AN EVOLUTIONARY GRAPH FRAMEWORK FOR

ANALYZING FAST-EVOLVING NETWORKS

by

Maduako Derek Ikechukwu

M.Sc. in Geospatial Technologies, 2012, jointly awarded by
University of Muenster Germany,
Universitat Jaume I, Spain and New University of Lisbon Portugal
B.Sc. in Geoinformatics and Surveying, University of Nigeria Nsukka, 2008

A Dissertation Submitted in Partial Fulfillment
of the Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in the Graduate Academic Unit of Geodesy and Geomatics Engineering

Supervisor: Monica Wachowicz, Ph.D., Dept. of Geodesy and Geomatics Engineering

Examining Board: Trevor R. Hanson, Ph.D., Dept. of Civil Engineering
Ian Church, Ph.D., Dept. of Geodesy and Geomatics Engineering
David Coleman, Ph.D., Dept. of Geodesy and Geomatics Engineering

External Examiner: Christophe Claramunt, Ph.D., NARI, France

This thesis is accepted by the Dean of Graduate Studies

THE UNIVERSITY OF NEW BRUNSWICK

April 2019
©Maduako Derek Ikechukwu, 2019
ABSTRACT

Fast-evolving networks by definition are real-world networks that change their structure, becoming denser over time since the number of edges and nodes grows faster, and their properties are also updated frequently. Due to the dynamic nature of these networks, many are too large to deal with and complex to generate new insights into their evolution process. One example includes the Internet of Things, which is expected to generate massive networks of billions of sensor nodes embedded into a smart city infrastructure. This PhD dissertation proposes a Space-Time Varying Graph (STVG) as a conceptual framework for modelling and analyzing fast-evolving networks. The STVG framework aims to model the evolution of a real-world network across varying temporal and spatial resolutions by integrating time-trees, subgraphs and projected graphs. The proposed STVG is developed to explore evolutionary patterns of fast-evolving networks using graph metrics, ad-hoc graph queries and a clustering algorithm. This framework also employs a Whole-graph approach to reduce high storage overhead and computational complexities associated with processing massive real-world networks. Two real-world networks have been used to evaluate the implementation of the STVG framework using a graph database. The overall results demonstrate the application of the STVG framework for capturing operational-level transit performance indicators such as schedule adherence, bus stop activity, and bus route activity ranking. Finally, another application of STVG reveals evolving communities of densely connected traffic accidents over different time resolutions.
DEDICATION

TO MY PARENTS AND MY WIFE, FOR THEIR UNCONDITIONAL SUPPORT, ENCOURAGEMENT AND INSPIRATION
ACKNOWLEDGEMENTS

I would like to take this opportunity to extend my sincere thanks to the people who made this work a success. First, I would like to thank my supervisor, Dr. Monica Wachowicz, for her advice, guidance, and encouragement throughout my Ph.D. program, and the financial support from her NSERC/Cisco Industrial Research Chair. Second, I want to thank Dr. Trevor Hanson, for his guidance and assistance throughout the program. My special thanks go to the members of People in Motion Lab for their support and assistance. My special thanks also are extended to the faculty members and staff of the Geodesy and Geomatics Engineering Department at the University of New Brunswick. I also want to give a special thanks to Sylvia Whitaker and Dr. Emmanuel Stefanakis for their support and encouragement. Finally, I would like to acknowledge the Tertiary Education Trust Fund (TETFund) of Nigeria and University of Nigeria Nsukka for the PhD scholarship.
Table of Contents

ABSTRACT ................................................................................................................................. ii
DEDICATION ................................................................................................................................ iii
ACKNOWLEDGEMENTS ............................................................................................................ iv
List of Tables ............................................................................................................................... vii
List of Figures ............................................................................................................................ viii

Chapter 1 Introduction ................................................................................................................. 1
  1.1 Background .......................................................................................................................... 4
  1.2 Scientific Contributions ....................................................................................................... 7
  1.3 Research Questions ............................................................................................................. 8
  1.4 Research Objectives .......................................................................................................... 9
  1.5 Structure of the Dissertation ............................................................................................. 9

Chapter 2
Space-Time Varying Graph for Modelling Places and Events in a Network ..................... 15
  2.1 Introduction ......................................................................................................................... 15
  2.2 Related Work ..................................................................................................................... 19
  2.3 Methodology ...................................................................................................................... 23
    2.3.1 Main STVG concepts .................................................................................................... 23
    2.3.2 Types of subgraphs ....................................................................................................... 25
    2.3.3 Graph Metrics ............................................................................................................... 27
  2.4 Experiment ........................................................................................................................ 30
  2.5 Discussion of the results .................................................................................................... 34
  2.6 Conclusions ......................................................................................................................... 42

Chapter 3 Transit Performance Assessment based on Graph Analytics ............................. 48
  3.1 Introduction ......................................................................................................................... 48
  3.2 Background ........................................................................................................................ 51
  3.3 The Space-Time-Varying Graph (STVG) ........................................................................... 54
    3.3.1 Subgraphs .................................................................................................................. 56
    3.3.2 Relevant graph metrics for the transit network analysis ............................................. 59
  3.4 Application of the proposed STVG to transit AVL data .................................................... 61
3.4.1 Datasets......................................................................................................................... 62
3.4.2 Automated Pre-Processing Steps .................................................................................. 63
3.4.3 Building the Graph Database ........................................................................................ 64
3.5 Discussion of Results......................................................................................................... 66
3.6 Conclusions...................................................................................................................... 76

Chapter 4
STVG: An Evolutionary Graph Framework for Analyzing Fast-evolving Networks .......... 84
4.1 Introduction......................................................................................................................... 84
4.2 Background and Related Work.......................................................................................... 86
4.3 Conceptual Framework for the STVG............................................................................... 92
  4.3.1 Whole-graph ............................................................................................................... 94
  4.3.2 Subgraphs ................................................................................................................. 95
  4.3.3 Projected Graphs ...................................................................................................... 97
  4.3.4 Capturing the Evolution of STVG ............................................................................ 99
  4.3.5 Graph metrics ......................................................................................................... 100
4.4 Case Study ..................................................................................................................... 102
  4.4.1 Data Description ...................................................................................................... 102
  4.4.2 Automated Pre-Processing Tasks ............................................................................... 103
  4.4.3 Building the Subgraphs ......................................................................................... 104
  4.4.4 Platial Subgraph ...................................................................................................... 105
  4.4.5 Mobility Subgraph ................................................................................................. 106
  4.4.6 Connectivity between Mobility and Platial Subgraphs .......................................... 108
  4.4.7 The Whole-graph .................................................................................................. 111
4.5 Discussion of the Results................................................................................................. 112
4.6 Conclusions...................................................................................................................... 117

Chapter 5 Summary and Conclusions................................................................................. 125
  5.1 Summary of the Research ............................................................................................. 125
  5.2 Research Contributions .................................................................................................. 126
  5.3 Future work..................................................................................................................... 132

Curriculum Vitae
List of Tables

Table 2.1 Comparison between GIS layer and graph data models. ................................................. 17
Table 2.2 Attributes of the traffic accident dataset .......................................................................... 31
Table 2.3 Variations in degree centrally over the years according to the Platial Subgraph nodes. ................................................................................................ 36
Table 2.4 Ranking Results. ............................................................................................................ 39
Table 3.1 Comparison between GIS, Relational Database and Graph Database Models. ............. 50
Table 3.2 Definitions of the graph metrics and their respective transit performance indicators. ... 61
Table 3.3 Pseudo code of the PageRank cypher query. .................................................................. 65
Table 3.4 Schedule adherence for each bus route. ........................................................................ 68
Table 3.5 Ranking the bus lines according to their centrality values............................................. 73
Table 3.6 Ranking the stops according to their centrality values. .................................................. 74
Table 3.7 Bus Stops showing the lowest activity levels. ............................................................... 75
Table 4.1 Capturing the Whole-graph evolution algorithm. .......................................................... 99
Table 4.2 Statistics of the Whole-graph. ...................................................................................... 112
### List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Linear structure of an Event Subgraph.</td>
<td>26</td>
</tr>
<tr>
<td>2.2</td>
<td>Linear structure of the Platial Subgraph.</td>
<td>26</td>
</tr>
<tr>
<td>2.3</td>
<td>Overview of the proposed STVG.</td>
<td>27</td>
</tr>
<tr>
<td>2.4</td>
<td>The constant connectivity between the Event Subgraph and Platial Subgraph.</td>
<td>33</td>
</tr>
<tr>
<td>2.5</td>
<td>The Influencing and Severity nodes and their respective edges in the Event Subgraph.</td>
<td>33</td>
</tr>
<tr>
<td>2.6</td>
<td>Connecting event nodes to the time-tree.</td>
<td>34</td>
</tr>
<tr>
<td>2.7</td>
<td>Evolutionary patterns of 20 streets.</td>
<td>35</td>
</tr>
<tr>
<td>2.8</td>
<td>Evolutionary patterns of 3 streets at different time resolutions.</td>
<td>37</td>
</tr>
<tr>
<td>2.9</td>
<td>Evolving graphs for accidents at 3 intersections.</td>
<td>38</td>
</tr>
<tr>
<td>2.10</td>
<td>Annual patterns of severe traffic accidents occurring during the weekends due to distraction.</td>
<td>40</td>
</tr>
<tr>
<td>2.11</td>
<td>Evolution patterns of communities.</td>
<td>41</td>
</tr>
<tr>
<td>3.1</td>
<td>Overview of the proposed STVG.</td>
<td>56</td>
</tr>
<tr>
<td>3.2</td>
<td>The nodes in the mobility space establish the connection to the time-tree.</td>
<td>59</td>
</tr>
<tr>
<td>3.3</td>
<td>The CODIAC Transit Network.</td>
<td>62</td>
</tr>
<tr>
<td>3.4</td>
<td>Different trip patterns of the bus route 52 from a mobility space perspective.</td>
<td>70</td>
</tr>
<tr>
<td>3.5</td>
<td>Observed patterns of trip 52-49 across two consecutive days.</td>
<td>71</td>
</tr>
<tr>
<td>3.6</td>
<td>Hourly patterns of stops activities.</td>
<td>75</td>
</tr>
<tr>
<td>4.1</td>
<td>Proposed STVG Evolutionary Framework.</td>
<td>93</td>
</tr>
<tr>
<td>4.2</td>
<td>Connectivity between the subgraphs and the Time-tree.</td>
<td>95</td>
</tr>
<tr>
<td>4.3</td>
<td>Platial Subgraph.</td>
<td>106</td>
</tr>
<tr>
<td>4.4</td>
<td>Mobility Subgraph.</td>
<td>107</td>
</tr>
<tr>
<td>4.5</td>
<td>The connectivity between the subgraphs.</td>
<td>109</td>
</tr>
<tr>
<td>4.6</td>
<td>The Time-tree.</td>
<td>111</td>
</tr>
<tr>
<td>4.7</td>
<td>The Overview of the Whole-graph.</td>
<td>112</td>
</tr>
<tr>
<td>4.8</td>
<td>Evolution of Graph Density and Volume at hourly temporal resolution.</td>
<td>113</td>
</tr>
<tr>
<td>4.9</td>
<td>Evolution of Graph Density and Volume at monthly temporal resolution.</td>
<td>114</td>
</tr>
<tr>
<td>4.10</td>
<td>Evolution of Graph Density and Volume at daily temporal resolutions for December 2016.</td>
<td>115</td>
</tr>
<tr>
<td>4.11</td>
<td>Evolution of Network volume at daily temporal resolution for Nov and Dec 2016.</td>
<td>116</td>
</tr>
<tr>
<td>4.12</td>
<td>Hourly evolution of Avg. Path Length vis-a-vis graph density and volume.</td>
<td>117</td>
</tr>
</tbody>
</table>
# List of Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFC</td>
<td>Automatic Fare Collection</td>
<td>48</td>
</tr>
<tr>
<td>APC</td>
<td>Automatic Passenger Counts</td>
<td>48</td>
</tr>
<tr>
<td>CSV</td>
<td>Comma Separated Values</td>
<td>31</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
<td>103</td>
</tr>
<tr>
<td>GTFS</td>
<td>General Transit Feed Specification</td>
<td>48</td>
</tr>
<tr>
<td>ID</td>
<td>Identity</td>
<td>32</td>
</tr>
<tr>
<td>IoT</td>
<td>Internet of Things</td>
<td>4</td>
</tr>
<tr>
<td>NB</td>
<td>New Brunswick</td>
<td>102</td>
</tr>
<tr>
<td>PG</td>
<td>Projected Graphs</td>
<td>2, 97</td>
</tr>
<tr>
<td>ST</td>
<td>Spatio-Temporal</td>
<td>5</td>
</tr>
<tr>
<td>STAG</td>
<td>Spatio-Temporal Aggregated Graph</td>
<td>5</td>
</tr>
<tr>
<td>STRG</td>
<td>Spatio-Temporal Region Graph</td>
<td>21</td>
</tr>
<tr>
<td>STVG</td>
<td>Space-Time-Varying-Graph</td>
<td>ii</td>
</tr>
<tr>
<td>TVG</td>
<td>Time-Varying Graphs</td>
<td>19</td>
</tr>
<tr>
<td>UDF</td>
<td>User Defined Functions</td>
<td>32</td>
</tr>
</tbody>
</table>
Chapter 1 Introduction

Modelling and analyzing topological relationships in real-world networks such as in transportation, communications and sensor networks, has been a cornerstone of many research efforts in Geographic Information Systems (GIS) given the role these networks play in enabling modern society. The challenge is that existing approaches for analyzing these dynamic networks, which is primarily based on the “snapshot approach”, can become overwhelmed by the processing complexity and storage requirements when faced with the analysis of a fast-evolving network. Fast-evolving networks generate graphs whose topology evolves swiftly at shorter temporal resolutions. Their graphs continuously grow as new nodes and edges emerge through time. These networks are now ubiquitous; for example, traffic accidents can happen in an instant, with significant network disruptions in a matter of seconds. Similarly, transit operations can be impacted by delays which can propagate throughout a network over time. The future promises even faster evolving networks as the Internet of Things (IoT) becomes mainstream. There is a need for a new analytical approach of capturing evolutions tailored for fast-evolving networks.

The snapshot approach commonly cited in literature to represent changes in a network is associated with high storage overhead and complex query processing especially when capturing the evolution of the network across various spatial and temporal resolutions. These challenges are associated to replication of the entire graph from one snapshot to another, in particular when modelling at shorter temporal resolutions. For example, in the case study of the traffic accident network described in Chapter 2, a total of 52,560 hourly
snapshots, 2,190 daily snapshots, 312 weekly snapshots, 72 monthly snapshots and 6 yearly snapshots would have to be generated to explore the evolutionary process. In the case of the transit network described in Chapters 3 and 4, where the network changed every 5 seconds, it would be computationally infeasible to generate the millions of graph snapshots for the 18-month period of data. An approach that reduces the replication of an entire graph from one time-window to another is therefore needed in order to avoid high storage overhead in analyzing fast-evolving networks.

This research work developed a Space-Time Varying Graph (STVG) framework that circumvents replication of the entire graph from one time-window to another and utilizes time-dependent queries for uncovering evolutionary patterns in fast-evolving networks at various spatial and temporal resolutions. The underlying graphs of the STVG framework is developed based on an integrated Whole-graph, Subgraphs and Projected graphs approach to avoid replication of an entire graph from one time point to another. This approach provides robustness and compactness needed to reduce high storage overhead and computational complexities associated with the snapshot approach. The Whole-graph of the STVG framework can be described as a graph $G = (S, T)$, where $S = (N, E)$ is a digraph and $T = (N, E)$ is a hierarchical tree graph such that $N$ is a set of nodes and $E$ is a set of edges. $S$ is composed of subgraphs and each subgraph is a subset of the whole-graph $G$. Multiple subgraphs $S'$ can be conceptualized such that $S = (S'_1, S'_2, \ldots S'_n)$ representing different non-metric spaces (e.g. Event Space, Mobility Space, Geographic Space). However, only one subgraph $S'_i$ is connected $T$ (Time-tree) which is proposed to represent the natural levels of a timestamp. Essentially, $T$ is a hierarchical (tree) structure where a root node is followed by year nodes, month nodes, day nodes, and further temporal resolutions. Each node of this
subgraph $S_i$ must be linked to a leaf-node of $T$. This is an important constraint to support any kind of time-dependent queries and optimal processing time.

The Projected graphs are vital for retrieving nodes and edges that define the state of the Whole-graph at varying temporal resolutions, time-windows and based on subgraphs of interest. Projected graphs present the key graphs on which graph metrics are computed for evolutionary analytics of the Whole-graph $G$. From the vertex-centric standpoint, the evolution of the Whole-graph $G$ materializes in the time-varying evolution of its Projected graphs. Depending on the network and application, varying resolutions of time can be defined to appropriately capture the evolutionary behavior of the graph. The evolutionary analytics is based on the research premise that graph metrics such as shortest path (weighted and unweighted), degree, betweenness and PageRank centrality, and clustering algorithms such as Louvain can be used to uncover evolutionary patterns. At the extreme, each time resolution could correspond to the smallest time unit in the dataset or to the time between any two consecutive modifications of the graph. In some cases, every Projected graph of the whole-graph $G$ becomes equivalent to an evolving process of a network.

The implementation of the proposed STVG framework was carried out using two case studies, namely, traffic accident and transit networks. These networks were selected because of the data availability, but also because they represent the expected behavior of fast-evolving networks. The datasets have different data rates which generate varying subgraphs and projected graphs across different spaces (event, geographic and mobility spaces) and temporal resolutions.

The traffic accident dataset is from the Florida Department of Highway Safety and Motor Vehicles for Brevard County covering the period of 2010 to 2015. It contains
information on 1,048,575 accidents that were collected at different temporal resolutions. The case study also includes the road network shapefile of the county. The transit dataset contains the Automatic Vehicle locations (AVL) feeds from the CODIAC transit network in Greater Moncton, New Brunswick, Canada. The road network dataset containing information about the bus routes, bus stops, street segments, street intersections, civic addresses was obtained from the GeoNB online service. A total of 59,617 completed bus trips from the AVL feeds pulled from the buses every 5 seconds into a for a period of 18 months (from June 2016 - December 2017) was used in this case study.

It is important to point out that STVG framework presents the potential for a wider applicability in modelling other real-world networks, such as the Internet of Things (IoT), social networks and communication networks.

1.1 Background

Time-varying graphs have been extensively explored in modelling slowly evolving networks including web citation and co-authorship network (Huang, et al. 2017, Khurana and Deshpande, 2016, Khurana, 2016, Tong et al. 2008), online social networks (Rossi et al. 2013, Y. Yang et al. 2014, J. Yang and Leskovec, 2011, Pereira et al. 2016, Koloniari and Pitoura, 2013), biology and disease networks (Leventhal et al. 2015, Magnien and Tarissan, 2015, Starnini et al. 2013, Jalili et al. 2016) as well as in slowly-changing communication flow networks (Blondel et al. 2008, Yarlagadda et al. 2015, Landesberger et al. 2015, Glacet et al. 2016). The graph evolution was modelled as a snapshot sequence of a graph. Each graph snapshot was a static representation of a valid state of a graph at a particular time point (Cheung et al. 2015). Changes in the topological relationships were also represented as a
sequence of graph snapshots (Zaki and Attia, 2016). The snapshot approach has been successfully applied to small network sizes because of the convenience it affords users to analyze the properties of an evolving graph based on discrete versions of consecutive time points or time windows (Liu & Yu, 2010). However, some limitations arise when using the snapshot approach for large size networks, ranging from high storage overhead due to the replication of an entire graph every time there is an update to expensive computation of queries across the snapshots (Huo & Tsotras, 2014).

It is also important to point out the focus of previous approaches on changes over time with no account of the changes of the networks in the space dimension. A few attempts can be found in the literature looking at graph evolution across space and time. This is because modelling the evolution of topological relationships of a network in varying space and temporal dimensions is a challenging task, especially where the space dimension includes both metric (e.g. Euclidean space) and non-metric spaces (e.g. Platial, mobility, social and economic spaces).

