ЛИТЕРАТУРА

1. *Егорушкин О. И., Колбасина И. В., Сафонов К. В.* О совместности систем символьных полиномиальных уравнений и их приложении // Прикладная дискретная математика. Приложение. 2016. № 9. С. 119–121.
2. *Egorushkin O. I., Kolbasina I. V., and Safonov K. V.* On solvability of systems of symbolic polynomial equations // Журн. СФУ. Сер. Матем. и физ. 2016. Т. 9. Вып. 2. С. 166–172.
3. *Глушков В. М., Цейтлин Г. Е., Ющенко Е. Л.* Алгебра. Языки. Программирование. Киев: Наукова думка, 1973.
4. *Salomaa A. and Soitolla M.* Automata-Theoretic Aspects of Formal Power Series. N.Y.: Springer Verlag, 1978.
5. *Семёнов А. Л.* Алгоритмические проблемы для степенных рядов и контекстно-свободных грамматик // Доклады АН СССР. 1973. № 212. С. 50–52.
6. *Сафонов К. В., Егорушкин О. И.* О синтаксическом анализе и проблеме В. М. Глушкова распознавания контекстно-свободных языков Хомского // Вестник Томского госуниверситета. 2006. Приложение № 17. С. 63–67.
7. *Сафонов К. В.* Об условиях алгебраичности и рациональности суммы степенного ряда // Матем. заметки. 1987. Т. 41. Вып. 3. С. 325–332.
8. *Safonov K. V.* On power series of algebraic and rational functions in C^n // J. Math. Analysis Appl. 2000. V. 243. P. 261–277.

УДК 519.682 DOI 10.17223/2226308X/13/32

АЛГОРИТМ РЕШЕНИЯ РАСШИРЕННОЙ ПРОБЛЕМЫ СИНТАКСИЧЕСКОГО АНАЛИЗА

В. В. Кишкан, К. В. Сафонов

Уточняется формулировка расширенной проблемы синтаксического анализа: разработать беступиковый алгоритм, который позволяет установить, может ли данный моном быть выведен при помощи системы продукций, образующих грамматику контекстно-свободного языка программирования, а также описать сразу все выводы этого монома, если такие существуют. Описание вывода монома состоит в следующем: определить, какие продукции из грамматики языка, сколько раз и в каком порядке применяются, что равносильно построению всех деревьев вывода. Предложен алгоритм решения расширенной проблемы синтаксического анализа, основанный на иерархии маркированных скобок; маркировка скобок показывает, за какой продукцией они закреплены, и позволяет проследить порядок их использования. Алгоритм имеет простую программную реализацию, дана также оценка сложности алгоритма.

Ключевые слова: *расширенная проблема синтаксического анализа, контекстно-свободный язык, сложность алгоритма.*

При разработке перспективных языков программирования, в том числе предназначенных для обеспечения работы суперкомпьютеров, включая квантовые, возникает необходимость исследовать контекстно-свободные языки (кс-языки) и контекстно-свободные грамматики (кс-грамматики). Один из аспектов связан с проблемой синтаксического анализа (разбора) выражений, написанных на языке программирования. Обычно для синтаксического анализа используются специальные программы — парсеры, разработанные применительно к тому или иному языку программирования и основанные на определённых алгоритмах разбора. Однако в ситуации, когда разра-

батывается и тестируется новый язык программирования, никаких парсеров ещё нет. Когда необходимо провести синтаксический анализ некоторого выражения относительно совокупности грамматических правил, находящихся в стадии разработки, могут быть полезными различные алгоритмы, в том числе имеющие высокую сложность — как правило, анализируются выражения ограниченной длины, и в этом случае высокая сложность алгоритма не играет решающей роли. Если длина тестируемой программы не слишком велика, сложность алгоритма синтаксического разбора может быть даже выше экспоненциальной — важно лишь, чтобы алгоритм был вполне конструктивным и допускал простую программную реализацию.

Отметим, что практически все известные в настоящее время языки программирования являются кс-языками, порождёнными кс-грамматиками, и потому кс-язык является адекватной математической моделью любого языка программирования, в которой правильным программам отведена роль мономов кс-языка.

