УДК 519.163

**Д.В. Кручинин**

# Модификация метода построения алгоритмов комбинаторной генерации на основе применения производящих функций многих переменных и приближенных вычислений

Предложен модифицированный метод построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ, который отличается от оригинального и его модификаций применением комплексного метода получения явных выражений коэффициентов производящих функций многих переменных для нахождения выражения функции мощности комбинаторного множества, в том числе определяемого несколькими параметрами. Также предложенный модифицированный метод отличается применением приближенных вычислений и двоичного поиска для поиска выбранного сына ИЛИ-узла, что позволяет снижать вычислительную сложность алгоритмов генерации по рангу. С целью апробации предложенного модифицированного метода построения алгоритмов комбинаторной генерации разработаны новые алгоритмы ранжирования и генерации по рангу для комбинаторного множества самонепересекающихся решеточных путей на плоскости. В данном случае применение двоичного поиска позволило сократить в среднем количество требуемых вычислительных операций и получить лучшее значение вычислительной сложности по сравнению с исходной версией алгоритма.

**Ключевые слова:** комбинаторная генерация, производящие функции многих переменных, приближенные вычисления, двоичный поиск, решеточный путь, алгоритм ранжирования, алгоритм генерации по рангу.
**DOI:** 10.21293/1818-0442-2021-25-1-55-60

Структура многих информационных объектов может быть представлена в виде иерархической или рекурсивной зависимости. Для представления и кодирования таких информационных объектов достаточно хорошо подходит применение древовидных структур данных. В свою очередь, это приводит к возможности описания исследуемого информационного объекта с помощью формального комбинаторного множества, для которого применимы различного рода алгоритмы комбинаторной генерации [1–3].

Комбинаторное множество – это конечное множество, элементы которого имеют некоторую структуру и имеется процедура построения элементов этого множества. Комбинаторная генерация – раздел наук на стыке информатики, комбинаторики и дискретной математики, в рамках которого исследуются методы и алгоритмы генерации элементов комбинаторных множеств. В качестве базовых типов алгоритмов комбинаторной генерации можно выделить алгоритмы последовательной генерации, алгоритмы ранжирования, алгоритмы генерации по рангу и алгоритмы генерации случайных объектов [3].

Можно выделить несколько общих подходов к построению новых алгоритмов генерации, например, поиск с возвратом [4], ECO-метод [5] и др. [6–8]. Представленная работа посвящена исследованию метода построения алгоритмов комбинаторной генерации на основе применения структур деревьев И/ИЛИ [9–11]. Данный метод может быть применен для последовательной генерации, а также ранжирования и генерации по рангу комбинаторных объектов. Метод основан на представлении комбинаторного множества в форме структуры дерева И/ИЛИ, количество вариантов которого совпадает со значением функции мощности.

**Модификация с точки зрения применения производящих функций многих переменных**

Если для исследуемого комбинаторного множества $A$ не известно выражение функции мощности $f$, принадлежащей алгебре $\{\mathbb{N},+,\times,R\}$, то метод на основе деревьев И/ИЛИ не может быть применен для разработки алгоритмов комбинаторной генерации. Для решения данной проблемы предлагается применение математического аппарата теории производящих функций, а именно: применение разработанного комплексного метода получения коэффициентов производящих функций многих переменных.

Производящие функции являются одним из основных инструментов, применяемых в современной комбинаторике [12, 13]. Если рассмотреть комбинаторное множество $A_n$, определяемое одним параметром $n$ (например, размерность каждого объекта комбинаторного множества), тогда функция мощности $f(n)=|A_n|$ может быть определена через коэффициенты обыкновенной производящей функции

$$F(x)=\sum_{n\geq 0} f_n x^n=\sum_{n\geq 0} f(n)x^n=\sum_{n\geq 0}|A_n|\,x^n.$$

Если комбинаторное множество определяется набором из нескольких параметров $n,m,\ldots,l$, тогда функция мощности $f(n,m,\ldots,l)=|A_{n,m,\ldots,l}|$ может быть определена через коэффициенты производящей функции многих переменных

$$F(x,y,\ldots,z)=\sum_{n\geq 0}\sum_{m\geq 0}\ldots\sum_{l\geq 0} f_{n,m,\ldots,l}\,x^n y^m\cdots z^l=$$

$$=\sum_{n\geq 0}\sum_{m\geq 0}\ldots\sum_{l\geq 0} f(n,m,\ldots,l)\,x^n y^m\cdots z^l=$$

$$=\sum_{n\geq 0}\sum_{m\geq 0}\ldots\sum_{l\geq 0}|A_{n,m,\ldots,l}|\,x^n y^m\cdots z^l.$$