Previous research work includes the Topological Aggregated Graph (TAG) that was based on the snapshot approach and initially proposed by Betsy and Shashi (2009) and later used by Rocha (2017) for modeling changes in transportation networks. Costes et al. (2015) have extended the TAG model to a Spatio-Temporal Aggregated Graph (STAG), where temporal snapshots are created by a matching algorithm in such a way that nodes and edges containing homologous timestamps are aggregated into ST-nodes and ST-edges respectively. One of the main advantages of this approach is the automated detection of evolutionary patterns due to the appearance and disappearance of nodes and edges.
Considering spatial geometry, Sadahiro (2001) proposed a graph data model to represent changes in the size and direction of polygons at different times. By overlaying two temporally neighboring layers (i.e. snapshots), the graph data model empirically analyzes changes in market areas of a convenience-store chain in Tokyo, Japan. More recently, Cheung et al. (2015) proposed a similar graph data model to represent a sequence of snapshots of a landscape using a time series. Nodes represent centroids of landscape patches (e.g. lattices) and edges represent the adjacency relationship between them.

Finally, Mondo et al. 2013, developed a graph-based approach to represent evolving entities in space and time. The model makes a distinction between filiation (derivation and continuation) and spatial relationships between entities. The spatio-temporal graph model was implemented across snapshots of time using an extended relational database for the implementation.

Many real-world networks however, do not evolve at the same rate: some evolve slowly while others evolve swiftly (Aggarwal & Subbian, 2014). For example, in transit networks, significant changes in the graph can occur every second, whereas in co-authorship networks changes might take place over months or years. The challenge is that for fast-evolving networks, where the evolutionary analytics relies on the power of representing the evolution in the network across smaller temporal resolutions, the snapshot approach presents high computation and storage problems (Liu and Yu, 2010). Therefore, the Whole-graph approach appears to be more suitable for modeling large and fast-evolving networks to reduce replication of the entire graph across time and affords computation of time-dependent queries at various temporal resolutions.
The Whole-graph approach was utilized in Pereira et al. (2016) to analyze evolving centrality metrics in a dynamic Twitter graph. However, the time-dimension of the graph is based on timestamp attributes of the nodes and edges of the graph. Each node and edge in the graph have a valid time at which they existed in the graph. Varying-time windows of weekly, fortnightly and monthly windows were used in the analysis. The disadvantage is that having timestamps as attributes of the nodes and edges is inefficient in cases where the network grows frequently with billions of nodes and edges to scan through. The whole-graph approach was also utilized in burst area discovery based on evolving top-k changes in a stream of fast bipartite graph evolutions of users and stories on the Digg.com website (Liu and Yu, 2010). The authors, however, utilized the Haar wavelet tree to reduce high computation complexity associated with a fast-evolving graph stream. Top-k burst areas are computed incrementally from small hop size to large hop size of time windows.

1.2 Scientific Contributions

The proposed STVG framework conveys the following scientific contributions in view of the above discussed background:

- This research work developed an approach to reduce high storage overhead and computational complexities associated with the snapshot approach. Previous studies are still focused on the snapshot approach, and more research is needed to explore the Whole-graph approach for real-world networks, especially fast-evolving networks. With the advent of new technologies such as the IoT networks, a massive amount of IoT data will be generated every second hindering the possibility of using the snapshot approach.
The STVG framework is based on modelling subgraphs for capturing the evolution of different spaces (metric and non-metric spaces) of fast-evolving networks. Exploring the dynamics of the topological relationships between subgraphs is a significant challenge, demanding that the data scientists involved develop complex conceptual frameworks capable of modelling and analyzing fast and large volume of heterogeneous data. The STVG framework proposed in this research work is a step forward to dealing with this challenge.

There are still open research issues in the field of fast-evolving networks, especially when attempting to analyze a massive whole-graph across different temporal resolutions. Evolutionary patterns are scale dependent and users should be able to determine the time granularity at query time. This thesis is an attempt to outline the main components of a suitable framework that can support ad-hoc graph queries and graph algorithms across temporal resolutions.

1.3 Research Questions

This research explored four questions:

- How can an STVG framework be developed for modelling and analyzing fast-evolving networks?

- How can a graph database be used to implement an STVG framework that captures the network’s space and time evolution with less storage overhead?

- How do graph metrics evolve in time and space as a function of the dynamics of fast-evolving networks?
What are the practical applications of an STVG framework?

The answers to the questions are summarized in Chapter 5 of this thesis, and the research work carried out in answering the questions are presented in Chapters 2, 3 and 4.

1.4 Research Objectives

The overall goal of this thesis was to develop a framework suitable for representing and carrying out evolutionary graph analytics of fast-evolving networks across spatial and temporal resolutions. The objectives in support of the goal were to:

- Develop a STVG framework for fast-evolving networks;
- Implement the STVG framework in a native Graph database;
- Carry out graph processing and ad-hoc query retrieval involving topological and attribute properties of the network from the graph database;
- Evaluate the STVG framework by carrying out a transit performance assessment, an evolutionary analytics of the transit network and a traffic accident analysis.

1.5 Structure of the Dissertation

This dissertation is article-based, containing the following papers as chapters:

  
Chapter 2 presents the proposed STVG framework (research objectives 1, 2, 3 and 4). The Whole-graph approach is described by combining directed and bipartite subgraphs with a time-tree for representing the topological relationships between places and events across time. The proposed STVG is used to explore evolutionary patterns of traffic accidents using graph metrics, ad-hoc graph queries and a clustering algorithm.

Chapter 3 presents the application of the STVG framework to conduct a transit performance assessment across different spatial and temporal resolutions (research objectives 2, 3 and 4). The mobility and geographic spaces of transit networks are modelled for processing a large volume of Automatic Vehicle Location data feeds generated by the bus transit network serving the communities of Greater Moncton, NB, Canada.

Chapter 4 presents the STVG framework is proposed to model the dynamics of a fast-evolving network and reduce high storage overhead in massively changing graph where new nodes and edges arrive every second (research objective 1, 2, 3 and 4). It supports the capability to build projected graphs at different time-windows and analyze their metrics across varying temporal resolutions.
Chapter 5 presents the summary of the research methodology, answers to the research questions and future research work.

References


Chapter 2

Space-Time Varying Graph for Modelling Places and Events in a Network

2.1 Introduction

Topological relationships can generate a network structure that allows us to represent the concepts of connectivity and adjacency between objects. Connectivity describes how objects are linked to each other to form a network, meanwhile adjacency describes whether two objects are next to each other within a network. According to Curtin (2007), topological relationships were one of the earliest representations in GIS, and objects belonging to a network structure have been usually broken down into arcs, nodes, and polygons that are stored in relational database tables, enforcing planar topology on data and topological layers.

A network structure in GIS might consist of one or more topological layers. In particular, another approach for handling connectivity in GIS is to compute on-the-fly the Euclidean distance between two arbitrary nodes. Topological relationships are not static, and they generate different evolutionary patterns that can be ordered by time. Integrating time introduces a specific challenge because there is a lack of a unified network structure in GIS that can represent topological relationships at varying temporal resolutions since the table-oriented storage is not suitable to effortlessly explore the underlying topological relationships across time (Mondo et al. 2013). In contrast, graph data models define a network structure which provides a more natural way to represent the topology of relationships in data mainly because nodes can represent any abstraction of the real-world, including objects independent...
of their cartographic representation (i.e. point, line or polygon), events at different temporal resolutions, and moving entities that change their topological relationships over time.

In this case, topological relationships are stored using the index-free adjacency method where each node of a network structure carries information about its neighbors, and as a result, a traversal query of an edge from a node is independent of the database size and the query can run at constant time (De Virgilio et al. 2014). This ensures that -- irrespectively of the volume of edges and nodes stored in a graph database -- graph query processing can be performed at any temporal resolution, especially for applications where data size increases over time (Angles & Gutierrez, 2017).

Therefore, the role of topological relationships in building network structures is one of the main characteristics that differentiates graph data models from GIS layer models. Table 2.1 highlights other fundamental distinctions regarding to graph metrics and topological queries. Shortest path, degree, betweenness and PageRank are graph metrics that can be used to uncover evolutionary patterns in networks. Finally, topological queries in GIS have been designed based on the topological arrangement of objects in geographic space that is well-known as the 9-intersection model (Egenhofer, 1993, Chen, et al., 2001). On the other hand, the needs of different applications have led to a variety of topological modelling and query languages in graph databases such as graph-based object queries (e.g. GraphDB, G-log, GOOD), unstructured queries (e.g. StruQL, Lorel), and networks (e.g. Cypher, SoQL).
Table 2.1. Comparison between GIS layer and graph data models.

<table>
<thead>
<tr>
<th>Concept</th>
<th>GIS Layer Model</th>
<th>Graph Data Model</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Building a network</td>
<td>Slow because topological relationships are primarily computed on-the-fly or stored in a topological layer.</td>
<td>Fast because topological relationships are built using the index-free adjacency.</td>
<td>Sadahiro (2001)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Curtin (2007),</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Baas (2012)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Miler et al. (2014)</td>
</tr>
<tr>
<td>Topological Queries</td>
<td>Slow and cumbersome when having many objects and relationships belonging to different GIS layers.</td>
<td>Efficient because graph databases are computationally designed to handle many entities and relationships.</td>
<td>Vicknair et al. (2010)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ciglan et al. (2012)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ivarsson et al. (2014)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Cheung et al. (2015)</td>
</tr>
</tbody>
</table>

Few attempts can be found in the literature in modelling the dynamics of topological relationships using graph data models. One example includes the Topological Aggregated Graph (TAG) that was initially proposed by Betsy and Shashi (2009) and it was also utilized by Rocha (2017) to keep track of changes using timestamps attached to each node and edge of a transportation network. In their TAG, the network structure does not change, meaning the number of edges and nodes are constant through time, only the properties of the nodes and edges are updated. Although this model has less storage requirements, queries based on connectivity and adjacency relationships using centrality metrics and path algorithms require predicates to assess if a node exists at a specific time. Costes et al. (2015) have also extended the TAG model to a Spatio-Temporal Aggregated Graph (STAG), where temporal snapshots...
are created by a matching algorithm in such a way that nodes and edges containing homologous timestamps are aggregated into ST-nodes and ST-edges respectively. One of the main advantages of this approach is the automated detection of evolutionary patterns due to the appearance and disappearance of nodes and edges. It has been successfully applied to investigate the evolutionary patterns of city street networks.

From a GIS perspective, Sadahiro (2001) proposes a graph data model to represent changes in the size and direction of polygons at different times. By overlaying two temporally neighboring layers (i.e. snapshots), the graph data model empirically analyzes changes in market areas of a convenience-store chain in Tokyo, Japan. More recently, Cheung et al. (2015) proposes a similar graph data model to represent a sequence of snapshots of a landscape in a time series. Nodes represent centroids of landscape patches (e.g. lattices) and edges represent the adjacency relationship between them. The time dimension is implemented as a stack of graph snapshots, and it allows the identification of changes in landscape patches that would not be possible to be detected using GIS.

This Chapter proposes a space-time varying graph that is based on the Whole-graph approach where a network structure evolves in time and space in such a way that evolutionary patterns are due to the changes in the connectivity and adjacency relationships among nodes of a network. However, it is very challenging to build such a graph data model because space does not only imply metric spaces (e.g. Euclidean space), but in addition, it includes non-metric spaces (e.g. Platial space) which have properties that vary under continuous evolution. The space-time varying graph has a set of arbitrary links in which the concepts of connectivity and adjacency are based on the existence of local interactions between places and events across time. Couclelis (1999), after reviewing some historical roots to space and
time concepts from many perspectives originating from the fields of philosophy, physics, mathematics, and geography, has pointed out that the representation of non-metric spaces is more generic and effective because they do not depend on metric properties such as shape, size, and distance. An experiment is used to demonstrate the importance of space-time varying graphs as the fundamental representation underpinning the complex interactions between non-metric spaces at varying temporal resolutions, and, in this way, to explore its potential to advance traffic accident analytics which has been traditionally carried out mainly based on spatial point pattern analysis in GIS.

2.2 Related Work

Graphs have been extensively used to model networks and topological relationships ranging from neurobiology and transportation to urban structural dynamic and statistical physics (Strogatz, 2001, Curtin 2007, Sadahiro 2001, Barabási, 2009). In particular, Time-Varying Graphs (TVGs) have been used to model changes happening in the network structure of graphs based largely on three approaches: the snapshot approach, the log file method and the Whole-graph approach (for a comprehensive survey please refer to Zaki and Attia, 2016).

In the snapshot approach, any topological change is captured as a sequence of graph snapshots, that is, \( G_{[t_1-t_n]} = \{G_1, G_2, G_3, \ldots, G_n\} \). Each graph snapshot \( G_i \) is a static representation of a valid state of a network at a time point \( t_i \) (Cheung et al., 2015). A snapshot consists of a set of nodes \( N \) and a set of edges \( E \), and their respective properties at a valid time instance \( t_i \), \( G_i = (N, E, t_i) \). This method can be applied to small network sizes where changes are periodic and occur according to a-priori known discrete time-window.
However, some limitations arise when using the snapshot approach for large size networks, ranging from high storage overhead due to the replication of an entire graph every time there is an update to expensive computation of queries across the snapshots. Many strategies have been proposed to address the storage overhead issue by using a set of deltas that store only the changes that are needed to construct a snapshot. The most efficient compression approach is known as SM-FVF (Ren et al. 2011) which creates clusters of k snapshots, where a compressed graph has no redundancy among delta files of the same cluster, but it still has redundancy between delta files of different clusters.

The log file method consists of a modified version of the snapshot approach whereby the most current snapshots are kept in a database while each historical snapshot is stored into a log file which is associated with a time interval. This method models changes as
\[ G_{[t_1-t_n=curr]} = [G_{[t_{curr}]}, L, M] \]
where \( G_{t_{curr}} \) represents the current version of a graph, \( L \) is a set of log files that represent the historical versions that have occurred during a time interval \( L_{[t_1-t_n=curr]} \) and \( M \) is a set of materialized snapshots. However, this method requires a trade-off between the redundancy ratio and log file size. Koloniari, Souravlias & Pitoura (2013) and Khurana & Deshpande (2016) have applied this method in social and citation networks since the number of nodes remain quasi-static in comparison to the creation of new edges. Nevertheless, compression techniques have been used to handle the log file sizes.

Finally, the Whole-graph approach handles the evolution based on a temporally-indexed graph that grows and updates continuously or discretely, that is \( G_{[t_1,t_n]} = [N_{[t_1-t_n]}, E_{[t_1-t_n]}] \), where \( N_{[t_1-t_n]} \) and \( E_{[t_1-t_n]} \) are sets of all node and edge instances respectively. Each node or edge has a timestamp attached to it depicting its valid time in the graph. The advantage of this approach is that it requires less storage overhead, but it is
inefficient in terms of query processing time due to the timestamp being stored as a property of each node and edge of a graph.

While many different methods for building time-varying graphs have been proposed in the literature, studies on space-time varying graphs are scarce and have been predominately focused on the snapshot approach (Cheung et al. 2015, Thibaud et al. 2013, Mondo et al. 2013, Stell et al. 2011). Earlier utilization of the graph snapshot approach for analyzing topological changes can be found in the study of polygon distribution of convenience stores in a market area in Tokyo by Sadahiro (2001). In this study, spatial polygon features are represented as nodes in each graph snapshot while a temporal edge is used to link respective polygons between snapshots. Changes in the polygon distribution of the convenience stores are observed based on the topological changes of the graph over the snapshots. Lee et al. (2005) later propose the Spatio-Temporal Region Graph (STRG) for representing topological objects (nodes) and their relationships (edges) extracted from video sequences. A STRG is generated for each frame and is decomposed into its subgraphs representing both non-metric and metric spaces for matching and indexing respectively. It has been a pioneering attempt in representing not only spatial features of video objects, but also the temporal relationships among them.

A variation of this snapshot approach has been adopted by Landesberger et al. (2015) for analyzing space-varying flow data over a time interval. They propose a Mobility Graph having a pre-defined discrete set of geographic locations (nodes), and a set of edges (i.e. people moving from one location to another). Flow is obtained by clustering the trips that were represented as edges connecting an origin with a destination node. Weights were also given to the edges to represent the magnitude of flow. The approach was implemented using
Call Data Records from a telecommunication company and georeferenced tweets. The results corroborate the scientific understanding of the benefits of using graphs to represent spatial relationships across different time steps. Stell et al. (2011) earlier had used bipartite graph snapshots to represent the evolution of shared ancestry where ‘father’, ‘mother’ and ‘child’ can be represented as nodes in each graph snapshot and the edges are used to represent shared ancestry. The work also went further to use the same concept to study the evolution in the movement of groups of people from one suburb to another within a city. The entities represented as nodes are groups of people and the filiations between them (as people move from one group to another) are represented as the edges (bipartite relationships) over time snapshots.

Specifically, in social network analysis, the graph snapshot approach has been also used to study periodic behavior (Lahiri et al. 2008, Shekhar and Oliver, 2010, Rossi et al. 2013) where the nodes represent the geographic location of friends and the edges represent their social relationships which are weighted by the Euclidean distances between them. Changes in the topological structure of the graphs are the major interest as they represent the friendship behavior that emerges from friends connecting and disconnecting across graph snapshots.

The proposed STVG is an attempt to adapt the Whole-graph approach by intuitively representing non-metric spaces where places and events can be represented as nodes, and edges are used to represent the topographic relationship among these nodes over varying temporal resolutions. It combines directed and bipartite graphs with a time-tree for representing the complex interaction between places and events across time. This offers robustness to the STVG in the sense that it reduces replication of the same graph components.
and it supports time-dependent queries using a time point or a time interval of the time-tree leaf nodes.

2.3 Methodology

The proposed STVG is based on the Whole-graph approach for representing entities of non-metric spaces (e.g. places, events, mobility, people and so on) and their topological relationships (e.g. adjacency and connectivity) that are not static. Subgraphs (such as illustrated in Figure 2.1) are used to facilitate the conceptual modelling of the evolution of topological relationships where nodes in each subgraph are entities in a meaningful non-metric space. However, in a graph database, the subgraphs are actually stored as one evolving whole-graph. It is also important to point out that subgraphs are not graph snapshots since they require complementary connectivity (at any point in time, the nodes belonging to one subgraph are linked to the nodes belonging to another subgraph) as well as eventual adjacency (at any point in time, a target node will be sequentially linked to a source node in the same subgraph).

2.3.1 Main STVG concepts

The STVG can be described as a whole-graph \( G = (S, T) \), where \( S = (N, E) \) is a digraph and \( T = (N, E) \) is a hierarchical tree graph such that \( N \) is a set of nodes and \( E \) is a set of edges. \( S \) is composed of subgraphs and each subgraph is a subset of the whole-graph \( G \). Multiple subgraphs \( S' \) can be conceptualized such that \( S = (S'_1, S'_2, \ldots, S'_n) \) representing different non-metric spaces. However, only one subgraph \( S'_i \) is connected \( T \) which is proposed to represent the natural levels of a timestamp (Ivarsson et al. 2014). Essentially, \( T \) is a hierarchical (tree) structure where a root node is followed by year nodes, month nodes, day nodes, and further
temporal resolutions. Each node of this subgraph \( S'_i \) must be linked to a leaf-node of \( T \). This is an important constraint to support any kind of time-dependent queries and optimal processing time. As a result, the STVG enables the fast computation of graph metrics, avoiding the limitations accruable to the static graphs such as Dual and Primal graph models (Añez et al.1996).

Nodes belonging to the same subgraph \( S'_i \subset S \) are linked by eventual adjacency edges between an existing node (source node) and a future node (target node) to be created in a graph database. These edges can be weighted using an associated numerical value that is a non-negative integer obtained by computing, for example, the Euclidean distance between the source and target nodes if they represent two locations of a moving object. The weights can also be computed using the elapsed time between them. Nodes belonging to different subgraphs \( S'_i \subset S \) and \( S'_j \subset S \) are linked by complementary connectivity edges since for every node in a subgraph there is always a corresponding node in another subgraph to which it is related.

