Quantum Algorithms for Colouring of Graphs

Muskan Saha,[1,*] Bikash K. Behera,[2,†] and Prasanta K. Panigrahi[2,‡]

[1] *Department of Physical Sciences, Indian Institute of Science Education and Research Bhopal, Indore Bypass road, Bhopal 462066, Madhya Pradesh, India*

[2] *Department of Physical Sciences, Indian Institute of Science Education and Research Kolkata, Mohanpur 741246, West Bengal, India*

Colouring of graphs is one of the most important concepts in graph theory and has a number of practical applications in the field of computer science such as making schedule, colouring of maps, networks, etc. Though there are approaches to find whether a graph is colorable or not, however till now, there have been no quantum algorithms checking a graph's colorabilty along with colouring the graph. Here, in the present work, we propose a new quantum algorithm using the principle of quantum entanglement for 2- and 3-vertex colouring. With basic quantum gates, we design quantum circuits that first check whether a given graph is 2 or 3 colorable and then accordingly perform the necessary operations to colour it.

Keywords: IBM Quantum Experience, Graph Colouring

I. INTRODUCTION

Quantum resources and quantum algorithms are well known for fast information processing and handling bulky data easily, which is why it overpowers classical algorithms [1]. Most well known quantum algorithms e.g., Grover's [2] and Shor's algorithm [3] have been extensively used in variety of quantum information processing tasks. Quantum principles such as superposition and entanglement are used to tackle the today's most difficult (NP hard) problems e.g., factoring of large integers [4, 5], solving travelling salesman problem [6, 7], optimization problems [8], simulation of large and complex molecules [9] etc. Quantum algorithms are run on realistic model of quantum computers such as provided by IBM Quantum Experience platform (IBM QE) [10], D-wave quantum computers [11], trapped ion quantum computer (IonQ) [12] to name a few. IBM QE has recently gained popularity by providing the 5-qubit and 14-qubit quantum chips to the community and making easily accessible through QISKit. A number of quantum information processing tasks in the field of quantum simulation [13–16], quantum machine learning [17, 18], quantum error correction [19–22], quantum information theory [23–25], quantum cryptography [26], quantum algorithms [27, 28], quantum optimization problems [6], quantum games [29], designing quantum communication devices [30, 31] are tested on the real chips and feasible results are obtained.

In quantum computation, we generally denote the quantum state of a qubit by $|\psi\rangle$ using Dirac's bra-ket notation, which can be in superposition of both $|0\rangle$ and $|1\rangle$, $|\psi\rangle = a|0\rangle + b|1\rangle$. Here a and b are complex amplitudes and $|0\rangle$ and $|1\rangle$ are the basis states. Quantum states are always normalized, hence $|a|^2 + |b|^2 = 1$. With a n-qubit system, we have 2^n possible states. In quantum world, states of qubits can be entangled, known as quantum entanglement. It is the phenomena of linking together/ correlation of the states of two or more qubits even when they are spatially separated. This concept becomes very important when dealing with things that have a strong relation in some way or the other. For example, in vertex colouring [32], this relation is the existence of a common edge between two vertices, where, two vertices can be given the same colour if and only if they are not adjacent (i.e. they do not share the same edge). In quantum computing, ancilla qubits are used to extract information from entangled states that enable to perform useful tasks. A quantum state collapses to a single state as soon as measurement is done and is no longer in superposed state [33]. Various operations can be performed using quantum gates (which are Pauli matrices) namely, i) X gate: transforms $|0\rangle$ to $|1\rangle$ and vice-versa, ii) Y gate: transforms $|0\rangle$ to $\imath|1\rangle$ and $|1\rangle$ to -$\imath|0\rangle$ iii) Z-gate: it is a phase flip gate. iv) Hadamard gate: it creates superposition states, v) CNOT gate: performs not operation on second qubit only if first qubit is in $|1\rangle$ state, vi) anti-CNOT gate: performs not operation on second qubit only if the first qubit is in $|0\rangle$ state. These are some of the basic quantum gates that we use in quantum computation.

