



# Realization of a Method for Calculating Bell Polynomials Based on Compositae of Generating Functions

Vadim S. Melman, Yuriy V. Shablya\*, Dmitry V. Kruchinin and Alexander A. Shelupanov

*Faculty of Security, Tomsk State University of Control Systems and Radioelectronics, Tomsk, Russia*

\*Corresponding author: [syv@keva.tusur.ru](mailto:syv@keva.tusur.ru)

**Abstract.** In this paper different computational methods for calculating partial and  $n$ -th complete Bell polynomials are considered. As one of the new methods, the authors propose to use the method for calculating Bell polynomials, which is based on compositae of generating functions. This method was realized by the authors in the form of a library for Wolfram Mathematica and compared with the built-in methods of Wolfram Mathematica and Maple. The results of the comparison demonstrate the advantage of the realization of the new method over the existing ones in spending time and memory for calculating.

**Keywords.** Bell polynomial; Calculation; Generating function; Composition; Composita; Library, Wolfram Mathematica

**MSC.** 11C08; 05A15; 97N80

**Received:** July 31, 2018

**Accepted:** December 28, 2018

Copyright © 2018 Vadim S. Melman, Yuriy V. Shablya Dmitry V. Kruchinin and Alexander A. Shelupanov. *This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.*

This article was submitted to “*International Conference on Mathematics*” organized by Fatih Sultan Mehmet Vakıf University, Topkapı (Cevizlibağ) Campus, Istanbul, Turkey during July 3-6, 2018.

Academic Editors: Dr. Kenan Yildirim, Mus Alparslan University, Turkey

Dr. Gumrah Uysal, Karabuk University, Turkey

Dr. Reza Abazari, University of Tabriz, Iran

## 1. Introduction

Bell polynomials are introduced in [2] and named in honor of the mathematician E. T. Bell. Partial Bell polynomials can be defined by the following generating function [25]:

$$\sum_{n,k \geq 0} B_{n,k}(x_1, x_2, \dots, x_{n-k+1}) \frac{t^n}{n!} u^k = \exp \left( \sum_{m \geq 1} x_m \frac{t^m}{m!} u \right).$$

The Bell polynomials are widely used for solving various problems of mathematical analysis, algebra, statistics, probability theory and combinatorial analysis. Many scientists have studied the Bell polynomials, such as L. Comtet [5], J. Riordan [25], S. Roman [26], and others.

Nowadays the Bell polynomials are also helpful, and they continue to be used in different mathematical problems. For example, in [3] the authors enumerate all colored partitions made by noncrossing diagonals of a convex polygon into polygons whose number of sides is congruent to  $b$  modulo  $a$  and give an explicit representation in terms of the partial Bell polynomials for the number of such partitions. In [23] the authors connect several explicit formulas for the partial Bell polynomials with the Bessel polynomials, apply the obtained formulas to give new expressions for the Catalan numbers and compute arbitrary higher order derivatives of elementary functions such as the sine, cosine, arcsine, arccosine, square root, logarithm, and exponential. Using some properties of the Bell polynomials, in [33] the authors establish two explicit formulas for the Motzkin numbers, the generalized Motzkin numbers, and the restricted hexagonal numbers. In [24] the authors use the explicit formulas of the Bell polynomials for deriving explicit formulas for the Euler numbers and polynomials, and in [31] the authors establish various mixed Euler sums and Stirling sums and present a unified approach to determining the evaluations of unknown Euler sums. In [22] the authors derive explicit expressions for the Bell polynomials and apply these to find explicit formulas for derivatives of trigonometric, logarithmic, and exponential elementary functions. The derivation of several properties and identities of Bell numbers and polynomials is presented in [?, 8].

Also in recent decades, the binary Bell polynomials have found their application. In [22] the authors present the link between the Bell polynomials and the Hirota D-operators. After that, the binary Bell polynomials are used for obtaining bilinear forms of nonlinear evolution equations, which makes it possible to find their multiple-soliton solutions. These studies are applicable to the description of nonlinear phenomena in hydrodynamics and plasma physics [7, 18, 21, 29, 30].

To use Bell polynomials for some practical problems, it is necessary to obtain coefficients for the powers of  $x$ . It can be done using the following explicit expression, which describes the partial Bell polynomials:

$$B_{n,k}(x_1, x_2, \dots, x_{n-k+1}) = \sum \frac{n!}{j_1! j_2! \dots j_{n-k+1}!} \left( \frac{x_1}{1!} \right)^{j_1} \left( \frac{x_2}{2!} \right)^{j_2} \dots \left( \frac{x_{n-k+1}}{(n-k+1)!} \right)^{j_{n-k+1}}, \quad (1)$$

where the sum is taken over all sequences  $j_1, j_2, \dots, j_{n-k+1}$  ( $k = \overline{1, n}$ ) and

$$\begin{cases} j_1 + j_2 + \dots + j_{n-k+1} = k, \\ j_1 + 2j_2 + \dots + (n-k+1)j_{n-k+1} = n. \end{cases}$$

Sometimes in practical problems there is a need to use the sum of partial Bell polynomials, which is called the  $n$ -th complete Bell polynomial and has the following form:

$$B_n(x_1, x_2, \dots, x_n) = \sum_{k=1}^n B_{n,k}(x_1, x_2, \dots, x_{n-k+1}). \quad (2)$$