All nodes belonging to a subgraph \( S'_i \subset S \) have labels and properties. Labels assign roles (e.g. “Stopped” representing the state of a moving object) and objects (e.g. “Street” and “Intersection”). All nodes labeled using the same label should belong to the same subgraph \( S'_i \). Graph database queries can be used to search a set of labels of a subgraph \( S'_i \), instead of the whole-graph \( S \). Finally, multiple properties can also be attached to the nodes and edges of a subgraph \( S'_i \). They are attributes \((a_1 \ldots a_n)\) containing semantic information about real-world entities, spatial data types (e.g. points), and temporal types (e.g. duration of an event).

The evolutionary assumptions underlying the STVG = \((S, T)\) can be described based on the following:
- **Node Connectivity over Time**: At any point in time, at least one node belonging to a subgraph $S'_i$ can be reached by all the other nodes in $S$. This assumption is essential to support any graph query/algorithm whose input is spread over all nodes generated at different time instances, with at least one node capable of generating the output.

- **No Recurrence of Edges over Time**: An edge is created only once at any point in time. This assumption is critical to maintain the eventual adjacency relationships thus to perform any computation of a graph metric such as shortest path and centrality.

- **Constant Connectivity**: Only one subgraph $S'_i$ and $T$ are always linked at any point in time. There must exist an edge between a node belonging to a subgraph $S'_i \subset S$ and a leaf-node of $T$.

- **Eventual Adjacency**: There is always a future target node in subgraph $S'_i \subset S$ that a source node will be linked to.

### 2.3.2 Types of subgraphs

Different types of subgraphs can be defined depending on the number of real-world entities and non-metric spaces being considered. In this section, the ‘Event’ and ‘Platial’ subgraphs are introduced. First, the Event Subgraph represents any discrete sequence of events that occurs at different time points. The nodes are semantically labelled to identify a real-world event they represent. A target node and its respective source node are linked using a NEXT edge in the Event Subgraph (Figure 2.1). This is important for triggering the subgraph evolution where $n_{st1}$ represents the first instance of an event node in the subgraph.
and \( n_{st_n} \) is the last instance of an event node. Different types of events can be represented using this linear subgraph structure, modelling the sequential order of their occurrence using eventual adjacency topological relationships.

![Figure 2.1. Linear structure of an Event Subgraph.](image)

Second, the Platial Subgraph is proposed to represent the sequence of places where the events have taken place in the geographic space such as roads, streets, intersections, bus stations, buildings and regional locations. It also represents a sequential order of nodes in \( S' \) using the NEXT edge as eventual adjacency topological relationships. Figure 2-2 illustrates the diagraph of a Platial Subgraph.

![Figure 2.2. Linear structure of the Platial Subgraph.](image)

Ideally, each node within the Event Subgraph has a corresponding node in the Platial subgraph. This complementary connectivity between these subgraphs in the STVG is semantically represented by the “LOCATED_AT” edges. Complementary connectivity is a fundamental topological relationship of STVG because independent of the number of subgraphs being represented in the STVG, all nodes are always connected to a time-tree using the “HAPPENED_AT” edges as shown in Figure 2.3.
2.3.3 Graph Metrics

One of the advantages of modelling every entity as a node in the proposed STVG is the ability to compute both node-centric and edge-centric graph metrics. While modelling time instances as nodes provides the ability to represent every time instant in the timestamp of the events as a node such that evolution of graph metrics can be tracked to any temporal resolution. Graph metrics such as shortest path (weighted and unweighted), degree, betweenness and PageRank centrality, and clustering algorithms such as Louvain can be used to uncover evolutionary patterns. They are further explained below.

**Degree Centrality:** This computes the in/out degree of a node. It usually provides information on how many neighbours a node has, and as a result, it helps us to identify the key nodes in a graph. For example, finding the most influential nodes (i.e. persons) in a social network. It is the simplest and easiest centrality metric to compute using a database query,
and therefore, it is the most popular metric used in graph analytics. It is defined as follows, let $A = (a_{ij})$ be the not symmetric adjacency matrix of a digraph $S$, the in/out degree centrality $D$ is given by

$$D(i) = \sum_{j=1}^{n} a_{ij} \quad (2.1)$$

where $a_{ij} = 1$ if there is a directed edge between nodes $i$ and $j$ and $a_{ij} = 0$ if there is no edge.

In the STVG, the use of degree centrality is expanded to not only finding key nodes, but also providing the ability to trace the key nodes at various temporal resolutions. This can be achieved by using both the eventual adjacency between nodes belonging to the same subgraph, and the complementary connectivity across the Place and Event subgraphs. For example, in the experiment described in Section 2-4, degree centrality is used in range queries with multiple MATCH-clauses to retrieve the key nodes at different time resolutions.

**PageRank Centrality:** This is an algorithm originally developed by Google that uses the hyperlink structure of the web to rank web pages according to the number of links to them (Brin & Page, 1998). It is similar to eigenvector centrality which scores nodes in a graph accordingly to their number of edges. In other words, the importance of any node depends on the number of nodes that link to it. However, a PageRank value is a relative score that considers a node as more important if it has more incoming edges than outgoing ones as well as the degree of the nodes that link to them. Accurate ranking enables the search for nodes that can reveal the evolution of a digraph $S$.

The PageRank of a node $i$ is given by

$$PR(i) = d/n + (1 - d). \sum_{j \neq i \in N} PR(j) / \text{OutDegree}(j) \quad (2.2)$$
where $d$ is called the damping factor which can be set between 0 and 1, $n$ is the total number of nodes in the digraph $S$ and Outdegree is the number of outgoing edges from node $j$. In the experiment, PageRank (NodeRank) is used to rank the causes (e.g. influencing factors) and severity factors of the traffic crashes.

**Louvain Community Detection Algorithm:** Louvain algorithm is a hierarchical clustering algorithm that has been applied to detect communities of densely connected nodes within large graphs in unsupervised way (Blondel et al. 2008). It ensures that nodes that belong to different communities are sparsely connected. It also relies on a heuristic to maximize modularity. The first step is a greedy assignment of nodes to communities where each node is a single community, and then the communities are progressively merged with neighboring ones based on the local edge density evolution. The aim is to obtain communities where edge density is high while intercommunity density remains low. The modularity is a scalar $m$ integer ranging from -1 to +1 which is defined as

$$Q = \frac{1}{2m} \sum_{ij} \left( A_{ij} - \frac{k_i k_j}{2m} \right) \delta(c_i, c_j) \quad (2.3)$$

where $C_i$ and $C_j$ are the communities of nodes $i$ and $j$ respectively, $A_{ij}$, denotes the edge weight between $i$ and $j$, $k_i$ and $k_j$ are the sum of the weights of the edges attached to nodes $i$ and $j$ respectively, $2m$ is the sum of all the edge weights in the graph and $\delta$ is a delta function. The Louvain algorithm is used in the experimental study to classify traffic accident events into communities with commonalities. These commonalities can be place, time or attribute factors. For example, identifying communities of traffic accident events with common weather conditions or other influencing factors such as alcohol and distraction and the evolution of these communities over time.
2.4 Experiment

Understanding where and when a traffic accident occurs and its pattern of evolution through time and space in a road network is of major interest to traffic accident agencies and transport authorities (Moons et al., 2009). In this section, the use of the STVG framework is demonstrated to identify, rank and analyze traffic crashes, their influencing factors and evolving patterns occurring at different temporal resolutions.

**Data description:** Three types of raw data were used to build the STVG. First, the traffic accident data of Brevard County from 2010 to 2015 provided by the Florida Department of Highway Safety and Motor Vehicles. It contains the information of about 1,048,575 accidents. Second, the shape file containing the road network was also used to extract the Platial Subgraph nodes. Table 2.2 describes the attributes recorded for each traffic accident.
**Table 2.2. Attributes of the traffic accident dataset**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FID</td>
<td>Feature Identifier</td>
</tr>
<tr>
<td>Crash_DT</td>
<td>Crash date</td>
</tr>
<tr>
<td>Crash_TM</td>
<td>Crash time</td>
</tr>
<tr>
<td>Age</td>
<td>Age of the driver</td>
</tr>
<tr>
<td>Crash_HOD</td>
<td>Crash hour of the day</td>
</tr>
<tr>
<td>Crash_DOW</td>
<td>Crash day of the week</td>
</tr>
<tr>
<td>Crash_MOY</td>
<td>Crash month of the year</td>
</tr>
<tr>
<td>Crash_Y</td>
<td>Crash year</td>
</tr>
<tr>
<td>Crash_WK</td>
<td>Crash week number of the</td>
</tr>
<tr>
<td>Fatalities</td>
<td>Number of fatality</td>
</tr>
<tr>
<td>Injury</td>
<td>Injury recorded</td>
</tr>
<tr>
<td>Alcohol_Related</td>
<td>Yes/No</td>
</tr>
<tr>
<td>Distraction_Related</td>
<td>Yes/No</td>
</tr>
<tr>
<td>Weather_Condition</td>
<td>Clear/Cloudy/Rain</td>
</tr>
</tbody>
</table>

**Data pre-processing tasks:** These tasks have been used to transform the raw data set into a suitable format for a graph database. In summary, the pre-processing tasks consisted of: (a) Platial and Event nodes extraction from the shapefiles; (b) semantic annotation of traffic accidents in such a way that “LOCATED AT” edges can be generated in a graph database, (c) computation of moves and stops by splitting the streets into segments (50-m length) as nodal units of analysis; and finally, (d) sequential ordering of the accidents using the timestamps. These tasks were implemented using Python function based on the ArcPy API. In summary, a total of 3 CSV files containing the street segments, intersections and annotated traffic accident data points were generated for building the STVG in the Neo4j graph database.

**Neo4j graph database:** The STVG was implemented in the Neo4j graph database management system. Neo4j is currently the most popular native graph database widely used for graph data management and analytics. The query language in Neo4j is called Cypher which can be used to implement ad-hoc queries, graph algorithms and User Defined Functions (UDFs). Graph metrics such as PageRank, degree centralities and other user-
defined functions can be encoded within the Cypher queries. The space-time varying computation of the graph metrics used in the experiment were coded in Cypher through a pipeline of query statements. In total, 2.006 million nodes and 6.38 million edges were generated in the Neo4j graph database. The STVG creation and the data uploading process took approximately 0.5 minutes to complete in a stepwise process as follows:

Step 1: Building the time-tree: The time-tree T is built containing all the time instances of the traffic accidents dataset from 2010-2015.

Step 2: Building the Event Subgraph: The event subgraph is built by importing the annotated traffic accident tuples as Traffic Crash nodes with their respective properties including crash ID, street segment ID, intersection ID, and Sequence. The ‘NEXT” edges between consecutive Traffic Crash nodes are created using a sequential order of occurrence.

Step 3: Building the Platial Subgraph: The Platial subgraph is built by importing the intersection points and the street segments as Intersection nodes and Street Segment nodes respectively into Neo4j. This step also includes the property nodes such as intersection ID, intersection name and street segment ID, and street segment name. The ‘NEXT” edges between consecutive Platial nodes are also created using their sequential order of occurrence. This step also involved creating the LOCATED_AT edges between the Event and Platial subgraphs as illustrated in Figure 2.4 below.
Step 4: Creating the influencing and severity factors nodes: The Weather nodes, Alcohol nodes and the Distraction nodes as well as the Injury and Fatality nodes with their properties were created in the Event Subgraph as shown in Figure 2.5.

Step 5: Building the relationship between the Events nodes and the leaf nodes of the time-tree. Finally, the mapping across place-time-event of the STVG is built by creating the ‘HAPPENED_AT’ edges (Figure 2.6).
2.5 Discussion of the results

The proposed STVG is useful for many applications in which the topological relationships in the data is as important as the data itself. In this Section, the STVG is demonstrated by answering the following questions:

- What are the evolutionary patterns of traffic accidents over different time resolutions?
- Why do the traffic accidents occur?
- Are there densely connected communities of traffic accidents?

What are the evolutionary patterns of traffic accidents over different time resolutions?

The degree centrality has been utilized in a graph query to compute the frequency of traffic crash occurrences at any given street and intersection at varying temporal resolutions using the complementary connectivity between the Platial and the Event subgraphs. A sample Cypher statement used to retrieve the streets where there has been the highest occurrence of traffic accidents on a yearly basis is shown in Query 1 below. The evolutionary
patterns are illustrated in Figure 2.7. The street nodes in the Platial Subgraph with the highest values of degree centrality represent the streets where there have been more traffic accidents over the 6-year period.

Cypher query

```
WITH range (2010,2015) AS Hour
MATCH(y:Year)-[:HAS_MONTH]-(m:Month)-[:HAS_DAY]-(d:Day-[:HAS_HOUR]-:h:Hour) <- [r1: HAPPENED_AT]-(cr:Crashes)
WITH cr, y
MATCH p = (cr:Crash)-[r:NEXT*..]-(cr:Crash)
UNWIND NODES (p) [1..-1] as n
WITH n, y, NODES (p) as nodes FILTER (node in nodes WHERE (node:Street)) as Str
RETURN y.yearid, DISTINCT(str), size((str)-()-n) as degree,
toFloat(degree.y-degree.y+1) as Delta ORDER BY degree'
```
and $D_{(t_{i-1})}$ is the degree centrality at $t_{i-1}$. The results are shown in Table 2.3 where positive values of $\Delta D$ indicate an increase in the frequency of traffic crashes, and negative values of $\Delta D$ indicate a decrease in the frequency of traffic crashes. The occurrence of traffic accidents has decreased between the years of 2010 and 2011 years for most of the streets (i.e. Babcock, Eau Gallie, Harbor City, Wickham, Malabar, Emerson, Palm Bay, Atlantic, New Haven, I-95, Minton, Clearlake, Fiske, Cocoa, Dairy, and Cheney), meanwhile their occurrence escalated in the coming years as also illustrated in Figure 2.8.

Another interesting example is that I95 and Babcock road have maintained a record high every year through the 6-year period and maintained a constant positive increase through the years except in 2011 when I95 road had an 18.5% drop from the previous year (2010). Accidents on Babcock Road have dropped by 81.8%. This information can help traffic managers search for other sources of data that could potentially explain these evolutionary patterns and hopefully help them to understand what has caused the low frequency of accidents in 2011 in contrast with the high frequency of accidents in 2013.

### Table 2.3. Variations in degree centrally over the years according to the Platial Subgraph nodes.

<table>
<thead>
<tr>
<th>PLACES</th>
<th>2010</th>
<th>$\Delta D$</th>
<th>2011</th>
<th>$\Delta D$</th>
<th>2012</th>
<th>$\Delta D$</th>
<th>2013</th>
<th>$\Delta D$</th>
<th>2014</th>
<th>$\Delta D$</th>
<th>2015</th>
<th>$\Delta D$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Highway</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>195</td>
<td>973</td>
<td>-180</td>
<td>793</td>
<td>2</td>
<td>795</td>
<td>182</td>
<td>977</td>
<td>39</td>
<td>1016</td>
<td>113</td>
<td>1129</td>
<td></td>
</tr>
<tr>
<td>Highway 1</td>
<td>435</td>
<td>80</td>
<td>515</td>
<td>-8</td>
<td>507</td>
<td>17</td>
<td>524</td>
<td>52</td>
<td>576</td>
<td>32</td>
<td>608</td>
<td></td>
</tr>
<tr>
<td>Highway A1a</td>
<td>194</td>
<td>17</td>
<td>211</td>
<td>-13</td>
<td>198</td>
<td>84</td>
<td>282</td>
<td>15</td>
<td>297</td>
<td>-61</td>
<td>236</td>
<td></td>
</tr>
<tr>
<td>Road</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Babcock</td>
<td>891</td>
<td>-729</td>
<td>162</td>
<td>382</td>
<td>544</td>
<td>468</td>
<td>1012</td>
<td>149</td>
<td>1161</td>
<td>21</td>
<td>1182</td>
<td></td>
</tr>
<tr>
<td>Atlantic</td>
<td>589</td>
<td>-202</td>
<td>387</td>
<td>-101</td>
<td>286</td>
<td>299</td>
<td>585</td>
<td>23</td>
<td>608</td>
<td>163</td>
<td>771</td>
<td></td>
</tr>
<tr>
<td>Malabar</td>
<td>435</td>
<td>-357</td>
<td>78</td>
<td>179</td>
<td>257</td>
<td>279</td>
<td>536</td>
<td>111</td>
<td>647</td>
<td>-80</td>
<td>567</td>
<td></td>
</tr>
<tr>
<td>Harbor City</td>
<td>482</td>
<td>-378</td>
<td>104</td>
<td>188</td>
<td>292</td>
<td>237</td>
<td>529</td>
<td>40</td>
<td>569</td>
<td>128</td>
<td>697</td>
<td></td>
</tr>
</tbody>
</table>
Using the time-tree of STVG, the average of degree centrality can also be used to compare the evolutionary patterns over different temporal resolutions as shown in Figure 8. Three streets have been used as an example to illustrate the hourly (Figure 2.8a), weekly (Figure 2.8b) and monthly patterns (Figure 2.8c). The results are revealing the highest peaks of accidents occurring during between 12pm and 4pm, week days, as well as during the months of August and September.

<table>
<thead>
<tr>
<th>Street</th>
<th>Babcock</th>
<th>Wickham</th>
<th>Atlantic</th>
<th>Babcock</th>
<th>Wickham</th>
<th>Atlantic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cocoa</td>
<td>157</td>
<td>-68</td>
<td>89</td>
<td>35</td>
<td>124</td>
<td>117</td>
</tr>
<tr>
<td>Clearlake</td>
<td>190</td>
<td>-81</td>
<td>109</td>
<td>43</td>
<td>152</td>
<td>70</td>
</tr>
</tbody>
</table>

37
The frequency of accidents at intersections can also be visualized as an evolving graph as shown in Figure 2.9. The top 3 intersections where the accidents occur in the County are the Wickham/Sarno intersection (301 accidents), the Eau Gallie/ Harbor City intersection (276), and the Minton/ Emerson intersection (269) for the last 6 years.

![Figure 2.9. Evolving graphs for accidents at 3 intersections.](image)

**Why do the traffic accidents occur?**

The PageRank centrality was used to rank the traffic accident influencing and severity factors. The Cypher query ranks the factors based on the number of out coming edges from the Event nodes to the Influencing factor nodes (i.e. Weather, Alcohol and Distraction nodes) as well as Severity Nodes (i.e. Fatality and Injury nodes).

The results show that approximately 73% of traffic accidents in the County occurred in clear weather conditions throughout the 6-year period. However, cloudy and rainy weather conditions have been also ranked second and third respectively as the most influencing factors in the occurrence of traffic accidents (Table 2.4). Moreover, the traffic accidents
caused by distraction on the part of the drivers has maintained a steady rise from 2010 to 2015 while the influence of alcohol has dropped by 21.2% in 2011. Unfortunately, it has shown a steady rise from 2011 to 2015 with up to a 37.7% increase in 2015. The severity factor ranking indicates that approximately 98% of the traffic accidents have caused more injuries than fatality cases throughout the 6-year period.

