Секция 6

МАТЕМАТИЧЕСКИЕ ОСНОВЫ ИНФОРМАТИКИ И ПРОГРАММИРОВАНИЯ

УДК 519.1

ЭЛАСТИЧНОСТЬ АЛГОРИТМОВ

В. В. Быкова

Современная индустрия программного обеспечения и средств информационной безопасности компьютерных систем диктует необходимость развития специальных методов анализа и классификации алгоритмов. В теории сложности вычислений классификация алгоритмов традиционно осуществляется с точки зрения вычислительной сложности — трудоемкости продуцируемых алгоритмами вычислительных процессов. При этом вычислительная сложность алгоритма формально описывается функцией временной сложности $t(n)$, отражающей максимальное количество элементарных шагов, которое необходимо алгоритму для достижения запланированного результата в зависимости от n — длины входа алгоритма [1]. Обычно ограничиваются рассмотрением поведения функций сложности в асимптотике при стремлении n к бесконечности, а изложение результатов ведется в терминах O-большое и o-малое [2]. До недавнего времени алгоритмы подразделяли на низкозатратные (полиномиальные) и высокозатратные (экспоненциальные). В сегодняшней программной инженерии используется пять сложностных классов алгоритмов. Выделены субполиномиальные (быстрые) алгоритмы из полиномиального класса, субэкспоненциальные и гиперэкспоненциальные алгоритмы из экспоненциального класса. Субполиномиальные и субэкспоненциальные алгоритмы — область повышенного интереса современных криптографических систем [3]. Использование непосредственного асимптотического оценивания для распознавания всех пяти сложностных классов алгоритмов в большинстве случаев сопряжено с трудностями вычислительного характера.

В данной работе в качестве меры вычислительной сложности алгоритмов взята эластичность функций $t(n)$. Относительно функций $t(n)$ сделан ряд естественных допущений. Во-первых, полагается, что $t(n)$ — монотонно неубывающие функции, областью значений которых выступает множество неотрицательных действительных чисел, а областью определения — множество неотрицательных целых чисел. Во-вторых, допускается отступление от дискретности изменения n (с формальной заменой n на x), т. е. предположение о том, что аргумент x непрерывен, а необходимые значения $t(n)$ вычисляются в целочисленных точках $x = n$. В-третьих, рассматриваемое множество функций ограничивается семейством $\mathfrak{L}$ — «по-существу положительных» логарифмически-экспоненциальных функций. Установление указанных границ рассматриваемого множества функций обеспечивает существование эластичности и возможность сравнения любых двух функций сложности алгоритмов по скорости роста.

Эластичный (греч. *elastikos*) — упругий, гибкий. С физической точки зрения эластичность — это свойство вещества оказывать механическое сопротивление силе, которая на него воздействует, и принимать исходную форму после спада данной силы. Противоположность пластичности. Изучается в теории упругости. С экономической

точки зрения эластичность — это характеристика изменения одного показателя (например, спроса) к другому показателю (например, цене товара). Используется в эконометрике для анализа производственных функций. С математической точки зрения эластичность $E_x(y)$ — коэффициент пропорциональности между темпами роста величин $y = t(x)$ и x — дифференциальная характеристика функции $y = t(x)$, определяемая как предел отношения относительного приращения этой функции к относительному приращению аргумента:

$$E_x(y) = \lim_{\Delta x \to 0} \left(\frac{\Delta y}{y} : \frac{\Delta x}{x} \right) = \frac{x}{y} \lim_{\Delta x \to 0} \frac{\Delta y}{\Delta x} = \frac{x}{y} y' = x(\ln y)' = \frac{(\ln y)'}{(\ln x)'}.$$

Таким образом, если $E_x(t)$ — эластичность функции временной сложности $y = t(x)$, то это означает, что при повышении значения x (длины входа алгоритма) на один процент значение t (время выполнения алгоритма) увеличится приблизительно на $E_x(t)$ процентов.

В работе [4] доказано, что семейство монотонно неубывающих, «по-существу положительных» $\mathfrak{L}$-функций можно разбить на классы *Subpoly, Poly, Subexp, Exp, Hyperexp*, для которых свойственно специфическое поведение эластичности на бесконечности. Это позволяет формально описать пять современных сложностных классов алгоритмов. Класс быстрых алгоритмов — множество алгоритмов с функциями сложности $t(x) \in$ *Subpoly*. Таким алгоритмам присуща тождественно нулевая или бесконечно малая эластичность. Класс полиномиальных алгоритмов — множество алгоритмов с $t(x) \in$ *Poly* и асимптотически постоянной эластичностью $E_x(t)$. Класс субэкспоненциальных алгоритмов — алгоритмы, для которых $t(x) \in$ *Subexp*. Эластичность $E_x(t)$ субэкспоненциального алгоритма — бесконечно большая величина, такая, что $1 \prec E_x(t) \prec x$. Для подобного алгоритма темп роста времени выполнения значительно выше темпа роста длины входа. Класс экспоненциальных алгоритмов — это алгоритмы с $t(x) \in$ *Exp*. Для них эластичность $E_x(t) = O(x)$ — бесконечно большая величина, асимптотически пропорциональная линейной функции. Функции с аналогичной эластичностью описывают законы естественного роста: скорость увеличения функции прямо пропорциональна ей самой. Класс гиперэкспоненциальных алгоритмов — множество алгоритмов с $t(x) \in$ *Hyperexp* и $x \prec E_x(t)$. Темп роста гиперэкспоненциальных функций настолько высок, что не укладывается в законы естественного роста. В настоящее время алгоритмы класса *Hyperexp* практически неосуществимы при большой длине входа.