Getting Bell polynomials in the form of a polynomial using (1) and (2) has a large computational complexity. Algorithms based on recurrence formulas are also used to obtain Bell polynomials. For example, it can be [26]

$$k B_{n,k}(x_1, x_2, \dots) = \sum_{i=1}^n \binom{n}{i} x_i B_{n-i, k-1}(x_1, x_2, \dots),$$

$$B_{n,k}(x_1, x_2, \dots) = \sum_{i=k-1}^{n-1} \binom{n-1}{i} x_{n-i} B_{i, k-1}(x_1, x_2, \dots),$$

or the recurrence formula [32]

$$B_{n,k}(x_1, x_2, \dots) = \sum_{i=1}^{n-k+1} \binom{n-1}{i-1} x_i B_{n-i, k-1}(x_1, x_2, \dots). \quad (3)$$

In [20] the following expression for the partial Bell polynomials is proved:

$$B_{n+1, k+1}(x_1, x_2, \dots, x_{n+1}) = \sum_{i=0}^{n-k} \binom{n}{i} x_{i+1} B_{n-i, k}(x_1, x_2, \dots, x_{n-i}).$$

Recurrence relations also exist for the  $n$ -th complete Bell polynomials, for example, the following relation given in [10]:

$$B_n(x_1, \dots, x_n) = \sum_{k=1}^n \binom{n-1}{k-1} x_k B_{n-k}(x_1, \dots, x_{n-k}).$$

But methods that are based on algorithms used recurrence relations also have a great computational complexity, which is estimated as  $O(n^2)$  [10]. This is a significant drawback of such algorithms. Hence, one of the problems associated with the use of Bell polynomials in practice is the complexity of calculating the coefficients for the powers of  $x$ . Therefore, the search for new computational methods for getting Bell polynomials is an important task.

Automation of calculations is also an important task. For example, in [4] the authors consider the effectiveness of practical implementation of integration methods for ordinary differential equations and compare new methods with the built-in methods of MATLAB. In [17] the authors realize a library for Wolfram Mathematica for efficient evaluation of multivariate residues based on methods from computational algebraic geometry. In [27] the authors realize a library for Wolfram Mathematica, that performs abstract vector calculus computations and is able to reduce three-dimensional scalar and vector expressions of a very general type to a well defined standard form. This library was applied for the automation of the calculation of high-order Lagrangians for the single particle guiding center system in plasma physics. In [9] the authors

realize a library for Wolfram Mathematica, that is dedicated to the study of various modeling, analysis, synthesis problems for nonlinear control systems.

In this paper the authors realize the method for calculating Bell polynomials based on compositae of generating functions [11] and compare this method with built-in methods of mathematical packages.

The article has the following structure:

- Section 2 presents computational methods for getting Bell polynomials.
- Section 3 defines the composita of a generating function and demonstrates its connection with the partial Bell polynomials.
- Section 4 describes a library for Wolfram Mathematica for calculating Bell polynomials, which is realized by the authors.
- Section 5 shows the main results of the comparison of methods for calculating Bell polynomials.
- Section 6 summarizes the results of the research.

## 2. Computational Methods for Calculating Bell Polynomials

Nowadays mathematical packages are widely used for solving different mathematical tasks. Getting Bell polynomials can be realized not only using explicit and recurrent formulas, but also applying mathematical packages. There are several built-in functions for calculating Bell polynomials in such software as Wolfram Mathematica, Maple, and Sage. In MATLAB there is a custom implementation of a method for calculating the partial Bell polynomials based on (3). Table 1 describes the functions of mathematical packages that can be used for calculating Bell polynomials.

**Table 1.** Functions of mathematical packages for calculating Bell polynomials

Mathematical package	Function	Description
Wolfram Mathematica	BellY	Return the partial Bell polynomial for the given values of $n$ and $k$ and the sequence $\{x_1, \dots, x_{n-k+1}\}$
Maple	IncompleteBellB	Return the partial Bell polynomial for the given values of $n$ and $k$ and the sequence $\{x_1, \dots, x_n\}$
	CompleteBellB	Return the $n$ -th complete Bell polynomial for the given value of $n$ and the sequence $\{x_1, \dots, x_n\}$
Sage	bell_polynomial	Return the partial Bell polynomial for the given values of $n$ and $k$
MATLAB	IncompleteBellPoly	Return a matrix of the partial Bell polynomials for the given values of $n$ and $k$ and the sequence $\{x_1, \dots, x_n\}$

Also there are other computational methods for calculating Bell polynomials. For example, in [19] a method for obtaining the matrix of partial Bell polynomials based on the analytic derivation of a polynomial is presented. In this paper special attention is paid to the specification of a polynomial, which includes coefficients, indices and powers of polynomial terms. It allows us to store polynomials in RAM, access to the element by its number, and make an exhaustive search.

### 3. Composita of a Generating Function and Partial Bell Polynomials

In this paper we realize the method for calculating Bell polynomials based on compositae of generating functions [11]. Previously, the compositae of generating functions were applied for obtaining explicit expressions of polynomials [13, 14] and for generating primality criteria that are used for distinguishing prime numbers from composite numbers [16].