**Table 2.4. Ranking Results**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Clear Weather</td>
<td>1672.72</td>
<td>831.11</td>
<td>1173.90</td>
<td>1523.37</td>
<td>1669.04</td>
<td>1987.19</td>
</tr>
<tr>
<td>Cloudy Weather</td>
<td>337.11</td>
<td>191.25</td>
<td>314.10</td>
<td>451.05</td>
<td>453.03</td>
<td>452.12</td>
</tr>
<tr>
<td>Rain</td>
<td>166.48</td>
<td>98.30</td>
<td>140.25</td>
<td>180.93</td>
<td>265.60</td>
<td>250.11</td>
</tr>
<tr>
<td>Distraction</td>
<td>5.569</td>
<td>95.72</td>
<td>137</td>
<td>189.83</td>
<td>249.75</td>
<td>240.49</td>
</tr>
<tr>
<td>Other</td>
<td>52.24</td>
<td>17.43</td>
<td>82.46</td>
<td>263.88</td>
<td>178.37</td>
<td>129.69</td>
</tr>
<tr>
<td>Alcohol</td>
<td>43.44</td>
<td>34.26</td>
<td>39.15</td>
<td>45.82</td>
<td>46.46</td>
<td>47.14</td>
</tr>
<tr>
<td>Fog, Smog, Smoke</td>
<td>2.7</td>
<td>3.79</td>
<td>4.53</td>
<td>3.40</td>
<td>7.8</td>
<td>8.01</td>
</tr>
<tr>
<td>Severe Crosswinds</td>
<td>0.15</td>
<td>0.28</td>
<td>1.043</td>
<td>0.7335</td>
<td>1.05</td>
<td>2.07</td>
</tr>
<tr>
<td>Blowing Sand, Soil, Dirt</td>
<td>0.15</td>
<td>0.28</td>
<td>0.405</td>
<td>0.58</td>
<td>0.81</td>
<td>0.15</td>
</tr>
<tr>
<td>Sleet/Hail/Freezing Rain</td>
<td>0.11</td>
<td>0.28</td>
<td>0.405</td>
<td>0.36</td>
<td>0.79</td>
<td>0.15</td>
</tr>
<tr>
<td><strong>Severity Factors</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Injuries</td>
<td>633.19</td>
<td>507.54</td>
<td>623.37</td>
<td>773.25</td>
<td>857.27</td>
<td>924.85</td>
</tr>
</tbody>
</table>

The topological relationships of the STVG graph can also be exploited to understand evolutionary patterns of co-relationships such as, for example, “How many ‘severe’ accidents occur on weekends across the years due to ‘distracted’ drivers?” This type of query in a GIS or relational database involves several JOIN operations over snapshots of time which are always inefficient in query run-time performance (Baas, 2012). In contrast, such a
query takes a set of few statements as illustrated below. The results are shown in Figure 2-10.

Cypher query 2

```
WITH range (2010,2015) AS Year
FOREACH(y IN Year |
MATCH(y:Year)-[:HAS_MONTH]-(m:Month)-[:HAS_DAY]-(d:Day)-
[:HAS_HOUR]-(h:Hour )<-[r1:HAPPENS_AT]-(cr:Crashes)
WITH cr
MATCH(sv:Severity_Factor)<-[HAS]-(cr:Crashes)-[:RELATED_TO]-
>(inf:Inf_Factor{Distraction:"Y"})
WHERE cr.WDay = 'Saturday' OR cr.WDay = 'Sunday'
RETURN COUNT(cr) AS Number_of_Crashes
```

![Figure 2.10. Annual patterns of severe traffic accidents occurring during the weekends due to distraction.](image)

**Figure 2.10.** Annual patterns of severe traffic accidents occurring during the weekends due to distraction.

**Are there densely connected communities of traffic accidents?**

The Louvain clustering algorithm is called within a Cypher query with the goal of detecting traffic accident communities that are densely connected to each other based on the influencing factors and the places where the accidents have occurred. These communities were computed using the nodes of the Event subgraph. The traffic accidents have formed communities as a result of their common relationship to an influencing or severity factor or
based on co-location and these community patterns that evolve over time. A sample Cypher query used to call the algorithm in the Neo4j graph database is shown below.

**Cypher query 3**

```cypher
CALL algo.louvain.stream(
  'MATCH (cr:Crashes) RETURN id(cr) as id',
  'MATCH (cr1)-[: RELATED_TO]-(inf:InF_Factor)<-[:RELATED_TO]-(cr2:Crashes)
  RETURN id(cr1) as source, id(cr2) as target',
  {graph:'cypher'})
YIELD nodeId, community
MATCH (cr:Crashes) WHERE id(cr) = nodeId
RETURN community as community id, count(*),
as communitysize, collect(cr.CrashID) as members
ORDER by communitysize DESC;
```

Figure 2.11 shows the graph visualization of the communities with commonalities in influencing factors that have emerged in 2010, 2014 and 2015. Some communities have emerged due to the accidents having an influencing factor such as the weather condition (rainy, cloudy and so on) in common meanwhile other communities have emerged by having more than one weather conditions and other influencing factors in common. Each community has at least a similarity different from that of other communities, this pattern evolves from one year to another.

![Graph Visualization of Communities](image)

**Figure 2.11.** Evolution patterns of communities.
2.6 Conclusions

The proposed STVG adopts a Whole-graph approach that reduces replication of the entire graph from one time-window to another and provides the support for time-dependent queries for uncovering evolutionary patterns at various temporal resolutions. Every place and event were represented as a node in the STVG in order to allow a seamless computation of both node-centric and edge-centric metrics using subgraphs independent of being a diagraph or a bipartite subgraph. The evolution of topological relationships is represented by eventual adjacency and complementary connectivity where nodes in each subgraph are entities in a meaningful non-metric space. Similarly, modelling every time instance as a node provides the ability to track the evolution of the whole-graph at any temporal resolution. The conceptual notions of eventual adjacency and complementary connectivity to represent topological relationships in a graph data model also provides a network structure where each node carries information about its neighbors. This is one of the main advantages of graph data models over GIS layer models.

The STVG was not devised for modelling geometric evolution of entities of a geographic space, for example, representing a polygon source node that splits into two target nodes as two new polygons. At any point in time, every node belonging to a Platial Subgraph will be linked to a subsequent node for representing evolution and dynamics of a network rather than representing a geometrical evolution of a specific node. The same reasoning was applied to modelling the Event Subgraph, where events are, at any point in time, sequentially linked to each other and subsequently linked to the leaf nodes of the time-tree. The STVG was developed to help us obtain a better understanding the evolutionary patterns of eventual adjacency between places and events from a network perspective. Moreover, complementary
connectivity is a fundamental topological relationship of the STVG because independent of the number of subgraphs being generated for a real-world network, a bipartite relationship between two subgraphs will always exist, allowing us to perform any computation of a graph metric such as shortest path and centrality.

The traffic accident data set was used to demonstrate the benefits and limitations of the STVG. We believe that STVG has a wider applicability in modelling complex real-world networks, using other types of data such as mobility data, social media data and health data. More research work is needed to study the benefits/limitations of the proposed STVG, and other data sets will provide more empirical evidence towards this end.

In the future, the research work will focus on ensemble graph metrics to quantify growth, transition and decay phases using topological relationships. We will also advance the STVG by integrating other subgraphs to the Platial and Event Subgraphs. Some examples include Mobility and Social Subgraphs. The applicability of subgraphs is expected to become more vital for edge computing where geographically distributed resources will perform tasks such as data pre-processing and subgraph creation on the fly and closer to where the data is being collected.

References


http://doi.org/10.1007/978-3-540-89930-3_7


http://doi.org/10.1145/2433396.2433479


http://doi.org/10.1068/b2752


Chapter 3 Transit Performance Assessment based on Graph Analytics

3.1 Introduction

General Transit Feed Specification (GTFS) datasets have been among the most widely used data sets for transit network analysis because they are freely available and constantly updated (Hadas, 2013), (Bast et al. 2014), Fransen et al. (2015), Farber et al. (2016)); however, they have limited utility as data are derived from planned schedules rather than actual departure and arrival times. It is not possible to use these data to understand actual deviations caused by Systematic errors in travel, departure and arrival time as well as other uncertainties caused by operational delay, service interruptions and unrealistic schedules (Wessel et al. 2016).

In contrast, real-time transit data feeds such as Automatic Vehicle Location (AVL), Automatic Passenger Counts (APC) and Automatic Fare Collection (AFC) have had much greater potential. They have been used for computing more advanced performance indicators ranging from schedule-adherence estimation (Furth et al. 2003, Reddy et al. 2009, Mandelzys and Hellinga 2009), schedule coordination planning and links rerouting (Utsunomiya et al. 2006, Kee et al. 2008, Seaborn at al. 2009), Hammerle et al. (2005) and Liao (2011) and in predicting on-time performance of transit systems (Cevallos et al. 2011).

While the volume of transit data has considerably increased, analytical techniques have remained largely consistent, typically relying on Geographic Information Systems (GIS) and relational database models such as those described in Mandelzys and Hellinga
The challenge is that there is a growing recognition that this analytical approach may be a barrier to timely analysis of real-time performance measures based upon massive amounts of transit-generated data (Fayyaz et al. 2017). Pelletier et al. (2011), for example, have argued that continuous processing of massive transit feeds available on individual trips is a limitation of relational GIS data models, reinforcing the similar conclusions by Berkow et al. (2009).

Transit agencies that are looking to discover hidden trends for transit performance may be at the limit of what can be learned from typical GIS and relational database models. Agencies have also indicated the need of new analytical approaches for continuously computing performance indicators such as schedule adherence, vehicle-kilometers, and person-kilometers for every individual run, route, or day. Ma and Wang (2014) state that analytical techniques based on GIS or relational database models are not an optimal solution for two reasons: First, GIS and relational database models lack a generic network structure capable of integrating massive transit feeds; and second, the slow performance of GIS and relational database models in computing transit performance indicators hampers their use in processing massive transit feeds.

Graph data models have become the state-of-the-art for analyzing massive volume data to uncover hidden patterns and offer several advantages when compared with a relational database approach (Angles & Gutierrez, 2017). Graph data models fundamentally unify connectivity with observation in any massive dataset that can be represented in a network structure. Graph databases have at least an edge between two entities (nodes) and each node carries the information about its neighbors with no existence of any global index between nodes. Graph data models are also very scalable irrespective of data size such that
a traversal of an edge from one node to another is independent of the size of data and can run in constant time. This ensures that network analysis can be performed on graph databases (when compared to relational databases) and is suitable for cases where data size increases rapidly (De Virgilio et al. 2014). Table 3.1 summaries the advantages and disadvantages of graph models over GIS and relational database models using the concepts of data scalability, network structure, network metrics, and in-memory capabilities.

Table 3.1. Comparison between GIS, Relational Database and Graph Database Models.

<table>
<thead>
<tr>
<th>Concept</th>
<th>GIS Model</th>
<th>Relational Data Model</th>
<th>Graph Data Model</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network structure creation</td>
<td>Slow (topology is based on points and lines)</td>
<td>Not available</td>
<td>Fast (topology is based on nodes and edges)</td>
<td>Miler et al. (2014) Baas (2012) Curtin (2007)</td>
</tr>
<tr>
<td>Network Metrics (e.g. shortest path, centrality, page ranking)</td>
<td>Limited (only shortest path is available)</td>
<td>Limited and inefficient (programming scripts are needed to compute network metrics)</td>
<td>Efficient (network metrics are embedded in graph queries)</td>
<td>Miler et al. (2014) Baas (2012) Vicknair et al. (2010) Curtin (2007)</td>
</tr>
</tbody>
</table>

This Chapter presents an STVG capable of supporting the computation of operational-level performance indicators based on graph metrics on a large volume of transit data feeds. The Chapter also demonstrates how “graph queries” can be used to compute operational-level performance indicators such as schedule adherence, bus stop activity and bus route activity ranking for a transit network.
3.2 Background

The use of graph data models and analysis of connectivity have been of interest in transportation (Calatayud et al. 2016). Two traditional types of network structures have been developed to represent the connectivity of a transit network (Añez et al. 1996). First, the primal graph model represents a network structure using a geographic space where transit routes are edges (links) and transit stops are nodes (vertices). The primal graph model was used in Hossain et al. (2013) for representing flight connections as nodes and the average number of monthly flights as links, building a mesh network structure. The degree and betweenness centrality metrics were used to estimate the resilience of Australian air transit network, which was a relatively small graph with a maximum of 1000 flights and 100 connections. These centrality metrics provide a relative measure of importance of any given node in the network, detail discussion of these metrics can be found in Section 3.3. Similar work was carried out by Wang et al. (2011) building a primal graph model where the airports are represented as nodes and their flight routes are represented as links. Using the Chinese air transit network, the degree centrality of an airport (city) was computed based on the number of connections from it to other airports. Finally, Kuzkin (2016) utilized a primal graph data model to formalize the representation of the connectivity between the geographic space of urban electric transport in large cities in Ukraine, where transit stops were represented nodes and routes were represented as edges. Graph metrics such as the average degree of nodes, the average shortest paths, and clustering coefficients were used to help transit managers to understand transit network development; however, they were not used to compute transit performance indicators.
Derrible and Kennedy (2011) and Derrible (2012) analyzed ridership of a metro transit system based on graph connectivity metrics for forecasting ridership increase or decrease associated with opening of new metro lines or closing existing lines in the network. These two studies were based on using the primal graph model for representing the connectivity between geographic space of the network: transit stops (nodes) and transit routes (edges). Centrality metrics have been primarily utilized to identify which transit stops and their respective transit routes are in need of immediate attention for prioritizing investment in terms of maintenance and operations, in particular in multi-modal transit networks (Sarker et al. 2014), Mishra et al. 2012). A similar approach has been utilized in Welch and Mishra (2013) to measure transit equity integrating routes, schedules, socio-economic, demographic and spatial activity patterns as well as in transit connectivity visualization (Mishra et al. 2014).

Tsekeris and Souliotou (2014) proposed the use of different graph metrics including average degree centrality for computing the level of importance of a transit stop, betweenness centrality for computing a ratio between transfer stops to total stops, and the average path length and reduced network diameter to provide new insights for adjusting transit planning priorities. Their work was based on a primal graph model that was used to represent the geographic space of a transit network having transit stops as nodes and transit lines as edges connecting these nodes. This research work has not drawn transit performance indicators from graph metrics, but it has definitely paved the way for future research in this direction.

Recently, Wu et al. (2018) have introduced a new statistical metric named “node occupying probability” using primal graph models. This metric differs from betweenness centrality, network diameter and closeness centrality because these graph metrics are
computed based on shortest path routing which ignores the inherent characteristics of a transit network. Following this approach, X. Wang et al. (2017) have proposed the robustness performance indicator previously defined by Derrible and Kennedy (2010) for quantifying the robustness of a transit network in terms of the number of alternative paths in the network structure divided by the total number of the stops in the network. They have used this robustness performance indicator to compare the network design of 33 metro-systems worldwide based on a geographic space where the metro stops were nodes and metro lines were edges.

The second traditional type of network structure is known as dual graph models (Añez et al. 1996). In this case, nodes are used to represent transit routes and edges are used to represent the transit stops. However, the computation of graph metrics, especially the centrality metrics, would be ineffective, hindering their use for computing transit performance indicators.

The graph data models described above attempted to utilize graph metrics as indicators for transit system performance relying only on a geographic space with no representation of a mobility space which could have been obtained from AVL data sets. The challenge is that no graph data model has been developed that specifically applies graph metrics to data from AVL data from transit, while accounting for the massive amount of data that needs to be analyzed quickly. The studies have focused only on the network structure, the centrality metrics between transit lines and transit stops in the network. The changes of these metrics over time as a function of evolving mobility spaces on the network and the potential effects on the network itself have not been studied yet.
There is no account of the effects of a mobility space on the performance indicators of these transit networks, but it is expected that these effects could be better understood through a graph data model that encompasses both geographic and mobility spaces. For example, it should be possible to use the AVL feeds of each vehicle of a transit network to develop time-dependent paths. This can quickly identify the shortest and longest trips by an individual transit vehicle or a transit line within an entire transit network, and associate this variability to a cause, such as weather, traffic delay or congestion by time of day, and route disruptions such as accidents. Mishra et al. (2012) emphasize the role of graph metrics in assisting decision makers in prioritizing transit investments and deciding which transit stops and routes need immediate attention in regard to operation and maintenance, especially in rural and suburban regions where there is a scarcity of information on transit ridership, boarding and alighting. There is precedent for using AVL/AFC/APC data sets in transit system analysis, and there is precedent for using graph metrics for transit and transportation network analysis, but there does not yet appear to be an approach that combines the two approaches.

The research aims at developing an STVG framework that accounts for the Mobility and Platial spaces of transit networks yet is capable of processing a large volume of AVL data feeds. This is utilized in a case study of two weeks of AVL data feeds of Codiac Transpo, the transit agency serving metropolitan Greater Moncton in New Brunswick, Canada.

3.3 The Space-Time-Varying Graph (STVG)

We propose an STVG that accounts for the mobility and geographic spaces of transit networks yet is capable of processing a large volume of AVL data feeds. The STVG for the
dynamic transit network can be described as a Graph $G = (S, T)$, where $S = (N, E)$ is a digraph and $T = (N, E)$ is a hierarchical tree graph such that $N$ is a set of nodes and $E$ is a set of edges described in Figure 3.1. $S$ is composed of subgraphs and each subgraph is a subset of the $G$. Subgraphs can be used to facilitate the conceptual modelling of the evolution of the topological relationships where nodes in each subgraph are entities in the mobility and geographic space of the transit network. Multiple subgraphs $S'$ can be conceptualized such that $S = (S'_1, S'_2, ... S'_n)$ representing different spaces. However, only one subgraph $S'_i$ is connected to $T$ which is proposed to represent the natural levels of a timestamp in the transit feeds (Ivarsson et al. 2014). Essentially, $T$ is a hierarchical (tree) structure where a root node is followed by year nodes, month nodes, day nodes, and further temporal resolutions. Each node of this subgraph $S'_i$ must be linked to a leaf-node of $T$. This is an important constraint to support any kind of time-dependent queries and optimal processing time. As a result, the graph model enables the fast computation of graph metrics, avoiding the limitations accruable to the static graphs such as Dual and Primal graph models (Añez et al.1996).

Nodes belonging to the same subgraph $S'_i \subset S$ are linked by the “NEXT” edges between an existing node (source node) and a future node (target node) to be created in a graph database. These edges can be weighted using an associated numerical value that is a non-negative integer obtained by computing, for example, the Euclidean distance between the source and target nodes if they represent two locations of a moving vehicle. The weights can also be computed using the elapsed time between them. Nodes belonging to different subgraphs $S'_i \subset S$ and $S'_j \subset S$ are linked by connectivity edges since, for every node in a subgraph, there is always a corresponding node in another subgraph it is related to.

All nodes belonging to a subgraph $S'_i \subset S$ have labels and properties. Labels assign
roles (e.g. “Stop” representing the state of a moving object) and geographic entity (e.g. “Street” and “Intersection”). Graph database queries can be used to search a set of labels of a subgraph $S'_i$, instead of the entire graph $S$. Finally, multiple properties can also be attached to the nodes and edges of a subgraph $S'_i$. They are attributes ($a_1 ... a_n$) containing semantic information about the entities, spatial data types (e.g. points), and temporal types (e.g. duration of an event).

Figure 3.1. Overview of the proposed STVG.

3.3.1 Subgraphs

The mobility subgraph is used to represent the mobility space of the dynamic transit network. In this subgraph, the primary concept is a “Trip” that represents the trajectory of a bus vehicle belonging to a bus route of a transit network. In the graph model, a “Trip” consists of nodes that are known as “Origin” (i.e. a node representing the first location of a
trip), “Halt” (i.e. a node representing the location where the bus is not moving), “Move” (i.e. a node representing the location where a bus is moving), and “Destination” (i.e. a node representing the last location of a trip). The connectivity between these nodes is represented by the “NEXT” relationship such as *Origin-Move, Halt-Halt, Halt-Move, Halt-Destination, Move-Destination, and Move-Halt.*

These nodes are computed using the AVL feeds that are streamed into the graph database using the algorithm developed by Cao and Wachowicz (2017). The algorithm is designed to transform the actual location observations (i.e. the GPS coordinates from the AVL feeds) of a bus vehicle into a sequence of nodes such as “Moves” and “Halts” using a set of automated tasks described in Section 3.2. In other words, these nodes in mobility subgraphs are created according to the actual movement of a bus which will have an impact on the computation of the performance indicators.

In contrast, the Platial subgraph is used to represent the geographic space which consists of physical geographic places that represent a “Trip” of a bus vehicle belonging to transit network. In this case, a “Trip” is represented as a bus route that consists of a static “Bus Stop” (i.e. a designated place for passengers to board or alight from a bus), “Street Segment” (i.e. the transit segment between two bus stops belonging to a bus route), and “Street Intersection” (e.g. existing 3-way intersections or 4-way intersections of a bus route). More geographic places can be defined depending on the availability of transit service data.

The graph model is designed to transform the actual location observations (GPS coordinates) of a moving bus into a sequence of nodes such as “Bus Stop”, “Street Segment”, and “Street Intersection” using a set of automated tasks described in Section 4.4.2. In other words, the nodes represent geographic places in the graph model that are created according
to the actual observations of the movement of a bus. The edges represent the “NEXT” relationship between two consecutive geographic places of a trip.

