# A Task Taxonomy for Temporal Graph Visualisation

Natalie Kerracher, Jessie Kennedy, and Kevin Chalmers

**Abstract**—By extending and instantiating an existing formal task framework, we define a task taxonomy and task design space for temporal graph visualisation. We discuss the process involved in their generation, and describe how the design space can be 'sliced and diced' into multiple overlapping task categories, requiring distinct visual techniques for their support. The approach addresses deficiencies in the task literature, offering domain independence, greater task coverage, and unambiguous task specification. The taxonomy and design space capture tasks for temporal graphs, and also static graphs, multivariate graphs, and graph comparison, and will be of value in the design and evaluation of temporal graph visualisation systems.

**Index Terms**—Taxonomies, Graph/Network Data, Time Series Data

## 1 Introduction

Temporal graph visualisation is a rapidly growing area concerning the challenges involved in visually representing change in a graph over time. Understanding the mechanisms involved in temporal change in a graph is of interest to a wide range of disciplines, from social and biological sciences, to computer networking, telecoms and transportation, to business and marketing. While the application domain may differ, many of the underlying questions regarding the properties of the graph and mechanism of change are the same. However, despite recent efforts e.g. [1], [2], [3], and a growing number of visualisation tools, no suitable task taxonomy exists for temporal graph visualisation.

Task taxonomies play a vital role in the design and evaluation of visualisation systems, as they reveal and categorise the possible range of tasks that can occur. While a number of general visualisation task taxonomies exist ( [4], [5], [6], [7], [8]), data specific classifications are still required [7] to supply benchmark tasks in evaluation [9], [10], and to support the design process. Visualisation systems are frequently used to carry out Exploratory Data Analysis (EDA). Andrienko [11] makes a strong case for the need to specify tasks when designing for EDA: data analysis is task driven; tasks motivate the analysis, determine the choice of data and methods, and affect the interpretation of the results. However, in EDA, the questions are not always known at the outset; Fekete et al. [12] state that '*the exploratory process itself may influence the questions and tasks that arise*'. System designers must therefore anticipate potential questions and tasks, in order to make an informed decision regarding which tools to include and to ensure that a sufficiently wide range of tasks can be supported. Thus, we propose a task taxonomy and design space that reveals the possible range of tasks for temporal graph data.

The main contributions of this paper are as follows:

- An extension to an existing formal task framework, to handle graph data (Section 3).
- A temporal graph task taxonomy, produced by applying the extended framework to temporal graph data (Section 4).
- A design space of potential temporal graph tasks, produced by combining task dimensions using a series of matrix structures (Section 4.2).
- A discussion of useful task categories which can be 'sliced and diced' from these structures (Section 6).

We evaluate our framework in terms of methodology adopted and task coverage (Section 5), demonstrating more complete task coverage than existing task taxonomies. We begin by presenting previous work on visualisation task taxonomies and temporal graph tasks, and an overview of the Andrienko task framework on which our work is based.

## 2 Related Work

We discuss tasks in the temporal graph literature according to paper type: systems and technique papers, user studies evaluating visualisation techniques, and task taxonomy papers.

We recently surveyed 95 papers relating to temporal graph visualisation systems and techniques [13]. With a number of exceptions (e.g. [2], [14], [15]), the discussions of tasks in these papers are limited, and often couched in domain specific terms. For example, Erten et al.'s [16] tasks relate to a specific co-authorship network: *What were the hottest topics in computing in*

- *N. Kerracher is with the Institute for Informatics & Digital Innovation at Edinburgh Napier University, Edinburgh UK. E-mail: n.kerracher@napier.ac.uk*
- *J. Kennedy and K. Chalmers are also with the Institute for Informatics & Digital Innovation.*

*the 1990's? Which research areas are experiencing steady decline/rapid increase? Which research communities are open and well-connected?*, while Gloor and Zhao's [17] tasks refer to their interest in communication technologies: *Do social networks depend on the interaction technology? Does the same group of people exhibit different network attributes when interacting via telephone, email, face-to-face or other?* Additionally, systems often have a specific purpose, thus tasks are constrained to a particular subset e.g. Kang et al. [18] specifically focus on analysis of change in group membership of a pair of individuals over time. Munzner [19] notes that the use of the term 'task' in the visualisation literature is "deeply overloaded". This reflects our findings of the wide range of tasks in the literature specified at varying levels of abstraction and granularity.

User studies tend to employ only a limited number of the range of possible tasks. Ghoniem et al.'s [20] study of the readability of matrix and node-link representations formulates seven generic tasks for static graphs concerned with gaining an overview of a graph's structure, including estimating the number of nodes and edges in the graph, and finding particular nodes, links, and paths. They do not consider tasks involving node or edge attributes. Purchase and Samra's [21], and Archambault et al.'s [22] studies investigating the mental map involve five tasks, which consider global and local graph structures; the evolution of node degrees; node and edge appearance/endurance; growth in number of nodes; and the readability of paths over time. Farrugia and Quigley [23] distinguish four task categories based on a combination of level of analysis (global network overview vs local individual node level) with temporal search space (specified vs unspecified time period), in conjunction with the static graph tasks of Lee et al. [9], a selection of which they formulated for the dynamic context. However, they give only a few examples of the tasks.

Several general visualisation task taxonomies have been developed, such as Shneiderman's [5] task by data type taxonomy, and Amar et al.'s [4] taxonomy of low level tasks. Recent work has focussed on collating existing taxonomies [7], [8], however, there is still a need for data specific classification [7]. For static graphs, Lee et al. [9] extend [4], and categorise tasks into five groups: topology based, attribute-based, browsing, overview, and high level tasks. While they include the general question, *'how has the graph changed over time?'*, in their high-level task category, they do not elaborate on the sub tasks involved. Shneiderman and Aris [15] identify six challenges for network visualisations and a number of high priority tasks, such as counting nodes and links, finding structural metrics, and structural and attribute based tasks similar to those described by Lee et al. [9]. They do not consider temporal graph tasks.

There are three main taxonomies specifically for temporal graph tasks: Yi et al.'s [2] categorisation of visual tasks in temporal social network analysis; Ahn et al.'s [1] taxonomy for network evolution analysis; and Bach et al.'s [3] adaption of Peuquet's [24] spatio-temporal task framework for temporal graph data. We discuss these taxonomies and how they relate to our classification in Section 5.2.

Beck et al. [25] suggest matching dynamic graph visualisation tools to application requirements through the use of profiles, rating them with respect to the set of aesthetic criteria for dynamic graph visualisation. However, the task and technique domains have not yet been fully explored, specified, and categorised, therefore their methodology requires expert knowledge of the range of available techniques and possible tasks. We believe two key pieces are still needed to implement their methodology: a classification of the available tools and techniques, and a comprehensive classification of the task domain.

