UCRL-TR-233690

LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

Temporal Representation in Semantic Graphs

Justin J. Levandoski, Ghaleb M. Abdulla

August 14, 2007

This work was performed under the auspices of the U.S. Department of Energy by University of California, Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

Temporal Representation in Semantic Graphs

Justin J. Levandoski
Department of Computer Science and Engineering
University of Minnesota - Twin Cities
Minneapolis, MN
justin@cs.umn.edu

Ghaleb Abdulla
Center for Applied Scientific Computing
Lawence Livermore National Laboratory
Livermore, CA
abdulla1@llnl.gov

Abstract

A wide range of knowledge discovery and analysis applications, ranging from business to bilogical, make use of semantic graphs when modeling relationships and concepts. Most of the semantic graphs used in these applications are assumed to be static pieces of information, meaning temporal evolution of concepts and relationships are not taken into account. Guided by the need for more advanced semantic graph queries involving temporal concepts, this paper surveys the existing work involving temporal representations in semantic graphs.

1 Introduction

The use of graphs to model and represent relationships between entities is a very popular practice. However, most of these representations follow the traditional mathematical description of a graph. The use of *semantic graphs* has grown useful for representing relationships between entitites in datasets. Semantic graphs differ from the traditional mathematical definition through the implementation of *node* and *link* types. Thus, semantic graphs are useful not only for their structural properties, but also for the useful semantic information embedded within them.

Semantic graph representation are used in the Semantic Web [1], biological applications [35], and multimedia applications [43], to name a few. Recently, a general push within each of these semantic graph applications has been to reliably model *temporal* attributes. Modeling temporal data has been an active area of research within the datbase community for many years [46]. In fact, over forty temporal data models exist in the literature. However, simply applying one of these models to the case of semantic graphs is not straighforward due to the type of analysis (and hence the type of queries) that are posed against a semantic graph.

The purpose of this paper is to survey the current research literature with a focus on temporal representations in semantic graphs. Through this survey, we hope to delineate and classify the exhisting models and techniques used for querying temporal information in semantic graphs.

1.1 Semantic Graphs and Ontologies

At the core, a semantic graph follows the basic structure of a directed graph consisting of nodes and directed edges. However, whereas nodes and edges in a generic directed graph may be homogeneous nodes, semantic graphs imply the notion of *typed* nodes and edges. Specifically, nodes in a semantic graph each have a specific *type* (e.g., *city* or *person*); each type is sometimes referred to as a *concept*. Furthermore, each node may contain one or more *attributes*. At the very least, a single-attribute node contains an *identifier*, or *referent* (e.g., name of person). Other attributes give extra information about a node (e.g., a person's eye color).

The edges, or links, in a semantic graph imply a *semantic* or *conceptual* relation between nodes. Like nodes, links can also have types. For instance, the link *person* → *city* may imply a person lives in the city referenced by the link. In some semantic graph representations, a link between node types implicitly defines a relation; such a graph is termed *homogeneous*. In homogeneous semantic graphs, only *one* link type is allowed between nodes. For example, in a semantic graph representing communication networks, the link *person* → *person* implies one person has *contacted* another person. In homogeneous semantic graphs, edges are usually not labeled. Examples of homogeneous semantic graphs are social networks (i.e., person to person relationships) or communication networks (i.e., a person contacting another person through email or phone). When more link types are allowed in a semantic graph, such a graph is termed *heterogeneous*. In this scenario, the semantic relationship between nodes is explicity labeled. For example, in the multi-link setting the semantic relationship *person* → *city* may mean a person *lives in* or *works in* the specified city.

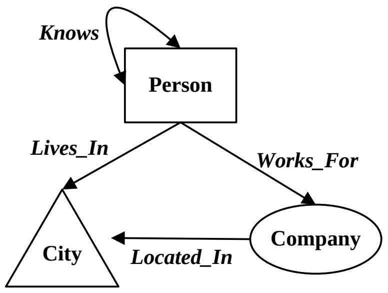

Figure 1. Semantic Graph Ontology

1.1.1 Ontologies

An important structure that explicity defines valid relationships between nodes in a semantic graph is an *ontology*. An ontology, also referred to as a *schema*, is generally defined as an auxillary graph that gives the valid edge types (relations) between node types. An example of a small ontology is given in Figure 1, that defines the valid relationships between three node types: *person*, *city*, and *company*. In this ontology, a person can: (1) *know* another person, (2) *live in* a city, and (3) *work for* a company. Furthermore, a company can be *located in* a city.

Given a set of relational data, the design of an ontology is reliant on the type of analysis that will be done on the graph. Thus, the amount of detail (number of node and edge types) captured in the graph can span from simple (e.g., a single node and link type) to complex (e.g., multiple node and link types). An example of a simple ontology is a social communication network involving only people (nodes), where connections imply communication between two people. An example of a more granular ontology is that of a movie database involving movies and actors. In the movie graph, actors could be connected to movies by either having a *lead role* or *supporting role*. Naturally, coarser ontologies lose information (i.e., detail) present in the finer models. Application domains also determine the amount of detail needed in the graph. For instance, graph analysis involving outlier detection calls for a high degree of granularity [5]. On the other hand, social network analysis may only be concerned with coarse relations, such as communication between people.

1.1.2 Semantic Graph Instances

Naturally, all *instances* of a semantic graph must conform to the structure defined by its *ontology*. A specific *instance* of the simple person - city - company ontology presented in Figure 1 is given in Figure 2. In this instance, a *person* Dave works for company Oracle (located in San Jose) and lives in San Rafael. Also, Dave knows another person John.

