Graph Drawing Approaches for Petri Net Models

ANTHONY SPITERI STAINES
Department of Computer Information Systems
University of Malta
MSIDA 2080
MALTA

Abstract: - This work presents a theoretical perspective how to apply graph drawing approaches to Petri Net models. Section 1 is the introduction to the problem. Section 2 is the background and the motivation. This part explains why graph drawing approaches are useful for Petri net modelling. Section 3 is the problem definition. This part explains the main problem. Section 4 defines the proposed solutions and examples are given. Section 5 discusses results, findings and assumptions. Section 6 gives some conclusions.

Key-Words: - Representation, Matrices, Ordinary Petri nets, System Modelling

Received: September 1, 2019. Revised: May 2, 2020. Accepted: May 19, 2020. Published: May 29, 2020.

1 Introduction

Graphs are useful tools for visualization. Their intended use in the field of computing and mathematics is vast. They can be used for formal representation. Graphs can be a fundamental component of computer modelling such as the diagrammatic notations found in software modelling and design [1]-[9]. They are useful to transform observations or mind concepts into actual representation models that can be communicated to different stakeholders. Graph research is not just aimed at acquiring skills to solve known problems but also at dealing with different situations where there is no proper expertise. Graphs can be used to create cognitive maps.

Graph drawing is a discovery process. It can be used to discover the existence of different models and patterns [1].

This work deals with the use of different forms of graph modelling and drawing techniques. Different techniques range from planar orientations, polyline drawings, acyclical representation, and others to tree like structures that can be reduced, redrawn or re-improved. Different techniques and algorithms are usable for compaction [6]-[10].

There are several ways for representing or drawing a graph. This sounds simple but many intricate complexities are involved in doing so. Even the ‘simple’ task of inserting a node or an edge in a model is actually not a simple task at all. Several algorithms ranging from simple to complex ones are involved in this job. Several insertion rules or strategies are available [1]. The insertion techniques can use dummy, literal, clause, etc. type of edges. Similar approaches could be used for nodes. Graph drawings can be heavily connected.

There is the possibility for having cycles or a cycle free drawing. The planarity or the orthogonality of the graph affects the shape and characteristics of the drawing [1]-[9]. The connectedness of the graphs determine how the representation is and impacts the structure of the graph. Sources and sinks in the graph determine the graph type. Polyline drawing approaches are useful for visibility representation. Rooted trees are special cases of graph drawings. Different algorithms like divide and conquer, radial drawing, etc. can be used for drawing rooted trees. The concepts of miniminality and maximalty, imply that when drawing a system different configurations exist [1]-[6]. It is possible to show the minimal form of the drawing or else the maximal form. In the minimal form there is reduced detail however the notation is easier to understand, visualise and remember. It can be aesthetically more pleasant and tidier. On the other hand the maximal principle implies that as much detail as possible is captured in the notation.

This work is limited to Petri net graph drawings [12]-[16]. Petri nets are conventionally bipartite digraphs that fall within the representation of digraphs. They can be conventionally represented using many of the graph drawing mainstream approaches. Which type of approach is best suited to drawing Petri net models? This is a difficult question to answer for the simple reason that every model is different. Simple Petri net drawings can take a variety of different shapes and forms.

Structural representation is an important topic in computing. Structural complexity deals with formal methods but it is not necessarily limited just to formal languages. Complexity can be evidenced in architectures, patterns, computer related structures and why not graphs? This is because graph models represent system structures. Complexity is related to abstraction and representation too. The proper drawing of models is a reflection of the understanding of complexity. Some problems might be unsolvable, however they can still be represented graphically. Different models of computation are possible, graphical abstractions retain a universal characteristic and the models can withstand the test of time.

Abstraction is the tool used to deal with complexity. Abstraction involves paradoxically sacrificing many of the complex details and going for a more limited one. Complex models that reflect many details provide the suitable scenario for abstraction.

