$ — индикатор события A:

$$\mathbb{I}(A) = \begin{cases} 1\,, & \text{событие } A \text{ произошло;} \\ 0\,, & \text{событие } A \text{ не произошло.} \end{cases}$$

Описанный выше способ организации испытаний позволяет считать случайные величины $\chi_1, \ldots, \chi_n$ независимыми и одинаково распределенными, причем

$$\mathsf{P}(\chi_i = 1) = p = 1 - \mathsf{P}(\chi_i = 0)\,, \quad i = 1, \ldots, n\,.$$

Очевидно, что при этом число p имеет смысл вероятности того, что рассматриваемый линейный

участок программы будет задействован при случайном выборе значений входных параметров в соответствии с данным распределением вероятностей на множестве $\mathcal{X}$. Положим $q = 1 - p$. Ясно, что q — это вероятность неисполнения рассматриваемого линейного участка (фрагмента) программы. Будем считать, что если $q = 1$, то участок неактивен, неисполняем (и может быть удален из программы), а если $q = q_0 < 1$, то фрагмент активен. Предположим, что на вход подано n стандартных независимых наборов значений входных параметров. Положим

$$\nu = \sum_{i=1}^{n} \chi_i \,.$$

При этом если рассматриваемый участок программы активен (исполняем), то

$$\mathsf{P}(\nu = k) \equiv p_0(k) = C_n^k q_0^{n-k}(1 - q_0)^k \,, \qquad k = 0, \ldots, n \,, \quad (1)$$

а если рассматриваемый участок программы неактивен (неисполняем), то

$$\mathsf{P}(\nu = k) \equiv p_1(k) = \begin{cases} 1 \,, & k = 0 \,, \\ 0 \,, & k = 1, \ldots, n \,. \end{cases} \quad (2)$$

Таким образом, способ проверки исполняемости фрагмента программы может быть описан в терминах задачи проверки статистических гипотез. Имеется случайная величина ν, распределение которой при гипотезе $H_0 : \ q = q_0 < 1$ имеет вид (1), а при гипотезе $H_1 : \ q = 1$ — вид (2). По наблюдаемому значению величины ν требуется сделать вывод о том, какое распределение — (1) или (2) — случайная величина ν имеет в действительности. В данной работе рассматриваются два подхода к решению этой задачи: традиционный подход, основанный на стандартной постановке задачи проверки статистических гипотез, решение которой дается леммой Неймана–Пирсона [11], и подход, основанный на рассмотрении апостериорных вероятностей ошибок [11–13].

Сначала рассмотрим первый подход. При проверке неисполняемости линейного участка программы следует формально допустить возможность ошибок двух типов: (1) исполняемый участок может быть признан неактивным и (2) неисполняемый участок может быть признан активным. Вероятности этих ошибок соответственно обозначим α и β. Ошибка первого типа значительно более нежелательна, чем ошибка второго типа, поскольку она может привести к значительной потере функциональности и надежности программы после удаления признанного неисполняемым активного участка. Зафиксируем некоторое число $a > 0$ и потребуем, чтобы метод контроля неисполняемости удовлетворял условию $\alpha \le a$. При этом желательно (например, по экономическим соображениям), чтобы вероятность β была бы минимально возможной. Другими словами, методика контроля неисполняемости должна быть такой, чтобы средняя доля неисполняемых фрагментов, признанных активными, была минимальной при условии, что средняя доля исполняемых фрагментов, признанных неисполняемыми, не превосходит a.

Как известно, правило проверки статистических гипотез, удовлетворяющее указанному критерию оптимальности, имеет следующий вид (см., например, [9, с. 78]). Пусть $d(\nu) = (d_0(\nu),\, d_1(\nu))$ — вектор-функция такая, что

$$d_0(\nu) \ge 0 \,,\ d_1(\nu) \ge 0 \,; \quad d_0(\nu) + d_1(\nu) \equiv 1 \,.$$

При заданном значении ν условимся с вероятностью $d_0(\nu)$ принимать гипотезу H_0 и с вероятностью $d_1(\nu)$ принимать гипотезу H_1.