The notion of compositae of generating functions is described in [12, 15] and can be apply to ordinary generating functions. According to R. P. Stanley [28], ordinary generating functions are defined as follows:

**Definition 1.** An ordinary generating function of the sequence  $(a_n)_{n \geq 0}$  is the formal power series

$$A(x) = a_0 + a_1x + a_2x^2 + \dots = \sum_{n \geq 0} a_n x^n.$$

**Definition 2.** The composita  $F^\Delta(n, k)$  of the generating function  $F(x) = \sum_{n > 0} f_n x^n$  is the coefficient function of the generating function  $(F(x))^k = \sum_{n \geq k} F^\Delta(n, k) x^n$ .

The composita can be represented using compositions as follows:

$$F^\Delta(n, k) = \sum_{\pi_k \in C_n} f_{\lambda_1} f_{\lambda_2} \dots f_{\lambda_k},$$

where  $C_n$  is the set of all compositions of an integer  $n$ ,  $\pi_k$  is the composition  $n$  into  $k$  parts such that  $\sum_{i=1}^k \lambda_i = n$ .

Also, the composita can be represented using partitions as follows:

$$F^\Delta(n, k) = \sum \binom{k}{j_1, j_2, \dots, j_{n-k+1}} f_1^{j_1} f_2^{j_2} \dots f_{n-k+1}^{j_{n-k+1}},$$

where the sum is taken over all sequences  $j_1, j_2, \dots, j_{n-k+1}$  ( $k = \overline{1, n}$ ) and

$$\begin{cases} j_1 + j_2 + \dots + j_{n-k+1} = k, \\ j_1 + 2j_2 + \dots + (n - k + 1)j_{n-k+1} = n. \end{cases}$$

The following recurrence relation holds true for compositae:

$$F^\Delta(n, k) = \begin{cases} f(n), & \text{for } k = 1; \\ \sum_{i=1}^{n-k+1} f_i F^\Delta(n - 1, k - 1), & \text{otherwise.} \end{cases}$$

If we consider a partial Bell polynomial where  $x_i$  is the  $i$ -th derivative of a function  $y(x)$ , then we get the following expression for calculating the partial Bell polynomial [11]:

$$B_{n,k} = \frac{n!}{k!} Y^{\Delta}(n, k, x), \quad (4)$$

where

$$Y^{\Delta}(n, k, x) = \sum_{\pi_k \in C_n} \frac{y^{(\lambda_1)}(x)}{\lambda_1!} \frac{y^{(\lambda_2)}(x)}{\lambda_2!} \dots \frac{y^{(\lambda_k)}(x)}{\lambda_k!}$$

is the composita of the generating function

$$Y(x, z) = y(x+z) - y(x) = \sum_{n>0} \frac{y^{(n)}(x)}{n!} z^n. \quad (5)$$

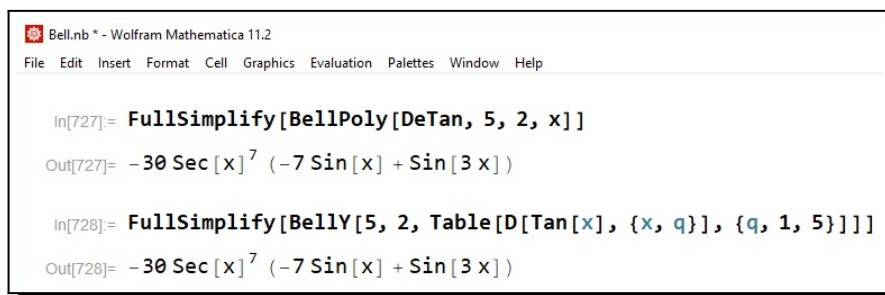
#### 4. Library for Wolfram Mathematica for Calculating Bell Polynomials

The method for calculating Bell polynomials based on compositae of generating functions was realized in the form of a library for Wolfram Mathematica. This library contains the following sections:

1. Compositae of generating functions  $F^{\Delta}(n, k)$ .
2. Rules for working with compositae  $F^{\Delta}(n, k)$ .
3. Compositae of generating functions  $Y^{\Delta}(n, k, x)$ .
4. Rules for calculating Bell polynomials based on compositae of generating functions.

The section “Compositae of generating functions  $F^{\Delta}(n, k)$ ” contains 150 functions for calculating compositae of different generating functions for polynomials and for rational, trigonometric, hyperbolic, logarithmic, exponential, and irrational functions. The section “Rules for working with compositae  $F^{\Delta}(n, k)$ ” contains 26 functions that describe the rules for calculating the sum, product, and composition of generating functions. The section “Compositae of generating functions  $Y^{\Delta}(n, k, x)$ ” contains 28 functions for calculating compositae of different generating functions in the form of (5) for rational, trigonometric, logarithmic, exponential, and irrational functions. The section “Rules for calculating Bell polynomials based on compositae of generating functions” contains functions that describe the rules for calculating partial Bell polynomials and  $n$ -th complete Bell polynomials.

