Time Complexity Analysis for Cullis/Radic and Dodgson's Generalized/Modified Method for Rectangular Determinants Calculations

Armend Salihu*,Halil Snopce*, Artan Luma*, Jaumin Ajdari*
South East European University, Tetovo, NORTH MACEDONIA.

Abstract

In this paper we present an analysis of the time complexity of algorithms based on Cullis/Radic Definition and Dodgson's Generalized/Modified Method for calculating rectangular/non-square determinants. We have identified the asymptotic time complexity of these algorithms, and that both algorithms have their advantages in relation to time complexity. From the time complexity analysis, we observed that the Cullis/Radic definition has an asymptotic time complexity of $O(C\binom{m}{n} \cdot m^3)$, while Dodgson's Generalized/Modified Method has an asymptotic time complexity of $O(2^{2m} \cdot (n-m)^2)$. Further, we noticed that in cases where the number of rows is less than or equal to half the number of columns, it is more appropriate to use the algorithm based on Dodgson's Generalized/Modified Method, while in cases where the number of rows is greater than half the number of columns, then Cullis/Radic Definition based algorithm is more suitable to use. Based on this analysis, we have also presented an algorithm which is a combination of these two algorithms and depending on the ratio between the number of rows and columns the rectangular determinant is calculated with the most appropriate method, for which we calculated the worst-case asymptotic time complexity as $O(\frac{n!}{((n/2)!)^2} \cdot \frac{n}{2}^3)$ while the best-case asymptotic time complexity is calculated as $O(n^3)$

Key Words: Rectangular determinants; time complexity; Dodgson's method; pivotal condensation; execution time.

1 Cullis/Radic and Generalized/Modified Dodgson's Method for Rectangular Determinants Calculation

The following presents the determinant calculation method based on the Cullis/Radic definition:

Theorem 1. *Let A be $m \times n$ a rectangular matrix:*

$$A_{m\times n} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}. \quad (1)$$

Its determinant, where $m \leq n$ is the sum (See: [4] [8]):

$$det(A_{m\times n}) = |A_{m\times n}| = \begin{vmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{vmatrix}$$

$$= \sum_{1<j_1<\cdots<j_m<n} (-1)^{r+s} \begin{vmatrix} a_{1j_1} & a_{1j_2} & \cdots & a_{1j_n} \\ a_{2j_1} & a_{2j_2} & \cdots & a_{2j_n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{mj_1} & a_{mj_2} & \cdots & a_{mj_n} \end{vmatrix}. \quad (2)$$

where $r = 1 + \cdots + m, s = j_1 + \cdots + j_m$.

Proof. See definition 1 in [8]. □

The following the pseudocode of the algorithm based on the above-mentioned method for calculating determinant of rectangular matrices.

P 1: Algorithm (det_A) based on Cullis-Radic method to calculate rectangular determinants

Step 1: Identify all combinations for determining m×m square determinants from columns combinations:
 if $m = n$
  Calculate square determinant with known methods.
 else

$$B = nchoosek(1:n,m);$$

Step 2: Identify all square determinants from the combination of columns:
 Create loop from 1 to total number of combinations (length of vector B)

$$D\{i\} = A(1:m, B(i,1:m)));$$

Step 3: Calculate determinants of square blocks from D
 Create loop from 1 to total number of combinations (length of vector B)

$$d = d + (-1)^\wedge(sum(1:m) + sum(B(i,1:m))) * SquareDet(D\{i\});$$

Step 4: Display the result of the determinant

* Faculty of Contemporary Sciences and Technologies, Emails: as28364@seeu.edu.mk, h.snopce@seeu.edu.mk, a.luma@seeu.edu.mk, and j.ajdari@seeu.edu.mk.

Theorem 2. *(Generalized Dodgson's formula) [2] Let A be $m \times n$ a rectangular matrix. Then for $p = min(m,n) \geq 2$, we have:*

$$
\begin{aligned}
&det\left(A_{\substack{1\leq i\leq m\\1\leq j\leq n}}\right)\cdot det\left(A_{\substack{i\neq m-1,m\\j\neq n-1,n}}\right)\\
&= \underset{(\varepsilon,p-1)}{det}\left(A_{\substack{1\leq i<m\\1\leq j<n}}\right)\cdot \underset{(\varepsilon,p-1)}{det}\left(A_{\substack{1<i\leq m\\1<j\leq n}}\right)\\
&- \underset{(\varepsilon,p-1)}{det}\left(A_{\substack{1\leq i<m\\1<j\leq n}}\right)\cdot \underset{(\varepsilon,p-1)}{det}\left(A_{\substack{1<i\leq m\\1\leq j<n}}\right)\\
&+ \underset{(\varepsilon,p)}{det}\left(A_{\substack{1\leq i\leq m\\1<j<n}}\right)\cdot \underset{(\varepsilon,p-2)}{det}\left(A_{\substack{1<i<m\\1\leq j\leq n}}\right)
\end{aligned}
\tag{3}
$$

Proof. See theorem 5.1 in [2]. □

In the following, we have developed the computer algorithm ($det\_Dodgson$) for theorem 1.

Since this method is applied for $m \geq 3$, and $m \leq n-2$, m-number of rows, n-number of columns of the matrix. The following is presented on the pseudocode of theorem 1.

P 2: Algorithm ($det\_Dodgson$) for generalized Dodgson method to calculate rectangular determinants

---

Step 1: Checking for conditions:
 if $m < 3$ or $m = n-1$
 Calculate rectangular determinant with known methods, like Laplace, Radic, Chios-like, etc.
 else if $m = n$
 Calculate square determinant with known methods.
 else
Step 2: Calculate submatrices:
 Calculate submatrices presented on theorem 1, calling $det\_Dodgson$ algorithm until the conditions of step 1 are met, as following:

$$
\begin{gathered}
d1 = det\_Dodgson(A(1:m-1,1:n-1));\\
d2 = det\_Dodgson(A(1:m-1,2:n));\\
d3 = det\_Dodgson(A(2:m,1:n-1));\\
d4 = det\_Dodgson(A(2:m,2:n));\\
d5 = det\_Dodgson(A(2:m-1,1:n));\\
d6 = det\_Dodgson(A(1:m,2:n-1));\\
d7 = det\_Dodgson(A(2:m-1,2:n-1));
\end{gathered}
$$

Step 3: After calculating submatrices, calculate the result of the determinant as following:

$$d = (d1*d4 - d2*d3 + d5*d6)/d7;$$

Recently, in 2022 we identified 9 different cases of Dodgson's generalization formula for rectangular determinant calculation, which is provided in theorem 3.