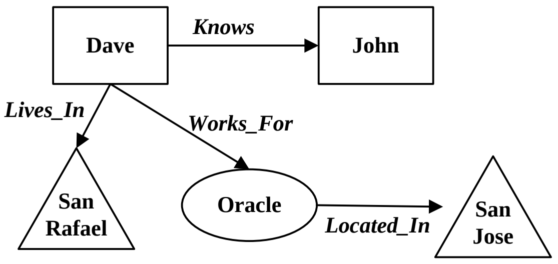

Figure 2. Semantic Graph Instance

1.2 Temporal DBMS Modeling

Within the database research world, much work has been done on modeling temporal attributes. One of the most important semantic property differences concerning time are that of *valid* and *transaction* time. Valid time refers to the real-world time in which an event occured or was observed. Transaction time refers to the time when a change was recorded (e.g., stored) in a database. For example, a mythical *concert* event could have a *valid* time of June 10th, 2007 from 1:00pm to 4:00pm, meaning the concert occured on this day between the given time period. However, the concert could have a different *transaction* time of June 20th, 2007, 5:00pm, representing the time that the event was stored in the database. In general, semantic graphs are used to model real-world relationships. Thus, throughout this survey we will be generally concerned with how *valid* time is represented in a semantic graphs. Discussion of *transaction* time will be explicitly denoted.

The database research literature contains many temporal data models; roughly forty at last count. Güting et al. [29] classify these models as extensions to the core DBMS model. In general, database tuples are represented as *facts* associated with a specific *timestamp* that describe when the fact is valid. The existing models can be classified within the following four categories:

1. Data Model Extended: examples are *relational* or *object-oriented*
2. Granularity of Facts: examples are a *tuple* or *relation*
3. Type of timestam: a chronon (i.e. instant), time interval, or a set of time intervals.
4. Time dimension: support for valid time, transaction time, or bitemporal (i.e., both valid and transaction time).

In general, a review of these temporal data models is outside the scope of this paper. Thorough overviews of these models, however, can be found in works by Zaniolo et al. [52] and Özsoyoglu et al. [37].

Taking a step outside of the temporal modeling universe, query languages for temporal DBMS systems have been proposed in the literature. The most prominent example is TSQL2 [46], an extension to the SQL-92 standard proposed specifically to query temporal attributes of data. The TSQL2 standard is *bitemporal*, satisfying queries on both *valid* and *transaction* time.

1.3 Focus of this Paper

The semantic graph model can be thought of as an abstraction from specific DBMS models and implementations. Thus, the work in this paper may overlap with specific concepts proposed for *temporal* DBMS modeling and lanugages. However, this paper focuses on temporal representations found in *graph* structures, where graph manipulations and analysis are a main factor in modeling and query processing.

To elaborate on these differences, consider the following queries where query 1 is an example of a generic spatio-temporal query, and query 2 represents a spatio-temporal query involving spatio-temporal information as well as other semantic relationships.

Query 1 *Find all vehicle tracks that go within 1 km of Facility X between time interval t1:t2*

Query 2 *Extract all vehicle and communication activity for Facility X for time period t1:t2; Find other facilities with similar vehicle/communication activity.*

Here, we see that query 1 asks about spatio-temporal aspects of vehicle tracks surrounding Facility X. Meanwhile, query 2 asks not only for vehicle track location during a specific time period, but also for communication activity for Facitility X. Furthermore, query 2 also requests a similarity search for other Facilities with similar vehicle and communication activity. Thus, query 2 goes beyond generic spatio-temporal query structure and uses analytical attributes regarding communication information as well as similarity measures. In this case, semantic relationships play a crucial role in answering query 2, and such relationships can naturally be represented as a semantic graph. Furthermore, the semantic graph representation is appropriate for semantic similarity queries, as graph matching algorithms can naturally be used to extract these similarities.

This paper will mainly focus on temporal representations in semantic graph with an eye toward efficiently answering queries similar to query 2 in the example above. However, discussions of temporal extensions to the ER model or specific temporal database implementations are certainly applicable. In fact, much existing work in temporal database modeling or query language extensions may overlap with query processing for semantic graphs. Where applicable, such work will be mentioned.

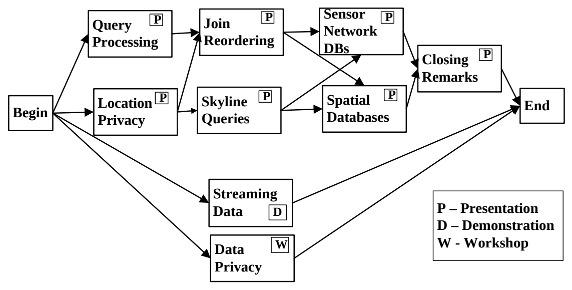

Figure 3. Sequential Representation

The outline and main organization of this paper is as follows. Section 2 will survey three temporal representations for semantic graphs. Specifically, we survey the Sequential Representation (Section 2.1), the Snapshote Representation (Section 2.2), and the Valid Edge Representation (Section 2.3). Next, Section **??** will introduce a set of temporal benchmark queries for semantic graphs, and evaluate each of the surveyed representations with respect to these queries. Finally, Section 4 will conclude this survey.

1.4 Targeted Queries

1.5 Targeted Ontologies

2 Temporal Representations

‘ This section surveys temporal representations for semantic graphs along with their application domains. Three main representations exist, namely: (1) *Sequential*, where time is modeled as a logical sequence in the graph. (2) *Snapshot*, where a version of the complete graph is stored at a specified time period. (3) *Valid-Edge*, where each edge (i.e., semantic relationship) is associated with an interval specifiying when it was valid in the modeled world. The goal of this section is to give the reader a technical overview of each representation by first introducing the model. Next, we will survey the different application domains as well as the developed information extraction methods for the representation. Finally, we will discuss database implementations that exist for the representation where appropriate.

2.1 Sequential Representation

Time in the sequential temporal representation is modeled primarily by a directed graph, where the graph itself represents a logical sequence. In the basic sense, the sequential model represents a timeline of events, usually with an absolute start and end point. Because of this constraint, the semantic relation between nodes in such a graph are constrained to model logical order. As an example, Figure 3 gives a graph representing a schedule of events at a database conference. In this example, the nodes of the graph represent sessions (e.g., presentations, workshops, or demonstrations), while the edges represent possible transitions between events (i.e., schedules) an attendee can follow. For instance, one possible schedule would be to attend the workshop on data privacy, denoted as: {*Data Privacy*}. Another possible schedule in this example involves presentations: {*Location Privacy, Join Reordering, Spatial Databases, Closing Remarks*}. Each of these schedules is a valid temporal sequence for the graph in Figure 3.