It is important to point out that these two subgraphs are complementary to each other in the network. Their relationship is complementary because for each node in the Mobility subgraph there will be always an equivalent node in the Platial subgraph. This is represented by the “LOCATED_AT” relationship that exists between, for example, an “Origin” node (Mobility subgraph) and “Bus Stop” node (Platial subgraph), or a “Halt” node (Mobility Context) and a “Bus Stop” node (Geographic Context). In terms of similarities, they build a hybrid topological structure where connectivity is always represented by the “NEXT” and “LOCATED_AT” relationships between nodes which represent any entity of a transit network.

Independently of the space being represented in the graph model, all nodes are connected to a Time subgraph (Time-tree) using the temporal relationship named “HAPPENS_AT” that plays an important role to represent the time dimension in the graph model. In the graph model, the time subgraph is represented as a time-tree and not as timestamps property associated with the nodes or edges (See Figure 3.2 below). Every node of the mobility subgraph is connected to leaf node of the time subgraph using the timestamps. The time-tree is a hierarchical multi-level temporal indexing structure representing the natural levels of a timestamp (Ivarsson et al. 2014). It consists of a “Root” node that connects all the “Year” nodes, and in turn, each “Year” node connects to the twelve “Month” nodes of a year, and so on.
Figure 3.2. The nodes in the mobility space establish the connection to the time-tree.

The hierarchical temporal indexing structure makes time-dependent graph queries to return query results very fast. A time-dependent query firstly arrives at the time-instant node or range of time-instant nodes of the query and traverses to all events or nodes that are linked to it without having to scan through all the nodes of the large graph.

3.3.2 Relevant graph metrics for the transit network analysis

One of the advantages of the proposed graph data model over the existing transit graph models is that one can use a user-specified query that can provide information about the operational performance of a transit network using centrality graph metrics. They are explained in more detail below.

Degree Centrality: is the total number of edges a node has with other nodes in the network. The higher the number of edges the more influential the node is in the network. The degree of a node $i$ is given by $C_D(i) = \sum_{j=1}^{n} a_{ij}$ where, element $a_{ij} = 1$ when there is a direct link between nodes $i$ and $j$ and $a_{ij} = 0$, when there is no link. For example, a graph query using degree centrality is proposed to be used to compute bus stop activity. The highest the degree value will indicate the highest bus stop activity.
PageRank Centrality: considers nodes as more important if they have more incoming edges than outgoing ones as well as the importance of the nodes that link to them. The Page Rank of a node $i$ is given by

$$PR(i) = \frac{d}{n} + (1 - d) \sum_{j \neq i \in N} \frac{PR(j)}{OutDegree(j)}$$

where “d” is called the damping factor which can be set between 0 and 1, $n$ is the total number of nodes in the graph and OutDegree is the number of outgoing edges from node $j$. This centrality measure is similar to eigenvector centrality, which scores nodes in the network according to the number of connections to high-degree nodes (Brin and Page 1998). PageRank and Degree centrality metrics are used to quantify transit activities at the bus stops and bus routes, stop frequency and so on, which are usually carried out by rigorous field surveys (Finch et al., 2013).

Edge path metrics: are one of the most common metrics in a graph data model. In transit, it is a path depending on how much time it takes a moving bus to travel from node to node. For example, the average shortest path length of a trip from an origin to a destination can be defined as:

$$\langle d \rangle = \frac{1}{N(N-1)} \sum_{i<j \in N} d_{ij}, \text{ where } d_{ij} \text{ is the number of edges for the shortest path from node } i \text{ to node } j.$$ A user-specified query identifies the shortest and longest paths from a source node to a target node using the weighted edges based on the Dijkstra’s shortest-path from source node $i$ to target node $j$ ($w$, $i$ and $j$ are temporally valid at time $t_i$), while all edges in $p$ are valid at time point $t_i$ (Dijkstra, 1959). When the source and target nodes are the origin and destination of a moving bus, it is possible to retrieve all the trips with the shortest and longest durations. This information is used to evaluate the schedule adherence performance of each trip on an hourly, daily, monthly, and yearly basis. An edge path metric can also be
used for computing transit accessibility indicators since they require the calculation of travel
time and distance traveled across all bus stops pairs at any given time-of-day. This approach
offers a considerable improvement over relational database models, which Fayyaz et al.
(2017) identify as computationally expensive and time consuming for computing all shortest
paths on origin-destination (OD) pairs.

Table 3.2 summarizes some examples of operational-level performance indicators
that illustrate the potential of these graph metrics using only AVL data sets in the absence of
AFC and APC data sets.

<table>
<thead>
<tr>
<th>Graph Metrics</th>
<th>Definition</th>
<th>Transit Performance Indicator</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edge Path</td>
<td>The length of an ordered sequences of nodes from origin node to destination node.</td>
<td>Schedule Adherence</td>
<td>The level of efficiency of the service for maintaining the published schedule (Cevallos et al., 2011).</td>
</tr>
<tr>
<td>Node Degree</td>
<td>Total number of edges a node has with other nodes in the network.</td>
<td>Bus stops and Routes Activity Index</td>
<td>Overall transit activity at any given stop or route at a given timestamp or period of time. (Finch et al., 2013)</td>
</tr>
<tr>
<td>Node PageRank</td>
<td>Ranks nodes in a network according to their degrees and connection to high degree nodes.</td>
<td>Bus stops and Routes Activity Index</td>
<td>A measure used to gauge overall activity at a bus stop or route compared to activity among bus stops and route across the transit system at a given timestamp or period of time (Finch et al., 2013).</td>
</tr>
</tbody>
</table>

3.4 Application of the proposed STVG to transit AVL data

We applied the graph model to data obtained from real-world AVL transit data in
order to explore the new types of performance measures generated through this analysis.
Two weeks of AVL transit feed data, representing 631 bus stops and 206 trips, were provided by Codiac Transpo, the bus-based transit provider in Greater Moncton, New Brunswick, Canada.

### 3.4.1 Datasets

The AVL feeds were pulled from the buses every 5 seconds into a PostgreSQL/PostGIS database. In addition, freely available GIS datasets containing information about the bus routes, Timestamp of the data point stop locations, street segments, street intersections locations and civic addresses were obtained from GeoNB online service (http://www.snb.ca/geonb1/e/index-E.asp). Figure 3.3 provides a map overview of the CODIAC transit network.

![Figure 3.3. The CODIAC Transit Network.](image-url)
3.4.2 Automated Pre-Processing Steps

The preprocessing consists of six automated steps:

Step 1: Categorization of Halts/Moves: In this task, a fixed distance value between two consecutive points is used for determining “Halts” and “Moves”. This value was empirically determined as being 15-m for a transit network. If the distance between the previous point and the current point is more than 15m, the current point is annotated as a “Move”. In contrast, if the distance is less than 15m, the current point is annotated as a “Halt”. These distances are Euclidian distances; network distance could be considered but for a short distance of 15m, Euclidian distance can be used for this purpose.

Step 2: Street Name Annotation. The objective of this task is to annotate the “Moves” and “Halts” according to the streets nomenclature of a transit network. For this task, a query on PostgreSQL database is run to retrieve the names of the streets where a “Move” or “Halt” is located. This is a non-trivial step because the GPS coordinates usually have 10m of accuracy in urban areas (Salarian, Manavella, and Ansari, 2015).

Step 3: Stop Identification. The objective of this task is to associate a “Bus Stop” node of the Geographic Context with a “Halt” node of the Mobility Context. It is important to point out that the algorithm also needs to verify the direction of a moving bus (e.g. eastbound and westbound) in order to identify the “Bus Stop” that a “Halt” is actually located.

Step 4: Street Intersection Identification. Here the street intersections are tagged to each GPS coordinate. The algorithm creates a circular zone with a radius of 30m (determined empirically) for each street intersection. The “Halts” and “Moves” that are located inside the circular zone are tagged with the intersection identifier.
Step 5: Origin/Destination Trip Identification. During this step, the algorithm identifies the origin and destination of a trip. We tag each first AVL feed of a bus trip as origin, and each last AVL feed of a bus trip as destination. The remaining AVL feeds are then sequentially indexed.

Step 6: Bus Trip Categorization. Bus trips are categorized in terms of a (Route Number-Run Number-Run-day), where (50-12-9) would represent Route #50 during the 12th run of the day and 9 is 9th day of the month. The date had to be concatenated to the trip number, so the trips could have unique ids in the database.

It is important to point out that these preprocessing tasks might vary accordingly to the type of data being used to build the graph data model. The steps were implemented using a Python algorithm and they have been developed by Cao and Wachowicz (2019). At the end of this preprocessing process, a CSV file is automatically generated containing all the data needed for the STVG implementation in Neo4j graph database.

3.4.3 Building the Graph Database

The STVG was implemented in the Neo4j graph database management system. Neo4j is currently the most popular native graph database widely used for graph data management and analytics. The query language in Neo4j is called Cypher, which can be used to implement ad-hoc queries, graph algorithms and User Defined Functions (UDFs). Graph metrics such as PageRank, degree centralities and other user-defined functions can be encoded within the Cypher queries. The computation of the various graph metrics used in this case study were encoded in Cypher through a pipeline of query statements as shown in the PageRank example.
Table 3.3. The two weeks of AVL data stream generated 4.5 million nodes and edges and in the Neo4j database.

Table 3.3. Pseudo code of the PageRank cypher query.

```
WITH range (6,23) AS Hour

Compute the PageRank for every hour incrementally
FOREACH (hour IN Hour |
CALL PageRank (){/
Write the values to the nodes as their property ‘name’
Write: True,
Property: ‘PageRank’,
Find the “halts” nodes which are located at the Bus stops
(Node:'MATCH (bs:Stops) RETURN id(bs) AS id',
Relationship:'MATCH (d: Day) -[: HAS_HOUR] -(h:Hour)< -[r1: HAPPENS_AT] -(st:Halts)
WITH st
MATCH (st)-[r]-(bs: Stop)}
Return the values of PageRank as the following node definition
RETURN id(st) AS source, id(bs) AS target, count (*) AS PageRank ORDER BY PageRank'}))
```

The STVG creation and the data uploading process took approximately less than 2 minutes to complete in a stepwise process as follows:

**Step 1:** *Building the time-tree subgraph:* The time-tree T is built containing all the time instances in the 2-weeks AVL dataset.

**Step 2:** *Building the Mobility Subgraph:* The Mobility subgraph is built by importing the pre-processed AVL feeds as “Moves” and “Halts” nodes with their respective properties. The ‘NEXT’ edges between consecutive Mobility subgraph nodes are also created using their sequential order of occurrence.
Step 3: Building the Platial Subgraph: The Platial subgraph is built by importing Bus stops, Intersections and the street segments as Bus stop, Intersection and Street Segment nodes respectively into Neo4j. This step also includes the property nodes such as BusStopID, Intersection ID, intersection name and street segment ID, and street segment name. The ‘NEXT” edges between consecutive Platial nodes are also created using their sequential order of occurrence. This step also involved creating the LOCATED_AT edges between the Mobility and Platial subgraphs.

Step 4: Building the connectivity between the Mobility nodes and the leaf nodes of the time-tree. Finally, the mapping across place-time-mobility is built by creating the ‘HAPPENED_AT’ edges that establishes the connectivity between the tree and mobility subgraph at various level of time granularity. In total, 1,057,018 edges were created for “HAPPEN_AT” relationship in the database.

3.5 Discussion of Results

The shortest and longest path metrics were calculated using the graph model which permitted a calculation of bus route running time at the network level. This was compared to GTFS data to assess schedule adherence over the course of two weeks (See Table 3.4 for an overview). The results have shown that 10% of the bus routes were completed on time and the majority 80% of bus routes have a range of dispersion between 5 minutes to up 31 minutes. This indicates a service fluctuation due to buses departing early or arriving late too frequently. This implies that the actual schedule (entire trip run) is not meeting the planned schedule along the majority of the bus routes of the transit network, although it could also indicate an unreasonable performance threshold of the planned schedule. This is probably
the case of bus routes 51 and 52 that presented infeasible performance values which need to be closely monitored over time in order to determine the causes of the delays that could range from weather conditions, congestion and traffic accidents.

It is also interesting to point out that path graph metrics can also be used to identify bus routes with similar adherence performances independently from being complementary service areas or having different service zones (residential, commercial, or industrial). Some examples of bus lines having similar performance levels serving different service zones can be described as follows:

- Bus routes 64 and 94 have similar adherence performance despite the fact that bus route 64 is serving the downtown area of Moncton and bus route 94 is serving a residential area in Dieppe.

- Bus routes 63 and 95 are serving an industrial area in Moncton and providing access to the St-Anselme Rotary Park in Dieppe respectively. They exhibit similar adherence performance over the two weeks.

- Bus route 50 is a circular line in Moncton and bus route 68 serves the rural community along the Salisbury road, with both having the same performance level.

- Bus routes 62 and 71 are serving two residential areas in Moncton and possess the same adherence performance.

As for the complementary services, the bus routes 60 and 67 have similar performance levels in contrast to the bus routes 50 and 52 where the schedule adherence of bus line 51 was very poor over 2 weeks. The analysis from shortest trip and longest trip running time computations shows that bus route 51 and 52 do not adhere to schedule in opposite directions as shown in Table 3.4. Route 50 runs too early while 52 runs too late.
Querying using both the geographic and the mobility spaces of the graph data model has provided more insights about the schedule adherence performance of a specific bus route. The graph analytics revealed that the bus route 52 might need schedule adjustments for handling headways between trips, especially during the peak periods. Bus route 52 is a complementary circular line providing access to the Main Street in Moncton and connection to most of the other bus routes. Figure 3.4 illustrates the different trip patterns of the mobility space that have emerged from its shortest and longest trips at peak hours (8am and 6pm) and at 4pm. The results show how the 6pm-longest trip and the 4pm-longest trip have a similar
number of moves (i.e. green nodes in the graph) and halts (i.e. red nodes in the graph) but present a discrepancy of 10 minutes in terms of schedule adherence. In contrast, the 8am-longest trip has a significant difference in the number of halts and moves in comparison to the 6pm-longest trip, but in this case, both trips present the same performance. Moreover, the shortest-trips are showing similar patterns of moves and halts for the 8am-shortest trip, 4pm-shortest trip and 6pm-shortest trip despite them having staggered trip times of service ranging from 14 minutes up to 18 minutes, which can make it difficult for passengers to use this bus route as a shared service.
Figure 3.4. Different trip patterns of the bus route 52 from a mobility space perspective.

Figure 3.5 is a graph visualization of the trip connectivity pattern using the nodes from the geographic space and the mobility space of the transit network made by a bus trip (52-49) from its origin to its destination at 8 am in the morning across 2 consecutive days. Red nodes represent halts, green nodes represent moves, blue nodes represent streets, grey nodes represent bus stops, and pink and purple nodes represent the origin and destination of
the trip. This pattern is constantly changing based on the overall traffic situation in the route. This graph visualization is the result of a Cypher query that makes it easier for transit managers to understand the trip behavior overtime and uncover the reasons for varying performance in schedule adherence. The visualization shows the halts that were made on the bus stops, the streets and at intersections at various points in time. Where and when the trip experienced some halts due to traffic congestion can also be spotted from the trip pattern connectivity visualization. For example, the trip on the 9th of June 2016 experienced more halts (suspension of movement) on Main Street than on the previous day, 8th. Transit managers could visualize the dynamic behavior of this trip over time in addition to the attribute information of these entities. Integrating structural information and observational values of the graph entities exposes more insight about the dynamics of performance analysis of the network.

Figure 3.5. Observed patterns of trip 52-49 across two consecutive days.
Finally, it is interesting to analyze the results generated by ranking the bus lines according to their stop activities as shown in Table 3.5. Transit managers can decide based on this information which bus lines to merge or discard to reduce less busy routes or to optimize ridership. The bus routes 51, 52 and 94 have the bus stops with the highest activity values and they are the most connected within the network. Bus route 95 has a very small coverage area and low activity level as shown Table 3.7. This bus route can be removed and added the coverage to route 94. This reduces the cost of running the route and brings route 94 to activity levels similar to those of route 51 and 52. Bus route 71 and 70 can be merged to become a single route to increase performance. The two routes are within a coverage area and do have intersections that could form the merger. Bus route 66 is the lowest in terms of activity ranking but cannot be removed because it covers the outer part of the city.
Table 3.5. Ranking the bus lines according to their centrality values.

<table>
<thead>
<tr>
<th>Bus Line</th>
<th>PageRank</th>
<th>Degree</th>
</tr>
</thead>
<tbody>
<tr>
<td>51</td>
<td>5.832</td>
<td>206</td>
</tr>
<tr>
<td>52</td>
<td>5.203</td>
<td>186</td>
</tr>
<tr>
<td>94</td>
<td>3.556</td>
<td>116</td>
</tr>
<tr>
<td>63</td>
<td>2.876</td>
<td>101</td>
</tr>
<tr>
<td>64</td>
<td>2.876</td>
<td>101</td>
</tr>
<tr>
<td>62</td>
<td>2.813</td>
<td>98</td>
</tr>
<tr>
<td>61</td>
<td>2.799</td>
<td>97</td>
</tr>
<tr>
<td>60</td>
<td>2.589</td>
<td>88</td>
</tr>
<tr>
<td>61B</td>
<td>2.075</td>
<td>69</td>
</tr>
<tr>
<td>93</td>
<td>2.016</td>
<td>68</td>
</tr>
<tr>
<td>50</td>
<td>1.927</td>
<td>64</td>
</tr>
<tr>
<td>95</td>
<td>1.876</td>
<td>42</td>
</tr>
<tr>
<td>67</td>
<td>1.315</td>
<td>41</td>
</tr>
<tr>
<td>68</td>
<td>1.257</td>
<td>39</td>
</tr>
<tr>
<td>81</td>
<td>1.206</td>
<td>39</td>
</tr>
<tr>
<td>71</td>
<td>1.181</td>
<td>38</td>
</tr>
<tr>
<td>70</td>
<td>1.136</td>
<td>33</td>
</tr>
<tr>
<td>80</td>
<td>0.83</td>
<td>25</td>
</tr>
<tr>
<td>66</td>
<td>0.441</td>
<td>11</td>
</tr>
</tbody>
</table>

The transit network has in total 631 bus stops and 206 total trips for the 2 weeks of data collection, showing an average degree centrality of 197.60. The degree of centrality utilizes both in-edges and out-edges for its computation. Table 3.6 enlists the top-20 bus stops according to the values of degree centrality. The higher the degree is, the higher the probability of a stop having a high transit activity based on the network connectivity. In the sample of 2 weeks, three bus stops have shown the strongest transit activity. They were: (a) Plaza Boulevard serving as a connector for 6 bus routes, (b) Champlain Place serving as a connector for 8 bus routes, and (c) 1111 Main Street serving as a connector for 5 bus routes.
It is expected that the activity patterns will be constantly changing based on the high/low passenger activity (on/off).

Table 3.6. Ranking the stops according to their centrality values.

<table>
<thead>
<tr>
<th>Stop</th>
<th>PageRank</th>
<th>Degree</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plaza Blvd (Walmart)</td>
<td>1072.053</td>
<td>16836</td>
</tr>
<tr>
<td>Champlain Place</td>
<td>1065.031</td>
<td>16110</td>
</tr>
<tr>
<td>1111 Main</td>
<td>969.563</td>
<td>14194</td>
</tr>
<tr>
<td>54 Highfield</td>
<td>613.454</td>
<td>1587</td>
</tr>
<tr>
<td>77 Weldon</td>
<td>346.816</td>
<td>1542</td>
</tr>
<tr>
<td>Riverview Place</td>
<td>311.011</td>
<td>1129</td>
</tr>
<tr>
<td>655 Main (City Hall)</td>
<td>218.35</td>
<td>990</td>
</tr>
<tr>
<td>1045 Main (Highfield Side)</td>
<td>217.395</td>
<td>984</td>
</tr>
<tr>
<td>353 St George at Weldon</td>
<td>217.394</td>
<td>967</td>
</tr>
<tr>
<td>269 Weldon</td>
<td>213.547</td>
<td>967</td>
</tr>
<tr>
<td>859 Main at Lutz</td>
<td>210.402</td>
<td>909</td>
</tr>
<tr>
<td>135 MacBeath</td>
<td>209.756</td>
<td>884</td>
</tr>
<tr>
<td>357 St George at Weldon</td>
<td>201.156</td>
<td>862</td>
</tr>
</tbody>
</table>

Figure 3.6 shows the hourly deviations of transit activity for the same top-3 bus stops. Different activity patterns have emerged due to the variation of the degree centrality values. The lowest transit activity of the Plaza Boulevard bus stop occurred at 3pm when the Champlain Place has shown the highest transit activity. The transit activity at the 1111 Main only exceeds the Plaza Boulevard’s activity at 7pm and 10pm. This implies the need for understanding in detail the periodic evolution at the right temporal granularity (e.g. day or hour) during a long period of time (for example, a year). This insight can be used for supporting effective decisions about stops for, example in prioritizing and channeling resources appropriately and timely.
Finally, for a period of 2 weeks, the stops in Table 3.7 have recorded close to zero activity. These bus stops also require attention and re-evaluation of their positions in the network to achieve overall better transit activity performance within the network. This provides useful insight on low demand for non-time point stops that probably require optimization or repositioning to achieve higher activity and performance of the overall system.