Colouring of graph primarily means colouring of vertices of the graph. Vertex colouring is defined as labelling the vertices with colours such that no two adjacent vertices (having an edge whose end points are these vertices) get the same colour. While colouring, we aim at using minimum colours as possible. The problem is NP hard but still we have several classical algorithms for the same, namely greedy colouring algorithm [47], backtracking method [34, 35], etc. After thoroughly understanding existing classical algorithms, we approach the problem to solve and propose quantum algorithms for the same. 2-colouring means colouring of the graph with two colours such that no two adjacent vertices get the

* muskan17@iiserb.ac.in
† bkb18rs025@iiserkol.ac.in
‡ pprasanta@iiserkol.ac.in

same colour and similarly, 3-colouring means colouring of graph with three colours such that no two adjacent vertices are assigned the same colour. D'Hont [36] proposed an approach to check whether a graph is 2-colorable or 3-colorable, however, there was no quantum algorithm till date, to color a graph after checking the graph's colorability. Our proposed algorithm, checks the colorability of a graph whether it is 2- or 3-colorable, then colors it accordingly. Here, we provide some of the examples of graphs which are 2- and 3-colorable and can be colored using our algorithm.

II. GRAPH THEORY

In graph theory, a graph G is a triple consisting of a vertex set V(G), an edge set E(G), and a relation that associates with each edge two vertices (not necessarily distinct) called its endpoints. If there is an edge connecting two vertices, the vertices are said to be adjacent to each other. A loop is an edge whose endpoints are the same. Multiple edges also exist that connect two vertices more than once. The degree of a vertex is defined as the number of edges incident on the vertex. δ denotes minimum degree in a graph and Δ denotes maximum degree in a graph. The vertices of a graph can be connected [37] or independent [38]. A graph is said to be connected if there is an edge from each vertex to every other vertex. In a disconnected graph, we have several connected components. A graph can be directed as well, where each edge shows directed relation between the vertices. While coloring we ignore the direction, i.e. we consider underlying graph of a directed graph.

A k-coloring [39, 40] of a graph G is a labeling f : V(G) $\rightarrow$ S, where$|S|$ = k. The labels are colors. A k-coloring is proper if adjacent vertices have different colors. The chromatic number $\chi(G)$ is the least k such that G is k-colorable. While colouring, we ignore loops i.e we don't consider graph with loops. A planar graph is a graph that can be drawn on a plane with no edges crossing. Such a representation is a planar embedding of G. A particular planar embedding of a planar graph is called a plane graph. Faces of a plane graph are the regions surrounded by the edges. For a planar graph, we have Eulers formula for checking whether the graph is planar or not, i.e. n - e + f = 2, where n denotes number of vertices, e denotes number of edges and f denotes number of faces. For a disconnected graph, we have n - e + f = 1 + C(G), where n, e, and f have their usual meaning and C(G) denotes the number of components in the graph G. We have a theorem relating chromatic number and maximum degree of a graph, $\chi(G)$ = Δ(G)+1. The dual graph G of a plane graph G is a plane graph whose vertices correspond to the faces of G. The edges of G correspond to the edges of G as follows: if e is an edge of G with face X on one side and face Y on the other side, then the endpoints of the dual edge e $\in$ E(G) are the vertices x, y of G that represent the faces X, Y of G. Dual of a plane graph is always a plane graph and has same number of edges as in the original graph. Colouring of a graph has many applications, like for setting up time-table [41], colouring of maps, pattern matching [42], scheduling [43], designing seating plans [44], solving Sudoku [29, 45] etc.

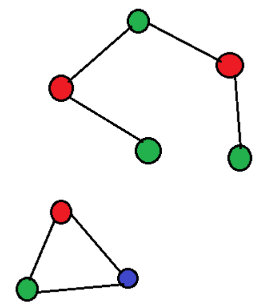

FIG. 1. **Graph Coloring.** Here we have a graph with two components. One of them is 2-colorable and other is 3-colorable .