### 2.1 The Andrienko Framework

The Andrienko framework [11] consists of a data model and task framework. The task framework (Section 2.1.2) takes a functional approach to specifying tasks. There are two components to every task: the target (unknown information) to be obtained, and the constraints (known conditions) that information needs to fulfil; a task therefore involves finding a target given a set of constraints. The data model (Section 2.1.1 ) identifies the data items that can participate in tasks as a target or constraint. We illustrate the concepts of the Andrienko framework with reference to an example author publication data set.

#### *2.1.1 The Andrienko Data Model (ADM)*

The ADM divides a data structure into two component types: referential or characteristic (Figure 1, top). For any particular data set, the referential components (aka referrers) are how the data is being categorised: in our temporal publication collection, we have chosen to deliberately collect information by year and by author (Figure 1, bottom). Characteristic components are what data is then being measured using this categorisation: in this case publication count and the department to which authors belong. Referential components tend to be time, space and/or populations (objects) because these are the most common dimensions by which data is structured. The ADM also includes the notion of a data function between referential and characteristic components; this is a simple look-up mapping, i.e. $f_{\text{department}}(\text{A}, 2014) = \text{Computing}$.

Both referential and characteristic components have different intrinsic properties - described in the ADM as "relations" between elements - dependent on their underlying type e.g. time is ordered, continuous, and distances can be calculated, so 2014 is 2 years later than 2012. Populations of objects such as authors and departments are unordered and discrete, that is we

cannot say A is less than B, and halfway between A and B makes no sense. These relations also determine reference subsets, e.g. the ordering relations in time determine the set of time points which fall within the interval 2010-2014. Thus we have intervals and cycles in time; areas and lines in space; and groups of objects in a population. These subsets can also have relations between them e.g. time interval 2008-2012 overlaps with 2010-2014.

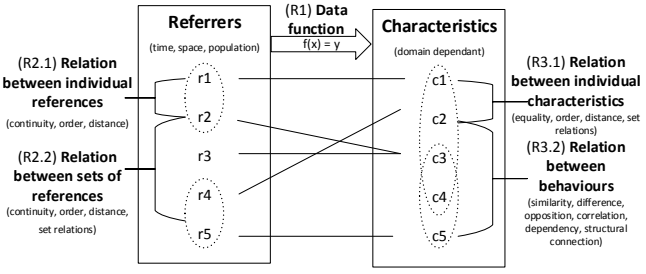
Behaviours are configurations of characteristic values which are decided by the relations within the associated set of references. For example, the ordering relations between time points determine the order in which the corresponding attribute values appear: an author's publication count in 2005, comes before that of 2006. A simple metaphor is to think of ordering a data table by the values in one column, which then decides the configuration of values in another column. Patterns are subjective constructs that describe the *"essential features of a behaviour... in a substantially shorter and simpler way than specifying every(thing)"* [11] p.85. For instance, in Figure 1, the pattern of the publication count for author A between 2012 and 2014 is that it increases; it decreases for B in the same time period. A pattern thus may be something a user observes (as in the previous example), or a rule learned by analytical means e.g. a line of best fit found by automated analysis. Finally, relations can exist between behaviours (and by extension, patterns) e.g. the behaviour and pattern of A's publication count is opposite to that of B.

To summarise: the data items that may participate in tasks are **individual references** (a year, a point in space, an object); **reference sets** (a time interval, an area in space, a set of objects); **individual characteristics** (attribute values - 10, first, red, x-small); **behaviours** - the configuration of a set of characteristics with respect to the reference set, which can be described by a **pattern** (e.g. a temporal trend, a distribution in space, frequency of values in a population); and **relations**, of which there are five types (see Figure 2): [R1] between references and characteristics (the data function); within the referential component, [R2.1] between individual references (order, distance, continuity) and [R2.2] between reference sets (as for R2.1, plus set relations); within the characteristic component, [R3.1] between individual characteristics (data dependent - equality, order, distance, set relations) and [R3.2] between behaviours (similarity, difference, opposition, correlation, dependency and structural connection).

| Referential Component | Characteristic Component |
|---|---|
| A Reference Value | A Characteristic Value |
| A Reference Value | A Characteristic Value |
| ... | ... |

| Author | Year | Publication count | Department |
|---|---|---|---|
| A | 2014 | 5 | Computing |
| A | 2013 | 2 | Computing |
| A | 2012 | 1 | Computing |
| B | 2014 | 2 | Computing |
| B | 2013 | 3 | Biology |
| B | 2012 | 4 | Biology |
| ... | ... | ... | ... |

Fig. 1: The ADM divides data into referential and characteristic components




Summary of relations between different referrer types

| | Time | Space | Population | Graph |
|---|---|---|---|---|
| Order | X | | | |
| Distance | X | X | | X |
| Continuity | X | X | | |
| Linking | | | | X |

Fig. 2: Relations between components of the Andrienko data model (extension highlighted in table)

### 2.1.2 The Andrienko Task Framework (ATF)

Tasks in the ATF are distinguished based on the data items (Section 2.1.1) that participate in them. Firstly, in terms of the level of analysis: elementary tasks involve individual data items (individual characteristics and references) while synoptic tasks involve sets of items considered together as a unified whole (i.e. reference sets and behaviours of attribute values). Synoptic tasks are further divided into descriptive tasks (concerned with describing the data) and connectional tasks (concerned with finding connections between phenomena - see below). Secondly, tasks are distinguished depending on which data items participate as targets or constraints, giving rise to three main task types: lookup, comparison and relation seeking. A summary of the task categories and examples are given in Figure 3.

In **lookup** tasks, referential and characteristic components participate as targets or constraints. On elements, lookup involves finding a characteristic given a reference (*direct lookup*) or references given a characteristic (*inverse lookup*). On sets, finding the pattern associated with the behaviour of an attribute over a reference set (*behaviour characterisation*), and inversely, finding the set of references corresponding to a given pattern (*pattern search*). In **comparison**, the task target is the relation between two or more specified components; either between characteristics or patterns (*direct comparison*), or references or reference sets (*in-*

| Task type | | Target | Constraint | Elementary (example) | Synoptic (example) ** |
|---|---|---|---|---|---|
| **Lookup** | Direct | characteristic | referential | Which department did Author A belong to in 2012? | What is the trend in Author A's publication counts 2012-2014? |
| | Inverse | referential | characteristic | Which author(s) had more than 4 publications in any year? | Find authors who move frequently between departments. |
| **Comparison** | Direct | relation | characteristic* | Compare the publication counts of Authors A and B in 2014. | Compare the trend in Author A's publication counts for 2012-2014 with the trend for 2009-2012. |
| | Inverse | relation | referential* | Did Author A's highest publishing count occur before or after his lowest? | Compare the time periods over which Author A's publication counts were increasing with the time periods over which they were decreasing. |
| **Relation seeking** | | characteristic/ referential* | relation | Find the year in which Author B moved departments (i.e. consecutive years where Author B belonged to two different departments). | Find authors with similar patterns in movement between departments. |

*at least one of these components is found via a lookup task **examples given are for descriptive tasks

Fig. 3: Summary of the ATF. Tasks are distinguished according to the combinations of data components which participate: individual elements or sets of elements (elementary/synoptic tasks) and whether they participate as targets or constraints (task types).