Demonstration of the correct work of the library is presented in Figure 1, which shows calculating the partial Bell polynomial using the function *BellPoly* for  $n = 5$ ,  $k = 2$  and the sequence  $(x_1, x_2, \dots, x_{n-k+1})$ , where  $x_i$  is the  $i$ -th derivative of the function  $y(x) = \tan(x)$ . Figure 1 also shows the calculation of the same polynomial by the built-in functions of Wolfram Mathematica. It is clear that the results of the calculations are completely the same.



```

Bell.nb * - Wolfram Mathematica 11.2
File Edit Insert Format Cell Graphics Evaluation Palettes Window Help

In[727]:= FullSimplify[BellPoly[DeTan, 5, 2, x]]
Out[727]:= -30 Sec[x]^7 (-7 Sin[x] + Sin[3 x])

In[728]:= FullSimplify[BellY[5, 2, Table[D[Tan[x], {x, q}], {q, 1, 5}]]]
Out[728]:= -30 Sec[x]^7 (-7 Sin[x] + Sin[3 x])

```

**Figure 1.** Calculating the partial Bell polynomial for the function  $y(x) = \tan(x)$ .

In this example, the following functions of the library are used:

- The function *CompositaTan* calculates the composita of the generating function  $\tan(x)$  for the given  $n$  and  $k$ .
- The function *DeTan* calculates the composita of the generating function in the form of (5) for function  $y(x) = \tan(x)$  for the given  $n$ ,  $k$  and  $x$ .
- The function *BellPoly* calculates the partial Bell polynomial for the given  $Y^\Delta(n, k, x)$ ,  $n$ ,  $k$  and  $x$ .

## 5. Comparison of Methods for Calculating Bell Polynomials

The method for calculating Bell polynomials based on compositae of generating functions was compared only with the built-in methods of Wolfram Mathematica and Maple. The built-in method of Sage and the method described in [19] do not allow us to get Bell polynomials where  $x_i$  is the  $i$ -th derivative of a function  $y(x)$  because in these methods it is not possible to enter a sequence  $(x_1, x_2, \dots, x_{n-k+1})$ . The built-in method of MATLAB is based on a recurrence relation and its computational complexity is known.

Wolfram Mathematica and Maple are proprietary software, and their built-in algorithms are not freely available. Therefore, the comparison of the methods for calculating Bell polynomials was carried out using the data obtained by conducting an experiment. Time and memory spent on calculations were used as comparison criteria.

The comparison of methods for calculating Bell polynomials includes the following steps:

1. The preparation of a test set of functions for testing.
2. The measurement of time and memory spent on calculations for all methods and for the same functions.
3. The comparison of the obtained results.

For testing we prepare a test set of functions that consists of three parts. Each part is distinguished by the complexity of its functions. The first part of the test set consists of elementary functions. The second part of the test set consists of complex functions that are obtained from two elementary functions and one operation on them (addition, multiplication, composition). The third part of the test set consists of complex functions that are obtained from

three elementary functions and two operations on them. Each part contains 10 functions, the test set is given in Table 2.

**Table 2.** Test set of functions for testing

No.	First part	Second part	Third part
1	$\sin(x)$	$\sin(\tan(x))$	$\sin\left(\tan\left(\frac{1}{x}\right)\right)$
2	$\tan(x)$	$\ln(x) + xe^x$	$\tan(\ln^2(x))$
3	$\sqrt{x}$	$\cos(2x - x^2 + x^3)$	$e^{\frac{1-x}{1+x}} + \tan(x)$
4	$xe^x$	$\sqrt{x} + \frac{1-x}{1+x}$	$\ln(x) + e^x + \frac{x}{\sqrt{1-x^2}}$
5	$\frac{1-x}{1+x}$	$\sqrt{\tan(x)}$	$\sqrt{4x - x^2 + 2x^3} + \frac{1-x}{1+x}$
6	$\frac{1}{x}$	$\frac{1}{1-x-x^2} + x^2$	$\ln\left(\sqrt{\frac{1}{x}}\right)$
7	$\sqrt{1-x^2}$	$5x - 3x^2 + \ln(x)$	$\frac{1-x}{1+x} + \tan^2(x)$
8	$\arccos(x)$	$\sqrt{4x - x^2 + 2x^3}$	$\frac{1}{x+3x^2-2x^3} + x^2$
9	$\sec(x)$	$e^x + x + \frac{1}{x}$	$\sqrt{\ln(2x - x^2 + x^3)}$
10	$-3x + 2x^2 + 5x^3$	$\frac{1-\sqrt{x}}{1+\sqrt{x}}$	$\frac{1-\sqrt{x}}{1+\sqrt{x}} + e^x$

The testing was performed on a computer with the following characteristics: Intel Core i3-6100U (2.3 GHz), 12 GB DDR4 RAM. For each function from the test set, we measured time and memory spent on calculating  $n$ -th complete Bell polynomials by each method for  $n$  in the interval from 10 to 70 in increments of 10. To measure time and memory, we used the functions *Timing* and *ByteCount* for Wolfram Mathematica, *time* and *Usage* with the option *bytes used* for Maple. To calculate  $n$ -th complete Bell polynomials, we used the functions *BellY* for Wolfram Mathematica and *CompleteBellB* for Maple.

Table 3 shows the average values of time spent on calculations for each part of the test set and for the entire test set.