Ideally the OCCAM razor principle of simplicity can be applied in this work. Simplification is not a simple task and requires considerable amount of knowledge. This is evidenced by the vast amount of literature in the topic and the multitude of algorithms that can be used for drawing and re-drawing these structures. Abstraction deals with decomposing the problem into smaller counterparts that can be solved individually.

It is impossible to cover all the styles and possibilities that exist in just one paper. Possibly not even several books would do justice to the problem. So this work is a small attempt that tries to strike a balance between drawing Petri nets and their possible graph representations.

2 Background and Motivation

Petri nets are useful formalisms that share a dual identity having both a mathematical and graphical representation. Petri nets [14]-[15] share several common features with digraphs [1],[2]. They are useful for representing static and dynamic structures. They have found extensive use in literature and in various fields. Graph drawings are like syntactic glue that hold the system properties together [1]-[11].

There are several classes of Petri nets that follow different definitions and conventions. These classes range from elementary or simple classes to more structured and complex ones like coloured Petri nets and higher order abstractions of Petri nets. Drawing Petri nets is an easy task if the Petri net is limited in size and scope. Petri nets are widely used in literature and have found their place in different fields of science ranging from computing, networking, electrical engineering and even biology and chemistry (e.g. to show properties and reactions of equations).

In literature, normally no particular style is considered when the Petri nets are drawn. So they are drawn using an ad-hoc approach which is normally acceptable for the work being presented.

This work tries to give ideas on how Petri nets that are drawn can be aesthetically improved using different drawing styles or approaches. The user is given leeway for experimentation when drawing the Petri net. By no means is the user restricted to using the ideas given in this work. This work should serve as a source of inspiration and ideas that would allow the user to come up with a better forms and ideas for presenting the models purely from the drawing point of view. From the mathematical point of view the Petri net that is drawn would remain unmodified. I.e. this work focuses on the visual style of the Petri net.

As already stated it is impossible to cover all the material and ideas related to graph drawing in a book let alone a simple paper. Graph drawings are suitable for improving visualisation as t here are several different styles that can be used to represent graphs. Petri net drawings in some instances are very close to circuit schematic layouts. However the end result and approval of the drawing lies in the user perception. User perception cannot be measured or validated easily. E.g. why is a certain structure more attractive than another even if both are representative of the same thing? Having a good layout will enhance the readability of the structure. There is the concept of added value when drawing a graph. A graph that is perfectly drawn adds value to the work it is presented in. Users need to keep these principles in mind when using Petri net drawings.

A balanced approach needs to be found between the aesthetical drawing of Petri nets and the overdrawing or over engineering of these models.

3 Problem Definition

As structures for visualisation and modelling, graph representations and structures are useful tools for visualisation and modelling. Graphs have been widely used in the field of requirements engineering and software engineering [1],[2].

The drawing and representation of Petri nets as graphs is not simple. The attractiveness of the drawing can be based on principles of layout, simplification, abstraction and aesthetics. This will have a g reat impact on the usability. Most users prefer to use a model that looks nice. Edges and nodes that are properly represented add value to the model. There are no standard established procedures for Petri net drawing. A mixture of methods and several styles can be used. From the graph drawing point of view, Petri net drawing is not ‘optimized’. Sometimes the edges can be in part curves or straight lines, etc. The node size could vary. There are several drawing styles for Petri nets. These can be seen in literature. Combined with this there are several graph drawing methods [1]-[7]. Which are the best for representing and drawing Petri nets? The Petri net diagram can be quite complex so simplification or reduction of the diagram could be of significant importance.

The problem definition is based on the fact that certain classes of graphs can be described recursively.

i) Petri net graphs might exhibit tree like properties. Thus drawing algorithms used for drawing, redrawing and compacting trees might be useful. In this context recursive rewinding techniques could be used.

ii) Decomposition of Series Parallel Digraphs. Petri nets could exhibit properties of series parallel digraphs. These structures can be decomposed to find elements in parallel or in series and condense them into a single element.

iii) A layered redrawing method can be used. Here the Petri net structure is a tree that is decomposed into various subtrees. These can be reconnected later. The process can be called node or subtree re-arrangement.

iv) Radial Drawing. Radial drawing allows for the redrawing or re-arrangement of the Petri net nodes and edges round concentric circles that show radial principles.

v) Rooted trees approach. In this approach the Petri nets are drawn as rooted trees.