*verse comparison*). **Relation seeking** is the opposite of comparison, where the target is the components associated with a given relation. Further variations of comparison and relation seeking tasks are specified depending on additional task constraints (see Section 4.1). All of these tasks involve the data function either alone (lookup) or as a subtask (comparison and relation seeking).

One final, but important, set of tasks are the connection discovery tasks. These tasks are also attribute based, but do more than describe the occurrence of phenomena (as with descriptive tasks); their aim is to find indications of possible connections or relations within or between phenomena. In these tasks, we are interested in two or more behaviours with respect to each other; these are termed 'mutual' behaviours, and can be described using one of three 'linkage patterns': correlation, dependency or influence, or structural connection (the interplay of two components e.g. a trend over time and variation over seasons). Connection discovery tasks can be formulated for each of the three main task types. Figure 7, based on Aigner et al. [26] illustrates the structure of the task framework.

#### 2.1.3 Limitations of the Andrienko Framework

Although the Andrienko framework is intended to be applicable to all types of data, it does not consider graph data. In our author publications data set, we may wish to extract and consider a co-authorship network (Figure 4).

Modelling co-authorship edges proves difficult under the existing framework, the problematic question being: *what type of data item is an edge*? We suggest these be modelled as relations between references. However, the relations between references of the main referrer types considered under the ADM (see table in Figure 2) are insufficient to describe edges. To illustrate: in our co-authorship network, authors are references. As they are clearly neither temporal nor spatial in nature, population is the remaining option for referrer type. The elements of a population referrer are discrete, unordered, and without distance. While these relations are appropriate when considering an unconnected set of objects, they are not sufficient to capture the co-authoring relations (edges) which exist between authors. We therefore extend the ADM by introducing a new referrer type (graph) and a new type of relation (linking), to model the edges of a graph. As a result of this extension we also introduce a set of structural tasks for use with the graph referrer type (see Section 3.2).

| Author | Year | Publications | Publication count | Department |
|---|---|---|---|---|
| A | 2014 | **a**, **b**, c, d, e | 5 | Computing |
| A | 2013 | **f**, g | 2 | Computing |
| A | 2012 | h | 1 | Computing |
| B | 2014 | **a**, **b** | 2 | Computing |
| B | 2013 | **f**, i | 3 | Biology |
| B | 2012 | j, k, l, m | 4 | Biology |
| ... | ... | | ... | ... |

Fig. 4: A co-authorship network can be extracted based on authors who have publications in common

## 3 Extending the Andrienko Framework for Graph Data

### 3.1 Extension to data model

We introduce a new type of referrer, 'graph', the elements of which are nodes. The graph referrer is distinguished from space, time, and population, by the type of relations which exist between its elements (table, Figure 2): graph is discrete, unordered, with distances. Additionally, nodes in a graph are related by edges. We therefore make a further extension to the framework by introducing a new type of relation between references, 'linking', which exists only between the elements of the graph referrer type and are specified by the edge set. A linking relation may

exist between two nodes, have a direction, weight, and possibly domain specific properties. Unlike the other types of relation, linking relations are not fixed, and may change over time in terms of their existence, weight, and domain properties. Indirect connections, or transitive relations, also exist between elements, and can be described by the chain of linking relations. They have the same properties as direct linking: existence, a direction, and possibly a weight/domain property (or some aggregated notion of weight based on the weights of the individual linking relations), plus a distance between elements. The distance relation in graphs is closely related to linking relations, as it is also specified by the edge set: distance between two elements can be defined as the geodesic distance and may take into account edge weights.

Treating edges as relations allows us to include them in tasks in the same way that we treat order and distances in time. For example, in inverse comparison we find the relations between references: *did Author A have his highest publication count before or after his lowest?* asks us to find which year came first (ordering relation) and possibly by how long (distance relation). In the graph case, *are the authors with the highest and lowest publication counts connected?* involves finding the linking and distance relations between two nodes.

As for the other referrer types, subsets of the graph referrer can be determined by the relations that exist between references. Order determines the time points belonging to a time interval, and analogously, linking relations between nodes form 'graph objects', which include Lee et al.'s [9], graph specific objects: paths, groups, connected components, and clusters; and any subgraph. While the nodes of the graph referrer are unordered, an additional ordering relation is present when dealing with paths and directed graphs.

The relations between graph objects are linking relations, distance, and the set relations (include, overlap, disjoint). For example, two clusters may be linked; their elements may overlap or be entirely disjoint; or one cluster may include other sub clusters. We highlight the extensions to the referrer types of the ADM in the table in Figure 2.

Behaviours (configurations of attribute values) are in part determined by the relations which exist between references. In the same way that temporal trends in attribute values are determined by the ordering relations between time points, in the graph case, behaviours involve the distribution of attribute values over the graph structure, as determined by the linking relations between nodes. Order, continuity, and distance relations are fixed: the order of time points never changes, the distance between two locations remains static; in contrast, linking relations between nodes are not predetermined, and may change over time. We therefore make one final extension to the ADM and introduce a new type of data item - structural behaviours - which are closely related to the original attribute based idea of behaviours. **Structural behaviours** are the configurations of references (nodes) as determined by the linking relations between them; they capture the idea that the same set of nodes may be connected together in different ways, producing different structures. **Structural patterns** describe structural behaviours and include clusters, cliques, motifs and network structures (small world, scale-free etc.).