Таким образом, если для исследуемого комбинаторного множества известна производящая функция (в том числе производящая функция многих переменных), которая определяет значения функции мощности данного комбинаторного множества, то ее коэффициенты $f_{n,m,\ldots,l}$ определяют значения самой функции мощности $f(n,m,\ldots,l)$. Тогда, получив явное выражение коэффициентов производящей функции, становится известным явное выражение $|A_{n,m,\ldots,l}|$ для функции мощности соответствующего комбинаторного множества. Подходящим средством для решения данной задачи является разработанный комплексный метод получения коэффициентов производящих функций многих переменных [14].

**Модификация с точки зрения применения приближенных вычислений и двоичного поиска**

Анализ общего алгоритма генерации по рангу варианта дерева И/ИЛИ показывает, что часть его вычислений направлена лишь на поиск позиции $k$ выбранного в варианте дерева И/ИЛИ сына ИЛИ-узла среди всех его сыновей (строки 17–22 Алгоритма 2 в [11]). В данном случае происходит вычисление частичных сумм вида

$$S_k = \sum_{i=1}^{k} w(s_i^{(z)})$$

и выполняется поиск значения параметра $k$, для которого выполняется условие

$$S_{k-1} \le l < S_k. \quad (1)$$

Если в структуре дерева И/ИЛИ присутствует ИЛИ-узел с количеством сыновей, которое линейно зависит от некоторого параметра $n$ исследуемого комбинаторного множества, то для худшего случая выполнения алгоритма генерации по рангу его вычислительная сложность увеличивается в $n$ раз. То есть только для поиска значения параметра $k$ (без учета всех остальных действий) оценка вычислительной сложности алгоритма генерации по рангу примет линейный вид. Для уменьшения вычислительной сложности предлагается реализовать:

1. Поиск значения параметра $k$ на основе методов приближенных вычислений. В данном случае на основе формул вычисления значения $w(s_i^{(z)})$ или частичных сумм $S_k$ находится значение $k^*$, являющееся приближенным решением неравенства (1):

$$k^* = \mathtt{SolveApprox}(S_{k-1} \le l < S_k).$$

Далее исследуется точность полученного приближенного решения $k^*$ и производится поиск точного решения $k$ в пределах заданной окрестности $\delta$. Тогда оценка вычислительной сложности поиска значения параметра $k$ изменится с $O(n)$ на $O(n^*+\delta)$, где $O(n^*)$ определяет оценку вычислительной сложности для функции $\mathtt{SolveApprox}$. Если $n^*+\delta \ll n$, то получаем значительное уменьшение количества выполняемых операций.

2. Поиск значения параметра $k$ на основе метода двоичного поиска. В данном случае предполагается наличие явной формулы для вычисления частичных сумм $S_k$. Запишем следующее неравенство для частичных сумм:

$$0 < S_1 < S_2 < \ldots < S_{n-1} < S_n.$$

Далее воспользуемся методом двоичного поиска для поиска точного решения неравенства (1)

$$k = \mathtt{BinarySearch}(S_{k-1} \le l < S_k).$$

Тогда оценка вычислительной сложности поиска значения параметра $k$ изменится с $O(n)$ на $O(\log_2 n)$. Если вычислительная сложность расчета частичной суммы $S_k$ не превышает вычислительную сложность расчета значения $w(s_i^{(z)})$, то количество выполняемых операций уменьшается.

Обобщая все предлагаемые дополнения к оригинальному алгоритму, получаем модифицированный метод построения алгоритмов комбинаторной генерации, который можно представить в виде последовательности следующих шагов:

1. Если известно выражение функции мощности $f(n,m,\ldots,l) = |A_{n,m,\ldots,l}|$ комбинаторного множества $A_{n,m,\ldots,l}$, принадлежащее алгебре $\{\mathbb{N},+,\times,R\}$, то переход на шаг 4.