This work is a theoretical experiment that tries to understand the different possibilities of graph representation methods that can be applied to Petri net representation. Thus it should be considered that it only provides ideas.

4 Proposed Solutions

As this work is an experiment several graph drawings will be classified and represented. The proposed solutions have been derived from robust approaches used in the field of graph drawing and representation methods.

This work is considered to be a theoretical experiment trying to identify and classify several graph drawing and representation methods. The solutions or proposed methods are as follows:

i) Compact Petri net matrix form for graph drawing [17]
ii) Topological shape metrics drawing approach
iii) Visibility shape metrics drawing approach
iv) Hierarchical drawing approach
v) Bi-graph drawing approach

The techniques mentioned above are given in general form. Some steps have also been changed. I.e. representation of the graph is done using the matrix given in 4.1.

The following examples are given using the following methods:

i) Tree like structures using recursive technique for winding down the model
ii) Decomposition of the tree like structures. I.e. reducing the levels in the tree
iii) Decomposition of Series Parallel like digraphs
iv) Layering or redrawing tree like digraphs
v) Radial drawing of Petri net structures
vi) Horizontal – Vertical drawing of Petri net like structures

4.1 Drawing Petri nets as Tree like Structures

In this method, Petri nets are drawn using tree like structures. Tree like structures are very common in computer organization. The advantage of using tree like structures is the decomposition layout. Practically, tree like structures represent some form of hierarchy. The top node represents the starting point. Nodes and edges are added going down to the bottom of the tree. This type of approach is a top-down approach.

4.2 Reducing the Levels in the Petri net Tree Like Structure

Here the tree like structure is reduced or redrawn by eliminating the levels in the tree. This can be done in several ways. The idea is mainly that of compaction. Compaction implies that a level of the tree is eliminated by adding the components of that level to a level above hence eliminating the level below. This is shown in fig. 1 and 2. Fig. 2 shows the use of a recursive winding technique. Fig. 3 shows the result.