In summary, to handle graph data, the data model is extended with a new referrer type, graph, whose elements are discrete, unordered, with distances, and has a new type of relation - linking - between its elements. As linking relations in the graph referrer are not fixed, we introduce structural behaviours which are described by structural patterns, and capture variation in graph structure. A summary of how data model terms apply to graphs is given in Figure 5.

| Data model term* | Graph term |
|---|---|
| A reference | node |
| (*linking*) relation | edge |
| A characteristic | an attribute value |
| A reference set | a set of nodes |
| *Structural behaviour* | graph objects e.g. path, cluster, subgraph etc. |
| *Structural pattern* | a cluster, clique, small world network etc. |
| A behaviour | distribution of attribute values over the graph |

*additions to ADM shown in italics

Fig. 5: Relating data model terms to graph terms

### 3.2 Extension to task framework

All of the data analysis tasks in the ATF involve the data function i.e. they always require at least one lookup task involving attribute values. A number of 'pure' forms of relational tasks involving only the referential component of the data are described, however, these are not considered to be data analysis tasks. Under the ADM, where relations between references are fixed, it never makes sense to perform tasks without reference to the data function; answering *how are years 1980 and 1981 related?* does not provide us with any new insight: 1980 is always prior to 1981 and we do not require a data set to know this. However, in our extended model, the non-fixed, possibly temporally changing linking relations between elements of the graph referrer introduce a level of unpredictability into the data set; it therefore makes sense to ask *are nodes p and q of the graph connected to one another?* or *what is the distance between nodes p and q?* We also require tasks which allow us to investigate the structural behaviours, such as *find connected components*, *to which cluster does node p belong?* or *how does the connection between nodes p and q change over time?* We include these pure forms of relational tasks for the graph case and refer to them as **structural tasks** (Figure 6). We base the task categories on those of the original framework. In addition, there is a set of tasks involving lookup, comparison, and relation seeking on edges e.g. *find*

| | Elementary | Synoptic* |
|---|---|---|
| **Lookup** | n/a | **Structural behaviour characterisation**: find a pattern to describe the configuration of connections between a set of graph elements, such as a particular motif or graph structure. E.g. *what is the co-authoring pattern of authors in the Computing department?*<br>**Structural pattern search**: find the set of graph elements associated with a given pattern of connections. E.g. *which authors belong to the small densely connected cluster?* |
| **Comparison** | **Find connections between elements**: given nodes, find whether/in what way they are connected e.g. *did Authors A and B co-publish?* | (Direct) **Find the relation between structural patterns** (similar/different/opposite) associated with given sets of graph elements e.g. *compare the co-authoring pattern of authors in Biology with that of the Computing department.*<br>(Inverse) **Find the relation between the sets of graph elements** associated with given patterns (linking, distance, set relations) e.g. *how are the two largest co-authoring clusters related?* |
| **Relation Seeking** | **Find elements connected in a given way**: The linking relation between graph elements is given, which may be specified generally e.g. whether or not any connection exists between objects; or specifically, including the distance, direction, and/or domain attributes. In addition, a node may also be specified. E.g. *find authors with whom Author A co-published*. | **Find structural patterns related in a given way** e.g. *find instances of the same network motif, or find other co-authoring clusters similar to that of co-authoring cluster A*.<br>**Find subsets of graph elements associated with given patterns which are related in a given way** e.g. *find closely connected co-authoring clusters.* |

**Note there are many permutations of these tasks, mirroring those of the original attribute based tasks, outlined in Section 4.1.*

Fig. 6: Structural tasks: tasks involving only the structure of the graph (categories based on the ATF).

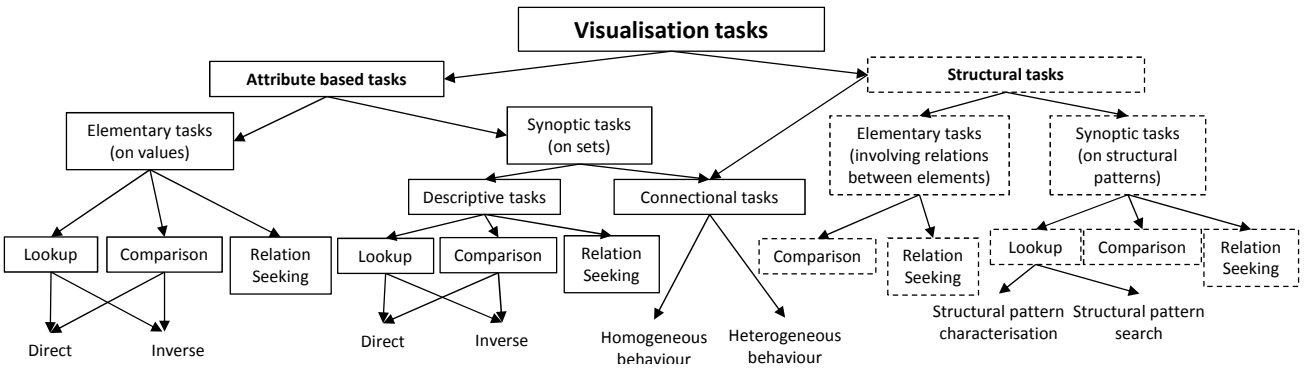



Fig. 7: Structure of the ATF, based on Aigner et al.'s [26] (p74) drawing of the Andrienko task model organised into a taxonomy, redrawn and extended to include structural tasks (indicated by dotted lines - see Section 3.2)

*co-authoring relations with a weight of 4*; *compare the co-authoring relationship between Authors A and B with that of B and C*; *find pairs of authors with similar co-authoring relationships*. However, tasks involving relations in this way do not exist within the ATF and we do not wish to add more task categories than necessary; in this case we treat relations as references and use the attribute based tasks of the ATF e.g. direct comparison to compare attributes associated with an edge or path. The extended task framework is shown in Figure 7.

The non-fixed relations between references of the graph referrer do not have major implications for the descriptive tasks of the original framework; we simply consider linking relations when formulating comparison and relation seeking tasks. For example, *do authors with the highest publication counts co-publish?*, is an inverse comparison task: we perform lookup tasks to find the authors, then find the linking relation between them. The non-fixed relations do, however, introduce additional possibilities to the set of connection discovery tasks: investigating the effect of graph structure on attribute values, and vice versa; how structural patterns at one point in time affect structural patterns at a later time; or the relationship between the changing structural patterns of two different networks. We discuss these in Section 4.3.

## 4 Temporal Graph Task Taxonomy

Having extended the Andrienko framework to handle graph data, we now seek to use it to elucidate the range of possible tasks involved in exploring *temporal* graph data. As discussed, task categories in the Andrienko framework are intentionally generic in order to be utilised with any type of data. The approach follows Bertin [27] in classifying tasks on the basis of the structure of the data, considering the level of analysis (synoptic/elementary distinction), and type of data item (referential components, characteristic

components, relations) participating as either task targets or constraints. We now make one further data-based distinction (essentially a sub-classification of synoptic tasks) for temporal graph data, classifying tasks according to the combinations of referential components which participate: time points, time intervals, graph elements, graph subsets (Figure 8).

This produces four classes of tasks involving very different data items, which will likely require significantly different visual representations. The classes capture the Andrienko elementary/synoptic distinction, along with three variants of synoptic tasks: elementary tasks (Q1), tasks considering graph subsets (Q2), temporal subsets (Q3), and both graph and temporal subsets (Q4). Each combination also has different characteristic components and relations associated with it. In particular, the behaviours which participate in tasks are very different. There are eight behaviours in total, four attribute based (A) and four structural (S):

**Q2** tasks involve the behaviour of an attribute over a set of nodes at a single time (A2) e.g. the distribution of an attribute value (such as publication count) over the network; and the configuration of nodes based on the linking relations between them, at a single time (S2) e.g. clusters, cliques, motifs, co-authoring groups.