The temporal properties in the sequential representation are explicit in the graph, i.e., the graph itself is a model of time. Therefore, the structure of a sequantial graph lends itself to any application that depends on analyzing or modeling order of events. Such applications are querying multimedia information [43], biological networks [35], sequence processing [41, 42, 40], and business intelligence [31]. These applications will be discussed in detail later in this section.

Queries against graphs using this representation usually involve extracting information about temporal sequences. As an example, a simple query against the graph given in Figure 3 is: *Find a graph (schedule) that contains presentations on location privacy and join ordering, and possibly other presentations.* The answer, in this case, would involve the subgraphs: (1) {*Location Privacy, Join Reordering, Sensor Network DBs, Closing Remarks*} (2) {*Location Privacy, Join Reordering, Spatial Databases, Closing Remarks*}. Both subgraphs represent a temporal sequence that satisfy the constraints of the query.

In addition to explicit temporal order, the sequential model can be expanded to contain information relating to a specific timescale (i.e., date and/or time). This is done by simply adding attributes to the nodes (events, in this case) that represent a temporal value. In the field of temporal modeling, two straightforward methods apply for representing time values as attributes: timestamp or interval [27]. In timestamping, a single time value is added to the event representing either its start time or length (in units). As an example, in Figure 3 the presentation "Query Processing" can have a timestamp of {45 mins} representing its duration, or {9:00 AM} representing the time that it began. On the other hand, an interval contains the begin and end information for the event. For instance, "Query Processing" can have interval attributes of {Begin: 9:00 AM} and {End: 9:45 AM}. Similarly, edges can contain temporal information, represented by timestamps or intervals, representing transition time.

2.1.1 Applications

This section will highlight several application areas where the sequential temporal model is used. It should be noted that the applications highlighted here are by no means a comprehensive list, and serve the purppose of emphasizing the importance and applicability of the sequential temporal representation. In the subsequent section, we will cover specific methods for querying such graphs that model sequential temporal events.

Multimedia The area of multimedia retrieval and mining naturally uses the sequential representation in order to model text, video, audio, and image presentations. Sheng et al. [43] uses sequential semantic graphs in order to model transitions in multimedia presentation graphs. A multimedia presentation graph is tantamount to the conference schedule graph given in Figure 3, except it models valid transistions in a multimedia presentation. The nodes in the multimedia graph can have different types (video, text, audio, image), much like the conference schedule (presentation, demonstration, workshop).

The work by Chen et al. [12] uses Augmented Transition Networks (ATN) in order to model timelines of various multimedia presentations. In this representation, nodes in the graph represent arbitrary states in a multimedia presentation where multiple presentations may be streaming simultaneously. Edges in the graph represent both the duration between states and the valid presentations playing during that time period. This Multimedia Augmented Transition Network (MATN) model was created in order to allow fast multimedia search in databases. Later work used the MATM model as a data mining framework in order to extract information from traffic video sequences [13]. Much like the MATM model, the work of Day et al. [17] proprosed an object-oriented approach to semantic modeling of video data, where a state in the transition network is represented by multiple nodes that represent objects in a video clip. In such a representation, queries posed on the graph deal with temporal interaction between objects in a video stream.

Bioinformatics While not strictly temporal in nature, the sequential model is relevant to bioinformatic applications where directed graphs are used to model concepts in molecular biology. Such examples are metabolic pathways signaling transduction pathways, or gene regulation [35]. In this graph representation, researchers often want to analyze whole graphs searching for specific sequences (e.g., for protein pathway homology searches) [44]. Such queries are tantamount to the database conference schedule query given earlier in Figure 3.

Sequence Processing At the core, a linear sequence of events is modeled as a directed graph. Thus, many applications in sequence processing either implicity or explicity operate on graphs using the sequential temporal representation. For instance, the work of Harada et al. [31] formulated the problem of detecting sequential patterns of events for supporting business applications. Also, sequence processing has been studied extensively in the realm of database management systems [40, 41, 42]. Such work was a precursor to the multiple stream management systems built to handle real-time data [3].

Graph Mining A subdiscipline of the graph mining research area is very relevant to the sequential representation. Specifically, algorithms have been developed to find "interesting" substructures in large graphs. The SUBDUE system [32] was built specifically to search for similar substructures in large graphs. This work is based on the minimum description length and optional background knowledge [16]. Such work has been applied to chemical compound analysis and CAD circuit analysis. However, substructure discovery can translate directly to finding interesting temporal patterns in semantic graphs using the sequential representation.

2.1.2 Information Extraction Methods

The sequential temporal representatiassdafon lends itself to many general querying and graph matching methods because of its inherent representation as a directded graph. Thus, many general graph-based query and pattern matching algorithms may be used to extract information from this model [25]. One categorization of graph-matching algorithms is that of *structural vs. semantic* matching. For temporal sequences, both query types apply. Structural matching is based solely on the *physical* structure of the graph [48, 50]. An example of a structural query using Figure 3 could be: *Find all conference schedules that contain nodes with only two incident edges*. Semantic matching combines both node and edge attributes as well as physical graph structure to answer queries for conceptually similar graphs [15]. In the case of the sequential temporal representation, this type of query translates into finding similar conceptual temporal transitions. For example, a semantic query on the graph given in Figure 3 is: *Find all possible schedules involving only presentations, where each presentation has two possible transitions*. Here, both semantic attributes (e.g., presentation) and structure (e.g., two possible transitions) are used to query the graph. Other general query and matching methods that apply are the concept of *exact* and *inexact* matching, as well as *optimal* and *approximate* solutions. A comprehensive overview of graph matching algorithms can be found in [25].