**Table 3.** Average values of time spent on calculations

Computational method	First part	Second part	Third part	Entire test set
Library	3.82	13.50	106.34	41.22
Wolfram Mathematica	24.05	24.45	124.92	57.81
Maple	79.18	407.04	1518.83	668.35

Figures 2 – 5 show graphs of the dependence of time spent on calculations on  $n$  for each part of the test set and for the entire test set. The secondary vertical axis shows the time spent on calculations for Maple because it is much greater than for the other methods.

On the basis of the obtained results, it can be noted that the increase in the complexity of used functions for calculating Bell polynomials increases the time spent on calculations. There is the smallest increase in time for Wolfram Mathematica (5.19 times) and the greatest increase



in time for the library (27.84 times). In absolute values, the average value of time spent on calculations for the library is smaller than for the other methods.

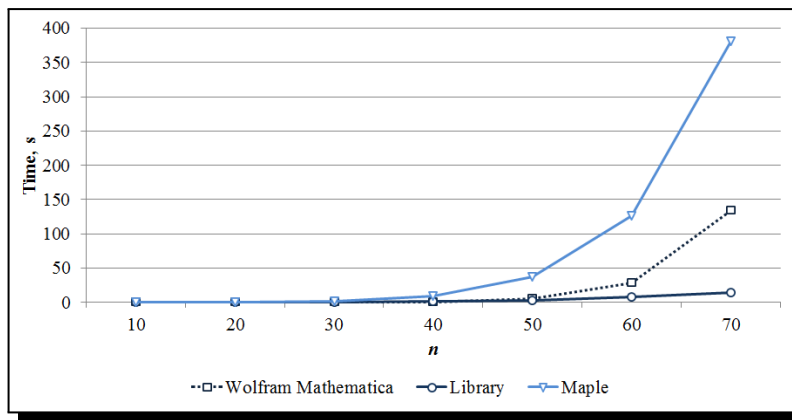


Figure 2. Dependence of time spent on calculations on  $n$  for the first part of the test set

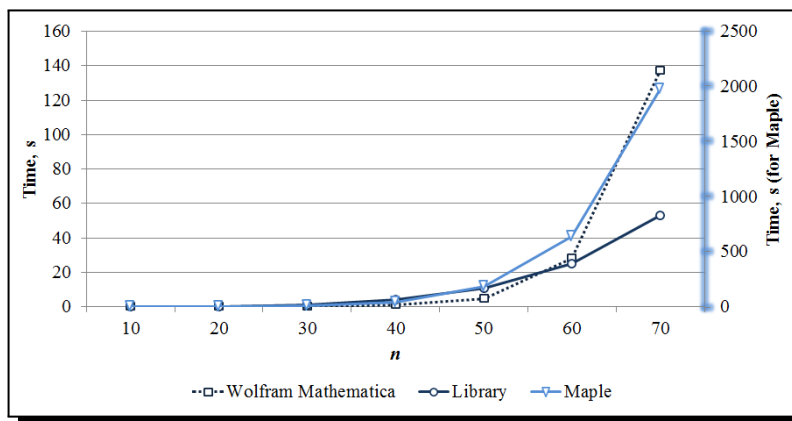


Figure 3. Dependence of time spent on calculations on  $n$  for the second part of the test set

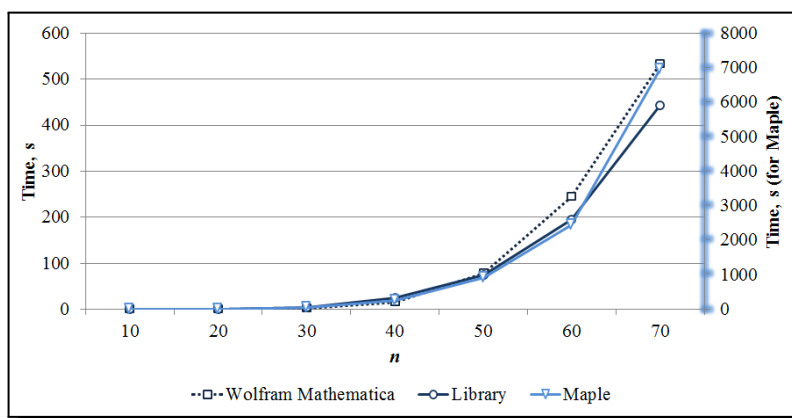


Figure 4. Dependence of time spent on calculations on  $n$  for the third part of the test set

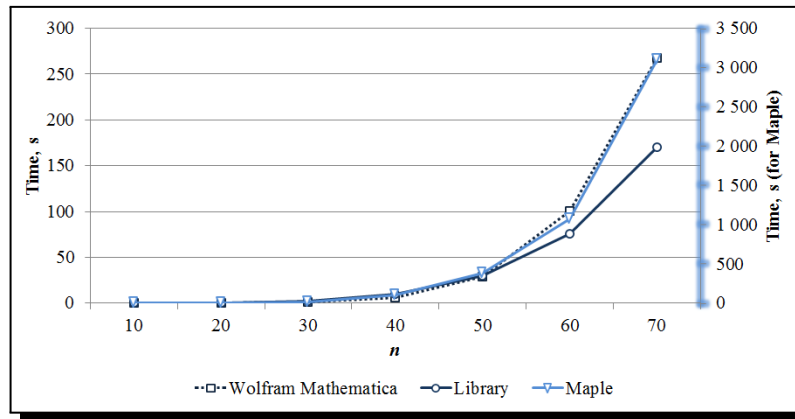


Figure 5. Dependence of time spent on calculations on  $n$  for the entire test set

Table 4 shows the average values of memory spent on calculations for each part of the test set and for the entire test set.