**Q3** involves the behaviour of an attribute of an individual graph element (a node, edge, or graph object) over time (A1) e.g. a temporal trend in the attribute of a node such as an individual author's publication count over time; and the behaviour of linking relations between two graph elements over time (S1) e.g. the pattern of change in connectivity between two nodes over time, such as the temporal pattern of co-authorship between two authors.

**Q4** has four possible behaviours associated with it: (A3) the behaviour of the temporal trends (described by A1) distributed over the graph e.g. the distribution of individual temporal trends in author publication counts, over the graph ( [14] is an example of visualising such behaviours). (A4) the behaviour of the distribution of the attribute values over the graph (as in A2), over time e.g. the change in distribution of research group affiliation over the co-authorship network, over time. (S3) the behaviour of the collection of behaviours in S1 i.e. the aggregate pattern of all linking relations between pairs of graph objects over time, or the distribution of individual temporal behaviours over the graph e.g. the distribution of temporal trends in co-authorship between pairs of authors, over the network ( [28] is an example of visualising this behaviour). (S4) the configurations of nodes (i.e. S2), over time e.g. the evolution of the structure of the co-authorship network over time. Applying the three task categories to each quadrant produces the main categories of (descriptive) tasks in our task taxonomy; we summarise these in Table 1. It is clear that very different visual approaches will

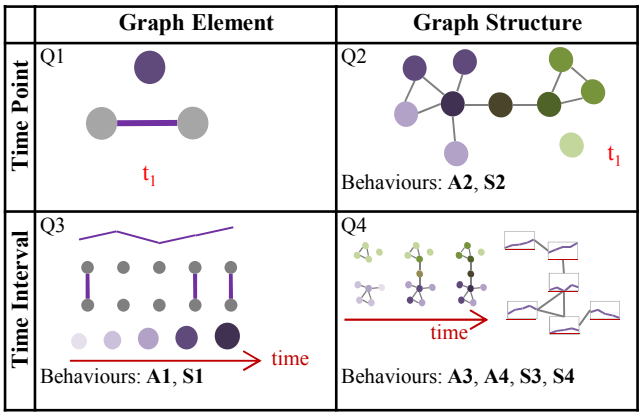



Fig. 8: Data items organised according to referential components

be required to carry out the same task type in each quadrant. For example, direct comparison in Q2 involves comparing attribute distributions of graphs, in Q3, comparing temporal trends in individual attribute values, and in Q4 comparing evolution of attribute distributions over the graph over time or distributions of temporal trends in attribute values over graphs.

While some of these tasks (such as those of Q3) are not obviously temporal graph tasks, *any* of these tasks may be required when exploring temporal graph data, and tasks from each quadrant may be necessary to contribute to our overall understanding of the data. Andrienko discuss behaviours as 'partial' (e.g. A1, A2) - associated with individual data items - and 'aspectual' (e.g. A3, A4) - which consider only certain aspects of the 'overall' behaviour (i.e. all behaviours over the entire data set). They demonstrate that two aspectual behaviours are neither equal to one another nor the same as the overall behaviour, thus we obtain only partial understanding of the overall behaviour and underlying phenomena through their study (see [11] pp.99-107). This underlines the importance of including tasks from all four quadrants when exploring temporal graph data, and as a result, the likely need for different techniques in order to support them. It also points to the need for tools to help us piece together our partial findings.

Note it is possible that a distinction based on the combinations of referential components involved could also be useful when classifying tasks for other types of data involving multiple referrers.

### 4.1 Sub-variations within task categories

As discussed, the data items participating as targets and constraints distinguish the lookup, comparison, and relation seeking task categories. The Andrienko framework discusses in-depth a number of variants within these categories arising from specifying additional data items as constraints, or from particular properties of the data items participating in the task.

| | Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|---|
| **Direct lookup/ behaviour characterisation** | Find a graph object's attribute value at a single time point | Describe the pattern of attribute values associated with a set of nodes, at a single time point; Describe the structural pattern for a given set of nodes, at a single time point | Describe the temporal trend of a node's attribute value; Describe the pattern of connectivity between a pair of nodes, over time | Describe the attribute distribution over the graph/subgraph over time; Describe the structural pattern of the graph/subgraph over time; Describe the distribution of temporal trends in node attributes, over the graph; Describe the distribution of temporal trends in connectivity between pairs of nodes, over the graph |
| **Inverse lookup/ pattern search** | Find the graph object(s)/time point(s) associated with an attribute value | Find the set(s) of nodes associated with a given attribute pattern...; Find the set(s) of nodes associated with a given stuctural pattern...and/or the time point(s) at which the pattern occurs | Find the node(s) having a particular temporal trend in attribute value...; Find the node(s) having a particular pattern of connectivity...and/or the timeperiod(s) over which the pattern occurs | Find the graph (subset(s)) and/or time interval(s) over which one of the above patterns occurs |
| **Direct comparison** | Compare attribute values (resulting from direct lookup tasks) | Compare patterns of attribute values over the graph; Compare structural patterns | Compare temporal trends in attribute values; Compare patterns of connectivity over time | Find the graph (subset(s)) and/or time interval(s) over which one of the above patterns occurs |
| **Inverse comparison** | Compare graph objects/time points (resulting from inverse lookup); Find the relation (connectivity) between graph objects | Compare the time points at which patterns of attribute values, or structural patterns, occur; Compare the sets of nodes associated with particular attribute/structural patterns (this includes finding linking relations) | Compare the time periods over which patterns occur; Compare nodes having particular trends in attribute values; Compare pairs of nodes having particular patterns of connectivity | Find attribute patterns (and possibly the corresponding nodes/time point(s)) related in a given way; Find structural patterns related in a given way |
| **Relation seeking** | Find attribute values (and possibly the corresponding graph element(s)/time point(s)) related in the given way; Find nodes related (connected) in the given way | Find attribute patterns (and possibly the corresponding nodes/time point(s)) related in a given way; Find structural patterns related in a given way | Find temporal trends in attribute values (and possibly the corresponding node(s)/time periods) which are related in a given way; Find temporal trends in connectivity (and possibly the corresponding node(s)/time periods) which are related in a given way | Find patterns related in a given way (and possibly the corresponding graph subsets/timeperiods) |

TABLE 1: Summary of descriptive tasks for temporal graphs

Temporal graph data involves two referrers, which compounds the possible variations. For example, in inverse lookup, where the target is the referential component and the constraint is an attribute component, one or other of the referrers can also be specified, giving three task variations: e.g. *find the years in which author A had more than six publications*; *find the authors who had more than six publications in 2010*; *find any author at any time who had more than six publications* (variations for the synoptic tasks can also be constructed e.g. *find the time period in which author A had an increasing trend in publication count* etc.). As all comparison and relation seeking tasks in the original framework involve at least one lookup task, it is possible to construct a wide variety of tasks simply based on the combinations of differently specified lookup tasks involved e.g. *compare the years in which author A had more than six publications* or *compare the authors who had more than six publications in 2010*. The framework also considers the possibility of comparison with a specified value i.e. where only one lookup task is involved (*compare author A's publication count in 2010 with the average number of publications (5)*).