Положим

$$L = L(\nu) = \frac{p_0(\nu)}{p_1(\nu)} \,, \ \ G_i(\ell) = \mathsf{P}(L < \ell | H_1) \,, \ i = 0, 1 \,.$$

Согласно фундаментальной лемме Неймана–Пирсона оптимальное правило имеет вид:

$$d(\nu) = d(L) = (d_0(L),\, d_1(L)) = \\ = \begin{cases} (0,\, 1) \,, & L < \ell_1 \,, \\ (r_1,\, 1 - r_1) \,, & L = \ell_1 \,, \\ (1,\, 0) \,, & L > \ell_1 \,, \end{cases} \quad (3)$$

где

$$\ell_1 = \sup\{\ell : \ G_0(\ell - 0) \le a\} \,;$$
$$r_1 = \frac{G_0(\ell_1) - a}{G_0(\ell_1) - G_0(\ell_1 - 0)} \,.$$

В рассматриваемом случае

$$L = L(\nu) = \begin{cases} q_0^n \,, & \nu = 0 \,; \\ \infty \,, & \nu > 0 \,; \end{cases}$$

$$G_0(\ell) = \begin{cases} 0 \,, & \ell < q_0^n \,; \\ q_0^n \,, & q_0^n \le \ell < \infty \,; \\ 1 \,, & \ell = \infty \,; \end{cases}$$

$$G_1(\ell) = \begin{cases} 0 \,, & \ell < q_0^n \,; \\ 1 \,, & \ell \ge q_0^n \,. \end{cases}$$

Возможны два случая: (A) $q_0^n \le a$ и (B) $q_0^n > a$. В случае (A) $\ell_1 = \ell_1^A \equiv \infty$ и потому согласно (3) заведомо активные участки ($\nu > 0$) с ненулевой вероятностью будут признаваться неисполняемыми,

что, очевидно, нежелательно. Поэтому с помощью выбора достаточно большого n надо добиваться, чтобы было выполнено неравенство $q_0^n > a$, т. е. чтобы имел место случай (B). В случае (B) имеем

$$\ell_1 = \ell_1^B = q_0^n\,;$$
$$\beta = 1 - G_1(\ell_1^B) + r_1\left[G_1(\ell_1^B) - G_1(\ell_1^B - 0)\right] = \\ = 1 - \frac{a}{q_0^n}\,.$$

При этом для фиксированного n вероятность β не может быть уменьшена. Однако с помощью надлежащего выбора n можно уменьшить β до величины, близкой к нулю. А именно: положим

$$n = n_1 = \max\left\{m:\ 1 - \frac{a}{q_0^m} \geq 0\right\} = \left[\log_{q_0} a\right]\,.$$

При таком выборе n, во-первых, имеет место ситуация (B), во-вторых, $q_0^n \approx a$, т. е. $\beta \approx 0$, и, в-третьих, $r_1 = 0$, т. е. если $\nu > 0$, то участок сразу признается активным.

При этом испытания можно прекратить после появления первого значения $\chi_i = 1$. Таким образом, число N испытаний, необходимых для проверки неисполняемости фрагмента программы по правилу, удовлетворяющему указанному критерию оптимальности, случайно, причем

$$\mathsf{P}(N = k|H_0) = q_0^{k-1}(1 - q_0)\,,\ \ k = 1, \ldots, n_1 - 1\,;$$
$$\mathsf{P}(N = n_1|H_0) = q_0^{n_1 - 1}\,.$$

Несложно убедиться, что

$$\mathsf{E}(N|H_0) = \frac{1 - q_0^{n_1}}{1 - q_0} \approx \frac{1 - a}{1 - q_0}\,.$$

При этом величина $\mathsf{E}(N|H_0)$ является минимально возможной среди всех последовательных критериев с $\alpha = a$ и $\beta = 0$, поскольку предложенный метод является урезанным последовательным критерием отношения вероятностей, оптимальным в смысле [12].