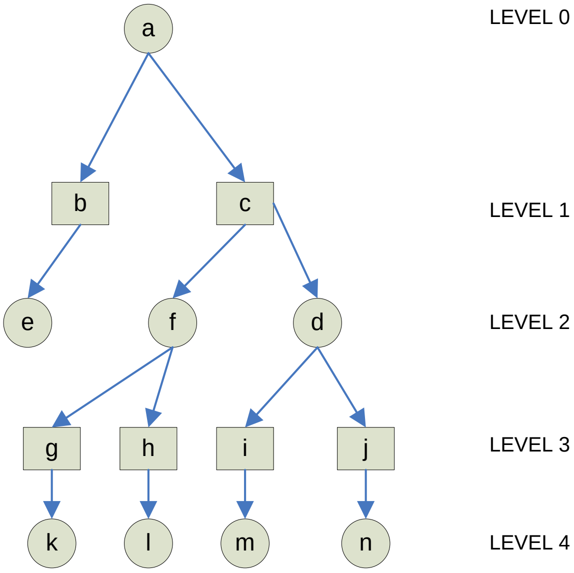


Fig. 1. Petri Net tree like structure

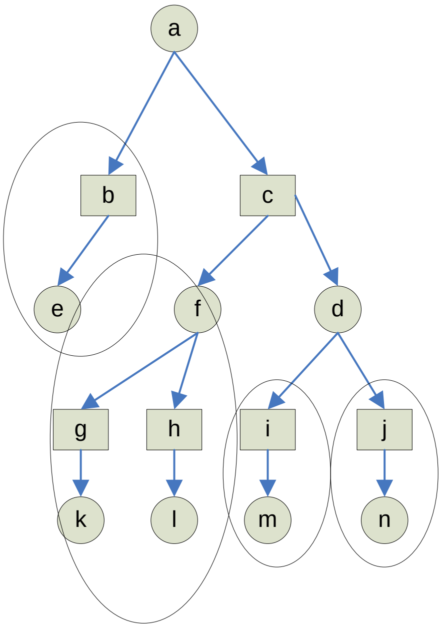


Fig. 2. Petri Net recursive winding technique

4.3 Petri Nets Drawn as Series Parallel Like Digraphs

Petri nets can be simply drawn as series parallel like digraphs. If this approach is used the Petri net resembles certain classes of UML 2 activity diagrams. UML 2 activity diagrams are supposedly based on Petri net like semantics and have many different notations to represent different types and system properties like streams and objects. The models given in the examples indicate that the techniques presented are also useful for activity diagrams. I.e. this work can also be extended to activity modelling.

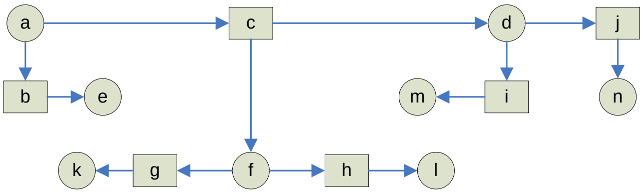


Fig. 3. Petri Net recursive redrawing result for fig.1.

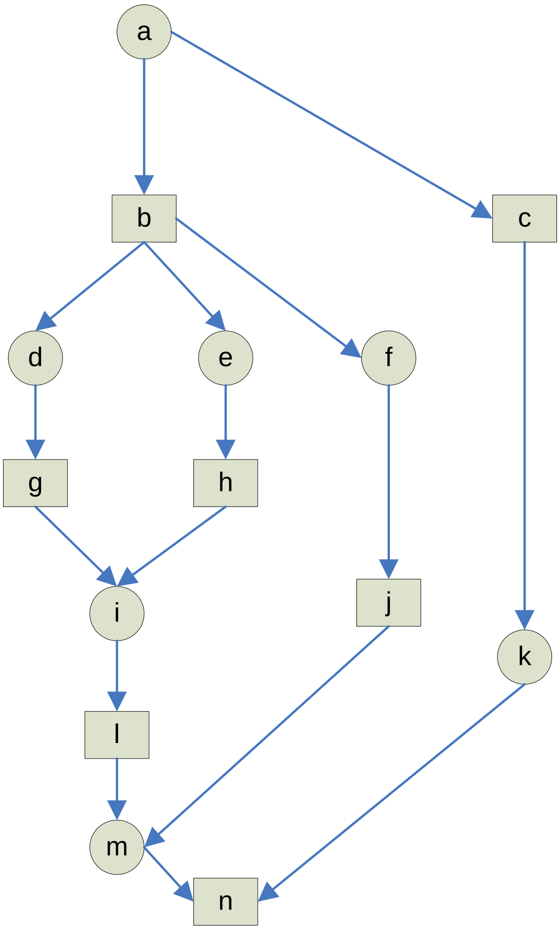
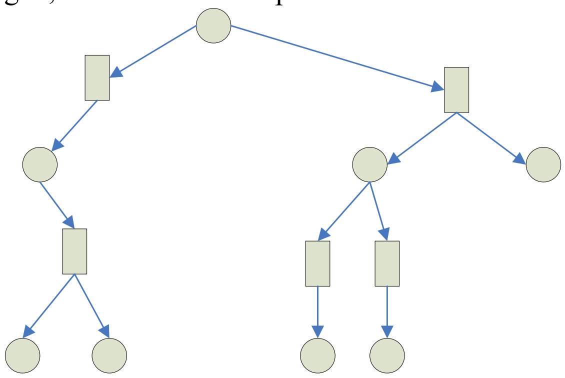


Fig. 4. Petri Net as series or parallel digraphs

4.4 Petri Net Drawing Layering or Redrawing

The layering or redrawing method implies that the structure which is possibly a tree is separated into two halves or other parts. Then the halves are rejoined or regrouped into a new structure that is more compact or better represented. There is also the possibility for node resorting or realignment. Fig. 4, 5 and 6 show this process.