In the case of multiple referrers, the framework describes variations in comparison and relation seeking tasks involving the same or two different referential components. Three variations are possible for temporal graph data: same graph component, different time (*compare author A's publication count in 2010 with their publication count in 2011*); different graph, same time (*compare author A and B's publication counts in 2010*); different graph, different time (*compare author A's publication count in 2010 with author B's publication count in 2011*). There is also the possibility of tasks involving comparison between different attributes (assuming the value domains are comparable) e.g. *compare the number of journal articles author A published in 2010 with the number of conference papers they published that year*.

These task variations - additional constraints, same or different time/graph components, and same or different attribute - can be combined, producing a huge variety of tasks. Importantly, task variants within the same task type may potentially require support from quite different visual techniques. We therefore sought a systematic way to investigate the possible variations in task, and a logical way to group together similar tasks (i.e. those likely to require similar visual techniques for their support). Our solution is to construct a task design space: we extend the quadrant categorisation (Section 4) by introducing further subdivisions to capture additional constraints, and whether the same or different referential components are involved in tasks, resulting in a set of 'task matrices'. The matrices are very large, with over 144 variations of attribute based tasks alone; we therefore here describe the approach taken and include the full task listing in the supplemental material (also at arXiv:1402.2867).

## 4.2 Temporal Graph Task Design Space

### 4.2.1 Attribute based tasks

We constructed three matrices, one for each of the main task types (lookup, comparison, relation seeking). Each matrix is divided into quadrants based on the time and graph components involved. This

| | | | | Graph | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Elements | | | | Subsets | | | |
| | | | | Both constraints | | One constraint, one target | Both are targets | Both constraints | | One constraint, one target | Both are targets |
| | | | | Same element | Different elements | | | Same subset | Different subsets | | |
| Time | Points | Both constraints | Same time | | | | | | 2 | | |
| | | | Different times | | 1a | | | | | | |
| | | One constraint, one target | | | | | | | | | |
| | | Both are targets | | | | | 1b | | | | |
| | Intervals | Both constraints | Same interval | | | | | | | | |
| | | | Different intervals | 3 | | | | | | | |
| | | One constraint, one target | | | | | | | | | |
| | | Both are targets | | | | | | | | | 4 |

Fig. 9: Comparison task matrix structure. Shaded cells indicate direct comparison (all other tasks are inverse). Task examples relate to numbered cells. For details of the formal notation used, see supplemental material.

**Example attribute based comparison tasks:**
**1a. Direct comparison** Compare the attribute values associated with two different nodes at two different times.
$?y_1, y_2, \lambda$: $f(\mathbf{t}_1, \mathbf{g}_1) = y_1; f(\mathbf{t}_2, \mathbf{g}_2) = y_2; y_1\ \lambda\ y_2$
**1b. Inverse comparison** Find the time points and nodes associated with two given attribute values and compare them.
$?t_1, t_2, g_1, g_2, \lambda$: $f(t_1, g_1) \in C'; f(t_2, g_2) \in C''; (t_1, g_1)\ \lambda\ (t_2, g_2)$
**2. Direct comparison** of the attribute patterns over two different subsets of the graph at the same time point.
$?p_1, p_2, \lambda$: $ß(f(x_1, x_2) \mid x_1 \in \mathbf{G}', x_2 = \mathbf{t}) \approx p_1; ß(f(x_1, x_2) \mid x_1 \in \mathbf{G}'', x_2 = \mathbf{t}) \approx p_2; p_1 \lambda p_2$
**3. Direct comparison** of the patterns of the same graph element over two different time intervals.
$?p_1, p_2, \lambda$: $ß(f(x_1, x_2) \mid x_1 = \mathbf{g}, x_2 \in \mathbf{T}') \approx p_1; ß(f(x_1, x_2) \mid x_1 = \mathbf{g}, x_2 \in \mathbf{T}'') \approx p_2; p_1\ \lambda\ p_2$
**4. Inverse comparison** of graph subsets and time intervals associated with given patterns:
$?G', G'', T', G'', \lambda, \psi$: $ß_G\{ß_T[ß(f(x_1, x_2) \mid x_2 \in T')] \mid x_1 \in G'\} \approx \mathbf{P}_1; ß_G\{ß_T[ß(f(x_1, x_2) \mid x_2 \in T'')] \mid x_1 \in G''\} \approx \mathbf{P}_2; T'\ \lambda\ T''; G'\ \psi\ G''$;
***OR*** $?G', G'', T', G'', \lambda, \psi$: $ß_T\{ß_G[ß(f(x_1, x_2) \mid x_2 \in G')] \mid x_1 \in T'\} \approx \mathbf{P}_1; ß_T\{ß_G[ß(f(x_1, x_2) \mid x_2 \in G'')] \mid x_1 \in T''\} \approx \mathbf{P}_2; T'\ \lambda\ T''; G'\ \psi\ G''$;

distinguishes the elementary and synoptic tasks, with elementary tasks appearing in Q1, and the three variations of synoptic tasks in Q2-4 (Figure 8). Each quadrant is then subdivided according to whether the time and graph components are specified (constraints) or unspecified (targets). This captures the inverse/direct task distinction in lookup and comparison tasks, with direct tasks appearing in the top left of each quadrant; comparison and relation seeking with a specified component also naturally emerges where all elements of one of the lookup tasks are specified.

In the comparison and relation seeking matrices, an additional subdivision is made relating to whether the same, or two different, temporal and/or graph components participate. The majority of tasks in these matrices can be formulated to involve the same or two different attributes (only those involving the same time and graph components cannot involve the same attribute, and we note these cases in the matrices).

We include an excerpt from the comparison task matrix (Figure 9) to show the structures used and variations in tasks within and between quadrants, along with examples of the formal notation used to describe tasks, full details of which is given in the supplemental material.

#### 4.2.2 *Structural Tasks*

As for the attribute based tasks, tasks are first divided based on the referential components involved. The structural elementary tasks (Q1) are more limited than their attribute based counterparts. They are distinguished according to the combinations of time/graph elements participating as targets or constraints (Figure 10). The variations of the synoptic tasks are the same as their attribute based counterparts, but involve structural behaviours and patterns associated with reference subsets in place of behaviours and patterns of attribute values. We therefore do not repeat the task matrices for these tasks.