Theorem 3. *[10] The pivot block $\underset{(\varepsilon,p-1)}{det}\left(A_{\substack{1<i<m\\1<j<n}}\right)$ of Bayat's formula can be any block of order $(m-2)\times(n-2)$ from the given determinant, and the following cases are:*

Case 1: Pivot block is: $\underset{(\varepsilon,p-1)}{det}\left(A_{\substack{1\leq i\leq m-2\\1\leq j\leq n-2}}\right)$

Case 2: Pivot block is: $\underset{(\varepsilon,p-1)}{det}\left(A_{\substack{1\leq i\leq m-2\\2\leq j\leq n-1}}\right)$

Case 3: Pivot block is: $\underset{(\varepsilon,p-1)}{det}\left(A_{\substack{1\leq i\leq m-2\\3\leq j\leq n}}\right)$

Case 4: Pivot block is: $\underset{(\varepsilon,p-1)}{det}\left(A_{\substack{2\leq i\leq m-1\\1\leq j\leq n-2}}\right)$

Case 5: Pivot block is: $\underset{(\varepsilon,p-1)}{det}\left(A_{\substack{2\leq i\leq m-1\\2\leq j\leq n-1}}\right)$

Case 6: Pivot block is: $\underset{(\varepsilon,p-1)}{det}\left(A_{\substack{2\leq i\leq m-1\\3\leq j\leq n}}\right)$

Case 7: Pivot block is: $\underset{(\varepsilon,p-1)}{det}\left(A_{\substack{3\leq i\leq m\\1\leq j\leq n-2}}\right)$

Case 8: Pivot block is: $\underset{(\varepsilon,p-1)}{det}\left(A_{\substack{3\leq i\leq m\\2\leq j\leq n-1}}\right)$

Case 9: Pivot block is: $\underset{(\varepsilon,p-1)}{det}\left(A_{\substack{3\leq i\leq m\\3\leq j\leq n}}\right)$

Proof. See theorem 3 in [10]. □

The pseudocode of each case from theorem 2 is like pseudocode presented in P 2, and changes in steps 2 for each case. For example, the pseudocode for case 1 is changed as following:

P 3: Modified algorithm (det_Blocks) based on theorem 2 (as example is considered case 1)

Step 1: Checking for conditions:
 if $m < 3$ or $m = n-1$
 Calculate rectangular determinant with known methods, like Laplace, Radic, Chios-like, etc.
 else if $m = n$
 Calculate square determinant with known methods.
 else
Step 2: Calculate submatrices:
 Calculate submatrices presented on theorem 1, calling $det\_Dodgson$ algorithm until the conditions of step 1 are met:

$$
\begin{gathered}
d1 = det\_Blocks(A(1:m-1,1:n-1));\\
d2 = det\_Blocks(A(1:m-1,[1:n-2 \quad n]));\\
d3 = det\_Blocks(A([1:m-2 \quad m],1:n-1));\\
d4 = det\_Blocks(A([1:m-2 \quad m],[1:n-2 \quad n]));\\
d5 = det\_Blocks(A(1:m-2,1:n));\\
d6 = det\_Blocks(A(1:m,1:n-2));\\
d7 = det\_Blocks(A(1:m-2,1:n-2));
\end{gathered}
$$

Step 3: After calculating submatrices, calculate the result of the determinant as following:

$$d = (d1 * d4 - d2 * d3 + d5 * d6)/d7;$$

The pseudocode presented in P 3 represents case 1 of theorem 2. However, the same algorithm can be used for each case of theorem 2, with changes in step 2 while selecting pivot block and reflecting that pivot block in each submatrix.

The above-mentioned theorem and pseudocode, has its advantage in cases of matrices with several zero elements. We have developed the algorithm that finds pivot block with highest number of zero elements, which is presented in pseudocode P 4 [10].

P 4: Find the block of order $(m-2) \times (n-2)$ with highest number of zero elements

Step 1: Insert the rectangular determinant A
Step 2: Calculate number of nonzero elements for each row/column
Initialize R for rows and C for columns
Create loop for i from 1 to m
Create loop for j from 1 to n
if $A(i,j) \neq 0$

$$R(i) = R(i) + 1;$$
$$C(i) = C(i) + 1;$$

Step 3: Find the best case with the highest number of zero elements
Initialize first case: $k = 1$
if $C(2) + C(n-1) < C(1) + C(n)$

$$k = 2;$$

else if $C(1) + C(2) < C(n-1) + C(n)$

$$k = 3;$$

if $R(2) + R(m-1) < R(1) + R(m)$

$$k = k + 3;$$

else if $R(1) + R(2) > R(m-1) + R(m)$

$$k = k + 6;$$

Step 4: Return best case

2 Time Complexity Analysis

In the following we present the time complexity analysis of the above-mentioned algorithms [7] [12] [9] [3].

Time complexity analysis of function (det_A) of algorithm P 1, based on Cullis-Radic method, is presented in Table 1.

Table 1: Time complexity of det_A function

Function: det_A		Cost	time
$[m,n] = size(A);$		$T_1 = const_1$	1
$d = 0;$		$T_2 = const_2$	1
if $m == n$ $d = det(A);$		$T_3 = n^3$	1
else	$B = nchoosek(1:n,m);$	$T_4 = const_4$	1
	for $i = 1 : length(B)$ d=d+ (-1)^(sum(1:m) +sum(B(i,[1:m]))) *det((A([1:m],[B(i,[1:m])]))); end	There are several methods to calculate square determinants with different time complexity, however we will be based on LU factorization method [16]: $T_4 = m^3$	$C\binom{n}{m}$

Based on Table 1, we have:

$$Total\_Cost = 1 \cdot T_1 + 1 \cdot T_2 + 1 \cdot T_3 + Max(1 \cdot T_4, C\binom{n}{m} \cdot T_4)$$
$$= 1 \cdot const_1 + 1 \cdot const_2 + 1 \cdot const_3 + Max(1 \cdot n^3, C\binom{n}{m} \cdot m^3).$$

Hence, the highest order is $C\binom{n}{m} \cdot m^3$. After eliminating constants, the asymptotic time complexity is $O(C\binom{n}{m} \cdot m^3)$.

Time complexity analysis of function $(det\_Dodgson)$ of algorithm P 2, based on Dodgson's generalized method provided by Bayat, is presented in Table 2.