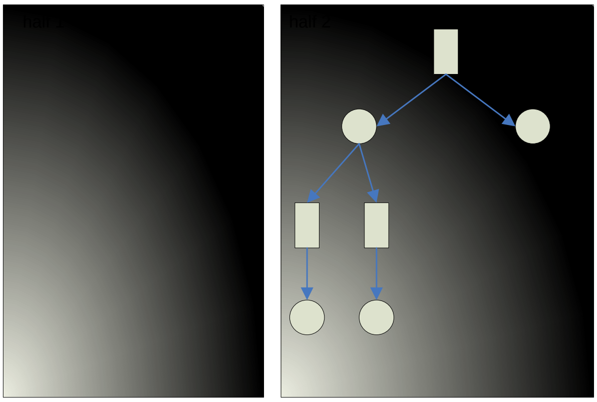


Fig. 5. Petri Net layering or redrawing

4.5 Radial Drawing of Petri Net Structures

Radial drawing of the Petri net structures implies that the Petri net nodes are aligned with concentric circles. This alignment will provide for radial representation or redrawing of the Petri net. The Petri net resembles a tree without a root. This is shown in fig. 7. Note that the radial structure of Petri Nets given in fig. 7 is just in essence a basic drawing layout (i.e. this is just a way of drawing the net).

4.6 Petri Nets Drawn as Horizontal – Vertical Drawings

In this approach the Petri net is drawn as a horizontal binary tree. The representational structure is quite neatly laid out and is very clear to understand. Obviously such a structure will take up more space to represent. For the radial drawing a generalized divide and conquer scheme where vertical and horizontal combinations are put together can be used. It is also possible to use a linear area with a constant aspect ratio when drawing the trees. These will provide more neatness. This is shown in fig. 8.