2. Если известно выражение производящей функции многих переменных $F(x,y,\ldots,z)$ для последовательности значений функции мощности $f(n,m,\ldots,l)$ комбинаторного множества $A_{n,m,\ldots,l}$, то применить комплексный метод получения явных выражений коэффициентов производящих функций многих переменных. Иначе дальнейшее применение метода построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ невозможно.

3. Если получено выражение функции мощности $f(n,m,\ldots,l)$ комбинаторного множества $A_{n,m,\ldots,l}$, принадлежащее алгебре $\{\mathbb{N},+,\times,R\}$, то переход на шаг 4. Иначе дальнейшее применение метода построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ невозможно.

4. На основе выражения функции мощности $f(n,m,\ldots,l)$ комбинаторного множества $A_{n,m,\ldots,l}$ построить структуру дерева И/ИЛИ $D$.

5. Определить биекцию $A_{n,m,\ldots,l} \leftrightarrow W(D)$ между элементами $a$ комбинаторного множества $A_{n,m,\ldots,l}$ и множества всех вариантов $v$ дерева И/ИЛИ $D$ в виде алгоритмов $\mathtt{ObjectToVariant}(a,D)$ и $\mathtt{VariantToObject}(v,D)$.

6. Определить биекцию $W(D) \leftrightarrow \mathbb{N}_{|W(D)|}$ между элементами множества всех вариантов $v$ дерева И/ИЛИ $D$ и конечного множества натуральных чисел $\mathbb{N}_{|W(D)|} = \{0,1,\ldots,|W(D)|-1\}$ в виде алгоритмов $\mathtt{RankVariant}(v,D)$ и $\mathtt{UnrankVariant}(r,D)$.

7. Если в структуре дерева И/ИЛИ $D$ присутствует ИЛИ-узел с количеством сыновей, которое зависит от параметров комбинаторного множества $A_{n,m,\ldots,l}$, то попробовать уменьшить вычислительную сложность алгоритма $\mathtt{UnrankVariant}(r,D)$ за счет применения методов приближенных вычислений или метода двоичного поиска для поиска выбранного сына ИЛИ-узла.

Предложенный модифицированный метод построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ отличается от оригинального метода и его модификаций применением разработанного комплексного метода получения явных выражений коэффициентов производящих функций многих переменных для нахождения выражения функции мощности комбинаторного множества, в том числе определяемого несколькими параметрами (отражено в действиях шага 2). Также предложенный модифицированный метод отличается применением методов приближенных вычислений и метода двоичного поиска для поиска выбранного сына ИЛИ-узла, что позволяет снижать вычислительную сложность алгоритмов генерации по рангу (отражено в действиях шага 7).

**Апробация модифицированного метода построения алгоритмов комбинаторной генерации**

Рассмотрим процесс разработки новых алгоритмов ранжирования и генерации по рангу на основе предложенного модифицированного метода построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ для следующего комбинаторного объекта: самонепересекающийся решеточный путь на плоскости из точки $(0,0)$ в точку $(n,n)$ с шагами вида $(0,1)$ (шаг вверх), $(1,0)$ (шаг вправо) и $(-1,0)$ (шаг влево) [15]. При этом остановимся на частном случае такого решеточного пути, содержащего всего $2(n+2)$ шагов, из которых $n$ шагов вверх, $n+2$ шага вправо и $2$ шага влево.

Данный комбинаторный объект предлагается кодировать в виде последовательности $a=(a_1,\ldots,a_{2n+4})$, где каждое $a_i$ представляет собой очередной выполненный шаг решеточного пути, при этом $a_i$ = "$N$" обозначает шаг вверх, $a_i$ ="$E$" – шаг вправо, $a_i$ ="$W$" – шаг влево.

Значения функции мощности данного комбинаторного множества формируют следующую целочисленную последовательность (последовательность A119578 в OEIS [16]):

$$0,2,18,120,700,3780,19404,96096,463320,2187900,\ldots$$

На рис. 1 показан пример всех возможных вариантов рассматриваемых решеточных путей при $n=2$.

Далее рассмотрим обобщение в виде комбинаторного множества таких самонепересекающихся решеточных путей из точки $(0,0)$ в точку $(k,k)$ с двумя шагами влево для всех $k \le n$.