Table 4. Average values of memory spent on calculations

Computational method	First part	Second part	Third part	Entire test set
Library	5.35	166.33	432.92	201.53
Wolfram Mathematica	6360.51	29141.23	243032.40	92844.71
Maple	3117.14	16004.49	187527.97	68883.20

Figures 6 – 9 show graphs of the dependence of memory spent on calculations on  $n$  for each part of the test set and for the entire test set. The secondary vertical axis shows the memory spent on calculations for the library because it is much smaller than for the other methods.

On the basis of the obtained results, it can be noted that the increase in the complexity of used functions for calculating Bell polynomials increases the memory spent on calculations. There is the smallest increase in memory for Wolfram Mathematica (38.21 times) and the greatest increase in memory for the library (80.92 times). In absolute values, the average value of memory spent on calculations for the library is smaller than for the other methods.

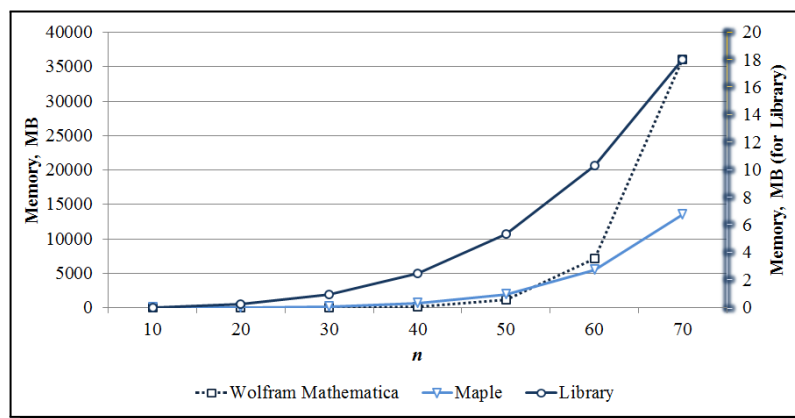


Figure 6. Dependence of memory spent on calculations on  $n$  for the first part of the test set

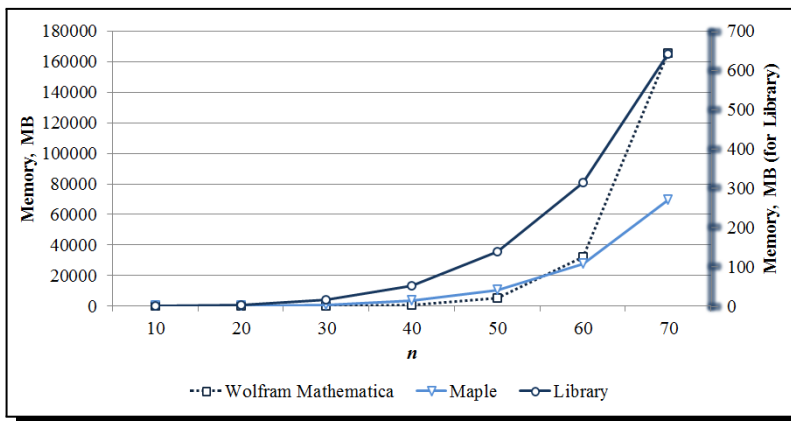


Figure 7. Dependence of memory spent on calculations on  $n$  for the second part of the test set

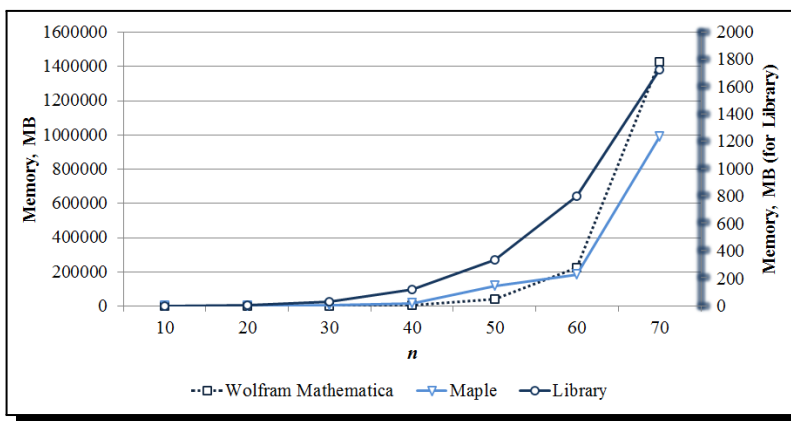


Figure 8. Dependence of memory spent on calculations on  $n$  for the third part of the test set