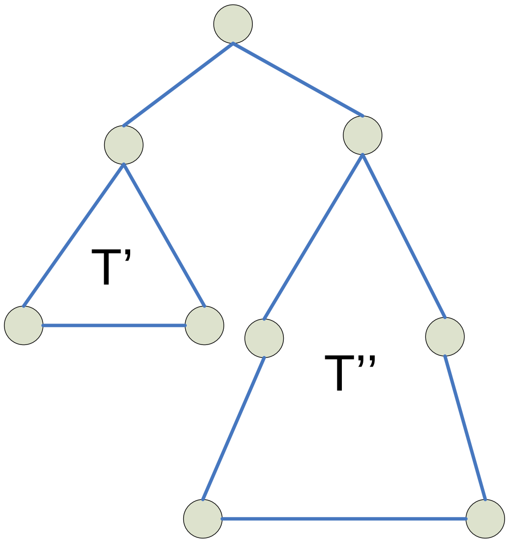


Fig. 6. Possible node resorting

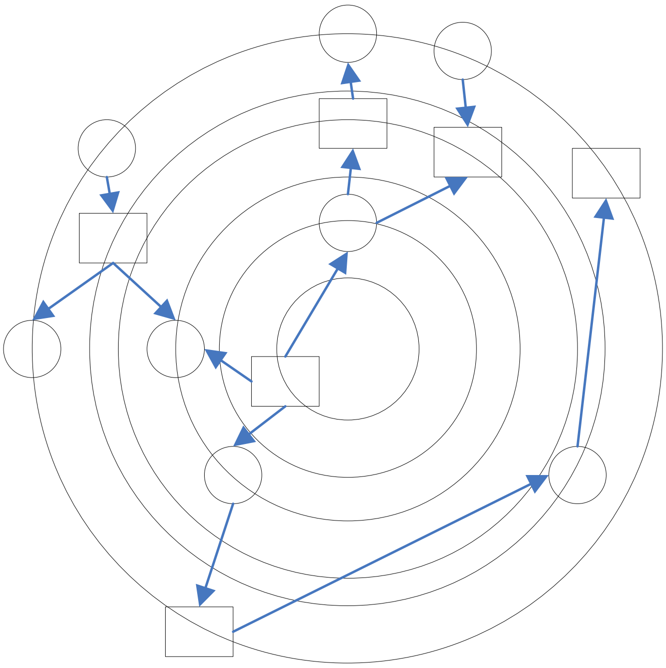


Fig. 7. Outline radial drawing of Petri Net structures

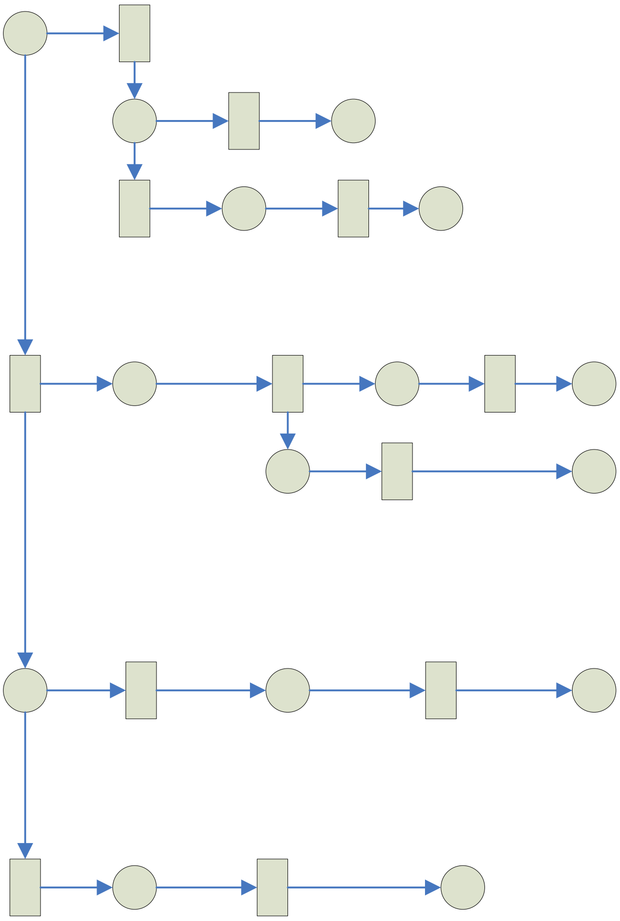


Fig. 8. Petri Net as horizontal-vertical drawing

5 Results, Findings and Assumptions

This work has focused on finding different ways how to redraw Petri nets. Techniques ranging from recursive winding techniques, decomposition of series parallel like structures and layering redrawing etc. have all been used to draw Petri net models.

It is noted that the compaction technique used in the decomposition of series parallel graphs might not be a correct reduction. This is because the main reason for showing this is just to give the idea how the Petri net can be

compacted. So the focus is not on the correctness of the approach.

The graphical illustrations depict the possibilities of how the Petri nets can be represented. It is possible to find several other possibilities. These are not discussed in this paper, however they do exist.

The graphs show balanced and well laid out diagrams. One of the main reasons for using these diagrams is to get a neat layout. The diagrams are fundamental types that show how Petri net modeling can be improved. There is a high level of ambiguity as to which diagrams to select. One diagram could be better than another diagram in a different circumstance.

E.g. the radial drawing could offer more compaction for a Petri net that has a small size. But if a Petri net has a larger size then this would not be so efficient for drawing it and a horizontal-vertical tree might be better.

The limitations of this work is that the focus has been on specific graph types. However there are several other graph types that are not mentioned and can be used. The Petri net structures that have been drawn are simple or toy models. Sometimes in the real world Petri net models can be quite complicated and elaborate. This would make it difficult to select the correct approach for modeling the Petri nets.

For drawing the graph models various software tools could be used. It is possible to use genetic algorithms for drawing the graphs.

6 Conclusions and Future Work

The topic of representing and redrawing Petri nets using graph techniques is a very interesting and important topic deriving its importance from the field of mathematics and graph theory.