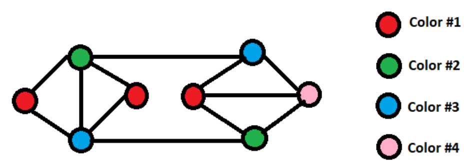

FIG. 2. **Greedy coloring approach.** In greedy coloring approach, colors are numbered in increasing order of preference and each time a new vertex is reached, it is coloured with color of the lowest number such that the constraints are satisfied.

III. CLASSICAL ALGORITHMS FOR COLOURING

A. Greedy colouring algorithm

The greedy coloring [46] relative to a vertex ordering $v_1, v_2, ..., v_n$ of V(G) is obtained by coloring vertices in the order $v_1, v_2, ..., v_n$, assigning to v_i the smallest-indexed color not already used on its lower-indexed neighbouring vertices. Proposition: If a graph G has degree sequence $d_1 > d_2 > ...>d_n$, then $\chi(G)<1 + max_i min(d_i, i-1)$.

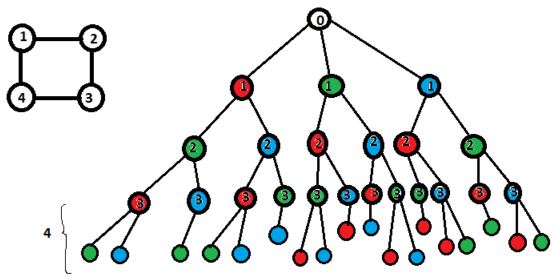

FIG. 3. **Coloring using backtracking.** A tree is obtained showing different possible coloring of a graph

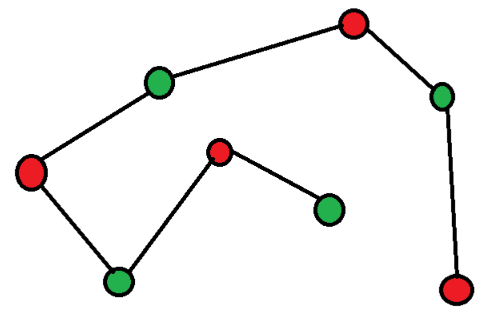

FIG. 4. **2-coloring of linear graph.** Alternate colors are given to non-adjacent vertices.

B. Backtracking k-coloring algorithm

While coloring of graph, there are several ways in which colors can be assigned to vertices of the graph. The different ways can be jotted down by backtracking, where a tree is formed as we proceed further for coloring vertices. In backtracking, we give a color to an initial vertex, then move to the next and color it according to constraint. In this manner, a sequence of colors is obtained, then from the last vertex (altering its any color with any other possible color that can be assigned), we again move back to the top root vertex and obtain another sequence. This is repeated at each node except root node and for each possible color of the node. Several other algorithms also exist for graph coloring [48, 49] that are preferable for some or the other graph. Existing well known Grover's [2] and Shor's algorithm [3] have also been modified and used for the same.

IV. COLORING OF VERTEX USING QUANTUM GATES

A. 2-coloring algorithm

2-coloring of a graph G means coloring of a graph with exactly two colors such that no two adjacent vertices get the same color. We show a linear arrangement and coloring of a grid (Fig. 5) with two colors. Trees are also 2-colorable (since they are bipartite). Here in Fig. 5, we begin coloring from any random vertex. The quantum circuit for coloring the above type of graphs is illustrated in Fig. 6. It can be observed that, initially all the qubits are in $|0\rangle$ state. Then a series of anti-controlled not gates have been applied to all the qubits. If the first qubit is in $|0\rangle$ state, then the first anti-controlled not gate will be applied, hence making the second qubit (assigning the second vertex a different color) in $|1\rangle$ state. Similarly, a series of $|01010101...\rangle$ state will be prepared after the execution of the whole quantum circuit, consequently coloring the whole vertices with alternative colors. The same quantum circuit will work if the initial state of the first qubit is changed to $|1\rangle$ state. Then a sequence of $|10101010...\rangle$ state can be obtained depending upon the number of vertices of the graph. It is to be noted that the circuit can be applied only to the graphs where all the vertices must be connected and 2-coloring should be applicable. A linear graph (Fig. 4), a cycle with even number of vertices, a 2D grid graph (Fig. 5) are the examples where the quantum circuit can be applied and 2-coloring of graphs can be achieved. A linear loop graph with odd number of vertices can be colored using the quantum algorithm given in the next Subsection IV B.