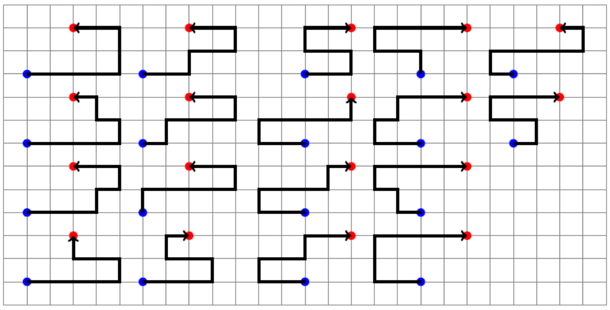

Рис. 1. Все возможные самонепересекающиеся решеточные пути из точки (0,0) в точку (2,2) с 2 шагами вверх, 4 шагами вправо и 2 шагами влево

Известна следующая функция мощности для множества самонепересекающихся решеточных путей из точки $(0,0)$ в точку $(n,n)$ с $n$ шагами вверх, $n+2$ шагами вправо и $2$ шагами влево

$$f_1(n)=\binom{2n}{n}\binom{n+1}{2}$$

для $n>0$ и $f_1(0)=0$. Тогда функция мощности для множества самонепересекающихся решеточных путей из точки $(0,0)$ в точку $(k,k)$ с $k$ шагами вверх, $k+2$ шагами вправо и $2$ шагами влево для набора всех значений $k \le n$ принимает следующий вид:

$$f_2(n)=\sum_{k=0}^{n} f_1(k)=\sum_{k=1}^{n}\binom{2k}{k}\binom{k+1}{2}. \qquad (2)$$

Так как функция мощности (2) принадлежит алгебре $\{\mathbb{N},+,\times,R\}$, то на ее основе можно построить структуру дерева И/ИЛИ (рис. 2).

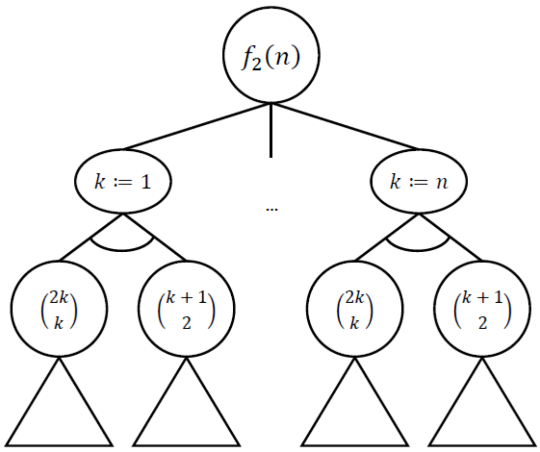


Рис. 2. Структура дерева И/ИЛИ для выражения (2)

Данная структура дерева И/ИЛИ основана на применении известной структуры дерева И/ИЛИ для комбинаторного множества сочетаний элементов. При этом левое поддерево И-узла показывает вариант самонепересекающегося решеточного пути из точки $(0,0)$ в точку $(k,k)$ с $k$ шагами вверх и $k$ шагами вправо, а правое поддерево И-узла определяет вариант встраивания в данный решеточный путь $2$ шага влево.

Для компактности представления каждый вариант дерева И/ИЛИ предлагается кодировать в виде последовательности $v=(k,v_1,v_2)$, в которой $k$ определяет метку выбранного сына ИЛИ-узла, помеченного $f_2(n)$ в варианте дерева И/ИЛИ; $v_1$ – соответствует варианту левого поддерева И-узла; $v_2$ соответствует варианту правого поддерева И-узла.

На основе полученной структуры дерева И/ИЛИ с помощью общих алгоритмов ранжирования и генерации по рангу вариантов дерева И/ИЛИ разработаны соответствующие алгоритмы для комбинаторного множества решеточных путей на плоскости (Алгоритмы 1 и 2). В данных алгоритмах также используются разработанные ранее алгоритмы ранжирования и генерации по рангу вариантов деревьев И/ИЛИ для множества сочетаний элементов.