В табл. 1 для $q_0 = 0{,}99$ приведены значения n_1 и $\mathsf{E}(N|H_0)$ в зависимости от a.

Для $q_0 = 0{,}999$, $q_0 = 0{,}9999$ и т. д. хорошие аппроксимации для n_1 и $\mathsf{E}(N|H_0)$ получаются из табл. 1 с помощью умножения соответствующих значений на 10, 100 и т. д.

Таблица 1 Значения n_1 и $\mathsf{E}(N|H_0)$ в зависимости от a (метод, основанный на лемме Неймана–Пирсона)

a	n_1	$\mathsf{E}(N\|H_0)$
0,05	300	95
0,01	460	99
0,005	530	99,5
0,001	920	99,99

4 Метод, основанный на оптимизации апостериорных вероятностей ошибок

Теперь рассмотрим подход, основанный на апостериорных вероятностях ошибок. Как уже отмечалось, величина α имеет смысл средней доли фрагментов, признанных неисполняемыми, среди исполняемых участков. С практической точки зрения намного важнее уметь управлять средней долей активных участков среди систем, признанных неисполняемыми (апостериорной вероятностью ошибки первого рода), которую будем обозначать γ. Правило различения двух простых гипотез, минимизирующее β при условии $\gamma \leq c$, где $c > 0$ — наперед заданное число, построено в работах [13–15]. Это правило имеет вид (3), где ℓ_1 и r_1 заменены соответственно величинами

$$\ell_2 = \sup\left\{\ell:\ \frac{G_0(\ell - 0)}{G_1(\ell - 0)} \leq \frac{c(1 - w)}{w(1 - c)}\right\}\,;$$
$$r_2 = \\ = \left[1 + \frac{w(1 - c)G_0(\ell_2 - 0) - c(1 - w)G_1(\ell_2 - 0)}{c(1 - w)G_1(\ell_2) - w(1 - c)G_0(\ell_2)}\right]^{-1}.$$

Здесь $w = \mathsf{P}(H_0)$ — априорная вероятность справедливости гипотезы H_0 — имеет смысл средней (ожидаемой) доли активных участков среди всех фрагментов программы (ребер графа $\{V, E\}$).

Возможны два случая:

(C) $q_0^n \geq c(1 - w)/(w(1 - c))$;

(D) $q_0^n < c(1 - w)/(w(1 - c))$.

Рассмотрим ситуацию (C). Здесь $\ell_2 = \ell_2^C \equiv q_0^n$, $\alpha = (1 - r_2)q_0^n$, $\beta = r_2$ и γ не зависит от r_2. Поэтому из соображений минимальности β следует положить $r_2 = 0$. При этом в силу условия $q_0^n \geq \\ \geq c(1-w)/(w(1-c))$ имеет место неравенство $\gamma \geq c$. Так как γ убывает с ростом n, то следует положить

$$n = n_2 = \max\left\{n:\ q_0^n \geq \frac{c(1 - w)}{w(1 - c)}\right\} = \\ = \left[\log_{q_0} \frac{c(1 - w)}{w(1 - c)}\right]\,.$$

При таком выборе n имеет место приближенное равенство $\gamma \approx c$ и для среднего числа испытаний $\mathsf{E}(N|H_0)$ справедлива аппроксимация $\mathsf{E}(N|H_0) \approx \\ \approx (w - c)/(w(1 - c)(1 - q_0))$.