Table 2: Time complexity of *det_Dodgson* function

<table>
<tr><th colspan="2">Function: det_Dodgson</th><th>Cost</th><th>Times</th></tr>
<tr><td colspan="2">[m,n] = size(A);</td><td>$T_1 = const_1$</td><td>1</td></tr>
<tr><td colspan="2">if m==n
d=det(A);</td><td>$T_2 = n^3$</td><td>1</td></tr>
<tr><td colspan="2">if m==n-1
$d = det\_Ones(A)$;</td><td>Based on Algorithm 2.2 (See [11]), transforms
determinant of order $(n-1)\times n$ to $n \times n$
by adding one row of elements equal to 1.
Square determinant's time complexity is $T_3 = O(n^3)$.</td><td>1</td></tr>
<tr><td colspan="2">else if $m < 3$
$d = det\_A(A)$;</td><td>As it is calculated the det_A time complexity is:
$T_4(3,n) = C\binom{n}{3}\cdot 3^3 = \frac{n\cdot(n-1)\cdot(n-2)\cdot(n-3)!}{3!\cdot(n-3)!}\cdot 3^3$
$= n\cdot(n-1)\cdot(n-2)\cdot 4.5 \approx n^3$.</td><td>1</td></tr>
<tr><td rowspan="3">else</td><td>$d1 = det\_Dodgson(A(1:m-1, 1:n-1))$;
$d2 = det\_Dodgson(A(1:m-1, 2:n]))$;
$d3 = det\_Dodgson(A(2:m, 1:n-1))$;
$d4 = det\_Dodgson(A(2:m, 2:n]))$;</td><td colspan="2">$T_{5-1}(m,n) = 4\cdot T_{5-1}(m-1,n-1)+1$,
$T_{5-1}(m-1,n-1) = 4\cdot T_{5-1}(m-1-1,n-1-1)+1$
$= 4\cdot T_{5-1}(m-2,n-2)+1$,
$T_{5-1}(m,n) = 4\cdot(4\cdot(T_{5-1}(m-2,n-2))+1+1$
$= 4^2\cdot T_{5-1}(m-2,n-2)+2$
for any k, we have:
$T_{5-1}(m,n) = 4^k\cdot T_{5-1}(m-k,n-k)+k$,
for $m-k=2 \Rightarrow k=m-2$,
$T_{5-1}(m,n) = 4^{m-2}\cdot T_{5-1}(1,n-m+2)+m-2$
Based on the first condition:
$T_{5-1}(2,n-m+2 = C\binom{n-m+2}{2}\cdot 2^3$
$\frac{(n-m+2)\cdot(n-m+1)}{2}\cdot 2^3 = 4\cdot(n-m+2)\cdot(n-m+1)$.
$T_{5-1}(m,n) = 4^{m-2}\cdot 4\cdot(n-m+2)\cdot(n-m+1)+m-2$
$\approx 4_{m-1}\cdot(n-m+2)\cdot(n-m+1)$.</td></tr>
<tr><td>$d5 = det\_Dodgson(A(2:m-1, 1:n))$;</td><td colspan="2">$T_{5-2}(m,n) = T5-2(m-1,n-1)+1$,
$T_{5-2}(m-1.n-1) = T_{5-2}(m-1-1,n-1-1)+1$
$= T_{5-2}(m-2,n-2)+1$,
$T_{5-2}(m,n) = T_{5-2}(m-2,n-2)+1+1 = T_{5-2}(m-2,n-2+2$,
for any k, we have:
$T_{5-2}(m,n) = T_{5-2}(m-k,n-k)+k$,
for $m-k=2 \Rightarrow k=m-2$,
$T_{5-2}(m,n) = T_{5-2}(2,n-m-+2)+m-2$.
Based on first condition:
$T_{5-2}(2,n-m+2) = C\binom{n-m+2}{2}\cdot 2^3 = \frac{(n-m+2)\cdot(n-m+1)}{2}\cdot 2^3$
$= 4\cdot(n-m+1)\cdot(n-m+1)$.
$T_{5-2}(m,n) = 4\cdot(n-m+2)\cdot(n-m+1)+m-2$
$\approx 4\cdot(n-m+2)\cdot(n-m+1)$.</td></tr>
<tr><td>$d6 = det\_Dodgson(A(1:m, 2:n-1))$;</td><td colspan="2">$T_{5-3}(m,n) = T_{5-3}(m,n-1)+1$,
$T_{5-3}(m,n-1) = T_{5-3}(m,n-1-1)+1 = T_{5-3}(m,n-2)+1$,
$T_{5-3}(m,n) = T_{5-3}(m,n-2)+1+1 = T_{5-3}(m,n-2)+2$,
for any k, we have:
$T_{5-3}(m,n) = T_{5-3}(m,n-k)+k$,
for $n-k=m+1 \Rightarrow k=n-m-1$,
$T_{5-3}(m,n) = T_{5-3}(m,n-n+m+1)+n-m-1$
$= T_{5-3}(m,m+1)+n-m-1$.
Based on first condition:
$T_{5-3}(m,m+1) = C\binom{m+1}{m}\cdot m^3 = (m+1)\cdot m^3 = m^4+m^3$.
$T_{5-3}(m,n) = m^4+m^3+n-m-1 \approx m^4$.</td></tr>
</table>

	$d7 = det\_Dodgson(A(2:m-1,2:n-1));$	$T_{5-4}(m,n) = T_{5-4}(m-2,n-2)+1,$ $T_{5-4}(m-2,n-2) = T_{5-4}(m-2-2,n-2-2)+1$ $= T_{5-4}(m-4,n-4)+1$ $T_{5-4}(m,n) = T_{5-4}(m-4,n-4)+1+1 = T_{5-4}(m-4,n-4)+2,$ for any k, we have: $T_{5-4}(m,n) = T_{5-4}(m-k,n-k) = \frac{k}{2},$ for $m-k=2 \Rightarrow k = m-2,$ $T_(m,n) = T_{5-4}(2,n-m+2)+\frac{m}{2}-2.$ Based on first condition: $T_{5-4}(2,n-m+2) = C\binom{n-m+2}{2}\cdot 2^3 = \frac{(n-m+1)\cdot(n-m+1)}{2}\cdot 2^3$ $= 4\cdot(n-m+2)\cdot(n-m+1).$ $T_{5-4}(m,n) = 4\cdot(n-m+2)\cdot(n-m+1)+\frac{m}{2}-2$ $\approx 4\cdot(n-m+2)\cdot(n-m+1).$	
$T_5 = T_{5-1}+T_{5-2}+T_{5-3}+T_{5-4} = 4^{m-1}\cdot(n-m+2)\cdot(n-m+1)+4\cdot(n-m+2)\cdot(n-m+1)+m^4+4$ $\cdot(n-m+2)\cdot(n-m+1 \approx 4^{m-1}\cdot(n-m+2)\cdot(n-m+1).$			1
$d = (d1*d4-d2*d3+d5*d6)/d7;$		$T_6 = const_6$	1