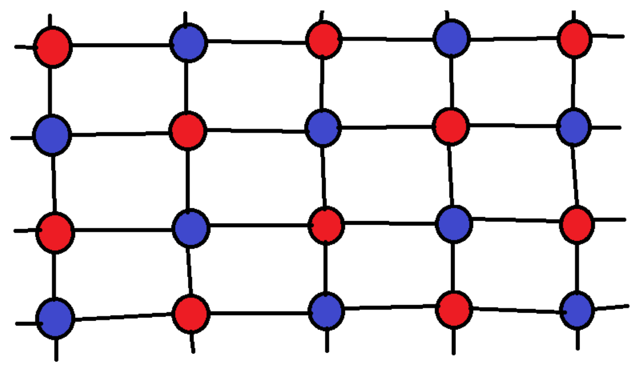

FIG. 5. **2-coloring of a 2D grid.** Alternate colors are given to non-adjacent vertices.

B. 3-coloring algorithm

3-coloring [50, 51] of a graph G is the coloring of graph with exactly three colors such that no two adjacent vertices are assigned the same color. Let us consider a graph with three vertices (Fig. 1) and as it can be seen from the figure, it is 3-colorable, we approach to color it using the quantum circuit depicted in Fig. 9. We take pairs of entangled states (entangled with either $\frac{|00\rangle+|11\rangle}{\sqrt{2}}$ or $\frac{|01\rangle+|10\rangle}{\sqrt{2}}$) representing the connection between the edges.

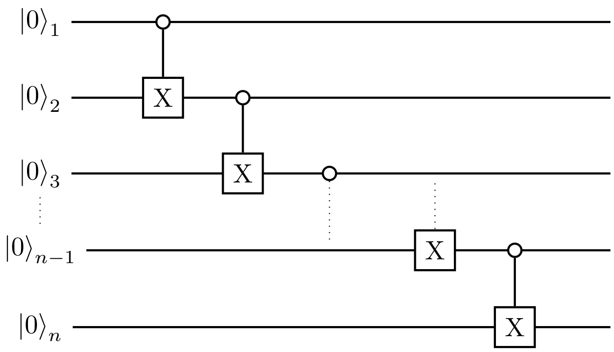

FIG. 6. **Quantum circuit for 2- coloring.** Here each qubit denote vertices, applying a gate is equivalent to applying color on the vertex, using control-not gate, we color vertices that do not have connection (i.e. do not share a common edge.

$\|\psi_{12}\rangle$	$\|\psi_{13}\rangle$	$\|\psi_{23}\rangle$	v_1	v_2	v_3
0	0	0	Red	Red	Red
0	0	1	Red	Red	Green
0	1	0	Red	Red	Green
0	1	1	Red	Red	Green
1	0	0	Red	Green	Red
1	0	1	Red	Green	Red
1	1	0	Green	Red	Red
1	1	1	Red	Green	Blue

TABLE I. Table representing the connection between the edges and colouring the vertices. The ancilla qubit states with corresponding edges are given and the vertices are colored accordingly.