Table 3.7. Bus Stops showing the lowest activity levels.

<table>
<thead>
<tr>
<th>Stop</th>
<th>PageRank</th>
<th>Degree</th>
</tr>
</thead>
<tbody>
<tr>
<td>312 Amirault (Irving)</td>
<td>0.15</td>
<td>1</td>
</tr>
<tr>
<td>505 MacNaughton</td>
<td>0.15</td>
<td>1</td>
</tr>
<tr>
<td>Cedarwood at Cypress</td>
<td>0.15</td>
<td>1</td>
</tr>
<tr>
<td>Southpine at Burgandy</td>
<td>0.15</td>
<td>1</td>
</tr>
<tr>
<td>1010 St. George</td>
<td>0.15</td>
<td>1</td>
</tr>
<tr>
<td>Across 180 Mapleton Rd</td>
<td>0.15</td>
<td>1</td>
</tr>
<tr>
<td>526 Salisbury</td>
<td>0.15</td>
<td>1</td>
</tr>
<tr>
<td>411 Chartersville</td>
<td>0.15</td>
<td>1</td>
</tr>
<tr>
<td>333 Chartersville</td>
<td>0.15</td>
<td>1</td>
</tr>
<tr>
<td>215 English Dr</td>
<td>0.15</td>
<td>1</td>
</tr>
</tbody>
</table>
3.6 Conclusions

Previous studies that utilized graph models approaches to compute transit performance have focused on primal graph models with emphasis only on the geographic space of a transit network. In this Chapter, STVG accounts for the mobility and geographic spaces of transit networks yet is capable of processing a large volume of AVL data feeds. We have implemented the STVG in Neo4j graph database to store the AVL data feeds and use the Cypher query to retrieve information about the operational level performance of a transit network. The STVG provides new insights about trip patterns according to geographic and mobility spaces.

We explored the effectiveness of the proposed STVG by using the AVL transit feeds generated by the public bus transit system in the City of Moncton, New Brunswick, Canada. The preliminary results have successfully demonstrated how graph metrics can be used to compute transit performance indicators such as schedule adherence, bus stop activity, and bus route activity ranking that are necessary for transit planning in terms of resource prioritization and ridership increase. Transit managers can also identify situations where bus lines need to be merged or discarded to reduce less trip activity and increase overall performance of a transit system. Other important transit performance indicators such as headway regularity and accessibility can be computed using the STVG, but this will require the availability of AFC and APC datasets. It is important to point out that the proposed STVG is not restricted to the case study but can be utilized for any transit network. The second takeaway from the study is the scalability in terms of large data management and processing of massive transit AVL feeds. This was practically impossible using previous primal graph model and technologies such as GIS and relational databases. The next phase of this study
includes scaling up the graph database using more than 1 year of AVL data feeds. The goal is to uncover the evolution pattern of the graph metrics and how they affect the behavior of the overall transit system over time.

References


http://doi.org/10.1016/0191-2615(95)00024-0


http://doi.org/10.1.1.109.4049


http://doi.org/10.1016/j.jtrangeo.2015.09.008


http://doi.org/10.1016/j.jtrangeo.2013.09.015


http://doi.org/http://dx.doi.org/10.3141/2121-06

http://doi.org/10.1016/j.tranpol.2014.01.005


Chapter 4

STVG: An Evolutionary Graph Framework for Analyzing Fast-evolving Networks

4.1 Introduction

Evolutionary graph analytics have attracted attention from many research communities with the main purpose of understanding the changing pattern of real-world networks through evolutionary analysis of graph metrics and dynamic interactions between entities. Graphs of real-world networks evolve as new nodes and edges continually appear and disappear in the structure but, more importantly, their metrics such as density, average path length and network diameter also evolve. Uncovering and understanding hidden patterns in an evolving network requires evolutionary analysis of the network over different temporal resolutions.

Subbian, 2014). For example, in transit and mobility networks where real-time monitoring and analysis may be required, nodes and edges are added to their graph on the temporal resolution of seconds, whereas in co-authorship networks, nodes and edges are added to their graph on the temporal resolution of months or years. Correspondingly, these cases require different approaches for the evolutionary analysis.

Networks whose graphs evolve slowly over time and space can be effectively analyzed through a sequence of graph snapshots where snapshots of the graph at two distinct times $t_1$ and $t_2$, can be the basis for change extraction. The entire graph can be replicated from one snapshot to another with few computational and storage limitations because of the coarse temporal resolution. The challenge is that for fast-evolving networks, where the evolutionary analysis relies on the explanatory power of representing the evolution in the network across smaller temporal resolutions, the snapshot approach can become ineffective due to high computation and storage demands required to deal with what may be many thousands of individual snapshots. An approach that reduces replication of the entire graph from one time-window to another to avoid high storage overhead even as the graphs grow at different temporal resolutions. The Whole-graph approach presents a method where the entire evolution of the graph through time is captured in an unabridged, but complete, graph that evolves compactly as nodes and edges appear and disappear in the graph. Each node or edge has a timestamp depicting its valid time in the graph. The valid time captures when the insertion of new nodes, edges and attributes or deletion of nodes, edges and their attributes are made in the graph.

In this Chapter, the underlying graph G in the STVG framework is modeled based on the Whole-graph approach where the network structure evolves in space and time in such a
way that evolutionary patterns are due to the dynamics in the connectivity and adjacency relationships among nodes in the network. The graph grows continuously as new nodes and edges emerge through time. The Whole-graph is composed of subgraphs including the Time-tree. Subgraphs are used to facilitate the conceptual modelling of connectivity between entities in distinct spaces of the real-word network. The subgraphs are connected to the Time-tree to keep track of the evolution of the Whole-graph. Projected graphs retrieve the state of the Whole-graph at any given temporal resolution. The evolution of the Whole-graph becomes visible in the space-time varying dynamics of the Projected graphs. Relevant graph metrics are used to retrieve Whole-graph evolution from its Projected graphs. The STVG therefore, is actualized from the Whole-graph, Subgraphs and Projected graph formalism.

The STVG framework is applied to model a dynamic transit network to uncover the evolving behavior of the network at various temporal resolutions. The dynamic interactions between places and mobility characteristics of moving vehicles in a transit network presents an evolving graph that evolves swiftly in structure and size (Casteigts et al. 2011). The analytical results show the potential of using the framework for transit and mobility trend analysis with possible application to time-dependent transit recommendation systems.

4.2 Background and Related Work

The most common approach for capturing evolutions in graph is the sequence of graph snapshots method where graph, $G_{[t_1-t_n]} = \{G_1, G_2, G_3, \ldots, G_n\}$. Each graph snapshot is a static graph that represents a valid state of the graph at time point. This approach is commonly used in many real-world networks because of the convenience it affords the users to analyze the properties of an evolving graph based on discrete versions of consecutive time
points or time windows (Liu and Yu, 2010). However, this approach is only efficient in small real-world networks that do not evolve frequently. When there is high frequency in the dynamics of the network, the snapshot approach possesses a high storage overhead in proportion to the graph size since static nodes and edges are constantly replicated across snapshots. Moreover, computing evolutionary queries across the snapshots is computationally intensive and complex (Liu and Yu, 2010).

The second approach is the Whole-graph approach which is based on an unabridged graph that evolves through time, that is $G_{[t_1,t_n]} = [N_{[t_1-t_n]}, E_{[t_1-t_n]}]$, where $N_{[t_1-t_n]}$ and $E_{[t_1-t_n]}$ are sets of all node and edge instances respectively. The evolution of the graph can be captured using time-dependent graph queries across various temporal resolutions. Each node or edge has a timestamp depicting its valid time in the graph. The valid time captures when the insertion of new nodes, edges and attributes or deletion of nodes, edges and their attributes are made in the graph. The evolution is derived by computing and correlating the dynamics in the graph through time.

Longer processing time involved in extracting time-dependent subgraphs from the Whole-graph can be a disadvantage of this approach unlike the convenience that snapshot approach provides through versioning of smaller graph volume at each time-window. This primarily occurs when the time dimension of a whole-graph is represented by timestamps that are stored as attributes of the nodes or edges. Updates and query processing take longer processing time because there is a massive number of node and edge attributes to be scanned. This bottleneck can be reduced by representing the time dimension as a graph such as using a time-tree (Ivarsson et al.2014). The advantages of the Whole-graph approach over the snapshot approach include no replication of static nodes and edges across time which leads
to less storage overhead. Also, longitudinal queries across time are less complex than in the snapshot approach because a simple time-dependent query can run through the entire graph, retrieve and compare the state of the graph across different temporal resolutions.

Previous research work has applied the snapshot approach to real-world networks such as the social, web-citation and co-authorship networks. In Huo and Tsotras (2014) a survey was carried out on time-evolving graphs of a social network with emphasis on the temporal metrics. They propose a general formalism to study the evolution of the temporal metrics based on a sequence of snapshots across time-windows. Graph temporal metrics such as density, average path, proximity, reachability time, centralities, clustering coefficient and conductance were discussed in this study.

In contrast, Huo and Tsotras, (2014) focused on the problem of efficient temporal shortest path queries on graph snapshots of a social network. The study utilized fixed time-window partitioning on a slowly-evolving graph to generate a sequence of snapshots on which the queries are applied, one at a time. They had a total of 165 daily graph snapshots of a social network graph and adjusted the query time-windows to 5-day and 25-day intervals. The concept of temporal shortest path in this study is noted to be useful for users historical trend analysis, for example, to discover how close two users were in the past and how their closeness has evolved over time. This work utilizes ad hoc query over a sequence of graph snapshots, but the major drawback is high storage overhead.

Some strategies have been proposed to address the storage overhead issue by using a set of deltas that store only the changes that are needed to construct a snapshot. The most efficient compression approach is known as SM-FVF which creates clusters of k snapshots, where a compressed graph has no redundancy among delta files of the same cluster, but it
still has redundancy between delta files of different clusters (Ren et al. 2011). Meanwhile, Gottumukkala et al. (2015) proposes the possibility of integrating Neo4J and Spark’s GraphX (big graph data technologies) in a workflow to optimize the storage, processing and visualization of big graphs. They utilized this method to carry out visual analytics of evolving social network in graph snapshots to track the evolutionary state of the graph including the structure and topology.

Lerman et al. (2010) worked on a different formulation of centrality metric for an evolving graph analysis that measures the number of paths that exist between source and destination nodes over time in a graph. Their model was based on sequence of graph snapshots and it was evaluated on a scientific papers’ citation graphs whose dynamics occurs slowly. It was concluded in this paper that centrality measures of some articles were under or over estimated by previous studies that did not take into account the evolutionary nature of the graph. The evolution of essential genes (central nodes) and their roles in a cell survival and development were studied in the work of Jalili et al. (2016), utilizing node centrality evolution in a sequence of graph snapshots of a biological network. Quattrociocchi et al. (2012) utilized the time-varying graph framework proposed by Casteigts et al. (2011) based on a sequence of graph snapshots to study the dynamics in coexistence of co-authorship and citation network. The graph snapshots reveal the evolution in a scientific network extracted from a portion of the arXiv repository covering a period of 10 years of publications in Physics. It shows that scientific networks are dynamic as nodes (i.e., individual scientists) join, participate, attract, compete, cooperate and disappear in the network which affects the shape and strength of the graph’s connectivity. They discovered how the selection process
of citations may affect the shape of the co-authorship network from a sparser and disconnected structure to a dense and homogenous one.

While the snapshot approach is associated with disadvantages of high storage overhead and computational complexity, the Whole-graph approach provides an alternative. The Whole-graph approach represents graph evolution not in sequence of static snapshots but as an unabridged graph that evolves compactly as nodes and edges appear and disappear in the graph. This approach was utilized in Pereira et al. (2016) to analyze evolving centrality metrics in a dynamic Twitter graph. The study reveals how closeness and betweenness centrality measures evolve in the follower/followee relationship. They found that the Twitter network is dynamic, users can assume or leave central roles in the graph through time. However, the time-dimension of the graph is based on timestamp attributes of the nodes and edges of the graph. Each node and edge in the graph have a valid time at which they existed in the graph. Varying-time windows of weekly, fortnightly and monthly windows were used in the analysis of the graph’s evolution.

As earlier mentioned, having timestamps as attributes of the nodes and edges is inefficient in cases where there are billions of nodes and edges to scan through. Meanwhile, Kumar et al. (2006) similarly examined the evolution of Flickr and Yahoo 360 social graph in parallel. They found out similarities in the evolution pattern and properties of the two different graphs. The whole-graph approach was also utilized in burst area discovery based on evolving top-k changes in a stream of fast bipartite graph evolutions of users and stories on the Digg.com website (Liu and Yu, 2010). The authors, however, utilized the Haar wavelet tree to reduce high computation complexity associated with a fast-evolving graph
stream. Top-k burst areas are computed incrementally from small hop size to large hop size of time windows.

Recently, Huang et al. (2017) proposed the Supra-adjacency matrix approach to define a temporal network used to analyze the evolution of important nodes in Enron email communication network and DBLP co-authorship network. They treated temporal networks as a special case of multi-layer network where the emerging graph G is defined as a graph of N nodes divided into T time-windows which therefore, creates a supra-adjacency matrix of NT x NT dimensions. This approach could be suitable for a slow evolving graph whose properties changes at longer time intervals, such as in co-authorship networks but will face massive computational challenges in a mobility graph that evolves, for example every 5 seconds. Supra-adjacency matrix created at seconds, minutes and even hourly time-windows for a period of 18 months or more will face huge computational overhead.

Overall, these studies represent a growing interest in evolutionary analytics of dynamic networks. However, more literature is found focusing on slow evolving social graphs, dealing primarily with small snapshots of graphs. Real-world networks such the transit network and communication network however present fast-evolving graphs that evolve at shorter temporal and spatial resolutions. We therefore, present an evolutionary graph analytics framework that is based on the Whole-graph approach to capture evolutionary patterns in a space-time varying graph (STVG) at various temporal resolutions. The STVG framework provides an approach to reduce high storage overhead and affords us the ability to extract projected graphs at different time-windows and analyze their metrics across varying temporal resolutions.
4.3 Conceptual Framework for the STVG

In this section, the STVG framework is described as consisting of three components as shown in Figure 4.1:

- **Whole-graph**: represents the entire content of a STVG in a given lifetime. The main purpose is to store all data based on an unabridged graph that grows and evolves through time rather than graph snapshots.

- **Subgraphs**: represent entities and their relationships in distinct non-metric spaces (e.g. places, events, mobility, people and so on) of the Whole-graph. The main purpose is to facilitate the conceptual modelling of the connectivity between entities in distinct spaces that generate the Whole-graph.

- **Projected graphs**: represent nodes and edges that are retrieved at varying time-windows and temporal resolution as well as based on subgraph of interest that will be used in the evolutionary analytics of the Whole-graph.

There are conceptual challenges in representing the dynamic behavior of a real-world network, especially where the evolutionary analysis relies on the explanatory power of representing interactions in the network across different temporal resolutions. An efficient approach should be such that reduces replication of the entire graph from one time-window to another to reduce high storage overhead. Essential evolutionary assumptions of the Whole-graph, subgraphs and the projected graphs must be put into account to uncover the dynamic properties of the Whole-graph using classical graph metrics. We therefore present the three components of the proposed STVG framework and their corresponding evolutionary assumption as formulated in (Casteigts et al., 2011).
Figure 4.1. Proposed STVG Evolutionary Framework.
4.3.1 Whole-graph

The Whole-Graph G represents the entire content in a given lifetime $T (t1 - tn)$ of the STVG, whose nodes and edges grow through time. G consists of all the nodes and edges through time, $t1 - tn$, $[N_{[t1-tn]}, E_{[t1-tn]}]$, where $N_{[t1-tn]}$ and $E_{[t1-tn]}$ are sets of all node and edge instances of the Whole-graph stored in a database. Conceptually, the Whole-graph G has two dimensions namely, the Space dimension and the Time dimension which are represented as $(S, T)$ where $S$ is a digraph and $T$ is a tree graph. $S$ is composed of subgraphs $(S'1, S'2, ... S'n)$, and each subgraph is a subset of the whole-graph G in the STVG. Each node or edge in $S$ has a timestamp depicting its valid time in the graph. This is used to capture the insertion of new nodes, edges and attributes or deletion of nodes, edges and their attributes and to track the evolution of the graph G.

The evolutionary assumptions underlying the Whole-graph G are described based on the following:

- *Vertex-centric evolution:* the evolution of the Whole-graph materializes in the dynamics within a node’s neighborhood. This is an important assumption to enable the capability to extract both global and local evolution of the graph’s properties.

- *Complete graph connectivity:* At any point in time, a node in G must have an edge connection with an existing node. This assumption is essential to support global reachability across the entire graph which is important in querying the global evolution of a real-world network at any point in time. This is one of the advantages of capturing evolution in a dynamic graph based on the Whole-graph formalism instead of as snapshots.
- **Periodicity of nodes**: The assumption holds in practice in many real-world networks. More specifically, it provides the ability to exploit STVG properties using nodal graph metrics that are generated by the dynamics of a network (e.g. recurrent existence of nodes). This assumption is also important in defining temporal resolutions at which the dynamics of graph can be discovered.

### 4.3.2 Subgraphs

Subgraphs \((S'_{1}, S'_{2}, \ldots S'_{n})\) are used to facilitate conceptual modelling of connectivity within and between entities in distinct spaces of a real-world network. However, in the database, the graph \(G\) is not stored in subgraphs but as a Whole-graph with complete connectivity between entities. It is also important to point out that subgraphs are not graph snapshots since they require constant and complementary connectivity (at any point in time, the nodes belonging to one subgraph are linked to the nodes belonging to another subgraph) as well as eventual adjacency (at any point in time, a target node will be sequentially linked to a source node in the same subgraph).

![Figure 4.2. Connectivity between the subgraphs and the Time-tree.](image)

95
Multiple subgraphs $S'$ can be defined such that $S = (S'_1, S'_2, \ldots, S'_n)$ consisting of nodes and edges that belong to a specific type entity in a real-world network such as places, people, events and so on. Nodes belonging to different subgraphs $S'_i \subset S$ and $S'_j \subset S$ are linked by complementary connectivity edges since for every node in a subgraph there is always a corresponding node in another subgraph it is related to. However, only one subgraph $S'_i$ is connected to $T$ which is another subgraph of $G$ (time-tree) that represents the natural levels of temporal resolutions. Nodes belonging to the same subgraph $S'_i \subset S$ are linked by eventual adjacency edges (“NEXT”) between an existing node (source node) and a future node (target node) as described in Figure 4.2.

Essentially, $T$ is a hierarchical (tree) structure where a root node is followed by year nodes, month nodes, day nodes, and further temporal resolutions. Such that, each node of a subgraph $S'_i$ links to a leaf-node of $T$. This is an important constraint to support the capturing of vertex-centric evolution of the graph $G$ at any defined temporal resolution. Similarly, every node in the subgraphs has a label and properties which are essential for label-based projections. Labels assign roles (e.g. “Moving” representing the state of a moving entity) and objects (e.g. “Street” and “Intersection”). Multiple properties can also be attached to the nodes and edges of a subgraph $S'_i$ which are attributes ($a_1 \ldots a_n$) containing semantic information about the entity.