Based on Table 2, we have:

$$Total\_Cost = 1\cdot T_1 + Max(1\cdot T_2, 1\cdot T_3, 1\cdot T_4, 1\cdot T_5) + 1\cdot T_6$$
$$= 1\cdot Const_1 + Max(1\cdot n^3, 1\cdot n^3, 1\cdot n^3, 1\cdot 4^{m-1}$$
$$\cdot(n-m+2)\cdot(n-m+1)) + 1\cdot Const_6.$$

Hence, the highest order is $4^{m-1}\cdot(n-m+2)\cdot(n-m+1)$. After eliminating constants and other lower grades, we can summarize the asymptotic time complexity as $O(2^2m\cdot(n-m)^2)$.

Time complexity analysis of function (det_Blocks) of algorithm P 3, based on modified Dodgson's generalized method, is presented in Table 3.

Table 3: Time complexity of *det_Blocks* function

Function: *det_Blocks*		**Cost**	**Times**
[m,n] = size(A);		$T_1 = const_1$	1
if m==n d=det(A);		$T_2 = n^3$	1
if m==n-1 $d = det\_Ones(A);$		Based on Algorithm 2.2 (See [11]), transforms determinant of order $(n-1)\times n$ to $n\times n$ by adding one row of elements equal to 1. Square determinant's time complexity is $T_3 = O(n^3)$.	1
else if $m < 3$ $d = det\_A(A);$		As it is calculated the det_A time complexity is: $T_4(3,n) = C\binom{n}{3}\cdot 3^3 = \frac{n\cdot(n-1)\cdot(n-2)\cdot(n-3)!}{3!\cdot(n-3)!}\cdot 3^3$ $= n\cdot(n-1)\cdot(n-2)\cdot 4.5 \approx n^3.$	1
else	$d1 = det\_Blocks(A(1:m-1,1:n-1));$ $d2 = det\_Blocks(A(1:m-1,2:n]));$ $d3 = det\_Blocks(A(2:m,1:n-1));$ $d4 = det\_Blocks(A(2:m,2:n]));$	$T_{5-1}(m,n) = 4\cdot T_{5-1}(m-1,n-1)+1,$ $T_{5-1}(m-1,n-1) = 4\cdot T_{5-1}(m-1-1,n-1-1)+1$ $= 4\cdot T_{5-1}(m-2,n-2)+1,$ $T_{5-1}(m,n) = 4\cdot(4\cdot(T_{5-1}(m-2,n-2))+1+1$ $= 4^2\cdot T_{5-1}(m-2,n-2)+2$ for any k, we have: $T_{5-1}(m,n) = 4^k\cdot T_{5-1}(m-k,n-k)+k,$ for $m-k=2 \Rightarrow k=m-2,$ $T_{5-1}(m,n) = 4^{m-2}\cdot T_{5-1}(1,n-m+2)+m-2$ Based on the first condition: $T_{5-1}(2,n-m+2 = C\binom{n-m+2}{2}\cdot 2^3$ $\frac{(n-m+2)\cdot(n-m+1)}{2}\cdot 2^3 = 4\cdot(n-m+2)\cdot(n-m+1).$ $T_{5-1}(m,n) = 4^{m-2}\cdot 4\cdot(n-m+2)\cdot(n-m+1)+m-2$ $\approx 4_{m-1}\cdot(n-m+2)\cdot(n-m+1).$	

	$d5 = det\_Blocks(A(2:m-1,1:n));$	$T_{5-2}(m,n) = T5-2(m-1,n-1)+1,$ $T_{5-2}(m-1.n-1) = T_{5-2}(m-1-1,n-1-1)+1$ $= T_{5-2}(m-2,n-2)+1,$ $T_{5-2}(m,n) = T_{5-2}(m-2,n-2)+1+1 = T_{5-2}(m-2,n-2+2,$ for any k, we have: $T_{5-2}(m,n) = T_{5-2}(m-k,n-k)+k,$ for $m-k=2 \Rightarrow k=m-2,$ $T_{5-2}(m,n) = T_{5-2}(2,n-m-+2)+m-2.$ Based on first condition: $T_{5-2}(2,n-m+2) = C\binom{n-m+2}{2}\cdot 2^3 = \frac{(n-m+2)\cdot(n-m+1)}{2}\cdot 2^3$ $= 4\cdot(n-m+1)\cdot(n-m+1).$ $T_{5-2}(m,n) = 4\cdot(n-m+2)\cdot(n-m+1)+m-2$ $\approx 4\cdot(n-m+2)\cdot(n-m+1).$	
	$d6 = det\_Blocks(A(1:m,2:n-1));$	$T_{5-3}(m,n) = T_{5-3}(m,n-1)+1,$ $T_{5-3}(m,n-1) = T_{5-3}(m,n-1-1)+1 = T_{5-3}(m,n-2)+1,$ $T_{5-3}(m,n) = T_{5-3}(m,n-2)+1+1 = T_{5-3}(m,n-2)+2,$ for any k, we have: $T_{5-3}(m,n) = T_{5-3}(m,n-k)+k,$ for $n-k=m+1 \Rightarrow k=n-m-1,$ $T_{5-3}(m,n) = T_{5-3}(m,n-n+m+1)+n-m-1$ $= T_{5-3}(m,m+1)+n-m-1.$ Based on first condition: $T_{5-3}(m,m+1) = C\binom{m+1}{m}\cdot m^3 = (m+1)\cdot m^3 = m^4+m^3.$ $T_{5-3}(m,n) = m^4+m^3+n-m-1 \approx m^4.$	
	$d7 = det\_Blocks(A(2:m-1,2:n-1));$	$T_{5-4}(m,n) = T_{5-4}(m-2,n-2)+1,$ $T_{5-4}(m-2,n-2) = T_{5-4}(m-2-2,n-2-2)+1$ $= T_{5-4}(m-4,n-4)+1$ $T_{5-4}(m,n) = T_{5-4}(m-4,n-4)+1+1 = T_{5-4}(m-4,n-4)+2,$ for any k, we have: $T_{5-4}(m,n) = T_{5-4}(m-k,n-k) = \frac{k}{2},$ for $m-k=2 \Rightarrow k=m-2,$ $T_{(}m,n) = T_{5-4}(2,n-m+2)+\frac{m}{2}-2.$ Based on first condition: $T_{5-4}(2,n-m+2) = C\binom{n-m+2}{2}\cdot 2^3 = \frac{(n-m+1)\cdot(n-m+1)}{2}\cdot 2^3$ $= 4\cdot(n-m+2)\cdot(n-m+1).$ $T_{5-4}(m,n) = 4\cdot(n-m+2)\cdot(n-m+1)+\frac{m}{2}-2$ $\approx 4\cdot(n-m+2)\cdot(n-m+1).$	
$T_5 = T_{5-1}+T_{5-2}+T_{5-3}+T_{5-4} = 4^{m-1}\cdot(n-m+2)\cdot(n-m+1)+4\cdot(n-m+2)\cdot(n-m+1)+m^4+4$ $\cdot(n-m+2)\cdot(n-m+1 \approx 4^{m-1}\cdot(n-m+2)\cdot(n-m+1).$			1
$d = (d1*d4-d2*d3+d5*d6)/d7;$		$T_6 = const_6$	1