Multiple language prototypes exist for querying temporal concepts embedded in graphs using sequential representation. One such language, GOQL, is based on the Object Query Language (OQL), a standardized object-oriented database query language [10]. GOQL extends the standard OQL framework in order to efficiently query temporal information in multimedia graphs [43]. Specifically, GOQL extends the OQL sequence operators in order to create the temporal operators *next*, *until*, and *connected* so queries involving relative order and sequences can be efficiently answered. The use of such a language can be seen in the following example query using the graph from Figure 3. The first query involves the *connected* operator by asking for *all conference events schedlued before "Sensor Network DBs"*

```
Select s
FROM g in Graph, sB, s, sE in g.Nodes p in paths(g,s,sB,sE)
WHERE g.name=''Conference Graph'',
sb.type=''Begin'' sE.name=''Sensor Network DBs''
p:sB connected s connected swl
```

In this code example, the where clause defines the specific stucture of the subgraph necessary to answer the query: a begin node, followed by all intermediate nodes eventually connected to the specific node "Sensor Network DBs". Sheng et al. [43] defined GOQL and presented theoretical analysis for query processing. However, no implementation was discussed. The theoretical basis for GOQL is found in *Computational Tree Logic* [23] and the temporal operators for *Propositional Lineary Temporal Logic* [40].

In the area of bioinformatics, PSQL extends SQL in order to efficiently query sequential subgraphs in biological networks [35]. PSQL was implemented on top of a purely relational database using simple *Node* an *Edge* tables to represent directed acyclic graphs for protien interactions. Specifically, the nodes in the PQL database represent typed biological entities (molecules or interations), e.g., genes, enzymes, etc. Homogeneous edges connect (1) molecules to interactions; meaning the molecule is necessary for an interaction to occur (2) interactions to molecules; meaning the molecule is a product of an interaction, and (3) interactions to other interactions. PSQL enriches standard SQL mainly by implementing path expressions. A very relevant and useful example of a PSQL query is *find all genes whose expression is directly or indirectly affected by a given compound L-Glutamate*. The visual version of this query is given in Figure 4, and the corresponding SQL-like query is as follows:

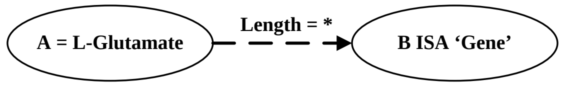

Figure 4. Biological Query

```
SELECT B
FROM A, B
WHERE A.name='L-Glutamate AND
A[-*]B and B ISA 'Gene'
```

The path expression [-*] in this example represents an acylic path of any length between nodes *A* and *B*. Similarly, a numeric value can be specified to find a path of a specific length.

While not strictly developed to query temporal sequences, QGraph is a full visual graph query language used to return subgraphs that match desired query patterns and semantics [6]. To query the graph, a QGraph user draws

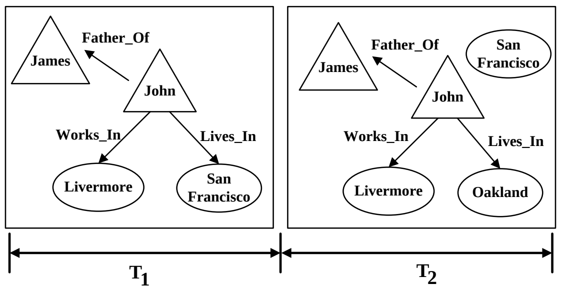

Figure 5. Snapshot Representation

a query graph, specifying both structure and semantics. QGraph then returns all matching subgraphs that match the user's query.

Several more languages exist that can be used to reliably query the sequential temporal representation. However, the graph query languages just mentioned serve as a good sample, as a comlete review of these graph languages is outside the scope of this paper. The important point to note is that since temporal information is inherent in the graph structure, many exisiting graphical query languages and graph matching algorithms can be used on the sequential representation.

2.1.3 Implementations

While a significant number of query methods exist for graphs representing temporal sequences, very few of the methods surveyed here have been implemented and tested in an actual prototype. No work surved in this section mentions experimental evaluations with regard to efficiency or runtime. Furthermore, with the exception of PSQL [35] and QGraph [6], no work surveyed in this section mentions an implementation of a query method.

2.2 Snapshot Representation

As the name suggests, time in the snapshot representation of semantic graphs is modeled by historical chronons, or "snapshots" instances, of a graph over a period of time. Thus, historical information about a particular semantic graph is available for analysis. Figure 5 gives an example of the snapshot temporal representation for a semantic graph involving a person "John". In the first snapshot T_1, "John" has three different semantic connections to other graph entities: (1) He is the *father of* another person "James", (2) He *works in* the city of Livermore, and (3) He *lives in* the city of San Francisco.

The snapshot representation lends itself well to discovering *dynamic* entities of a graph over time [11, 21]. In the realm of semantic graphs, the *dynamic* entities involve semantic relationships that can change over time. For instance, in snapshot T_1 in Figure 5, the semantic relationships *Works_In* and *Lives_In* can be considered dynamic, as the person "John" can live and work in different cities during his lifetime. In Figure 5 gives an example of such a change; between snapshot T_1 and T_2, a new semantic relationship was formed, showing that "John" now *lives in* "Oakland" instead of "San Francisco". On the other hand, the semantic relationship *Father_Of* between "John" and "James" is static, and will not change over time. Of course, the amount of *dynamic* and *static* elements in a semantic graph is a property of the domain being modeled.

Due to dynamic elements in a semantic graph, the snapshot representation is mostly applied to instances or problems involving changes over time to *all* or *part* of a graph. Questions that are posed against this representation generally involve knowledge discovery applications; a simple example of such a query is: *Find the largest area of change in graph A between time periods 1 and 3.* Such a query differs fundamentally from the sequential representation presented in Section 2.1. In the sequential representation, querying involves matching a specific pattern known beforehand, e.g., *Find all schedules that follow a pattern P*. Querying the snapshot representation generally assume no specific query pattern *a priori*. However, this does not disqualify the use of the snapshot representation to be used on matching algorithms involving a specific query pattern across one or multiple graphs in a series. In the subsequent section, we will cover specific applications for the snapshot representation.