Алгоритм 1. Алгоритм ранжирования вариантов дерева И/ИЛИ, представленного на рис. 2:

Rank( $v=(k,v_1,v_2)$, $n$ )
**begin**
  $l_1 :=$ RankC( $v_1$, $2k$, $k$ )
  $l_2 :=$ RankC( $v_2$, $k+1$, $2$ )
  $r := l_1 + l_2\binom{2k}{k} + \sum_{i=1}^{k-1}\binom{2i}{i}\binom{i+1}{2}$
  **return** $r$
**end**

Алгоритм 2. Алгоритм генерации по рангу вариантов дерева И/ИЛИ, представленного на рис. 2:

1 Unrank( $r$, $n$ )
2 **begin**
3   $k := 1$
4   $sum := 0$
5   **while** $sum + \binom{2k}{k}\binom{k+1}{2} \le r$ **do**
6     $sum := sum + \binom{2k}{k}\binom{k+1}{2}$
7     $k := k+1$
8   **end**
9   $r := r - sum$
10   $l_1 := r \bmod \binom{2k}{k}$
11   $l_2 := \mathrm{floor}\left(r / \binom{2k}{k}\right)$
12   $v_1 :=$ UnrankC( $l_1$, $2k$, $k$ )
13   $v_2 :=$ UnrankC( $l_2$, $k+1$, $2$ )
14   $v=(k,v_1,v_2)$
15   **return** $v$
16 **end**

В полученной структуре дерева И/ИЛИ присутствует ИЛИ-узел с количеством сыновей, которое зависит от параметров исследуемого комбинаторного множества (имеется $n$ сыновей ИЛИ-узла, помеченного $f_2(n)$). Следовательно, в алгоритме генерации по рангу (Алгоритм 2) выполняются вычисления частичных сумм следующего вида:

$$S_k = \sum_{i=1}^{k}\binom{2i}{i}\binom{i+1}{2}. \tag{3}$$

Таким образом, для худшего случая вычислительная сложность поиска выбранного сына ИЛИ-узла (определение значения параметра $k$ в строках 3–8 Алгоритма 2) равна $O(n^2)$. Данная оценка вычислительной сложности складывается из вычислительной сложности $O(n)$ для расчета биномиального коэффициента, а также из максимального количества итераций в цикле while, которое равно $n$.

Согласно шагу 7 предложенного модифицированного метода построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ, чтобы уменьшить вычислительную сложность алгоритма генерации по рангу, применим метод двоичного поиска для поиска выбранного сына ИЛИ-узла. Для этого воспользуемся следующей формулой для вычисления частичных сумм (3):

$$S_k = \frac{(2k+1)!}{3(k-1)!k!}.$$

Тогда получаем следующую модификацию алгоритма генерации по рангу вариантов дерева И/ИЛИ, представленного на рис. 2:

Алгоритм 3. Модификация алгоритма генерации по рангу вариантов дерева И/ИЛИ, представленного на рис. 2:

Unrank( $r$, $n$ )
**begin**
  $k :=$ BinarySearch( $S_{k-1} \le r < S_k$ )
  $r := r - S_{k-1}$
  $l_1 := r \bmod \binom{2k}{k}$
  $l_2 := \mathrm{floor}\left(r / \binom{2k}{k}\right)$
  $v_1 :=$ UnrankC( $l_1$, \$2k\$, $k$ )
  $v_2 :=$ UnrankC( $l_2$, $k+1$, $2$ )
  $v=(k,v_1,v_2)$
  **return** $v$
**end**

Применение метода двоичного поиска позволяет в среднем сократить количество требуемых операций для поиска точного решения неравенства (1). В результате для худшего случая вычислительная сложность поиска выбранного сына ИЛИ-узла станет равна $O(n\log_2 n)$, что является лучшим значением по сравнению с исходной версией алгоритма.

**Заключение**

Основным результатом данной статьи является предложенный модифицированный метод построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ. В данном случае модификация представлена, во-первых, с точки зрения применения производящих функций многих переменных для нахождения явных выражений функции мощности комбинаторного множества, а также с точки зрения применения приближенных вычислений и двоичного поиска для алгоритма генерации по рангу.