Based on Table 3, we have:

$$Total\_Cost = 1\cdot T_1 + Max(1\cdot T_2, 1\cdot T_3, 1\cdot T_4, 1\cdot T_5) + 1\cdot T_6$$

$$= 1\cdot Const_1 + Max(1\cdot n^3, 1\cdot n^3, 1\cdot n^3, 1\cdot 4^{m-1}\cdot(n-m+2)$$

$$\cdot(n-m+1)) + 1\cdot Const_6$$

.

Hence, the highest order is $4^{m-1}\cdot(n-m+2)\cdot(n-m+1)$. After eliminating constants and other lower grades, we can summarize the asymptotic time complexity as $O(2^2m\cdot(n-m)^2)$.

The time complexity similarly can be concluded for each 9 cases.

Calculation of asymptotic time complexity of algorithm P 4, which is used to identify the pivot block with highest number of zero elements is presented on Table 4.

Table 4: Time complexity of Most_Zero_Elements_Block function

Function: *Most_Zero_Elements_Block*	**Cost**	**Time**
[m,n] = size(A);	$T_1 = const_1$	1
for $i = 1 : m$ for $j = 1 : n$ if $A(i, j) \sim= 0$ $B(i) = B(i) + 1;$ $C(j) = C(j) + 1;$	$T_2(m,n) = m \cdot n$ Due to nested loop.	1
if $C(1) + C(2) < C(n-1) + C(n)$ $k = 1;$	$T_3 = const_3$	1
elseif $C(2) + C(n-1) < C(1) + C(n)$ $k = 2;$	$T_4 = const_4$	1
else $k = 3;$	$T_5 = const_5$	1
if $B(2) + B(m-1) < B(1) + B(m)$ $k = k + 3;$	$T_6 = const_6$	1
elseif $B(1) + B(2) > B(m-1) + B(m)$ $k = k + 6;$	$T_7 = const_7$	1

Based on Table 4, we have:

$$Total\_Cost = 1 \cdot T_1 + 1 \cdot T_2 + Max(1 \cdot T_3, 1 \cdot T_4, 1 \cdot T_5) + Max(1 \cdot T_6, 1 \cdot T_7) = 1 \cdot Const_1 + 1 \cdot m \cdot n + Max(1 \cdot Const_3, 1 \cdot Const_4, 1 \cdot Const_5) + Max(1 \cdot Const_6 + 1 \cdot Const_7).$$

After eliminating constants, we get the asymptotic time complexity of algorithm P 4 as $O(m \cdot n)$.

The analysis of the growth of time complexity graphically is presented on following graph for cases: number of columns from 50 to 54 and number of rows from 3 to 28.

As can be seen from Figure 1, the break point is on about half of number of columns.

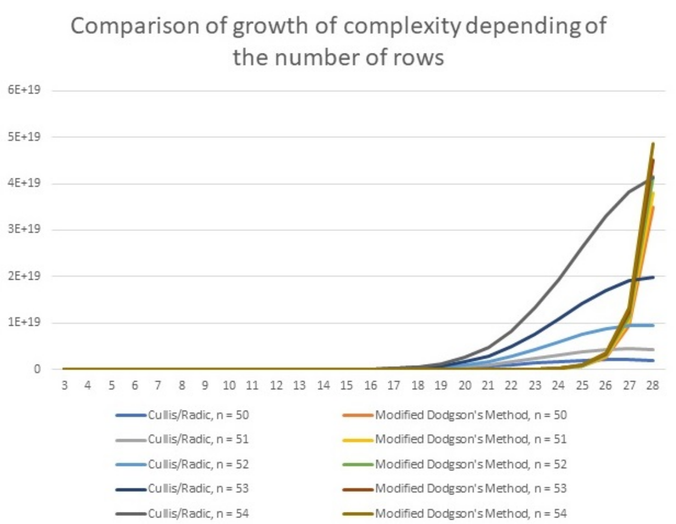

Figure 1: Comparison of growth of complexity depending on the number of rows, $50 \leq n \leq 54$, and $3 \leq m \leq 28$

Based on the analysis we can note that the Cullis/Radic definition (Algorithm P 1) is more efficient than the Dodgson's method (Algorithms P 2 and P 3) if the number of rows is higher than the half of number of columns, and in cases where the number of rows is lower or equal to half of number of columns, then the Dodgson's modified method is more efficient. Hence, we propose an algorithm which is a combination of both algorithms.