Классификация алгоритмов по асимптотике поведения эластичности функций сложности не разрушает прежней, традиционной классификации с полиномиальными и экспоненциальными алгоритмами, а лишь дополняет и уточняет ее. Свойства эластичности позволяют без особого труда вычислять ее для любой $\mathfrak{L}$-функции.

В экономических науках наибольший интерес вызывают эластичные и абсолютно (совершенно) эластичные зависимости. Функция временной сложности алгоритма описывает обратную зависимость, нежели производственная функция: если производственная функция — это связь «ресурсы→ объем», то функция сложности алгоритма — «объем исходных данных→ ресурсы». Поэтому естественно то, что в теории сложности вычислений эффективными считают быстрые (совершенно неэластичные) и полиномиальные (эластичные) алгоритмы. Алгоритмы с функциями сложности из классов *Subexp, Exp, Hyperexp* целесообразно называть абсолютно эластичными. В асимптотике темп роста времени выполнения абсолютно эластичных алгоритмов «взрывается»

даже при небольших изменениях объема исходных данных, поступающих на вход алгоритмов.

ЛИТЕРАТУРА

1. *Юдин Д. Б.* Математики измеряют сложность. М.: Книжный дом «Либроком», 2009. 192 с.
2. *Быкова В. В.* Математические методы анализа рекурсивных алгоритмов // Журнал СФУ. Математика и физика. 2008. № 1(3). С. 236–246.
3. *Василенко О. Н.* Теоретико-числовые алгоритмы в криптографии. М.: МЦНМО, 2006. 336 с.
4. *Быкова В. В.* Метод распознавания классов алгоритмов на основе асимптотики эластичности функции сложности // Журнал СФУ. Математика и физика. 2009. № 2(1). С. 48–61.

УДК 681.3

АНАЛИЗ РАБОТЫ ПРОГРАММЫ С РЕСУРСАМИ: ВЫЯВЛЕНИЕ НЕНАДЁЖНЫХ И НЕБЕЗОПАСНЫХ ОПЕРАЦИЙ[1]

В. В. Горелов

Современное программное обеспечение обладает большим числом разнообразных свойств. Их можно условно разделить на положительные — это те, которые позволяют решить задачу оптимальным методом, и отрицательные, которые мешают задуманному решению. Среди отрицательных свойств программы основными являются ненадёжность и небезопасность. Под надёжностью программы (или программно-аппаратного комплекса) подразумевается её способность, путём действия или бездействия, работать без причинения вреда вовне. Под безопасностью — устойчивость к внешнему воздействию, которое может нарушить работу программы. Данные понятия связаны друг с другом, но существуют разные методы для разработки надёжных и безопасных программ. Здесь предложен метод обнаружения (в процессе работы отлаживаемой программы) нежелательных с точки зрения надёжности и безопасности моментов её работы. Обнаружение подразумевает не только выявление факта нарушения правильной последовательности работы с ресурсами, но и предоставление разработчику детальной информации о конкретном месте ошибки в исходном коде программы. Кроме того, предоставляется информация о ходе её появления в случае не одномоментного (не в одном месте программы) её проистечения.

Уникальность предложенного метода в том, что число ресурсов, с которым он может работать, заранее не ограничено. Такой подход разработан потому, что для большого числа разных ресурсов был замечен общий класс однотипных ошибочных конструкций в программах. В частности, к этому классу отнесены следующие шаблонные ошибки: утечки ресурсов; использование ресурсов после их освобождения; повторные освобождения ресурсов; использование (неинициализированных) ресурсов без их предварительного захвата; использование ресурсов за их границами (относится к динамической памяти и адресному пространству); нарушение вызываемого механизма захвата и освобождения ресурсов (функция захвата возвращает идентификатор ранее захваченного и ещё не освобождённого ресурса) и другие.

Для решения задачи представления неограниченного количества ресурсов разработан язык описания ресурсов (для языка программирования Си). Для описания ресур-

[1]Работа выполнена в рамках реализации ФЦП «Научные и научно-педагогические кадры инновационной России» на 2009–2013 гг. (гос. контракт № П1010).