Также в качестве апробации метода разработаны новые алгоритмы ранжирования и генерации по рангу для комбинаторного множества самонепересекающихся решеточных путей на плоскости. В данном случае применение метода двоичного поиска для поиска выбранного сына ИЛИ-узла позволяет сократить в среднем количество требуемых вычислительных операций и получить вычислительную сложность $O(n\log_2 n)$, что является лучшим значением по сравнению с исходной версией алгоритма с вычислительной сложностью $O(n^2)$.

*Литература*

1. Кнут Д.Э. Искусство программирования. Комбинаторные алгоритмы. – М.: Вильямс, 2013. – Т. 4А, ч. 1. – 960 с.

2. Kreher D.L. Combinatorial algorithms: Generation, enumeration, and search / D.L. Kreher, D.R. Stinson. – USA: CRC Press, 1999. – 329 p.

3. Ruskey F. Combinatorial generation. Working version (1j-CSC 425/520), 2003. – 311 p. – URL: http://page.math.tu-berlin.de/~felsner/SemWS17-18/Ruskey-Comb-Gen.pdf, свободный (дата обращения: 01.01.2022).

4. Reingold E.M. Combinatorial algorithms: Theory and practice / E.M. Reingold, J. Nievergelt, N. Deo. – USA: Prentice Hall, 1977. – 433 p.

5. Barcucci E. Exhaustive generation of some lattice paths and their prefixes / E. Barcucci, A. Bernini, R. Pinzani // Theoretical Computer Science. – 2021. – Vol. 878–879. – P. 47–52.

6. Flajolet P. A calculus for the random generation of combinatorial structures / P. Flajolet, P. Zimmerman, B. Cutsem // Theoretical Computer Science. – 1994. – Vol. 132, No. 1–2. – P. 1–35.

7. Martinez C. A generic approach for the unranking of labeled combinatorial classes / C. Martinez, X. Molinero // Random Structures and Algorithms. – 2001. – Vol. 19, No. 3–4. – P. 472–497.

8. Рябко Б.Я. Быстрая нумерация комбинаторных объектов // Дискретная математика. – 1998. – Т. 10, № 2. – С. 101–119.

9. Кручинин В.В. Методы построения алгоритмов генерации и нумерации комбинаторных объектов на основе деревьев И/ИЛИ. – Томск: В-Спектр, 2007. – 200 с.

10. Кручинин В.В. Методы, алгоритмы и программное обеспечение комбинаторной генерации: дис. ... д-ра техн. наук. – Томск, 2010. – 387 с.

11. Shablya Y. Method for developing combinatorial generation algorithms based on AND/OR trees and its application / Y. Shablya, D. Kruchinin, V. Kruchinin // Mathematics. – 2020. – Vol. 8, No. 6. – Article 962.

12. Ландо С.К. Лекции о производящих функциях. – 3-е изд. – М.: Изд-во МЦНМО, 2007. – 144 с.

13. Wilf H.S. Generatingfunctionology. – Second edition. – USA: Academic Press, 1994. – 228 p.

14. Kruchinin D. Method for obtaining coefficients of powers of bivariate generating functions / D. Kruchinin, V. Kruchinin, Y. Shablya // Mathematics. – 2021. – Vol. 9, No. 4. – Article 428.

15. Gao S. Tackling sequences from prudent self-avoiding walks / S. Gao, K.-H. Chen // Proceedings of the 2014 world congress in computer science, computer engineering, and applied computing. – 2014. – P. 1–7.

16. Sloane N.J.A. The on-line encyclopedia of integer sequences. – URL: https://oeis.org, свободный (дата обращения: 01.01.2022).

**Кручинин Дмитрий Владимирович**
Канд. физ.-мат. наук, доцент каф. компьютерных систем в управлении и проектировании (КСУП)
Томского государственного ун-та
систем управления и радиоэлектроники (ТУСУР)
Ленина пр-т, д. 40, г. Томск, Россия, 634050
ORCID: 0000-0003-3412-432X
Тел.: +7 (382-2) 41-47-17
Эл. почта: kruchinindm@gmail.com

Kruchinin D.V.