P 3: Modified algorithm (*det_Blocks*) based on theorem 2 (as example is considered case 1)

Step 1: Checking for conditions:

if $m = n$

Calculate square determinant with known methods.

else if $m = n - 1$

Transform determinant to square determinant, by adding one row with elements equal to 1.

$$d = det\_Ones(A);$$

else if $m < 3$ or $m = n/2$

Step 2: Identify all square determinants from the combination of columns:

Create loop from 1 to total number of combinations

$$D\{i\} = A(1 : m, B(i, 1 : m)));$$

Step 3: Calculate determinants of square blocks from D

Create Loop from 1 to total number of combinations

$$d = d + (-1)^\wedge(sum(1 : m) + sum(B(i, 1 : m))) * SquareDet(D\{i\});$$

else

Step 4: Calculate submatrices:

Calculate submatrices presented on theorem 1, calling *det_Comb* algorithm until the conditions of step 1 are met:

$$d1 = det\_Comb(A(1 : m-1, 1 : n-1));$$
$$d2 = det\_Comb(A(A(1 : m-1, 2 : n));$$

$$d3 = det\_Comb(A(2:m, 1:n-1));$$
$$d4 = det\_Comb(A(2:m, 2:n));$$
$$d5 = det\_Comb(A(2:m-1, 1:n));$$
$$d6 = det\_Comb(A(1:m, 2:n-1));$$
$$d7 = det\_Comb(A(2:m-1, 2:n-1));$$

Step 5: Calculate the result of the determinant as following:

$$d = (d1*d4 - d2*d3 + d5*d6)/d7;$$

Step 6: Display the result of the determinant

Note: The algorithm P 5 can also be combined with algorithm P 3, with changes only in step 4, where in cases of several elements of original matrix equal to zero can be more efficient.

The worst-case time complexity of the above presented algorithm is where the number of rows is half the number of columns.

The asymptotic time complexity of the algorithm presented in P 5, is calculated in Table 5, where we have identified the worst-case and best-case time complexity as follows.

Table 5: Time complexity analysis of (det_Comb) function

Function: det_Comb		**Cost**	**Time**
$[m,n] = size(A);$		$T_1 = const_1$	1
if $m == n$ $d = det(A);$		$T_2 = n^3$	1
if $m == n-1$ $d = det\_Ones(A);$		Based on Algorithm 2.2 (See [11]), transforms determinant of order $(n-1)\times n$ to $n\times n$ by adding one row of elements equal to 1. Square determinant's time complexity is: $T_3 = O(n^3)$.	1
else if $m < 3$ $d = det\_A(A);$		As it is calculated the det_A time complexity is: $T_4(3,n) = C\binom{n}{3}\cdot 3^3 = \frac{n\cdot(n-1)\cdot(n-2)\cdot(n-3)!}{3!\cdot(n-3)!}\cdot 3^3 = n\cdot(n-1)\cdot(n-2)\cdot 4.5 \approx n^3.$	1
else if	$B = nchoosek(1:n, n/2);$	$T_5 = const_5$	1
	for $i = 1 : length(B)$ $d = d + (-1)\hat{}(sum(1:(n/2)) + sum(B(i,[1:(n/2)])))$ $*det((A([1:(n/2)],[B(i,[1:(n/2)])])));$	There are several methods to calculate square determinants with different time complexity, however we will be based on LU factorization method [16]: $T_6 = (\frac{n}{2})^3$	$C\binom{n}{n/2}$
else	$d1 = det\_Comb(A(1:n/2-1, 1:n-1));$ $d2 = det\_Comb(A(1:n/2-1, 2:n]));$ $d3 = det\_Comb(A(2:n/2, 1:n-1));$ $d4 = det\_Comb(A(2:n/2, 2:n]));$	$T_{7-1}(n/2,n) = 4\cdot T_{7-1}(n/2-1,n-1)+1,$ $T_{7-1}(n/2-1,n-1) = 4\cdot T_{7-1}(n/2-1-1,n-1-1)+1$ $= 4\cdot T_{7-1}(n/2-2,n-2)+1,$ $T_{7-1}(n/2,n) = 4\cdot(4\cdot T_{7-1}(n/2-2,n-2))+1+1$ $= 4^2\cdot T_{7-1}(n/2-2.n-2)+2,$ for any k, we have: $T_{7-1}(n/2,n) = 4^k\cdot T_{7-1}(n/2-k,n-k)+k,$ for $n/2-k=2 \Rightarrow k = n/2-2,$ $T_{7-1}(n/2,n) = 4^{n/2-2}\cdot T_{7-1}(2,n-n/2+2)+n/2-2$ $= 4^{n/2-2}\cdot T_{7-1}(2,n/2+2)+n/2-2.$ Based on first condition: $T_{7-1}(2,n/2+2) = C\binom{n/2+2}{2}\cdot 2^3 = \frac{(n/2+2)\cdot(m/2+1)}{2}\cdot 2^3$ $= 4\cdot(n/2+2)\cdot(n/2+1).$ $T_{7-1}(n/2,n) = 4^{n/2-2}\cdot 4\cdot(n/2+2)\cdot(n/2+1)+n/2-2$ $\approx 4^{n/2-1}\cdot(n/2+2)\cdot(n/2+1).$	