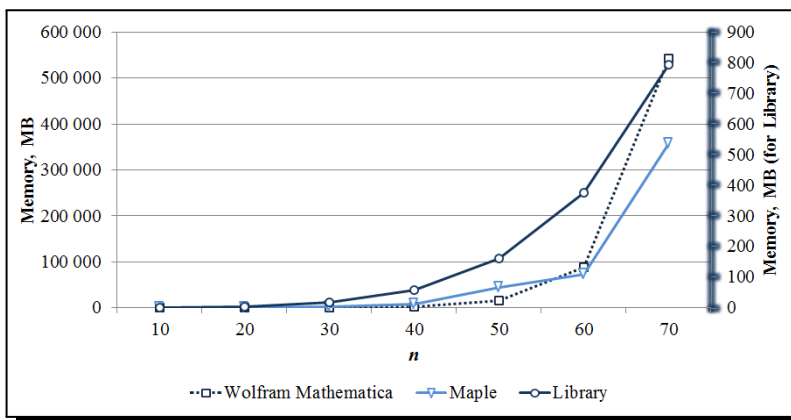
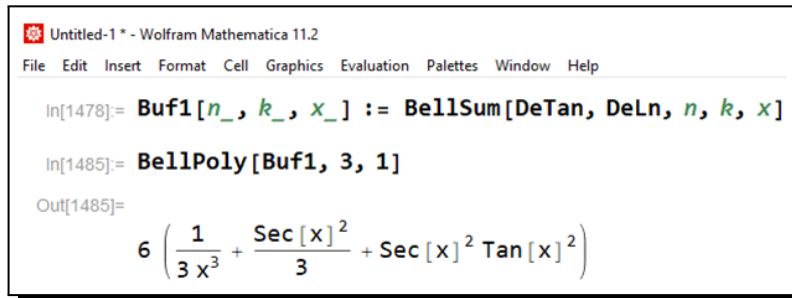


Figure 9. Dependence of memory spent on calculations on  $n$  for the entire test set

Thus, the method for calculating Bell polynomials based on compositae of generating functions spends less resources than the built-in methods of Wolfram Mathematica and Maple. One drawback of using the library is the need for the manual decomposition of complex functions

into elementary functions and operations on them. Figure 10 shows an example of calculating the partial Bell polynomial for the function  $y(x) = \tan(x) + \ln(x)$  with the manual decomposition. This drawback can be corrected by developing an algorithm for automatic decomposition of a given function.



```

Untitled-1* - Wolfram Mathematica 11.2
File Edit Insert Format Cell Graphics Evaluation Palettes Window Help

In[1478]:= Buf1[n_, k_, x_] := BellSum[DeTan, DeLn, n, k, x]

In[1485]:= BellPoly[Buf1, 3, 1]

Out[1485]=
6 ( 1/(3 x^3) + Sec[x]^2/3 + Sec[x]^2 Tan[x]^2 )

```

**Figure 10.** Calculating the partial Bell polynomial for the function  $y(x) = \tan(x) + \ln(x)$

## 6. Conclusion

In this paper we have considered different computational methods for calculating partial and  $n$ -th complete Bell polynomials. As one of the new methods, we propose to use the method for calculating Bell polynomials, which is based on compositae of generating functions. We have realized this method in the form of a library for Wolfram Mathematica and compared with the built-in methods of Wolfram Mathematica and Maple. The results of the comparison demonstrate the advantage of the realization of the new method over the existing ones in spending memory for calculating. Also there is the advantage of the new method over the built-in method of Maple in spending time for calculating.

## Acknowledgement

This work was funded by Russian Foundation for Basic Research and the government of the Tomsk region of Russian Federation (grants no. 18-41-703006 and no. 18-31-00201) and the Ministry of Education and Science of Russian Federation (government order no. 2.8172.2017/8.9, TUSUR).

## Competing Interests

The authors declare that they have no competing interests.

## Authors' Contributions

All the authors contributed significantly in writing this article. The authors read and approved the final manuscript.

## References

- [1] A. Bayad, Y. Simsek and H.M. Srivastava, Some array type polynomials associated with special numbers and polynomials, *Appl. Math. Comput.* **244** (2014), 149 – 157.