### 4.3 Connection Discovery Tasks

So far we have discussed only descriptive tasks. Connection discovery tasks involve relational behaviours (correlation, dependence etc.) and seek to find indications of possible relations between the parts of a single phenomenon (homogeneous behaviours) or between two or more phenomena (heterogeneous behaviours). Three variations of relational behaviours are given in the Andrienko framework; we discuss these for the case of temporal graphs. We also discuss relational behaviours involving graph structures. Tasks involving relational behaviours can be formulated for each of the three task types. As the Andrienko framework does not discuss how to handle multiple referrers in relational behaviours, we draw on the partial behaviours to guide us.

**Relational behaviour involving two (or more) different attributes of the same reference set**: Applied to temporal graphs, this task considers a relational behaviour between two different attributes and the same graph and temporal components. One example is the relational behaviour between two different attributes of the elements of the graph at a single time point

| | | Graph elements (nodes, graph objects) | | |
|---|---|---|---|---|
| | | **Both constraints** | **One constraint, one target** | **Both are targets** |
| **Time points** | **Both constraints** | **Find connections between elements (comparison)** (How) is graph element $\mathbf{g_1}$ connected to graph element $\mathbf{g_2}$ at the given time, **t**?<br>$?\lambda$: ($\mathbf{g_1}$, **t**) $\lambda$ ($\mathbf{g_2}$, **t**) | **Find elements connected in the given way (relation seeking)** Find the graph element(s) to which graph element $\mathbf{g_1}$ is connected in the given way at time **t**:<br>$? g_2 :$ ($\mathbf{g_1}$, **t**) **Λ** ($g_2$, **t**) | **Find elements connected in the given way (relation seeking)** Find graph objects which are connected in the given way at the given time:<br>$?g_1, g_2 :(g_1,$ **t**$)$**Λ** $(g_2,$ **t**$)$ |
| | **Both targets** | **Hybrid** Find the time points at which two given graph objects were connected in the given way:<br>$? t :$($\mathbf{g_1}$, $t$)**Λ** ($\mathbf{g_2}$, $t$) | **Find elements connected in the given way (relation seeking)** Find the graph element(s) to which graph element $\mathbf{g_1}$ is connected and the time(s) at which the connection(s) occur:<br>$? g_2, t :$ ($\mathbf{g_1}$, $t$) **Λ** ($g_2$, $t$) | **Find elements connected in the given way (relation seeking)** Find graph objects (and their associated time points) at any time that are connected in the given way:<br>$? g_1, g_2, t :(g_1, t)$**Λ** $(g_2, t)$ |

Fig. 10: Variants of the elementary structural tasks (comparison and relation seeking).

e.g. a correlation between indegree and out degree of all nodes. Further, we might consider the relational behaviour between the indegree and outdegree of the same node over time: we might look for some correlation or dependency in the two temporal trends e.g. an increase in outdegree followed by an increase in indegree. Finally, we might consider the relational behaviour between the two attributes for all graph objects at all time points.

**Relational behaviour involving two (or more) different attributes of different reference sets**: We could apply this to temporal graphs in two ways: (1) Where the reference sets are a graph over time and external events, we might investigate the relational behaviour between an attribute of a graph object and external events (in time), looking at how the attribute values in the graph are influenced by outside events over time. This may be of particular interest e.g. where some form of external intervention in the network is under observation, such as vaccination in a public health network. (2) Where the reference sets are two different temporal graphs, we might investigate the relations between two (possibly different) attributes of two different graphs over possibly different time periods. This behaviour may be of interest where we are investigating two different but related networks e.g. co-authorship networks from different domains, or the energy grid and a computer network. Moreover, we might not only be interested in attribute based behaviours, but also the relation between structural behaviours e.g. Gloor and Zhao [17] are interested in the relationship between networks constructed to reflect different communication mediums (face-face, telephone, email). In this case we might also wish to find some correlation between the structural behaviours of the network itself, for example, during times at which the email network is densely connected, the face-to-face network may be less so.

**Relational behaviour involving the same attributes of different reference subsets**: We might investigate the relations between behaviours over different parts of the graph or behaviours over different time periods (e.g. do particular patterns of values in one area of the graph, or over a particular time period, influence patterns in another?). We can formulate these tasks to consider correlation or influence between attribute values of: (i) different parts of the graph over the same time period; (ii) the whole graph or a graph subset during different time periods; (iii) different parts of the graph during different time periods.

The role of graph structure in relation to attribute values is also of interest: we may investigate whether particular structures influence attribute values or vice versa (e.g. Christakis and Fowler [29] investigate the influence of network structure on obesity). We may also be interested in relational behaviour between graph structures. For example, in social network analysis a number of theories surround tie formation e.g. Yi et al. [2] discuss preferential attachment, accumulative advantage, homophily, follow-the-trend, and multiconnectivity. In all cases we would look at how structural patterns in the graph at one point in time influence the structural patterns at another.

## 5 Evaluation

Various approaches to evaluating task taxonomies exist, including: evaluation with domain experts e.g. [1]; using the taxonomy to design a visualisation system e.g. [6]; case study approach (e.g [7] use task categories to describe the tasks which an existing system supports); evaluation against existing taxonomies e.g [7]. In our evaluation, we consider the methodology adopted in relation to other possible approaches, and assess task coverage against that of existing taxonomies.

### 5.1 Evaluating the methodology

We identify three main steps in constructing a task taxonomy: (1) generate the tasks (2) collate and order them (3) describe them, and discuss the design decisions taken when constructing our task taxonomy.

The first stage in creating a task taxonomy is to gather together the tasks. A number of approaches are described by Schulz et al. [8], including: surveying individuals to generate lists of tasks e.g. [9]; task analysis, involving observation of users of a visualisation; and inferring from existing systems the tasks which can be performed e.g. [1]. Our approach is

based on Andrienko's [11] modelling approach to task specification: data and tasks are represented using formal expressions indicating the task targets and constraints. Through manipulation of these expressions, all possible permutations of tasks are extricated. This approach has a number of advantages. Firstly, the other approaches have limitations in terms of task coverage: relying on user-generated tasks may skew them toward those of the user's domain, and is limited by the individual's perspective; task generation involving existing systems may limit tasks to those that can already be supported by existing visual approaches. In using a formal model, Andrienko is able to demonstrate completeness of the task framework with respect to their chosen data model and level of abstraction. Secondly, using a formal approach ensures task specification at a consistent level of granularity and abstraction, and domain independence, without requiring translation of the generated tasks.

The main drawback of using a formal approach to generate tasks is the lack of user involvement in the process, thus further work is required to establish which tasks are most important to users when carrying out analysis.