**Modification of the method for developing combinatorial generation algorithms based on the use of multivariate generating functions and approximations**

This article proposes a modified method for developing combinatorial generation algorithms based on AND/OR trees. The proposed method differs from the original method and its modifications by using the method of obtaining explicit expressions for the coefficients of multivariate generating functions to find the expression for the cardinality function of a given combinatorial set (including combinatorial sets defined by several parameters). Also, the proposed method is distinguished by the use of approximation methods and the binary search to find the selected son of the OR node. This reduces the computational complexity of the unranking algorithms. In order to test the proposed method, a new ranking and unranking algorithms were developed for a combinatorial set of self-avoiding lattice paths. In this case, the use of binary search reduced on average the number of required computational operations and a better computational complexity compared to the original version of the algorithm were obtained.

**Keywords:** combinatorial generation, multivariate generating functions, approximations, binary search, lattice path, ranking algorithm, unranking algorithm.
**DOI:** 10.21293/1818-0442-2021-25-1-55-60

*References*

1. Knuth D.E. *The Art of Computer Programming. Vol. 4A: Combinatorial Algorithms, Part 1*. USA, Addison-Wesley, 2011, 883 p.

2. Kreher D.L., Stinson D.R. *Combinatorial algorithms: Generation, Enumeration, and Search*. USA, CRC Press, 1999, 329 p.

3. Ruskey F. *Combinatorial Generation*. Working version (1j-CSC 425/520). 2003, 311 p. Available at: http://page.math.tu-berlin.de/~felsner/SemWS17-18/Ruskey-Comb-Gen.pdf (Accessed: January 01, 2022).

4. Reingold E.M., Nievergelt J., Deo N. *Combinatorial Algorithms: Theory and Practice*. USA, Prentice Hall, 1977, 433 p.

5. Barcucci E., Bernini A., Pinzani R. Exhaustive generation of some lattice paths and their prefixes. *Theoretical Computer Science*, 2021, vol. 878–879, pp. 47–52.

6. Flajolet P., Zimmerman P., Cutsem B. A calculus for the random generation of combinatorial structures. *Theoretical Computer Science*, 1994, vol. 132, no. 1–2, pp. 1–35.

7. Martinez C., Molinero X. A generic approach for the unranking of labeled combinatorial classes. *Random Structures and Algorithms*, 2001, vol. 19, no. 3–4, pp. 472–497.

8. Ryabko B.Ya. Fast enumeration of combinatorial objects. *Discrete Mathematics and Applications*, 1998, vol. 8, no. 2, pp. 163–182.

9. Kruchinin V.V. *Metody postroenija algoritmov generacii i numeracii kombinatornyh obektov na osnove derevev I/ILI* [Methods for developing algorithms for ranking and

unranking combinatorial objects based on AND/OR trees]. Tomsk, 2007, 200 p. (in Russ.).

10. Kruchinin V.V. *Metody, algoritmy i programmnoye obespecheniye kombinatornoy generatsii*. *Diss. doctor nauk* [Methods, algorithms and software for combinatorial generation. Doctor Diss.]. Tomsk, 2010, 387 p.

11. Shablya Y., Kruchinin D., Kruchinin V. Method for developing combinatorial generation algorithms based on AND/OR trees and its application. *Mathematics*, 2020, vol. 8, no. 6, Article 962.

12. Lando S.K. *Lectures on generating functions*. USA, American Mathematical Society, 2003, 148 p.

13. Wilf H.S. *Generatingfunctionology*. Second edition. USA, Academic Press, 1994, 228 p.

14. Kruchinin D., Kruchinin V., Shablya Y. Method for obtaining coefficients of powers of bivariate generating functions. *Mathematics*, 2021, vol. 9, no. 4, Article 428.

15. Gao S., Chen K.-H. *Tackling Sequences from Prudent Self-avoiding Walks*. Proceedings of the 2014 world congress in computer science, computer engineering, and applied computing, 2014, pp. 1-7.

16. Sloane N.J.A. The on-line encyclopedia of integer sequences. Available at: https://oeis.org/ (Accessed: January 01, 2022).

**Dmitry V. Kruchinin**
Candidate of Science in Mathematics and Physics,
Associated Professor, Department of Computer Control and Design Systems, Tomsk State University of Control Systems and Radioelectronics (TUSUR)
40, Lenin pr., Tomsk, Russia, 634050
ORCID: 0000-0003-3412-432X
Phone: +7 (382-2) 41-47-17
Email: kruchinindm@gmail.com