2.2.1 Applications

Since the snapshot representation allows changes of dynamic elements of a graph to be analyzed over time, graph mining is the major application for this representation. Graph mining, or *structural motif finding*, involves finding common or "interesting" patterns in a graph [9, 16, 34]. Some mining methods are constructed for finding interesting substructures in a single, static graph [16, **?**]. In this case, the snapshot model does not apply, as such methods would apply to the sequential method presented in Section 2.1. However, many mining methods consider changes across time [9, 8, 11, 21, 20, 19, 39, 45]; here, the snapshot representation naturally applies.

The application areas for graph mining on snapshots span a spectrum of disciplines that model data as a graph, both semantic and otherwise. For instance, mining of World Wide Web data alone has multiple subdisciplines, such as hyperlink analysis over time [21, 19], Web social networks [33], and, although not techically the Web, structured newsgroups [7]. Also, network communication topology graphs serve as a crucial testbed for mining and knowledge discovery [11, 20, 39].D

Much like the sequential temporal representation, query and mining methods for the snapshot representation in the literature are concerned with *structure*, or *structure and* *semantics*. In the snapshot representation, a dynamic *structural* change is soley based on changes to the physical structure of all or part of the graph over time. On the other hand, a *structural and semantic* change is based on changes to specific typed or attributed structures, i.e., nodes and semantic relationships in a semantic graph, over time.

2.2.2 Information Extraction Methods

Since the snapshot model can be considered a set of n graphs observed over T_n observation periods, many graph matching algorithms that were discussed in Section 2.1.2 can be applied to answer questions about a specific pattern query over time. An example of such a query is: *Does subgraph P occur in graph G in all snapshots between time periods 1 and 7?*. In this example, the graph matching algorithm for pattern P can ask for a solely structural change [48, 50], or it can have specific semantic information attached [15].

Within the discipline of graph mining, the concept of mining temporally evolving graphs is becomming a major topic within Web analysis. Previous work only considered static representations of the web, however, Desikan et al. [21] proposed the evolving graph representation and defined three levels of analysis for web graphs over time: (1) Single Node, (2) Subgraphs, and (3) Whole Graphs. Single node analysis covers properties of a single web page in this case. However, the concept can be generalized for a generic node in any semantic graph. An example of interesting single-node properties changing over time in Web mining are link structure, in-degree, out-degree, hub/authority score, and PageRank score. These properties model the "popularity" of a particular node over time. Subgraph analysis leads to interesting or abnormal patterns of behavior on the Web, and may lead to insight on upcoming trends or criminal behavior. Whole graph mining leads to analysis of basic graph properties such as size and density, and also derived properties such as hub and authority scores, in the case of the Web. Another use of temporal analysis on Web graphs are evolving PageRank scoring [19]. Recently, analysis Web links with semantic information has been studied in the area of Web mining [18] *without* temporal evolution.

In general, pattern mining of temporal changes in a snapshot series of graphs is defined as a theoretical framework by Borwardt et al. [8]. Citing the need to adequately study interacions in real-world systems, the authors define the concept of frequent pattern mining on dynamic graphs. The problem posed in this type of work is the ability to determine dynamic subgraphs that are (1) topologically similar, and (2) show similar behavior over time. Semantic relationships in this framework receive attention, as edge patterns are defined by the type of two nodes that create an edge. However, analysis of multiple semantic relationships between two nodes is not covered.

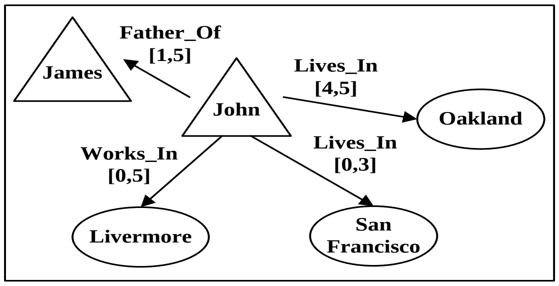

Figure 6. Snapshot Representation

The snapshot representation has also been used in research on visualizing changes in communication and social networks over time. Both of these network models are considered *homogeneous* semantic graphs. The work of Gloor et al. [26] developed a visualization browser for analyzing social links over time. This work was based around changes to social communication derived through emails and represented as graph snapshots. A dynamic visualization algorithm, named sliding time frame algoritm, was developed in order for users to define a valid time period (e.g., 5 days) where a series of changes in the graph can be visualized. Users can also choose to view historical data, i.e., the graph structure before the defined window. Ultimately, the aim of this work is to distinguish communication patterns over time in different netowrks by means of visualization.

Other specific mining methods that apply to the snapshot temporal representation are abnormality detection [45], spatio-temporal changes in graphs [11], and recovery algorithms for missing graph information [9]. While each of these methods is specific to an application, the underlying theme of detecting changes between a sequence of graphs is the same. Thus, the snapshot is heavily used by these an many other graph mining methods.

2.3 Valid Edge Representation

A third and very powerful temporal represenation in semantic graphs is the valid edge representation. The valid edge model involves attaching a *time interval* to each semantic relationship in the graph. The interval represents the time that an relationship is "valid" in the graph. Figure 6 gives an example of the valid edge representation for the semantic interactions of person "John", introduced previously in Section 2.2. Assumming the current time period is 5, we see that John lived in San Francisco from time period 0 to 3, and currently lives in Oakland, where he moved at time period 4. Similarly, time intervals are attached to all other semantic relationships in Figure 6.

The valid edge model is a very effective tool as it preserves the spirit of extensibility inherent in most universes modeled by semantic graphs. Thus, it is very powerful for

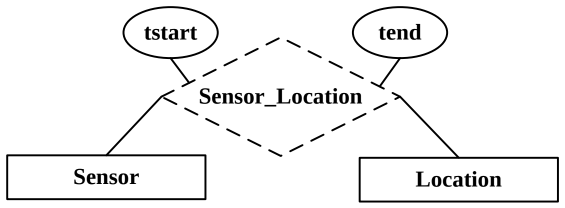

Figure 7. DRER Model

answering queries of the type “Find all instances where X is valid over time period [t_1,t_2]”. As a specific example involving Figure 6, we could ask “Find all cities where John lived between time periods 0 and 5”. Also, the valid edge model is inherently more efficient in answering the analysis queries just presented in comparison to the snapshot model. This efficiency is gained by not having to store an entirely new version of the graph, as in the snapshot model, every time an update is made to all or part of a graph. Updates in the valid edge model only involve the specific entity in the graph being updated. Also the valid edge model is generally more robust than the snapshot model, as a snapshot for a specific time instant can be reconstructed easily for all or part of a graph in valid-edge format.