<table>
<tr>
<td></td>
<td>$d1=det\_Comb(A(1:n/2-1,1:n-1));$
$d2=det\_Comb(A(1:n/2-1,2:n]));$
$d3=det\_Comb(A(2:n/2,1:n-1));$
$d4=det\_Comb(A(2:n/2,2:n]));$</td>
<td colspan="2">$T_{7-1}(n/2,n)=4\cdot T_{7-1}(n/2-1,n-1)+1,$
$T_{7-1}(n/2-1,n-1)=4\cdot T_{7-1}(n/2-1-1,n-1-1)+1$
$=4\cdot T_{7-1}(n/2-2,n-2)+1,$
$T_{7-1}(n/2,n)=4\cdot(4\cdot T_{7-1}(n/2-2,n-2))+1+1$
$=4^2\cdot T_{7-1}(n/2-2.n-2)+2,$
for any k, we have:
$T_{7-1}(n/2,n)=4^k\cdot T_{7-1}(n/2-k,n-k)+k,$
for $n/2-k=2\Rightarrow k=n/2-2,$
$T_{7-1}(n/2,n)=4^{n/2-2}\cdot T_{7-1}(2,n-n/2+2)+n/2-2$
$=4^{n/2-2}\cdot T_{7-1}(2,n/2+2)+n/2-2.$
Based on first condition:
$T_{7-1}(2,n/2+2)=C\binom{n/2+2}{2}\cdot 2^3=\frac{(n/2+2)\cdot(m/2+1)}{2}\cdot 2^3$
$=4\cdot(n/2+2)\cdot(n/2+1).$
$T_{7-1}(n/2,n)=4^{n/2-2}\cdot 4\cdot(n/2+2)\cdot(n/2+1)+n/2-2$
$\approx 4^{n/2-1}\cdot(n/2+2)\cdot(n/2+1).$</td>
</tr>
<tr>
<td></td>
<td>$d5=det\_Comb(A(2:n/2-1,1:n));$</td>
<td colspan="2">$T_{7-2}(n/2,n)=T_{7-2}(n/2-1,n)+1,$
$T_{7-2}(n/2-1,n)=T_{7-2}(n/2-1-1,n)+1=T_{7-2}n/2-2,n)+1,$
$T_{7-2}(n/2,n)=T_{7-2}(n/2-2,n)+1+1=T_{7-2}(n/2-2,n)+2,$
for any k, we have:
$T_{7-2}(n/2,n)=t_{7-2}(2,n/2-2)+n/2-2.$
Based on first condition:
$T_{7-2}(2,n/2-2)=C\binom{n/2-2}{2}\cdot 2^3=\frac{(n/2-2)\cdot(n/2-3)}{2}\cdot 2^3$
$=4\cdot(n/2-2)\cdot(n/2-3).$
$T_{7-2}(n/2,n)=4\cdot(n/2-2)\cdot(n/2-3)+n/2-2$
$\approx 4\cdot(n/2-2)\cdot(n/2-3).$</td>
</tr>
<tr>
<td></td>
<td>$d6=det\_Comb(A(1:n/2,2:n-1));$</td>
<td colspan="2">$T_{7-3}(n/2,n)=T_{7-3}(n/2,n-1)+1,$
$T_{7-3}(n/2,n-1)=T_{7-3}(n/2,n-1-1)+1=T_{7-3}(n/2,n-2)+1,$
$T_{7-3}(n/2,n)=T_{7-3}(n/2,n-2+1+1=T_{7-3}(n/2,n-2)+2,$
for any k, we have:
$T_{7-3}(n/2,n)=T_{7-3}(n/2,n-k)+k,$
for $n-k=n/2+1\Rightarrow k=n-n/2-1=n/2-1,$
$T_{7-3}(n/2,n)=T_{7-3}(n/2,n-n/2+1)+n/2-1$
$=T_{7-3}(n/2,n/2+1)+n/2-1.$
Based on first condition:
$T_{7-3}(n/2,n/2+1)=C\binom{n/2+1}{n/2}\cdot(n/2)^3=(n/2+1)\cdot(n/2)^3$
$=(n/2)^4+(n/2)^3.$
$T_{7-3}(n/2,n)=(n/2)^4+(n/3)^3+n/2-1\approx(n/2)^4.$</td>
</tr>
<tr>
<td></td>
<td>$d7=det\_Comb(A(2:n/2-1,2:n-1));$</td>
<td colspan="2">$T_{7-4}(n/2,n)=T_{7-4}(n/2-2,n-2)+1,$
$T_{7-4}(n/2-2,n-2)=T_{7-4}(n/2-2-2,n-2-2)+1$
$=T_{7-4}(n/2-4,n-4)+1,$
$T_{7-4}(n/2,n)=T_{7-4}(n/2-4,n-4)+2,$
for any k, we have:
$T_{7-4}(n/2,n)=T_{7-4}(n/2-k,n-k)+k/2,$
for $n/2-k=2\Rightarrow k=n/2-2,$
$T_{7-4}(n/2,n)=T_{7-4}(2,n/2+2)n/4-2.$
Based on first condition:
$T_{7-4}(2,n/2+2)=C\binom{n/2+2}{2}\cdot 2^3=\frac{(n/2+2)\cdot(n/2+1)}{2}\cdot 2^3$
$=4\cdot(n/2+2)\cdot(n/2+2).$
$T_{7-4}(n/2,n)=4\cdot(n/2+2)\cdot(n/2+1)+n/4-2$
$\approx 4\cdot(n/2+2)\cdot(n/2+1).$</td>
</tr>
<tr>
<td colspan="3">$T_7=T_{7-1}+T_{7-2}+T_{7-3}+T_{7-4}=4^{n/2-1}\cdot(n/2+2)\cdot(n/2+1)+$
$4\cdot(n/2-2)\cdot(n/2-3)+(n/2)^4+4\cdot(n/2+2)\cdot(n/2+1)\approx 4^{n/2-1}\cdot(n/2+2)\cdot(n/2+1).$</td>
<td>1</td>
</tr>
<tr>
<td colspan="2">$d=(d1*d4-d2*d3+d5*d6)/d7;$</td>
<td>$T_8=const_8$</td>
<td>1</td>
</tr>
</table>

Based on Table 5, we have:

$$Total\_Cost = 1 \cdot T_1 + Max(1 \cdot T_2, 1 \cdot T_3, 1 \cdot T_4, 1 \cdot T_5, C\binom{n}{n/2}$$
$$\cdot T_6, 1 \cdot T_7) + 1 \cdot T_8 = 1 \cdot Const_1 + Max(1 \cdot n^3, 1 \cdot n^3, 1 \cdot n^3,$$
$$1 \cdot Const_5 + C\binom{n}{n/2} \cdot (\frac{n}{2})^3, 1 \cdot 4^{n/2-1} \cdot (n/2+2) \cdot (n/2+1)) + 1)$$
$$+1 \cdot Const_8$$

.

Hence, the highest order is $C\binom{n}{n/2} \cdot (\frac{n}{2})^3$. After eliminating constants and other lower grades, we can summarize the worst-case asymptotic time complexity as $O(\frac{n!}{((n/2)!)^2} \cdot (n/2)^3)$.

While the best-case is $O(n^3)$, for $m = 3$, calculated as follows:

For Cullis/Radic we have:

$$Total\_Cost = 1 \cdot T_1 + 1 \cdot T_2 + 1 \cdot T_3 + Max(1 \cdot T_4, C\binom{n}{n/2} \cdot T_5)$$
$$= 1 \cdot Const_1 + 1 \cdot Const_2 + 1 \cdot Const_3 + Max(1 \cdot n^3, C\binom{n}{3} \cdot 3^3)$$

.

While,

$$Max(1 \cdot n^3, C\binom{n}{3} \cdot 3^3) = Max(1 \cdot n^3, \frac{n!}{3! \cdot (n-3)!} \cdot 3^3)$$
$$= Max(1 \cdot n^3, \frac{n \cdot (n-1) \cdot (n-2) \cdot (n-3)!}{3! \cdot (n-3)!} \cdot 3^3)$$

Since the n^3 is the highest order, the asymptotic time complexity is $O(n^3)$.

For generalized/modifed Dodgson's method, we have:

$$Total\_Cost = 1 \cdot T_1 + Max(1 \cdot T_2, 1 \cdot T_3, 1 \cdot T_4, 1 \cdot T_5) + 1 \cdot T_6$$
$$= 1 \cdot Const_1 + Max(1 \cdot n^3, 1 \cdot n^3, 1 \cdot n^3, 1 \cdot 4^{3-1} \cdot (n-3+2)$$
$$\cdot (n-3+1)) + 1 \cdot Const_6$$

.