В ситуации (D) критерий имеет те же недостатки, что и в ситуации (A). Поэтому с помощью

Таблица 2 Значения n_2 и $\mathsf{E}(N|H_0)$ в зависимости от c и w (метод, основанный на оптимизации апостериорных вероятностей ошибок)

	$a = 0,05$		$a = 0{,}01$		$a = 0{,}005$		$a = 0{,}001$	
w	n_2	$\mathsf{E}(N\|H_0)$	n_2	$\mathsf{E}(N\|H_0)$	n_2	$\mathsf{E}(N\|H_0)$	n_2	$\mathsf{E}(N\|H_0)$
0,05	—	—	164	80	233	91	394	99
0,1	73	52	238	91	308	96	468	99
0,2	155	78	319	96	388	98	549	99
0,3	208	87	372	98	442	99	602	100
0,4	252	92	416	99	486	99	646	100
0,5	292	94	457	99	526	100	687	100
0,6	333	96	497	100	566	100	727	100
0,7	377	97	541	100	610	100	771	100
0,8	430	98	595	100	664	100	825	100
0,9	511	99	675	100	745	100	905	100
0,95	585	99	749	100	819	100	979	100

надлежащего выбора n и c надо добиваться, чтобы имела место ситуация (C).

В табл. 2 для $q_0 = 0{,}99$ приведены значения n_2 и $\mathsf{E}(N|H_0)$ в зависимости от c и w.

Для $q_0 = 0{,}999$, $q_0 = 0{,}9999$ и т. д. хорошие аппроксимации для n_2 и $\mathsf{E}(N|H_0)$ получаются из табл. 2 с помощью умножения соответствующих значений на 10, 100 и т. д.

Литература

1. *Шалимов А. В.* Метод оценки частоты выполнения фрагментов кода последовательной программы // Моделирование и анализ информационных систем, 2010. Т. 17. Вып. 2. С. 122–132.
2. *Wu Youfeng, Larus J. R.* Static branch frequency and program profile analysis // 27th Annual Symposium (International) on Microarchitecture Proceedings, 1994. P. 1–11.
3. *Ball T., Larus J. R.* Optimally profiling and tracing programs // ACM Transactions on Programming Languages and Systems (TOPLAS), 1994. P. 1319–1360.
4. *Королев В. Ю., Смелянский Р. Л., Смелянский Т. Р., Шалимов А. В.* Об оценивании частоты выполнения фрагментов кода последовательной программы // Программирование, 2013 (в печати).
5. *Пальчун Б. П.* Метод испытаний программ на надежность // Функциональная устойчивость специального математического обеспечения автоматизированных систем. — М., 1989. С. 111–117.
6. *Парнов Е. И.* На перекрестке бесконечностей. — М.: Атомиздат, 1967. 464 с.
7. *Smelianski R. L., Alanko T.* On the calculation of control transition probabilities in a program // Inform. Process. Letters, 1984. No. 3.
8. *Смелянский Р. Л., Гурьев Д. Е., Бахмуров А. Г.* Об одной математической модели для расчета динамических характеристик программы // Программирование, 1986. Вып. 6.
9. *Королев В. Ю.* Предельные распределения для случайных последовательностей со случайными индексами и некоторые их приложения: Дисс. . . . докт. физ.-мат. наук. — М.: МГУ, 1994. 265 с.
10. *Королев В. Ю., Соколов И. А.* Основы математической теории надежности модифицируемых систем. — М.: ИПИ РАН, 2006. 108 с.
11. *Леман Э.* Проверка статистических гипотез. — М.: Наука, 1979.
12. *Вальд А.* Последовательный анализ. — М.: Наука, 1960.
13. *Королев В. Ю.* Наиболее мощные критерии проверки простой гипотезы против простой альтернативы с апостериорным уровнем значимости // Проблемы устойчивости стохастических моделей: Труды семинара. — М.: ВНИИСИ, 1985. С. 87–91.
14. *Королев В. Ю.* Различение двух простых гипотез с неопределенными решениями // IV Международная Вильнюсская конференция по теории вероятностей и математической статистике: Тезисы докладов. Т. 2. — Вильнюс: Институт математики и кибернетики АН Литовской ССР, 1985. С. 100–103.
15. *Акбулатов Н. А., Белокуров Д. В., Королев В. Ю.* Задачи проверки надежности БИС ЭВМ // Разработка и применение в народном хозяйстве ЕС ЭВМ: Тезисы докладов Всесоюзной школы-семинара (Кишинев, 1985). — М., 1985. С. 91–94.