2.3.1 Applications

The valid edge representation models time-sensitive semantic relations effectively. Thus, it is present in many different modeling paradigms spanning many applications.

RDF/Semantic Web Recently, there has been much research and development in the area of *Resource Description Framework (RDF)*, a metadata model and language proposed by W3C for building machine-readable semantic structures for data on the World Wide Web. Using the RDF structure, The Semantic Web hopes to effectively integrate semantics for such web activities as information integration, search, and analysis. In fact, only recently did RDF researchers formally define the valid-edge representation for the semantic Web [28]. Thus, research involving semantic relations in RDF data that is temporally correlated is relatively new.

The major application of the valid-edge model in RDF research falls in the area of geospatial and temporal semantic analytics. Until recently, much research on RDF semantic analysis applications focused on solely semantic relations. However, geo-spatial and spatiotemporal thematic data processing is now a major push for RDF applications. The underlying theme of this research is to effectively model the spatial and temporal relationships between entities in a semantic graph. Several high-level ontologies have been proposed to accomplish this task [2, 38, 51]

RFID/Mobile Data While not explicity tied to data processing and analysis in semantic graphs, data modeling for RFID data has much in common with the valid-edge representation. At a fundamental level, RFID data is *temporal* and *dynamic*. For instance, a very practical use-case for RFID chips is inventory tracking. Inventory items, with RFID chips attached, interact with readers that are responsible for transmitting an item’s information. This information is eventually stored in a standardized format, and queries against this data almost always involve temporal qualifiers (e.g., *Find all readers that interacted with item 1 in the last hour*). Thus, interactions in the RFID setting are always associated with a timestamp, and data modeling in this area use constructs resembling the valid-edge model [36, 49].

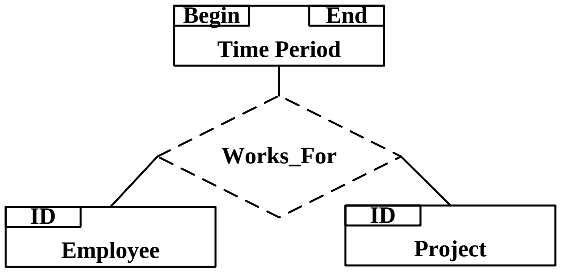

Figure 8. RAKE Temporal Model

Wang and Liu [49] aimed to create a model for RFID data such that data management systems can effectively support large-scale temporal changes. Specifically, the model aims to help data mangement systems suppor tracking and analysis queries over dynamic RFID data. This model is based on the identification of fundamental entities and relationships found in RFID systems; of importance in this survey is the dynamic relationships. Specifically, Wang and Liu point address dynamic temporal data found in RFID systems, such as observations, location change, and containment change of objects. They propose the Dynamic Relationship ER Model (DRER), a simple temporal extension to the ER model to overcome the constraint that all relationships in the ER model are static and current. The DRER model employs state-based dynamic relationships that contain two attributes denoting the *start* and *end* time of the state. Figure 7 gives an example of a state-based dynamic relationship model between a sensor and location. Here, we see that a sensor is associated with a *location* between a valid time interval. Similar to [49], Liu et. al [36] use the DRER to model the myriad use-case scenarios found in RFID environments.

Temporal Data Modeling Extending the ER to handle temporal data is not exclusive to RFID environments. In fact, the valid-edge representation can be found in other general-purpose temporal models. The Relationships, Attributes, Keys, and Entities (RAKE) model [24] makes use of *begin* and *end* timestamps in time-varying relationships. Figure 8 gives and example of the RAKE temporal modle, showing a relationship between an *Employee* and *Project* entities. The RAKE model contains more detailed constructs for other time-varying scenarios; as we can see, however, at its core the RAKE model heavily resembles the DRER model for state-based dynamic relationships. As another example, the Temporal ER Model (TER) [47], where temporal relationships are associated with a *lifetime*.

2.3.2 Information Extraction Methods

Within the area of Semantic Web research, information extraction and thematic and temporal analysis is of much interest. While not strictly temporal in nature, Angles et al. introduce the concept of querying RDF data from a graph database perspective [1]. This work moves beyond strictly modeling RDF concepts by discussing and initial definition of a strict database model and the design of query language primitives on such a model. Specifically, this work addresses the second of the three components of a data model defined by Codd [14]: (1) a collection of data structure types, (2) a collection of transformation operators and query lanaguage and (3) a collection of general integrity rules. Theoretical in nature, the main contribution of Angles et al. is a review of exisiting graph query lanaugages and a listing of graph primitives for RDF query languages, e.g., *paths and connectedness*, *aggregate functions*, and *neighborhoods*.

Moving into spatial and temporal analysis of RDF data, the work of Perry et al. [38] defines an semantic ontology for linking thematic (i.e., semantic), spatial, and temporal relationships, as shown in Figure 9. Here, the bold arrows represent a sub-class relationship, while the labeled lines represent non-hiearchichal relationships. We also see valid-edge intervals on the *located_at* and *occured_at* relationships. This ontology uses the concept of *Continuants*, that represent entities that persist over time (e.g., person or building), and *Occurants*, that represent events and processes (e.g., battle or concert). In this ontology, a *named place* continuant is a spatial entity, meaning its identity is strongly associated with space (e.g., building), whereas a *dynamic entity* is relatively mobile, and not strongly associated with space (e.g., person). An occurent, on the other hand, is always associated with a valid time and space.

The work by Perry et al. [38] also formalizes a set of spatial and temporal query operators over RDF triples and theoretically demonstrates their effectiveness. Example of temporal operators are: (1) *temporal_intersect*, selecting an interveral where the entire path (i.e., semantic relationship) is valid, (2) *temporal_range*, selecting an interval where a specific path unfolds, and (3) *temporal_restrict*, defining the properties of an entity at a given time period. The authors then extend these operators to the spatio-temporal domain.