Проблема синтаксического анализа мономов кс-языка возникла на заре теории языков программирования в 50–60-х годах прошлого века. Удивительно, но до настоящего времени в формулировке проблемы сохраняются разночтения, в связи с чем возникает необходимость уточнить её, а именно: рассмотрим кс-язык, порождённый кс-грамматикой, которая представляет собой систему правил вывода (продукций)

$$z_j \to\ q_{j1}(z,x),\ \ldots,\ z_j \to\ q_{jp_j}(z,x),\ j=1,\ \ldots,n, \tag{1}$$

где $q_{jk}(z,x)$ — мономы от некоммутативных символов алфавита $z_1\ldots,z_n,\ x_1,\ldots,x_m$ с числовым коэффициентом равным 1.

Символы $x_1,\ldots,x_m$ из второй группы называются терминальными символами и образуют словарь кс-языка. Применительно к языкам программирования терминальными символами являются цифры, буквы, вспомогательные знаки, а также состоящие из них «блоки», обозначающие, например, операторы языка программирования. Символы первой группы $z_1,\ldots,z_n$ называются нетерминальными, поскольку не присутствуют явно в тексте программ, а играют вспомогательную роль, участвуя в кс-грамматике как совокупности продукций, порождающих кс-язык.

По правилам грамматики формируются мономы от терминальных символов $x_1,\ldots,x_m$, которые интерпретируются как правильные предложения языка [1, 2]. Такие мономы рассматриваются как корректные, в отличие от произвольных мономов, которые могут не соответствовать правилам грамматики, а значит, являются некорректными.

Вывод корректных мономов кс-языка с помощью системы продукций (1) осуществляется так: продукции сначала применяются к начальному символу z_1, а затем к другим получающимся мономам неограниченное число раз и в любом порядке, что позволяют продуцировать новые мономы от терминальных и нетерминальных символов. Вывод заканчивается, когда получается моном только от терминальных символов — это и есть корректный моном языка, дальнейший вывод из него невозможен, поскольку продукции применимы только к нетерминальным символам. Все корректные мономы образуют соответствующий кс-язык.

В проблеме синтаксического анализа мономов кс-языка выделяют две составляющие: первая часть, называемая проблемой принадлежности или этапом синтаксического контроля, состоит в том, чтобы определить, принадлежит ли данный моном рассматриваемому кс-языку, т. е. может ли быть получен из начального символа z_1 при помощи продукций; вторая часть проблемы — описание синтаксической структуры монома. Такое описание понимается в литературе по-разному.

Так, различные авторы при постановке проблемы синтаксического анализа допускают следующие варианты: требуется разработать алгоритм для того, чтобы установить, какие правила подстановки и сколько раз использовались при выводе данного монома, при этом порядок использования правил подстановки не имеет значения; какие правила подстановки, сколько раз и в каком порядке использовались при выводе этого монома, т. е. построить хотя бы один из возможных выводов монома [2]. Как видно, для полного решения проблемы синтаксического анализа необходимо построить сразу все возможные выводы монома, если таких несколько.

Кроме того, исследователи уделяют большое внимание тому, чтобы разработать беступиковый алгоритм синтаксического анализа мономов кс-языка [1, с. 248]).

В связи с этим будем называть *расширенной проблемой синтаксического анализа* мономов кс-языка проблему разработки беступикового алгоритма, который позволяет установить, может ли моном быть выведен при помощи системы продукций кс-языка (решить проблему принадлежности), а также найти сразу все выводы этого монома; описание вывода монома будем понимать следующим образом: определить, какие продукции, сколько раз и в каком порядке применяются для вывода этого монома, что равносильно построению всех деревьев вывода.

Синтаксический анализ монома, проводимый в соответствии с этим алгоритмом, будем называть *расширенным синтаксическим анализом* (алгоритм 1).

Для решения расширенной проблемы синтаксического анализа рассмотрим расширенную систему уравнений Хомского — Шютценберже, которая имеет вид

$$z_j = Q_j^*(z,x,t) = t_{j1}[\ q_{j1}(z,x)\ ] + \ \cdots\ + t_{jp_j}[\ q_{jp_j}(z,x)\ ],\ j = 1,\ \ldots\ , n.$$

Решение этой системы можно получить методом последовательных приближений [2]:

$$z^{(k+1)}(x,t) = Q^*(z^{(k)}(x,t),x,t);\ k = 0,1,\ldots;\ z^{(0)} = 0. \tag{2}$$

В результате решение получается в виде формальных степенных рядов

$$z_j = z_j^*(x,t) = \sum_{i=0}^{\infty} \langle z_j^*, w_i \rangle\ w_i,\ j = 1,\ldots,n,$$

где w_i — мономы от символов $x_1,\ldots,x_m,t_{11},t_{12},\ldots,t_{np_n}$ с числовыми коэффициентами $\langle z_j^*, w_i\rangle$, содержащие также систему открывающихся и закрывающихся скобок.