- [2] E.T. Bell, Partition polynomials, *Ann. of Math. (2)* **29**(1/4) (1927-1928), 38 – 46.
- [3] D. Birmajer, J.B. Gil and M.D. Weiner, Colored partitions of a convex polygon by noncrossing diagonals, *Discrete Math.* **340**(4) (2017), 563 – 571.
- [4] V.P. Bubnov, A.S. Eremin, N.A. Kovrizhnykh and I.V. Olemskoy, Comparative study of the advantages of structural numerical integration methods for ordinary differential equations, *SPIIRAS Proceedings* **4**(53) (2017), 51 – 72.
- [5] L. Comtet, *Advanced Combinatorics*, D. Reidel Publishing Company (1974).
- [6] C. Gilson, F. Lambert, J. Nimmo and R. Willox, On the combinatorics of the Hirota D-operators, *Proc. R. Soc. Lond. A* **452**(1945) (1996), 223 – 234.
- [7] Y. Jiang, B. Tian, W.-J. Liu, M. Li, P. Wang and K. Sun, Solitons, Backlund transformation, and Lax pair for the  $(2+1)$ -dimensional Boiti-Leon-Pempinelli equation for the water waves, *J. Math. Phys.* **51**(9) (2010), 1 – 11.
- [8] D.S. Kim and T. Kim, Some identities of Bell polynomials, *Sci. China Math.* **58**(10) (2015), 1 – 10.
- [9] U. Kotta and M. Tonso, NLControl – a Mathematica package for nonlinear control systems, *IFAC-PapersOnLine* **50**(1) (2017), 681 – 686.
- [10] M.I. Krivoruchenko, Trace identities for skew-symmetric matrices, *Mathematics and Computer Science* **1**(2) (2016), 21 – 28.
- [11] V. Kruchinin, Derivation of Bell polynomials of the second kind, arXiv:1104.5065v1 (2011), 1 – 15.
- [12] D.V. Kruchinin and V.V. Kruchinin, A method for obtaining generating functions for central coefficients of triangles, *J. Integer Seq.* **15** (2012), 1 – 10.
- [13] D.V. Kruchinin and V.V. Kruchinin, Application of a composition of generating functions for obtaining explicit formulas of polynomials, *J. Math. Anal. Appl.* **404**(1) (2013), 161 – 171.
- [14] D.V. Kruchinin and V.V. Kruchinin, Explicit formulas for some generalized polynomials, *Appl. Math. Inf. Sci.* **7**(5) (2013), 2083 – 2088.
- [15] V.V. Kruchinin and D.V. Kruchinin, Composita and its properties, *Journal of Analysis and Number Theory* **2** (2014), 37 – 44.
- [16] D.V. Kruchinin, Y.V. Shablya, O.O. Evsutin and A.A. Shelupanov, Integer properties of a composition of exponential generating functions, in: *Proc. Intl. Conf. of Numerical Analysis and Applied Mathematics*, T. Simos and C. Tsitouras (eds.), American Institute of Physics Inc., 2017, pp. 1 – 4.
- [17] K.J. Larsen and R. Rietkerk, MULTIVARIATERESIDUES: A Mathematica package for computing multivariate residues, *Comput. Phys. Commun.* **222** (2018), 250 – 262.
- [18] P.-L. Ma, S.-F. Tian, L. Zou and T.-T. Zhang, The solitary waves, quasi-periodic waves and integrability of a generalized fifth-order Korteweg-de Vries equation, *Waves Random Complex Media* (2018), 1 – 17.
- [19] V.A. Melnikova, Algorithm for partition polynomials analytical derivation, *Systems, Methods, Technologies* **3**(19) (2013), 112 – 116.
- [20] P. Natalini and P.E. Paolo, Remarks on Bell and higher order Bell polynomials and numbers, *Cogent Mathematics* **3** (2016), 1 – 15.
- [21] D. Pelinovsky, J. Springael, F. Lambert and I. Loris, On modified NLS, Kaup and NLBq equations: Differential transformations and bilinearization, *J. Phys. A: Math. Gen.* **30**(24) (1997), 8705 – 8717.
- [22] F. Qi and M.-M. Zheng, Explicit expressions for a family of the Bell polynomials and applications, *Appl. Math. Comput.* **258** (2015), 597 – 607.

- [23] F. Qi, X.-T. Shi, F.-F. Liu and D.V. Kruchinin, Several formulas for special values of the bell polynomials of the second kind and applications, *J. Appl. Anal. Comput.* **7**(3) (2017), 857 – 871.
- [24] F. Qi and B.-N. Guo, Explicit formulas for special values of the Bell polynomials of the second kind and for the Euler numbers and polynomials, *Mediterr. J. Math.* **14**(3) (2017), 1 – 14.
- [25] J. Riordan, *An Introduction to Combinatorial Analysis*, Princeton University Press (2016).
- [26] S. Roman, *The Umbral Calculus*, Academic Press (1984).
- [27] J. Squire, J. Burby and H. Qin, VEST: Abstract vector calculus simplification in Mathematica, *Comput. Phys. Commun.* **185**(1) (2014), 128 – 135.
- [28] R.P. Stanley, *Enumerative Combinatorics*, Volume **2**, Cambridge University Press (1999).
- [29] J.-M. Tu, S.-F. Tian, M.-J. Xu, P.-L. Ma and T.-T. Zhang, Solitons, Backlund transformation, and Lax pair for the  $(2 + 1)$ -dimensional Boiti-Leon-Pempinelli equation for the water waves, *Computers and Mathematics with Applications* **72**(9) (2016), 2486 – 2504.
- [30] Y.-L. Wang, Y.-T. Gao, S.-L. Jia, G.-F. Deng and W.-Q. Hu, Solitons for a  $(2 + 1)$ -dimensional variable-coefficient Bogoyavlensky-Konopelchenko equation in a fluid, *Modern Phys. Lett. B* **31**(25) (2017), 1 – 18.
- [31] W. Wang, Euler sums and Stirling sums, *J. Number Theory* **185** (2018), 160 – 193.
- [32] F.S. Wheeler, Bell polynomials, *ACM SIGSAM Bulletin* **21**(3) (1987), 44 – 53.
- [33] J.-L. Zhao and F. Qi, Two explicit formulas for the generalized Motzkin numbers, *J. Inequal. Appl.* **2017**(1) (2017), 1 – 8.