Application prototypes have also been built for spatiotemporal properties found in RDF data. Hakimpour et al. [30] built the REST web service artifact that gathers distributed RDF data from multiple web sources. After this processing, the REST service allows uses to specify simple requests for events based on time, space, semantics, and cost (i.e., the cost ratio of time and space). When addressing cost, the application gives the user a simple slide scale to specify preference on the time or space dimension.

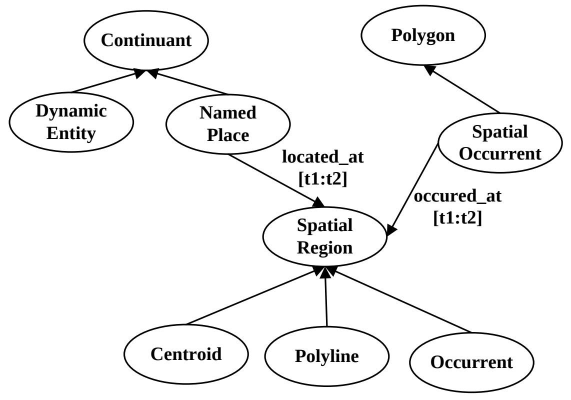

Figure 9. Spatiotemporal Semantic Ontology

In the area of RFID data processing, the ESL-EV query language [4] was designed as a stream query language with support for temporal events. ESL-EV is an SQL-like language that with syntax for detecting temporal event *sequences* for contiuously-updated RFID readings. The core of this language is based on a sequence operator *SEQ*, that detects a sequence of events from multiple streams. For example, the following ESL-EV code demonstrates how to detect if an RFID badge is scanned by a sequence three readers R1-R3:

```
Select R1.time, R2.time, R3.time
FROM R1,R2,R3
Where SEQ(R1,R2,R3)
AND R1.tid=R2.tid AND R1.tid=R3.tid
```

The ESL-EV language has further efficiency features such as sliding windows for valid time periods and multiple tuple pairing modes in order to cut down on the amount of historical information needed to detect an RFID sequence.

Similarly, the AUCQL [22] query language was developed for semistructured data, such as XML that could be used to represent graph data. The main draw behind AUCQL that differentiates it from other semi-structured query languages is its ability to handle properties of link (edge) models. One of many such properties is time.

3 Experimental Evaluation

4 Conclusion

References

[1] R. Angles and C. Gutierrez. Querying RDF Data from a Graph Database Perspective. *Proc. of 2nd European Semantic Web conference, ESWC*, pages 346–360, 2005.

[2] I. Arpinar, A. Sheth, C. Ramakrishnan, E. Usery, M. Azami, and M. Kwan. Geospatial Ontology Development and Semantic Analytics. *Transactions in GIS*, 10(4):551–575, 2006.

[3] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and Issues in Data Stream Systems. *Proceedings of the twenty-fi rst ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 1–16, 2002.

[4] Y. Bai, F. Wang, P. Liu, C. Zaniolo, and S. Liu. RFID Data Processing with a Data Stream Query Language. In *ICDE*, pages 1184–1193, Istanbul, Turkey, Apr. 2007.

[5] M. Barthelemy, E. Chow, and T. Eliassi-Rad. Knowledge Representation Issues in Semantic Graphs for Relationship Detection. *AAAI Spring Symposium on AI Technologies for Homeland Security*, 2005.

[6] H. Blau, N. Immerman, and D. Jensen. A Visual Language for Querying and Updating Graphs. Technical report, Technical Report 2002-037, Dept. of Comp. Sci., Univ. of Mass. Amherst, 2002.

[7] C. Borgs, J. Chayes, M. Mahdian, and A. Saberi. Exploring the Community Structure of Newsgroups. *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 783–787, 2004.

[8] K. Borgwardt, H. Kriegel, and P. Wackersreuther. Pattern Mining in Frequent Dynamic Subgraphs. *Proceedings of the Sixth International Conference on Data Mining*, pages 818–822, 2006.

[9] H. Bunke, P. Dickinson, C. Irniger, and M. Kraetzl. Recovery of Missing Information in Graph Sequences. *Pattern Recognition*, 39(4):573–586, 2006.

[10] R. G. G. Cattell, editor. *The Object Database Standard, ODMG-93.* Morgan Kaufmann, 1996.

[11] J. Chan, J. Bailey, and C. Leckie. Discovering and Summarising Regions of Correlated Spatio-Temporal Change in Evolving Graphs. *Data Mining Workshops, 2006. ICDM Workshops 2006. Sixth IEEE International Conference on*, pages 361–365, 2006.

[12] S. Chen and R. Kashyap. A Spatio-Temporal Semantic Model for Multimedia Database Systems and Multimedia Information Systems. *Knowledge and Data Engineering, IEEE Transactions on*, 13(4):607–622, 2001.

[13] S. Chen, M. Shyu, C. Zhang, and J. Strickrott. A Multimedia Data Mining Framework: Mining Information from Traffic Video Sequences. *Journal of Intelligent Information Systems*, 19(1):61–77, 2002.

[14] E. F. Codd. Data Models in Database Management. *SIGMOD Record*, 11(2), 1981.

[15] T. Coffman, S. Greenblatt, and S. Marcus. Graph-Based Technologies for Intelligence Analysis. *Communications of the ACM*, 47(3):45–47, 2004.

[16] D. J. Cook and L. B. Holder. Substructure Discovery Using Minimum Description Length and Background Knowledge. *JAIR*, 1:231–255, 1994.

[17] Y. Day, S. Dagtas, M. Iino, A. Khokhar, and A. Ghafoor. Object-Oriented Conceptual Modeling of Video Data. *Proceedings of the Eleventh International Conference on Data Engineering*, pages 401–408, 1995.

[18] C. DeLong, S. Mane, and J. Srivastava. Concept-aware ranking: Teaching an old graph new moves. In *ICDM Workshops*, pages 80–88, 2006.