Petri nets can be used to represent many different forms of systems and architectures. There are several classes of Petri nets and the higher order classes can be used for more advanced modeling and representations. Graphs are pictorial because they make it simpler to communicate at a glance.

The limitations of the work in this paper are that the techniques presented are fragmented making it difficult to select a p articular technique over another. Also the use of these techniques is time consuming so many users will prefer to use a standard drawing approach which implies that the Petri net is just drawn as it occurs.

Future work can be conducted to see which tools are available to construct these models. Other work can include identifying which models are suited for which purposes. Problems exist as to select the best drawing method.

References:

[1] G. Di Battista, P. Eades, R. Tamassia, I.G. Tollis, *Graph Drawing Algorithms for the Visualization of Graphs*, Pretence Hall, 1998.

[2] G. Di Battista, A. Garg, G. Liotta, R. Tamassia , E. Tassinari , F. Vargiu, An Experimental Comparison of Four Graph Drawing Algorithms, *Computational Geometry Theory and Applications,* Vol: 7, Elsevier, 1997, pp. 303-325.

[3] N. Chiba, K. Onoguchi, T. Nishizeki, Drawing Plane Graphs Nicely, T. *Acta Informatica*, Vol :22, 1985, pp. 187-201.

[4] K. Sugiyama, S. Tagawa, M. Toda, Methods for Visual Understanding of Hierarchical System Structures, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol: 11 , No: 2 , 1981, pp. 109-125.

[5] M. Eiglsperger, M. Kaufmann, M. Siebenhaller, A Topology-Shape-Metrics Approach for the Automatic Layout of UML Class Diagrams, · *Proceeding of SoftVis '03,* ACM, 2003, pp. 189-198.

[6] I. Córdoba, G. Varando, C. Bielza, P. Larranaga, A partial orthogonalization method for simulating covariance and concentration graph matrices, 2018.

[7] N. Gelfand, R. Tamassia, Algorithmic Patterns for Orthogonal Graph Drawing, *Graph Drawings*, 1998.

[8] R. Tamassia, G. Di Battista, C. Batini, Automatic Graph Drawing and Readability of Diagrams, *IEEE Transactions on systems, Man and cybernetics*, Vol: 18, no: 1, 1988, pp. 61-79.

[9] S. C. North, G. Woodhull, On-line Hierarchical Graph Drawing, *Proc. Of the 9th Int. Symposium on Graph Drawing*, 2001, pp. 232-246.

[10] M. Alqadah, G. Stapleton, J. Howse, P. Chapman, Evaluating the Impact of Clutter in Euler Diagrams, Proc. of 8th International Conference, Diagrams 2014 pp. 108-122.

[11] R. Milner, Pure bigraphs: Structure and dynamics, *Information and Computing*, Vol. 204, Issue 1, Elsevier, 2006, pp. 60-122.

[12] A. Spiteri Staines, Alternative Matrix Representation of Ordinary Petri Nets, WSEAS Transactions on Computers, Vol 18, 2019, pp. 11-18.

[13] A. Spiteri Staines, Implementing a M atrix Vector Transition Net, BJMCS, Vol. 4, Science Domain, 2014, pp. 1921-1940.

[14] T. Murata, Petri nets: Properties, Analysis and Applications, *Proc. of IEEE*, vol. 77, i ssue 4, 1989, pp. 541-580.

[15] M. Zhou, K. Venkatesh, Modeling, *Simulation, And Control Of Flexible Manufacturing Systems: A Petri Net Approach* (Series in Intelligent Control and Intelligent Automation), World Scientific, 1999.

[16] A. Spiteri Staines, Matrix Representations for Ordinary Restricted Place Transition Nets, *Transactions on Computers*, WSEAS, Vol 16, 2017,pp. 23-29.

[17] K.M. Abadir and J.R. Magnus, *Matrix Algebra*, Cambridge University Press, 2005

Creative Commons Attribution License 4.0 (Attribution 4.0 International, CC BY 4.0)