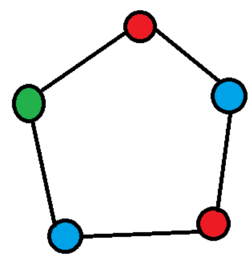

FIG. 7. **3- coloring of a cycle (odd cycle).** Any cycle with 2n+1 vertices is 3-colorable. Note: green vertex can neither be replaced by blue nor by red color. Hence, third color is needed.

For the above example, we consider three pair of entangled states $|\psi_{12}\rangle$, $|\psi_{13}\rangle$ and $|\psi_{23}\rangle$ to denote the edge connection between vertices 1 and 2, 1 and 3, and 2 and 3 respectively. It is to be noted that here we ignore loops and multiple edges in the graph. We take three ancilla qubits (initially prepared in the $|0\rangle$ state) to extract information about their (vertices) connection and store the information about entanglement (of qubits or vertices) in the ancilla qubits to use it for coloring purposes. Two controlled-not operations are then applied from the two-qubit state representing the edges to the ancilla qubits. After the operations, if the ancilla qubit is in $|0\rangle$ state, then the corresponding pair of vertices are entangled with $\frac{|00\rangle+|11\rangle}{\sqrt{2}}$ meaning there is no connection between the vertices. However, if the ancilla qubit is in $|1\rangle$ state, there is an edge between the corresponding vertices. As there are three ancilla qubits representing three possible edges (12,13,23), we have eight possible cases (if we consider labelled vertices) to first check the colorability of the graph and then make it color. The possible outcomes of the ancilla qubits with possible color combinations is presented in the Table I. According to the table, we can apply eight three-qubit controlled operations to color the graph, further, we can combine some of the inputs of the ancilla qubits as they give rise to the same color combination of the graph. For example, the input $|010\rangle$ and $|011\rangle$ have same color combination ($|001\rangle$ representing Red,Red,Green). Here instead of using three controlled operations on the three qubits (representing the three vertices), we can use two controlled operations for the ancilla qubits as their first two qubits' state is same. Following the same way, we have in total five controlled operations among which three are three-controlled and two are two-controlled operations on the three vertices to color them. We apply identity, NOT and Hadamard operations on the vertices to represent three different colors named as Red, Green and Blue. It can be mentioned that only for the last input $|111\rangle$, i.e., when all the vertices are connected, then 3-coloring graph is applicable. Finally, measurement is done on the three qubits and we can find the color assigned to the three vertices. We find that a graph with three vertices where all of the vertices are directly connected to each other is 3-colorable. The algorithm can be used to color some of the graphs given in Figs. 1, 7 and 8.

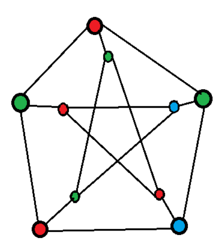

FIG. 8. **3-coloring of a graph**.