The evolutionary assumptions underlying the Subgraphs $S'_i$ are described based on the following:

- **Node Connectivity over Time**: At any point in time, at least one node belonging to a subgraph $S'_i$ can be reached by all the other nodes in $S$. This assumption is essential to support any graph query/algorithm whose input is spread over all
nodes generated at different time instances, with at least one node capable of
generating the output.

- *No Recurrence of Edges over time:* An edge is created only once at any point in
time. This assumption is critical to maintain the eventual adjacency relationships
thus to perform any computation of a graph metric such as shortest path and
centrality.

- *Constant Connectivity over time:* Subgraphs are in constant connectivity between
themselves, however, only one subgraph $S'_i$ and $T$ are always linked at any point
in time. There must exist an edge between a node belonging to a subgraph $S'_i \subseteq S$ and a leaf-node of $T$.

- *Eventual Adjacency:* There is always a future target node in subgraph $S'_i \subseteq S$ that
a source node will be linked to.

Conceptually, the subgraphs are essential for convenient definition nodes and edges
that are being projected for evolutionary analysis. The projected graphs can be locally
extracted from a single subgraph or globally from all the subgraphs.

4.3.3 Projected Graphs

Projected graphs are vital for retrieving nodes and edges that define the state of the
Whole-graph at varying temporal resolutions, time-windows and based on subgraphs of
interest. Projected graphs present the key graphs on which graph metrics are computed for
evolutionary analytics of the Whole-graph $G$. From the vertex-centric standpoint, the
evolution of the Whole-graph $G$ materializes in the time-varying evolution of its Projected
graphs, $PG(T)$. Depending on the network and application, varying resolutions of time can
be defined to appropriately capture the evolutionary behavior of the graph. At the extreme, each interval could correspond to the smallest time unit in the dataset or to the time between any two consecutive modifications of the graph (Santoro et al. 2011). In some cases, every Projected graph of the Whole-graph G becomes equivalent to an evolving graph model (Ferreira, 2004).

Time, $T$, of the Projected graph can represent temporal resolutions of the graph such as in seconds, minutes, hours, days, weeks, months and years. It can also represent time-windows such as a 2-week window, one-month window, 2-year window and so on. Time-windows can be defined as event-based such as to capture the evolution in the network through the period of a snow storm or other event.

The time-window can be a sliding window or a tumbling window depending on the application. A sliding window defines a time period that goes back in time from the present. For instance, a sliding window of one hour includes nodes and edges that have appeared in the last one hour. In a tumbling window, nodes and edges are grouped in a single window based on time. For example, in a tumbling window with a length of 2 hours, the first window (w1) contains nodes and edges that appeared between the zeroth and second hours. The second window (w2) contains nodes and edges that appeared between the second and fourth hours and the third window (w3) contains nodes and edges that appeared between the fourth and sixth hours. In this case, the Whole-graph G is projected every 2 hours, and none of the windows overlap; each Projected graph represents a distinct and unique state of the Whole-graph.

The evolutionary assumptions underlying the Projected graphs can be described based on the following:
- **Recurrence of nodes:** A node of a projected graph $PG(\mathcal{T})$ can reoccur in different projected graphs at different time windows, $wi, wj ... wk$, even in the case of tumbling time-windows. This assumption is important because some entities (nodes) in a real-world network (e.g. places) do not evolve but other entities and connectivity around them do evolve. The nodes will appear in more than one time-window because its neighborhood evolves across time.

- **No recurrence of edges:** In tumbling time-windows, an edge in a projected graph $PG(t_i)$ cannot reoccur in another projected graph because connectivity in each projected graph is distinct and unique.

### 4.3.4 Capturing the Evolution of STVG

Graph metrics are generally used to uncover the dynamics of the Whole-graph globally or dynamics in the graph locally over the Projected graphs. The process of capturing the evolution of the Space-time-varying graph involves a pipeline of operations. Table 4.1 is used to describe the algorithmic workflow of operations for capturing the evolutionary behavior of the Whole-graph based on the STVG framework.

**Table 4.1.** Capturing the Whole-graph evolution algorithm.

Define a time-window, $\mathcal{T} \subset T$ for analysis
- where $\mathcal{T}$ is the selected time-window
- and $T$ is the entire lifetime of the Whole-graph.
Define subgraph (s) of interest, $S' \subset S$
- where $S'$ is the selected subgraph (s)
- and $S$ is the entire space-dimension of the Whole-graph.
Define the temporal resolution $t_i \subset \mathcal{T}$ for iterative projections
FOREACH $t_i$ in $\mathcal{T}$
Project Graph, $PG_{(t_i)}$
-where $PG_{(t_i)}$ is a project graph at temporal resolution, $t_i$

FOREACH Graph $PG_{(t_i)}$

Compute Graph Metrics, $M_i$
-where $M_i$ is a graph metric (e.g. graph Density, Volume and Avg. path length)

FOREACH Consecutive $M_i$

Compare $M_i$ values, delta $\Delta$
-Identify patterns in $\Delta$ values

The algorithmic workflow can be implemented in a graph query to project the state of the Whole-graph and compute the graph metrics at varying temporal resolutions within any given time-window. Time-series analysis methods can be used thereafter to analyze the computed graph metrics values across time for evolutionary behavior of the graph (Fenn et al., 2011). For example, if the graph Density at time $t_i$ is $D_{t_i}$, at $t_j$ time $D_{t_j}$ and it continually evolves throughout the entire graph life-time, $(D_{t_i}, ..., D_{t_n})$.

4.3.5 Graph metrics

Graph metrics such as network density, network diameter and average path length have been utilized in this case study to retrieve the evolutionary pattern of the transit network structure at different temporal resolutions over monthly and yearly time-windows.

**Graph Density**: One of the most commonly used metrics for observing evolution in network structure is the graph density. In this experimental study, graph density of each Projected graph $PG_{(t_i)}$ in the STVG measures how close a given $PG_{(t_i)}$ is to a complete graph. That is, the ratio between the number of its edges and the number of all possible edges between n nodes.
Graph Density $D = \frac{|E|}{|N|^* (|N|-1)}$ (4.1)

However, in practical terms, true graph density $\rho(G) = \text{Mass}(G) / \text{Volume}(G)$, where the Mass of the graph $G$ is a total mass of its edges and nodes, and Volume ($G$) is a size-like graph characteristic of the graph (Kirgizov, 2013).

$\text{Mass}(G) = |E|$ which approximately estimates the Mass of the graph while;

$\text{Volume}(G), = |N| * (|N| - 1)$ which approximately estimates the Volume (Size) of the graph which is dependent on the number of nodes.

The evolution of the density of the Whole-graph can be observed by computing and observing the trend over the Projected Graph $PG_{(ti-tn)}$ through time, that is

$$\text{Density} = \frac{E_{(ti-tn)}}{|N_{(ti-tn)}|^* (|N_{(ti-tn)}|-1)}$$ (4.2)

**Average Path Length:** Average path length otherwise known as characteristic path length is also one of the most robust measures of network topology (Edwards, Hofmeyr, Stelle, & Forrest, 2012). It is denoted by $\langle d \rangle$ and can be used as a measure of the efficiency of travel or mass transit efficiency in the network.

For each $PG_i$, Average Path Length can be measured as:

$$\langle d \rangle = \frac{1}{N(N-1)} \sum_{i<j \land N} d_{ij}$$ (4.3)

where $d_{ij}$ is the number of edges for the shortest path from node $i$ to node $j$.

And the evolution of $\langle d \rangle$ can be observed through $PG_i$ over time $(ti - tn)$.

where the $\langle d \rangle$ of $PG_{(ti-tn)}$ can be defined as

$$\frac{1}{|N_{(ti-tn)}|^* (|N_{(ti-tn)}|-1)} \sum_{i<j \land N} d_{ij} (ti - tn).$$ (4.4)
4.4 Case Study

In this section, the case study and its implementation utilizing the STVG framework is described to carry out evolutionary graph analytics of a bus transit network.

4.4.1 Data Description

The datasets used for the implementation are the Automatic Vehicle locations (AVL) feeds from the Codiac bus transit network in Greater Moncton, New Brunswick, Canada and GIS shapefile dataset containing information about the bus routes, bus stops, street segments, street intersections, civic addresses obtained from GeoNB online service (http://www.snb.ca/geonb/e/index-E.asp). We extracted a total of 59,617 completed bus trips from the AVL feeds pulled from the buses every 5 seconds into a PostgreSQL/Post GIS database for a period of 18 months (from June 2016 - December 2017). The raw data set from the AVL feeds consist of the following attributes;

- vlr id: The ID of the data point in the vehicle location reports table
- route id vlr: The route ID in the vehicle location reports table
- route name: The route name
- route id rta: The route ID in the route transit authority table
- route nickname: The abbreviate of the route
- trip id br: The trip ID in the bid route table
- transit authority service time id: Transit authority service time ID
- trip id tta: Transit authority trip ID
- trip start: Start time of the trip
- trip finish: Finish time of the trip
- lat: Latitude
- lng: Longitude
- timestamp
4.4.2 Automated Pre-Processing Tasks

The data pre-processing tasks involve the steps that have been used to transform the raw datasets into a suitable format for building the STVG. These tasks were implemented by an automated data pre-processing algorithm developed by Cao & Wachowicz, (2019) which includes the following steps:

Step 1: Transformation of GPS points into Moves/Stops: The algorithm utilizes a fixed distance value between two consecutive points to determine if the bus was moving, “Moves” or stopped, “Stops”. This fixed distance was empirically determined as being 15 m for a transit network. If the distance between the previous point and the current point is more than 15 m, the current point is annotated as a “Move”. In contrast, if the distance is less than 15m, the current point is annotated as a “Stop”. These distances are Euclidian distances; network distance could be considered but, for a short distance of 15 m, Euclidian distance can be used for this purpose.

Step 2: Street Name Annotation. The algorithm annotates the “Moves” and “Stops” computed from the previous step with the street names if the event was performed on a street. For this task, a query on a PostgreSQL database is run to retrieve the names of the streets where a “Move” or “Stop” is located. This is a non-trivial step because the GPS coordinates usually are accurate to ±10m in urban areas (Salarian, Manavella, & Ansari, 2015).

Step 3: Bus Stop Identification. This step annotates the “Moves” and “Stops” with the “Bus Stop” names if the event took place within a bus stop in a 30 m buffer zone. It is important to point out that the algorithm also needs to verify the direction of a moving bus (e.g. eastbound and westbound) to identify the “Bus Stop” that a “Stop” or “Move” is located at.
Step 4: Street Intersection Identification. This step also annotates the “Moves” and “Stops” with the “Street intersection” names if this event took place within a street intersection. To determine if the mobility action is within a street intersection, the algorithm creates a buffer zone of a 30m radius (determined empirically) for each street intersection. The “Stops” and Moves” that are located inside a given buffer zone are annotated with the intersection identifier.

Step 5: Origin/Destination Trip Identification. The algorithm in this step identifies the origin and destination of each trip. The first GPS point of a bus trip located at a bus stop or station is tagged as the origin, and the last point of that same trip ID located at a bus stop or station is tagged as the destination. The GPS points between the origin and destinations points are sequentially indexed in order of occurrence.

Step 6: Bus Trip Labeling. The algorithm labels the trips in terms of (Route Number-Run Number-Run-day-Run Month-Run Year), where (50-10-23-12-2016) would represent Route #50 during the 10th run of the 23rd day of 12th month of 2016. The date had to be concatenated to the trip IDs, so they could have unique IDs in the database. A “Trip” here is a completed journey of a bus from an origin to a final destination.

At the end of this preprocessing pipeline, a CSV file is automatically generated containing all the data needed to build the STVG of the bus transit network in Neo4j graph database.

4.4.3 Building the Subgraphs

We define the ‘Platial’ and ‘Mobility’ subgraphs for generating the Whole-graph of the bus transit network which represents the connectivity between mobility events and places
over time. The pre-processed data sets in the CSV file are used to build the subgraphs of the bus transit network which generates a Whole-graph. The smallest temporal resolution at which nodes and edges were inserted into the subgraphs was 5 seconds, corresponding to the time-interval of the AVL feeds. This means evolution in the subgraphs can be tracked to the detail of every 5 seconds. We utilized the Neo4j graph database for the implementation. Neo4j is currently the most popular native graph database widely used for graph data management and analytics. The language in building and processing the graph in Neo4j is called Cypher, which is used to write the nodes and edges continuously into the graph and used for encoding graph metrics for evolutionary graph analytics.

4.4.4 Platial Subgraph

The Platial subgraph consists of a sequence of physical geographic places that represent a “Trip” of the bus within the transit network. In this case, a “Trip” is represented as a bus route that consists of a static “Bus Stop” (i.e. a designated place for passengers to board or alight from a bus), “Street Segment” (i.e. the transit segment between two bus stops belonging to a bus route), and “Street Intersection” (e.g. existing 3-way intersections or 4-way intersections of a bus route). More geographic places can be defined depending on the availability of dataset and application. In this case study, nodes are created in the Platial subgraph according to the actual locations of the bus trips at any given time. The “NEXT” edges represent adjacency relationships between two consecutive geographic places in a trip. Figure 4.3 illustrates the network structure of the Platial subgraph in the STVG of the bus transit network. A node in the Platial subgraph can reoccur in more than one Projected graph even in tumbling time-windows.
Cypher statements as shown in the example below were used to import the bus stop, intersection street segments points as Bus stop nodes, Intersection nodes and Street segment nodes respectively to the graph database. This step also includes the nodes properties such as intersection ID, intersection name and street segment ID, and street segment name.

Cypher Statement used to create Bus Stop nodes in the Platial Subgraph

```cypher
Load csv with headers from 'file:///busstops.csv' as csv
Create (bs:BusStop {BusStopID: csv.stop_id, sName: csv.stop_name, sLat: toFloat(csv.stop_lat), sLon: toFloat(csv.stop_lon), sParentStation: csv.parent_station, locType: csv.location_type});
```

### 4.4.5 Mobility Subgraph

The Mobility subgraph represents the discrete sequence of mobility events (moves and stops) of a moving bus in space and time. In this subgraph, as shown in Figure 4.4, the primary entity is the “Trip” that represents the trajectory of a moving bus within a bus route of a transit network. A “Trip” node is a composite node which is created by the sequence of “Move” (i.e. a node representing the location where a bus is in motion), and “Stop” (i.e. a
node representing the location where the bus is not in motion), nodes from an “Origin” (i.e. a node representing the first location of a trip) to a “Destination” (i.e. a node representing the last location of a trip). The connectivity between these nodes is represented by the “NEXT” adjacency relationship which is a space-time relation such as Origin-Move, Stop-Stop, Stop-Move, Stop-Destination, Move-Destination, and Move-Stop.

![Figure 4.4. Mobility Subgraph.](image)

Each node in the Mobility subgraph occurs at a specific point in time and adds a new node and their respective NEXT relation to the graph. These nodes are connected to the time-tree’s leaf nodes using their date and timestamps, starting from the year to the second leaf nodes. This is important for tracking the evolution of the subgraph. The “Origin” and “Destination” nodes of a trip have relationships to the “Trip” node labelled as “START_AT” and “ENDS_AT” respectively which gives meaning to the edges and useful for the evolutionary analysis of the trips. This subgraph is primarily responsible for the dynamics of the graph. In this case study, every 5 seconds, a node or more from the mobility subgraph is added to the Whole-graph.
The Mobility subgraph nodes were continuously loaded while concurrently establishing the connectivity with the Platial subgraph nodes. The cypher statements for importing the “Moves” and “Stops” nodes in the Whole-graph are shown below.

### Cypher statements used to create “Moves” and “Stops” nodes

<table>
<thead>
<tr>
<th><strong>“Moves” Nodes</strong></th>
<th><strong>“Stops” Nodes</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Load csv with headers from <code>file:///data/Moves.csv</code> as csv</td>
<td>Load csv with headers from <code>file:///data/Stops.csv</code> as csv</td>
</tr>
<tr>
<td>MERGE(r:Moves(MoveID: csv.moveid)) ON CREATE SET</td>
<td>MERGE (r:Stops (StopID: csv.stopid)) ON CREATE SET</td>
</tr>
<tr>
<td>r.TripID = csv.Tripid, r.Street=csv.streetname, r.BusStop=csv.Busstop, r.latitude=toFloat(csv.lat), r.longitude=toFloat(csv.long), r.Date=csv.Date, r.Time=csv.Time, r.Sequence=toInteger(csv.sequence), r.State=csv.state, r.year = toInteger(substring(csv.Date,6,4)), r.month = toInteger(substring(csv.Date,3,2)), r.day = toInteger(substring(csv.Date,0,2)), r.hour = toInteger(substring(csv.Time,0,2));</td>
<td>r.BusStop=csv.BusStop , r.latitude=toFloat(csv.lat), r.longitude=toFloat(csv.long), r.Date=csv.Date, r.Time=csv.Time, r.Sequence=toInteger(csv.sequence), r.State=csv.state, r.year = toInteger(substring(csv.Date,6,4)), r.month = toInteger(substring(csv.Date,3,2)), r.day = toInteger(substring(csv.Date,0,2)), r.hour = toInteger(substring(csv.Time,0,2));</td>
</tr>
</tbody>
</table>

#### 4.4.6 Connectivity between Mobility and Platial Subgraphs

Ideally, each node in the Mobility subgraph has a corresponding node in the Platial subgraph where it is located at in a given time instance. Platial and Mobility subgraph nodes are complementary in space and time. They are complementary because for each node in the Mobility subgraph there is always a corresponding node in the Platial subgraph on which a connection is established using a cypher statement as shown below and as described in Figure 4.5.
Cypher Statement used to create the edges between Mobility and Platial subgraphs.

\[
\text{MATCH (bs:BusStop), (st:Stops) WHERE bs.BusStopID = st.Busstop} \\
\text{MERGE (st)-[: LOCATED_AT]->(bs);} \\
\text{MATCH (ss:Streets), (st:Stops) WHERE ss.STNoSpace = st.Street} \\
\text{MERGE (st)-[: LOCATED_AT]->(ss);} \\
\text{MATCH (ss:Streets), (st:Moves) WHERE ss.STNoSpace = st.Street} \\
\text{MERGE (st)-[: LOCATED_AT]->(ss);} \\
\text{MATCH (bs:BusStop), (st:Moves) WHERE bs.BusStopID = st.Busstop} \\
\text{MERGE (st)-[: LOCATED_AT]->(bs);} 
\]

Figure 4.5. The connectivity between the subgraphs.

The connectivity between the two subgraphs is semantically represented as the “LOCATED_AT” edges that exists between the corresponding nodes. Independent of the number of subgraphs being represented in the graph model, all nodes are connected to a time-tree based on the temporal relationship between the nodes in only one of the subgraphs and the leaf nodes of the time-tree. The Mobility subgraph nodes are connected to the time-tree from top to the lowest level of temporal resolutions using, for example the Cypher statement
below to create the sequential “HAPPEN_AT” edges between the “Move” nodes and their corresponding time-tree leaf nodes.

Cypher Query: Sequential Connection of the Moves nodes to the Time-tree.

```cypher
MATCH (t:Moves) WITH t
MATCH (yy:Year {yearid:t.year}) WITH t,yy
MATCH (yy)-[r1]->(mm:Month {monthid:t.month}) WITH t,yy,mm
MATCH (mm)-[r2]->(dd:Day {dayid:t.day}) WITH t,yy,mm,dd
MATCH (dd)-[r3]->(hh:Hour {hourid:t.hour}) WITH t,yy,mm,dd, hh
MATCH (hh)-[r4]->(mm1:Minute {minuteid:t.minute})WITH t,yy,mm,dd,hh, mm1
MATCH(mm1)-[r5]->(ss:Second {secondid:t.second})WITH t,yy,mm,dd,hh,mm1,ss
CREATE (t)-[:HAPPENS_AT]->(ss);
```

The time-tree is the time-dimension of the Whole-graph which contains all possible time instances of every node in the subgraphs throughout the graph lifetime. We built a time-tree of a 2-year lifetime (2016-2017) in hierarchical and sequential order of temporal resolutions as described in Figure 4.6. The Cypher statements used to create the time-tree is described below.