[19] P. Desikan, N. Pathak, J. Srivastava, and V. Kumar. Incremental Pagerank Computation on Evolving Graphs. *International World Wide Web Conference*, pages 1094–1095, 2005.

[20] P. Desikan and J. Srivastava. Analyzing Network Traffic to Detect E-Mail Spamming Machines. *Proc. ICDM Workshop on Privacy and Security Aspects of Data Mining*, pages 67–76, 2004.

[21] P. Desikan and J. Srivastava. Mining Temporally Evolving Graphs. *Proc. of WebKDD*, pages 22–25, 2004.

[22] C. E. Dyreson, M. H. Böhlen, and C. S. Jensen. RFID Data Processing with a Data Stream Query Language. In *ICDE*, pages 1184–1193, Istanbul, Turkey, Apr. 2007.

[23] E. Emerson. Temporal and Modal Logic. *Handbook of theoretical computer science (vol. B): formal models and semantics table of contents*, pages 995–1072, 1991.

[24] S. Ferg. Modeling the Time Dimension in an Entity-Relationship Diagram. *4th International Conference on the Entity-Relationship Approach*, pages 280–286, 1985.

[25] B. Gallagher. Matching Structure and Semantics: A Survey on Graph-Based Pattern Matching. Technical report, Technical Report Lawrence Livermore National Laboratory, Livermore, CA, 2006.

[26] P. A. Gloor, R. Laubacher, Y. Zhao, and S. B. Dynes. Temporal Visualization and Analysis of Social Networks. In *NAACSOS Conference*, 2004.

[27] H. Gregersen and C. Jensen. Temporal Entity-Relationship Models-A Survey. *IEEE Transactions on Knowledge and Data Engineering*, 11(3):464–497, 1999.

[28] C. Gutierrez, C. Hurtado, and A. Vaisman. Introducing Time into RDF. *Knowledge and Data Engineering, IEEE Transactions on*, 19(2):207–218, 2007.

[29] R. Güting and M. Schneider. *Moving Objects Databases.* Morgan Kaufmann, 2005.

[30] F. Hakimpour, B. Aleman-Meza, M. Perry, and A. Sheth. *The Geospatial Web*, chapter Spatiotemporal-Thematic Data Processing in Semantic Web. Springer, London, 2007.

[31] L. Harada, Y. Hotta, and T. Ohmori. Detection of Sequential Patterns of Events for Supporting Business Intelligence Solutions. In *IDEAS*, pages 475–479, July 2004.

[32] L. B. Holder, D. J. Cook, and S. Djoko. Substructure Discovery in the SUBDUE System. In *AAAI*, pages 169–180, Seattle, WA, Aug. 1994.

[33] D. Jensen and J. Neville. Data Mining in Social Networks. *National Academy of Sciences workshop on Dynamic Social Network Modeling and Analysis*, 2002.

[34] M. Kuramochi and G. Karypis. Finding Frequent Patterns in a Large Sparse Graphs. *Data Mining and Knowledge Discovery*, 11(3):243–271, 2005.

[35] U. Leser. A Query Language for Biological Networks. *Bioinformatics*, 21(Suppl 2), 2005.

[36] S. Liu, F. Wang, and P. Liu. A Temporal RFID Data Model for Querying Physical Objects. Technical report, Technical Report: Time Center, 2007.

[37] G. Ozsoyoglu and R. Snodgrass. Temporal and Real-Time Databases: a Survey. *Knowledge and Data Engineering, IEEE Transactions on*, 7(4):513–532, 1995.

[38] M. Perry, F. Hakimpour, and A. Sheth. Analyzing Theme, Space, and Time: an Ontology-Based Approach. In *GIS*, pages 147–154, Arlington, VA, Nov. 2006.

[39] B. Pincombe. Anomaly Detection in Time Series of Graphs using ARMA Processes. *ASOR BULLETIN*, 24(4):2, 2005.

[40] J. Richardson. Supporting Lists in a Data Model (A Timely Approach). *Proceedings of the 18th International Conference on Very Large Data Bases*, pages 127–138, 1992.

[41] P. Seshadri, M. Livny, and R. Ramakrishnan. Sequence query processing. *ACM SIGMOD Record*, 23(2):430–441, 1994.

[42] P. Seshadri, M. Livny, and R. Ramakrishnan. SEQ: A model for sequence databases. *Data Engineering, 1995. Proceedings of the Eleventh International Conference on*, pages 232–239, 1995.

[43] L. Sheng, Z. Ozsoyoglu, and G. Ozsoyoglu. A Graph Query Language and its Query Processing. *Data Engineering, 1999. Proceedings., 15th International Conference on*, pages 572–581, 1999.

[44] T. Shlomi, D. Segal, E. Ruppin, and R. Sharan. QPath: A Method for Querying Pathways in a Protein-Protein Interaction Network. *BMC Bioinformatics*, 7(1):199, 2006.

[45] P. Showbridge, M. Kraetzl, and D. Ray. Detection of Abnormal Change in Dynamic Networks. *Information, Decision and Control, 1999. IDC 99. Proceedings. 1999*, pages 557–562, 1999.

[46] R. Snodgrass. *The Tsql2 Temporal Query Language*. Kluwer Academic Pub, 1995.

[47] B. Tauzovich. Towards temporal Extensions to the Entity-Relationship Model. *Proc. 10th. Int. Conf. on Entity-RelationshipApproach*, pages 163–179, 1991.

[48] J. Ullmann. An Algorithm for Subgraph Isomorphism. *Journal of the ACM (JACM)*, 23(1):31–42, 1976.

[49] F. Wang and P. Liu. Temporal Management of RFID Data. *Proceedings of the 31st international conference on Very large data bases*, pages 1128–1139, 2005.

[50] T. Washio and H. Motoda. State of the art of graph-based data mining. *ACM SIGKDD Explorations Newsletter*, 5(1):59–68, 2003.

[51] M. Yuan. Research Article Use of a Three-Domain Representation to Enhance GIS Support for Complex Spatiotemporal Queries. *Transactions in GIS*, 3(2):137–159, 1999.

[52] C. Zaniolo. *Advanced Database Systems*. Morgan Kaufmann, 1997.