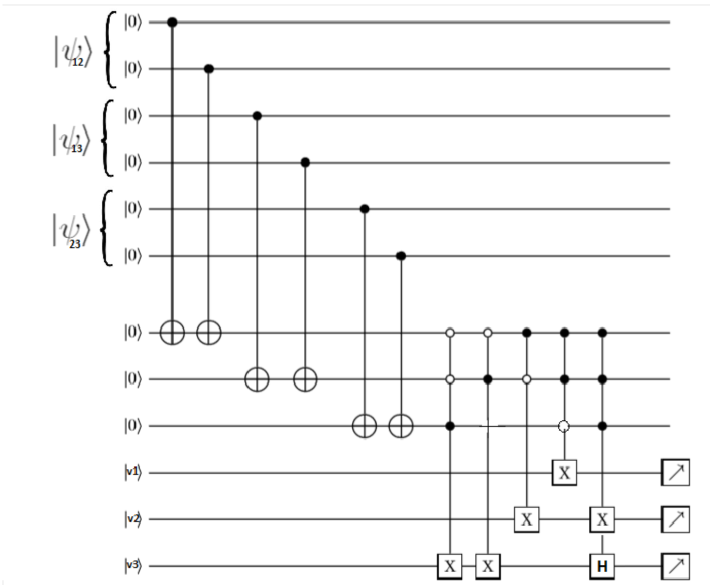

FIG. 9. **Quantum circuit for 3-coloring** Here each qubit denotes vertices (namely, 1, 2 and 3) edges joining these vertices are denoted by $|\psi_{12}\rangle$, $|\psi_{13}\rangle$ and $|\psi_{23}\rangle$ respectively. Information about connection between the vertices is stored in the following ancilla qubits. This information is used to color vertices namely, v_1, v_2 and v_3 in the graph. colors are represented by gates. our first color (label) is identity gate, second one is NOT gate and the third color is made by a Hadamard gate.

V. CONCLUSION

We have demonstrated coloring of graphs using basic quantum gates. We have shown 2-coloring and 3-coloring which can be further generalized to k- coloring. We have developed an algorithm to check 2-colorability and 3-colorabiltiy of a graph and then color the graph accordingly. Our algorithm can further be modified and can be used to prove an important theorem, namely four color theorem [53]. The approach used here can find applications such as scheduling [43], setting time table [41], coloring of maps, pattern matching [42] etc.

[1] M. A. Nielsen, I. Chuang, and L. K. Grover, Am. J. Phys. **70**, 558 (2002).
[2] Z. Jiayu, Z. Junsuo, X. Fanjiang, H. Haiying, and Q. Peng, IJMLC **4**, (2014).
[3] L. M. K. Vandersypen, M. Steffen, G. Breyta, C. S. Yannoni, M. H. Sherwood, and I. L. Chuang, Nature **414**, 883 (2001).
[4] A. Dash, D. Sarmah, B. K. Behera, and P. K. Panigrahi, arXiv:1805.10478
[5] X. Peng, Z. Liao, N. Xu, G. Qin, X. Zhou, D. Suter, and J. Du, Phys. Rev. Lett. **101**, 220405 (2008).
[6] K. Srinivasan, S. Satyajit, B. K. Behera, and P. K. Panigrahi, arXiv:1805.10928.
[7] H. Talbi, A. Draa, and M. Batouche, 2004 IEEE International Conference on Industrial Technology, 2004. IEEE ICIT '04.
[8] R. Kumar, A. Ranjan, and P. Srivastava, 2010 2nd International Conference on Education Technology and Computer.
[9] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Nature **549**, 242 (2017).
[10] IBM Quantum Experience, URL: https://quantumexperience.ng.bluemix.net/qx/community.
[11] D-Wave, URL: https://www.dwavesys.com/quantum-computing.
[12] IonQ, URL: https://ionq.co/
[13] O. Viyuela, A. Rivas, S. Gasparinetti, A. Wallraff, S. Filipp, and M. A. Martin-Delgado, npj Quantum Inf. **4**, 10 (2018).
[14] D. Aggarwal, S. Raj, B. K. Behera, and P. K. Panigrahi, arXiv:1806.00781.
[15] A. A. Zhukov, S. V. Remizov, W. V. Pogosov, Y. E. Lozovik, Quantum Inf. Process. **17**, 223 (2018).
[16] N. N. Hegade, B. K. Behera, and P. K. Panigrahi, arXiv:1712.07326.
[17] S. Dutta, A. Suau, S. Dutta, S. Roy, B. K. Behera, and P. K. Panigrahi, arXiv:1811.01726.
[18] S. Woerner and D. J. Egger arXiv:1806.06893.
[19] C. Vuillot, Quantum Inf. Comput. 18, 0949 (2018).
[20] D. Ghosh, P. Agarwal, P. Pandey, B. K. Behera, and P. K. Panigrahi, Quantum Inf. Process. **17**, 153 (2018).
[21] R. K. Singh, B. Panda, B. K. Behera, and P. K. Panigrahi, arXiv:1807.02883.
[22] S. Satyajit, K. Srinivasan, B. K. Behera, and P. K. Panigrahi, Quantum Inf. Process. 17, 212 (2018).
[23] D. Alsina and J. I. Latorre, Phys. Rev. A **94**, 012314 (2016).
[24] A. R. Kalra, N. Gupta, B. K. Behera, S. Prakash, and P. K. Panigrahi, arXiv:1707.09462.
[25] M. Swain, A. Rai, B. K. Behera, and P. K. Panigrahi, arXiv:1810.00874.
[26] B. K. Behera, A. Banerjee, and P. K. Panigrahi, Quantum Inf. Process. **16**, 312 (2017).