Also, in this case since the n^3 is the highest order, the asymptotic time complexity is $O(n^3)$.

3 Conclusions

In this paper we have analyzed the asymptotic time complexity of algorithms based on Cullis/Radic definition and generalized/modified Dodgson's Condensation method/s for rectangular determinant calculations. From the calculations we noted that the asymptotic time complexity for Cullis/Radic definition is $O(C\binom{n}{m} \cdot m^3)$, while for the generalized/modified Dodgson's Condensation method/s the asymptotic time complexity is $O(2^2 m \cdot (n-m)^2)$.

Further we have analyzed which complexity grows faster and tested for rectangular determinant of order for $50 \leq n \leq 54$, and $3 \leq m \leq 28$, and from analysis it is noted that the break point is on about half of number of columns compared to number of rows. In cases where the number of columns is less than the half of the number of rows, then the Dodgson's Condensation method/s are growing slower than the Cullis/Radic definition, otherwise the Cullis/Radic definition is growing slower. From this analysis we have proposed a combined algorithm where it calculates determinants with Cullis/Radic definition in cases where the number of columns is higher than the half of number of rows and calculates determinants with Dodgson's Condensation method/s in cases where the number of columns is lower than the half of number of rows.

From where we calculated the worst-case asymptotic time complexity as $O(\frac{n!}{((n/2)!)^2} \cdot (n/2)^3)$, while the best-case asymptotic time complexity is when the $m = 3$, and it is calculated as $O(n^3)$.

References

[1] A. Amiri, M. Fathy, and M. Bayat. "Generalization of Some Determinantial Identities for Non-Square Matrices Based on Radic's Definition." *TWMS Journal of Pure and Applied Mathematics*, 1(2):163-175, 2010.

[2] M. Bayat, "A Bijective Proof of Generalized Cauchy–Binet, Laplace, Sylvester and Dodgson Formulas." *Linear and Multilinear Algebra*, 2020.

[3] T. H. Cormen, C. E. Leiserson, R. L. Rivest, R. L., and C. Stein, *Introduction to Algorithms 4th Edition*. The MIT Press Cambridge, Massachuestts London, England, 2022.

[4] C. E. Cullis, *Matrices and Determinoids*. Cambridge: University Press, 1913.

[5] A. Makarewicz and P. Pikuta, "Cullis–Radic Determinant of a Rectangular Matrix Which Has a Number of Identical Columns." *Annales Universitatis Mariae Curie-Sklodowska*, 74(2):41-60, 2020.

[6] A. Makarewicz, P. Pikuta, and D. Szalkowski, "Properties of the Ddeterminant of a Rectangular Matrix." *Annales Universitatis Mariae Curie-Sklodowska*, 68(1):31-41, 2014.

[7] R. E. Neaplitan, *Foundations of Algorithms*. Cambridge: Jones and Bartlett Learning, 2015.

[8] M. Radic, "Definition of Determinant of Rectangular Matrix." *Glasnik Matematicki*, pp. 17-22, 1966.

[9] K. H. Rosen, *Discrete Mathematics and Its Applications 8th Edition*. New York: McGraw-Hill Education, 2019.

[10] A. Salihu, H. Snopce, J. Ajdari, and A. Luma, "Generalization of Dodgson's Condensation Method for Calculating Determinant of Rectangular Matrices." *2nd IEEE International Conference on Electrical, Computer and Energy Technologies (ICECET)*, Prague, 2022.

[11] A. Salihu and F. Marevci, ”Chio’s-like Method for Calculating the Rectangular (Non-Square) Determinants: Computer Algorithm Interpretation and Comparison.” *European Journal of Pure and Applied Mathematics*, 14(2):431-450, 2021.

[12] S. S. Skiena, *The Algorithm Design Manual*. London: Springer-Verlag London Limited, 2008.

[13] P. Stanimirovic and M. Stankovic, ”Determinants of Rectangular Matrices and the Moore-Penrose Inverse.” *Novi Sad J Math.*, 27(1):53-69, 1997.

[14] M. Stojakovic, ”Determinante Nekvadratnih Matrica.” *Vesnik DMNRS*, 1952.

[15] A. P. Sudhir, ”Generalisations of the Determinant to Interdimensional Transformations: A Review.” *arXiv:1904.08097v1*, 2019

[16] W. Xingbo and X. Yaoqi, ”How Difficult To Compute Coefficients of Characteristic Polynomial?” *International Journal of Research Studies in Computer Science and Engineering (IJRSCSE),*, 3(1):7-12, 2016.

Armend Salihu is a PhD Candidate at the South East European University, Faculty of Contemporary Sciences. In 2009 he has received bachelor degree at the University of Prishtina, Faculty of Electrical and Computer Engineering, while in 2017 he received master degree in the field of computer Science from the University for Business and Technology, Faculty of Computer Sciences and Engineering. Currently he is engaged as teaching assistant at the University of Prishtina, Faculty of Mathematics and Natural Sciences.

Artan Luma is a full professor at the South East European University. He has been graduated as Engineer of Informatics in Computer Science at the State University of Tetovo in 1997. He received Master of Electrical Engineering Science at the University of Prishtina, Faculty of Electrical and Computer Engineering in 2007. In 2011 he received PhD in Computer Science at the South East European University, Faculty of Contemporary Sciences and Technologies. His speciality is cryptography.

Halil Snopce is a full professor at the South East European University. He has been graduated in Mathematics in 1997 at the University of ”St. Cyril and Methodius”, Faculty of Natural and Mathematical Sciences. In 2007 he received master degree in Mathematics at the University of Tirana, Faculty of Natural and Mathematical Sciences, with the main topics in numerical analysis. In 2011 he has received his PhD in Computer Science and Applied Mathematics in the CST-Faculty at the South East European University. His speciality is parallel processing.

Jaumin Ajdari is a full professor at the South East European University. He has been graduated as Engineer in Applied Mathematics, as well as Master in Mathematics at the University of Zagreb, Faculty of Natural Science and Mathematics. He also received Master of Science in Mathematics at the University of Tirana, Faculty of Natural Sciences, Department of Numerical Mathematic and Parallel processing in 2006. In 2011 he has received his PhD in Mathematical Sciences at the University of Tirana, Faculty of Natural Sciences. His speciality is parallel processing.