Cypher statement for creating the time-tree.

```cypher
WITH range(2016, 2017) AS YEARS, range(1-12) as MONTHS, range() as Days, range(1,24) as Hours, range(1,60) as Minutes, range(1,60) as Seconds
FOREACH(year IN YEARS | MERGE (y:Year {yearid: year})
    FOREACH( month IN MONTHS | CREATE (m:Month {monthid: month})
        MERGE (y)-[:CONTAINS]->(m)
        FOREACH(day IN (CASE
            WHEN month IN [1,3,5,7,8,10,12] THEN range(1,31)
            WHEN month = 2 THEN
                CASE
                    WHEN year % 4 <> 0 THEN range(1,28)
                    WHEN year % 100 = 0 AND year % 400 = 0 THEN range(1,29)
                    ELSE range(1,28)
                END
            ELSE range(1,30)
        END) |
            CREATE (d:Day {value: day}
                MERGE (m)-[:CONTAINS]->(d)
                FOREACH( hour IN Hours | CREATE (h:Hour {hourid:hour})
                    MERGE (d)-[:CONTAINS]->(h)
                    FOREACH ( minute in Minutes | CREATE (mn:Minute {minuteid:minute})
                        MERGE (mn)<[:CONTAINS]->(h)
                    )
                    FOREACH ( seconds in Seconds | CREATE (ss:Second {secondid:second})
                        MERGE (ss)<[:CONTAINS]->(mn1)
                    )
                )
            ))
        )
    )
)
```

110
4.4.7 The Whole-graph

The Whole-graph is composed of the subgraphs and the time-tree. It presents the complete connectivity between the space and time dimensions of the network. Figure 4.7 represents the overview of two dimensions of the graph model, the space dimension, which consists of the subgraphs, and the time dimensions that consists of the time-tree. The connectivity between these dimensions enables the evolutionary analytics of the transit network graph. The time-tree does not only track the evolution of the space dimension, but it also facilitates efficient retrieval of time-dependent projected graphs. Table 4.2 describes the statistics of the Whole-graph in the graph database.
4.5 Discussion of the Results

From a global perspective, the trend in the graph density and volume as shown in Figure 4.8 reflects the bus transit network’s topology variations within a 2-year time window at hourly temporal resolution. This trend could be useful in observing increase or decrease in bus transit mobility activities over time, as well as trip density within the entire network through time. We could also observe if there is a significant correlation between global
network density and congestion density in the network through time. Congestions in the graph model represents where there are many of “Stops” on a street with a “Move”.

**Figure 4.8.** Evolution of Graph Density and Volume at hourly temporal resolution.

Figure 4.8 also reveals an evolutionary trend in the Whole-graph of the transit network such that the hourly network density, graph density with respect to traffic congestion and bus movement have the lowest values at peak transit hours (8am and 4pm). In a practical sense, the trend reveals the highest volume of transit mobility activities (“Moves” and “Stops”), trips and traffic congestion at these peak hours. In other words, between 8 am and 9 am in the mornings as well as 4 pm and 5 pm in the evenings, more nodes and edges are added to the Whole-graph of the transit network signifying a lot of mobility activities within the network during these times. The inverse relationship between graph density and graph volume is clearly depicted in Figure 4.9 where an increase in volume means decrease in density and vice versa. The evolutionary trend shows a strong correlation between the global network density of the graph, graph density with respect to traffic congestion and density with respect to the bus movement. Similar behavior is seen in the network volume, congestion and movement volume, as one increases, others increase as well.
In monthly temporal resolution, Figure 4.9 reveals an evolutionary behavior with peaks in August 2016, December 2016 and May 2017 in terms of movement, traffic congestion and network volume as well as density of the graph.

**Figure 4.9.** Evolution of Graph Density and Volume at monthly temporal resolution.

The Figures show that, from June to December 2016, graph density with respect to the entire network, movement and congestion reveal a zig-zag (up and down) alternate behavior different from what is observed in the same months in 2017. The graph volume with respect to congestion, movement as well as that of the entire network from June to December 2017, did not experience significant changes. The peaks observed in August 2016 and December 2016 did not reoccur in the following year, 2017. Figure 4.10 provides a closer
look at the behavior of the transit network in a monthly time-window (December 2016 peak) on a daily temporal resolution. The daily behavior of transit activity in December 2016 is compared to that of the previous month, November 2016 (Figure 4.11) to depict the differences in daily transit pattern and the reason for such a peak in December.

Figure 4.10. Evolution of Graph Density and Volume at daily temporal resolutions for December 2016.
The daily graph volume and density behavior for the December 2016 bus transit network activity peak depicts high volume of bus transit activity in many days of the month except the lows observed between Tuesday 6th to Friday 9th as well as on Monday 12th. There is also the weekend low such as on Sunday 18th, Saturday 24th and 31st of December 2016. Comparing December 2016 daily network volume with that of the previous month, November 2016, as shown in Figure 4.11, one could deduce the reason for the December peak. There is more transit network low volume in November with respect to that of December. From 12th of December to 30th, there are more mobility events in the network when compared to those of November. In graph topological sense, more nodes and edges are added to the Whole-graph in December than in November 2016.

![Network Volume: November vs December 2016](image)

**Figure 4.11.** Evolution of Network volume at daily temporal resolution for Nov and Dec 2016.

The evolutionary trend in the Average path length of the network would reflect the average trip length (travel time) over time. One analysis of interest would be to observe how evolving average path length correlates with evolving trip density/volume over time in the network.
Figure 4.12. Hourly evolution of Avg Path Length vis-a-vis graph density and volume.

It can be observed in Figure 4.12, that Average path length has a very strong correlation with graph density and volume. It increases with increase in graph volume and decreases with an increase in graph density. In a practical sense, average path length in this case study is the average trip length (travel time) from an origin to destination within the network. It is as expected that travel time (trip length) would increase when volume of congestion and mobility activity in the network increase, as observed in Figure 4.12. We can also observe in Figure 13 that the highest average trip lengths are seen at peak hours of the day (8am and 4pm) when it shows the highest traffic congestion and network volume.

4.6 Conclusions

Evolutionary graph analytics based on sequence of graph snapshots have been commonly utilized in literature mainly because of the convenience it affords the users to manage small-size graph in discrete versions and compare differences between snapshots across consecutive time windows as this approach may be suitable for small graphs and for cases where evolution do not occur at shorter temporal resolutions. It is associated with high storage overhead in proportion to the size of the evolving graph and the time intervals between snapshots because the entire graph is usually replicated from one snapshot to
another. Also, computing evolving queries across the snapshots is computationally intensive and complicated. These are major disadvantages especially, in cases where the graph is massive and evolves frequently at shorter temporal resolutions and where the evolutionary analysis relies on the explanatory power of representing the dynamics across different temporal resolutions.

We therefore, propose a framework based the Space-Time-Varying Graph (STVG) formalism which utilizes the Whole-graph approach to model the dynamics of the graph whose evolutions materialize in the time-varying evolution in its Projected graphs. The STVG framework provides an approach to reduce high storage overhead of massively changing graphs where new nodes and edges arrive every second. It affords the capability to extract projected graphs at different time-windows and analyze their metrics across varying temporal resolutions.

The framework was implemented for a transit case study using the AVL feeds of the bus transit network of Greater Moncton, New Brunswick, Canada which generated a Whole-graph of 44.2 GB in the database. In contrast, using the Snapshot method created 7,280 hourly snapshots, 732 daily snapshots, 18 monthly snapshots and 2 yearly snapshots in the graph database where the smallest snapshot amounted to 1.3 GB of graph. The total storage cost for over 18,000 snapshots of graph needed for our evolutionary graph analytics amounted to 23.4 TB, because the total storage and computational cost increase linearly with the number of snapshots, that is, total cost = cost per snapshot multiplied by the number of snapshots. Using the Snapshot method would have over 50,000 % increase in storage when compared to our Whole-graph approach.
The framework was implemented for a case study using the bus transit network of Greater Moncton, New Brunswick, Canada. Evolutionary analysis is based on graph density, volume and average path length on projected graphs at varying temporal resolution across different time windows. The analysis reveals evolutionary patterns in the overall network density of the graph, traffic congestion density as well as graph density with respect to bus movement at hourly, daily and monthly temporal resolutions. Similar patterns are observed in the evolutionary pattern of network volume, congestion and movement volumes. This type of analytics for any transit networks potentially provides an efficient way to uncover dynamics of the network as well as the dynamics in the network over space and time. The evolutionary pattern of the transit network properties such as average paths, network density and volume as a result of the dynamics in human mobility pattern may become vital for transit network optimization. Potential applications can be found in transit trend analysis as well as in time-dependent transit recommendation systems.

References


http://doi.org/10.1007/978-3-642-22450-8_27


http://doi.org/10.1103/PhysRevE.84.026109


http://doi.org/10.1145/1150402.1150476


Chapter 5 Conclusions

This chapter provides an overall summary of research contributions and achievements that have been fully described from Chapter 2 through 4. It also provides suggestions for future work using the developed STVG framework as a backbone.

5.1 Summary of the Research

This research developed a STVG framework for representing and analyzing evolutionary patterns in large and fast-evolving networks at different spatial and temporal resolutions. The STVG framework for evolutionary graph analytics of fast-evolving networks is developed based on the Whole-graph, Subgraphs and Projected graphs formalism such that the evolution of the Whole-graph and subgraphs materializes in the time-varying evolution of its Projected graphs. The STVG framework also provides an approach to reduce high storage overhead in massively changing graph where new nodes and edges arrive every second. It also allows us to extract Projected graphs and analyze their metrics across varying temporal resolutions. The STVG framework was utilized for evolutionary graph analytics of traffic accidents as well as in evolutionary graph analytics and transit performance assessment of the bus transit network of Greater Moncton, New Brunswick Canada.
5.2 Research Contributions

In summary, the contributions of the research are tied to answering the research questions as well as the overall contribution to Geographic Information Science.

How can a Space-Time-varying Graph be developed to model and analyze fast-evolving networks?

The STVG framework captures the evolution of a fast-evolving network by modelling a Whole-graph, Subgraphs and Projected graphs. The Whole-graph is the underlying graph G of the STVG which represents the entire dynamics of fast-evolving network in a given lifetime T. The Whole-graph is composed of Subgraphs including the time-tree. Subgraphs are used to facilitate the conceptual modelling of connectivity between entities in distinct spaces of the fast-evolving network. The Subgraphs are connected to the time-tree to keep track of the evolution of the Whole-graph. Projected graphs are used in the STVG framework to retrieve the state of the Whole-graph at any given temporal resolution. The evolution of the Whole-graph becomes visible through the evolution observed in the Projected graphs.

The proposed STVG framework is an attempt to adopt the Whole-graph approach to model evolving topological relationships between non-metric spaces over varying temporal resolutions in a fast-evolving network. The STVG combines directed and bipartite graphs with a time-tree to represent evolving topological relationships between non-metric spaces across time. This offers robustness to the STVG such that it reduces replication of the entire graph from one time point to another and supports ad-hoc graph queries to uncovering and analyze evolutionary patterns at various temporal resolutions. The conceptual notions of
eventual adjacency and complementary connectivity used in STVG to represent topological relationships in a fast-evolving network also provides a network structure where each node carries information about its neighbors.

**How can a graph database be used to implement an STVG framework that captures the network’s space and time evolution with less storage overhead?**

The STVG framework was implemented in the Neo4j graph management platform. Neo4j is currently the most popular native graph database widely used for graph data management and analytics. Graph metrics, algorithms and User Defined Functions (UDFs) have been developed in Neo4j and can be utilized within a graph query. The underlying Whole-graph of the STVG framework was implemented in Neo4j as an unabridged graph that grows through time rather than in graph snapshots. The modeling and implementation approach support a continuous growth of the Whole-graph through time without replicating the entire graph each time there is an update to the graph. This approach reduces the high storage overhead problem associated with the snapshot approach because there is no replication of the entire graph from one time point to another. Inbuilt graph metrics and algorithm in Neo4j (such as centrality, path, density, volume, Louvain etc.) were utilized in the graph queries to retrieve the evolution of Whole-graph from its Projected graphs. These graph metrics are robust and most commonly used for capturing evolution in network structure. They are less complex to compute, however, the path metrics (such as shortest path, average path length and network diameter) are computing memory intensive.
How do graph metrics evolve in time and space as a function of the dynamics of fast-evolving networks?

The STVG framework uncovered the evolution of relevant graph metrics of a traffic accident network and transit network over space and time. In the traffic accident network, the evolution of node-centric graph metrics such as degree and PageRank centrality were observed across different spatial and temporal resolutions of the network. Ad-hoc graph queries computed a changing delta (Δ) to quantify the evolution of the graph metrics across different temporal resolutions. The Louvain clustering algorithm was used to detect evolving communities in the network showing nodes that are densely connected to each other. Depending on the network and application, varying resolutions of time and space can be defined to appropriately capture the evolutionary behavior of network through the evolution of its graph metrics. It is observed that as the graphs of these networks evolve their metrics also evolve in space and time.

In the transit network, the analysis was based on the evolution of the graph density, volume and average path length at varying spatial and temporal resolution across different time windows. The analysis revealed evolutionary patterns in the overall graph density of the network at hourly, daily and monthly temporal resolutions. Similar patterns were observed in the evolutionary pattern of the graph volume. It was also observed that average path length has a very strong correlation with graph density and volume. It increases with increase in graph volume and decreases with an increase in graph density. This type of analytics for any transit network, potentially provides an effective way to uncover dynamics of the network as well as the dynamics in the network over space and time.
These metrics can be extended to capture and predict growth, transition and eventual connectivity in fast-evolving networks. For example, computing Pagerank through a deep random walk in the network across time can be used for future key nodes and edge prediction.

What are the practical applications of the STVG framework?

The STVG was firstly employed in a traffic accident network to identify and analyze locations with a high frequency of traffic accidents and their evolving pattern in space and time. Previously, most traffic accident analysis have mostly been based on GIS spatial point pattern analysis with more emphasis on the spatial constraints and no account of the temporal constraints of these events which leads to under or over-estimation of the analysis results. The STVG was used to overcome this challenge and have traffic accident patterns analyzed in space and time based on graph analytics of the relationship between geographic places in the road network and traffic accident events. Also, previous traffic accident clustering methods such as the KDE is based only on spatial constraint (Euclidean distance). The STVG, instead uses topological relationships for clustering which makes computation faster in massive datasets using Louvain clustering algorithm.

In addition, the STVG was utilized to carry out transit performance assessment and evolutionary analytics of a transit network. To achieve that, both geographic and mobility spaces were integrated to compute transit performance indicators (such as schedule adherence, bus stop and route activity levels) based on graph metrics (shortest path, degree and PageRank centrality) using a large volume of transit data feeds. This implementation was carried out using the AVL transit feeds generated by the public bus transit system in Greater Moncton, New Brunswick, Canada and the road network dataset. The preliminary
results demonstrate how graph metrics can be used in the STVG to compute transit performance indicators such as schedule adherence, bus stop activity, and bus route activity ranking that are necessary for transit planning in terms of resource prioritization and ridership increase.

The framework was also used to uncover evolutionary patterns in transit network where nodes and edges appear in the graph every 5 seconds for a period of 18 months. The analysis was based on graph density, volume and average path length on projected graphs at varying temporal resolution across different time windows. The study reveals evolutionary patterns in the overall network density of the graph, traffic congestion density as well as graph density with respect to bus movement at hourly, daily and monthly temporal resolutions. Similar patterns are observed in the evolutionary pattern of network volume, congestion and movement volumes. This type of analytics for any transit networks potentially provides an efficient way to uncover dynamics of the network as well as the dynamics in the network over space and time. The evolutionary pattern of the transit graph metrics such as average paths, network density and volume as a result of the dynamics in human mobility pattern may become vital for transit network optimization. Potential application can be found in transit trend analysis as well as in time-dependent transit recommendation systems.

These two networks have been useful to demonstrate the applicability of the STVG framework for modeling and analyzing fast-evolving networks. The STVG framework can be extended to modelling other real-world networks such as Internet of Things social and communication networks.
What are the overall contributions of the STVG to Geographic Information Science?

Modelling topological relationships in non-metric spaces at varying temporal resolutions has been a major challenge in Geographic Information Science. This research developed a Space-Time Varying Graph where topological relationships in non-metric spaces evolve across different temporal resolutions in such a way that the evolutionary patterns are due to the dynamics in connectivity and adjacency relationships within and between entities of non-metric spaces. The applicability of the STVG in modelling topological relationships in non-metric spaces whose properties vary under continuous evolution was demonstrated in both a fast-evolving traffic accident network and a transit network. In the traffic accident network, adjacency and connectivity relationships within and between Event space and Platial space were modeled as an STVG and utilized to advance traffic accident analytics beyond traditional spatial point pattern analysis in GIS as contained in Chapter 2. Meanwhile, similar topological relationships within and between Mobility and Platial spaces in a transit network were modeled and analyzed at varying temporal resolutions using the STVG. This approach was utilized to advance transit performance assessment and transit network evolutionary trend analysis beyond the existing methods as discussed in Chapter 3 and 4 of this thesis.

Finally, it is important to mention that, as a limitation, the STVG was not devised for modelling geometric changes in entities of a geographic space. For example, the STVG is not developed to represent the change in a polygon node that splits into two new polygon nodes in time $t_i$. Instead, at two different time points, a node belonging to a non-metric space is linked to its corresponding node to represent change and dynamics of a network rather
than representing the geometrical changes of a specific node. The STVG was developed to help us obtain a better understanding the evolutionary patterns of eventual adjacency and complementary connectivity between places and events in a network space. Complementary connectivity becomes the fundamental topological relationship in the STVG because independent of the number of spaces being represented in the network, a bipartite relationship between the spaces will always exist, giving us the ability to compute different types of graph metrics in the network.

5.3 Future work

In furthering this research, the next steps are:

- To further evaluate the framework by utilizing the STVG framework to model and analyze other fast-evolving networks such as Internet of Things, and communication networks. These are examples of networks that evolve swiftly in fractions of a second temporal resolution. The evolutionary assumptions utilized in developing the framework will be further evaluated using these networks to further guaranty the quality of this modeling approach. More constraints will be incorporated in the implementation of the framework to detect false positive and other errors in the dataset as quality control measures. The scalability of the STVG framework will be further examined by utilizing larger datasets.

- To develop ensemble graph metrics to quantify growth, transition and decay phases using topological relationships. For example, graph density, average path length and network diameter can be integrated as an ensemble graph metric to quantify and analyze growth of the network. Similarly, average degree centrality and PageRank
centrality can be combined in an algorithm to quantify transition and decay points in the network.

- To integrate other subgraphs to the Platial, Mobility and Event Subgraphs. For example, social and economic subgraphs which will represent additional non-metric spaces in the STVG framework. The use of subgraphs in modeling distinct spaces of a real-word network is also expected to become more vital for Internet of Things where geographically distributed sensors and computing resources will perform tasks such as data pre-processing and subgraph creation on the fly and closer to where the data is being collected.

- To extend the use of the STVG to compute other complex transit performance indicators such as headway regularity and accessibility with the availability of AFC and APC datasets. Modelling passenger counts, trip fares and bus location feeds as distinct subgraphs in the STVG will provide the capability to compute other transit performance indicators other than schedule adherence, bus stop and route activity ranking. More feedback from the transport domain experts will be needed to further evaluate the efficacy of the framework.
Curriculum Vitae

Candidate’s full name: Ikechukwu Derek Maduako

Universities attended:

B.Sc. in Geoinformatics and Surveying, University of Nigeria Nsukka, 2008
M.Sc. in Geospatial Technologies, 2012, jointly awarded by University of Muenster Germany, Universitat Jaume I, Spain and New University of Lisbon Portugal

Publications:

The following research publications are recorded by the author within the period of the PhD program.

Peer Reviewed Journal Papers:


Conference Papers and Poster Abstracts:


http://doi.org/10.1145/3132211.3132452


http://doi.org/10.1109/WF-IoT.2016.7845469


Fredericton Canada.