[27] R. P. Singh, B. K. Behera, and P. K. Panigrahi, DOI: 10.13140/RG.2.2.21343.10408.

[28] S. Gangopadhyay, Manabputra, B. K. Behera, and P. K. Panigrahi, Quantum Inf. Process. **17**, 160 (2018).

[29] A. Pal, S. Chandra, V. Mongia and B. K. Behera, DOI: 10.13140/RG.2.2.19777.86885.

[30] B. K. Behera, T. Reza, A. Gupta, P. K. Panigrahi, arXiv:1803.06530.

[31] B. K. Behera, S. Seth, A. Das, P. K. Panigrahi, arXiv:1712.00854.

[32] C. Cesare, Nature (2015).

[33] D. P. DiVincenzo, D. Bacon, J. Kempe, G. Burkard, and K. B. Whaley, Nature **408**, 339 (2000).

[34] E. Mota, A. Silva, and L. Sampaio, arXiv:1807.09034.

[35] A. Parihar, N. Shukla, M. Jerry, S. Datta, and A. Raychowdhury, Latin Am. Symp. Theor. Inf. **8392**, 433 (2014).

[36] E. D'Hondt, Theor. Comput. Sci. **410**, 302 (2009).

[37] L. Barriere, P. Flocchinib, F. V. Fomin, P. Fraigniaud, N. Nissee, N. Santoro, and D. M. Thilikosg, I **219**, 1 (2012).

[38] S. Oh, arxiv:1709.03678.

[39] A. Parihar, N. Shukla, M. Jerry, S. Datta and A. Raychowdhury, Sci. Rep. **7**, 911 (2017).

[40] J. S, Turner, J Alg, **9**, 63-82 (1988).

[41] A. Nanda, M. P. Pai, and A. Gole, IJMLC **2**, 4 (2012).

[42] M. Wang, and L. Zhu, Instr. Measur. Cir. Sys. **127**, 27 (2012).

[43] S. K Gupta and J. Kyparisis, Omega **15**, 207 (1987).

[44] P. Kumar Chaki and S. Anirban, ICCTICT, 320 (2016).

[45] A. Majumder, A. Kumar, N. Das and N. Chakraborty, IJCSNS **10**, 8 (2010).

[46] A. Kosowski and L. Kuszner, Conf. Ppr., 592 (2006).

[47] Yu Li, C. Lucet, A. Moukrim, and K. Sghiouer, Logistique et transports, Sousse, Tunisia. LT-027 (2009).

[48] L. Barba, J. Cardinal, M. Korman, S. Langerman, A. van Renssen, M. Roeloffzen, and S. Verdonschot, Comput. Sci. **10389**, 97 (2017).

[49] Borowiecki, Piotr, Sidorowicz, and Elzbieta, Fund. Inf. **114**, 105 (2012).

[50] N. Alon and N. Kahale, SIAM **26**, 1733 (1997).

[51] O. V. Borodina, A. N. Glebova, A. Raspaudb, and M. R. Salavatipour, J. Comb. Theor. **93**, 303 (2005).

[52] A. D. Corcolesa, J. M. Chow, J. M. Gambetta, C. Rigetti, J. R. Rozen, G. A. Keefe, M. B. Rothwell, M. B. Ketchen, and M. Steffen, Appl. Phys. Lett. **99**, 181906 (2011).

[53] L. Xiang, arXiv:0905.3713.