Алгоритм 1. Решение расширенной проблемы синтаксического анализа

Вход: моном (программа) w степени (длины) N.

1: Проводим N итераций метода последовательных приближений (2) для решения соответствующей расширенной системы уравнений Хомского — Шютценберже.

2: Перебираем все полученные в шаге 1 мономы степени N, определяя те из них, которые, с точностью до множителей t_{jk}, совпадают с мономом w.
Если таких мономов нет, то моном w вывести невозможно; если есть, то они дают решение расширенной проблемы синтаксического анализа в соответствии со следующим шагом.

3: Считываем все найденные в шаге 2 мономы слева направо, устанавливая иерархию маркированных скобок (по признаку сравнения скобок — внутренняя или внешняя) и определяя тем самым порядок применения продукций при выводе монома w.

Теорема 1. Сложность алгоритма 1 равна $O\left(Ng^{d^N}\right)$, где g и d — некоторые целые числа.

Несмотря на то, что сложность данного алгоритма высокая, он может быть использован для синтаксического анализа применительно к языку программирования, находящемуся в стадии разработки.

ЛИТЕРАТУРА

1. *Глушков В. М., Цейтлин Г. Е., Ющенко Е. Л.* Алгебра. Языки. Программирование. Киев: Наукова думка, 1973.
2. *Salomaa A. and Soitolla M.* Automata-Theoretic Aspects of Formal Power Series. N.Y.: Springer Verlag, 1978.

УДК 510.52 DOI 10.17223/2226308X/13/33

О ГЕНЕРИЧЕСКОЙ СЛОЖНОСТИ ПРОБЛЕМЫ ПРЕДСТАВИМОСТИ НАТУРАЛЬНЫХ ЧИСЕЛ СУММОЙ ДВУХ КВАДРАТОВ

А. Н. Рыбалов

Изучается генерическая сложность проблемы представимости натуральных чисел суммой двух квадратов. Эта проблема, восходящая ещё к Ферма и Эйлеру, тесно связана с проблемами факторизации целых чисел и распознавания квадратичности вычетов по составным модулям, для решения которых не известно эффективных алгоритмов. Доказывается, что, при условии трудноразрешимости этой проблемы в худшем случае и P = BPP, для её решения не существует полиномиального сильно генерического алгоритма. Сильно генерический алгоритм решает проблему не на всём множестве входов, а на подмножестве, последовательность относительных плотностей которого при увеличении размера экспоненциально быстро сходится к 1.

Ключевые слова: *генерическая сложность, суммы квадратов, диофантовы уравнения.*

Введение

Проблема представимости натуральных чисел суммой двух квадратов состоит в том, чтобы по любому заданному натуральному числу N определить, разрешимо ли в натуральных числах диофантово уравнение $x^2 + y^2 = N$. Эта задача восходит ещё к Ферма, который в 1640 г. сформулировал (см. [1, 2]) следующее красивое утверждение: любое простое число вида $p = 4n + 1$ представимо в виде суммы квадратов двух натуральных чисел. Эта гипотеза впоследствие была доказана Эйлером и называется теперь теоремой Ферма — Эйлера [1, 2]. В дальнейшем был получен критерий Ферма — Эйлера разрешимости диофантова уравнения $x^2 + y^2 = N$ для любого натурального N. Однако этот критерий сводит проблему к задаче факторизации (разложения на множители) целых чисел, которая на текщий момент считается трудноразрешимой [3]. Таким образом, критерий Ферма — Эйлера не может быть проверен эффективно (за полиномиальное от размера входа время).

Генерический подход к алгоритмическим проблемам предложен в [4]. В рамках этого подхода алгоритмическая проблема рассматривается не на всём множестве входов, а на некотором подмножестве «почти всех» входов. Такие входы образуют генерическое множество. Понятие «почти все» формализуется введением естественной меры на