Little attention is given in the literature to the task collation stage of constructing a taxonomy. Where task lists are generated by multiple individuals, some process of normalising tasks to a particular level of abstraction, and categorising like tasks together to produce general categories is required. During such a process subtle distinctions between tasks can be lost, and less frequently occurring, 'corner case' tasks may be discarded. By using task matrices (Section 4.2), we can categorise the tasks using a 'slice and dice' approach along the rows and columns: this is useful, as all of the tasks fall into more than one category. It also allows us to maintain the nuanced distinctions between tasks while showing high level categories (see Section 6).

The final stage is task description. Most task descriptions are verbal [8]. Use of a formal notation avoids ambiguity and allows highly nuanced distinctions to be made. Formal notation has the disadvantage that it may be difficult for newcomers to read, so we also include verbal task descriptions.

### 5.2 Evaluating task coverage

There are a number of tasks in our framework for which we can give real world examples and which existing frameworks are not able to capture: comparing the evolution of two graphs, in terms of their attribute distributions and/or structures, over time is comparison in Q4, and captures Gloor and Zhao's task, *does the same group of people exhibit different network attributes when interacting via telephone, email, face-to-face or other?*; tasks considering the distributions of temporal trends over the graph (Q4 behaviours A3 and S3), are reflected in the bioinformatics tasks of interest to Saraiya et al. [14] e.g. *How does the (temporal) behaviour of a particular graph vertex affect other vertices connected directly or indirectly to it?* We here compare task coverage in our taxonomy with existing works.

As Lee et al.'s taxonomy [9] is intended for static graphs, almost all tasks can be positioned in Q1 and Q2; only the high level task *how has the graph changed over time?* considers the temporal dimension. We utilised their notion of graph objects when discussing our data model. While their discussion of static tasks is comprehensive and offers many useful real-world examples, our taxonomy offers a systematic way to specify the possible permutations of these tasks, for example: for tasks involving attributes on nodes, the general description *find the nodes having a specific attribute value* does not consider the opposite, direct lookup task, *find the values of specific nodes*; similarly, their topological tasks are generally phrased for relation seeking, rather than comparison; comparison is only briefly mentioned for the whole graph case, omitting the possibilities amongst individual nodes or edges. They also separate their topology and attribute based tasks into distinct categories. Through the notion of behaviours, our framework makes clear the important relationship between attribute values and graph structure, and includes tasks involving attribute distributions over the graph. A few of their tasks do not fit into our structure, because they do not involve questions about the data. *Follow path* and *revisit* are visual tasks, while *give a meaningful name to a group* involves user interpretation.

Yi et al. [2] categorise visual tasks in temporal social network analysis by the level at which temporal change in the network can be analysed: nodal and dyad level (node or edge attributes, and associations between attributes) subgroup (based on connectivity or node attributes), and global level. They identify the general aspects of interest at each of these levels in relation to network evolution: the emergence, growth and dissolution of nodes and ties, the processes of subgroup formation, and global changes in the networks topology over time. They also note the importance of considering the relationship between attributes and graph structure.

Yi et al.'s levels of analysis are captured in our quadrants: Q1 and Q3 - node/dyad level; Q2 and Q4 - subgroup and global level. They identify a number of interesting questions relating to social network analysis which can be asked at each of these levels, however these are domain specific and specified at a high level: the range of lower level tasks (find, compare etc.) involved in these analyses are not discussed. In contrast, our taxonomy offers a domain independent, low-level breakdown of tasks involved in such analysis.

Bach et al. [3] recently adapted Peuquet's [24] spatio-temporal task framework for temporal graph

data. The original framework consists of three dimensions, when, where, and what; they redefine the where and what dimensions to capture the lack of fixed spatial positions in temporal graph data. Tasks are formulated based on two known dimension values, with the third dimension's value to be found.

This approach is the most similar to that of the Andrienko framework: the 'when' and 'where' dimensions are equivalent to the temporal and graph referrer respectively; 'what' is the attribute component. Tasks are also target and constraint based, capturing both direct and inverse lookup tasks. However, they do not consider relations and attributes as separate data items (these are incorporated into the 'what' and 'where' dimensions). This makes the task framework less complex, but means they are not able to capture comparison and relation seeking tasks, and higher level tasks such as finding correlations and dependencies. In addition, through use of the quadrants, our framework makes clear how the tasks can systematically be applied to different types of data items.

The most comprehensive taxonomy developed to date is Ahn et al.'s [1] taxonomy for network evolution analysis consisting of three dimensions: Entity, Property, and Temporal Feature. 'Entity' follows Yi et al. [2] in their distinction of levels of analysis; 'Property' distinguishes between structural attributes and domain properties. These two dimensions capture what should be observed. 'Temporal Features' explain how these items should be analysed: as 'Individual events' at single time points or 'Aggregate events' over a period of time.

We identified a number of limitations when applying this taxonomy to the design of a visualisation tool. All three dimensions largely describe data items that participate in tasks (nodes/groups/networks, attributes, occurrences of events at time points or over time intervals); a lack of explanation as to the tasks in which these items participate (find, compare etc.) results in ambiguity when applying the taxonomy. The framework also relies on five patterns which describe the 'shape of change' in attribute values over time (growth/contraction, convergence/divergence, stability, repetition, peak/valley). We would suggest that these patterns are too limited, as they are mainly applicable to temporal trends in individual, numeric attribute values, and are not sufficient to capture the relational nature of graph data i.e. changing patterns in graph connectivity and attribute distributions over the graph, over time. This is problematic as one of the key purposes of using graph visualisation is to describe structural change, and attribute values in the graph context. Finally, the taxonomy focusses on evolution, but does not consider the tasks involved in contextualising evolution (e.g. comparing other graph structures with the evolving structure of interest).

Our framework addresses these limitations as follows. We handle temporal change in properties with non-numeric data types through the more general Andrienko notion of pattern, under which Ahn et al.'s 'shapes of change' and 'rate of change' can be subsumed (note these are particularly useful when considering the range of patterns in Q3). We are also able to distinguish the different types of pattern applicable to temporal graph data using the notion of behaviours, and clearly show these using the quadrant view. In Section 6 we outline the tasks in our framework which consider structure and attributes in isolation, however, the majority of our tasks consider graph attributes in a structural context. The formal approach to task specification which we have adopted systematically specifies the operations which can be carried out on the data objects, and encompasses both 'evolutionary' and 'contextual' tasks (discussed further in Section 6).

## 6 Discussion

The matrices of the task design space are constructed based on the categories of the taxonomy, from which a number of overlapping sets of tasks can be easily 'sliced and diced'. The ability to partition tasks in this way is particularly useful when mapping visual techniques to categories of tasks which they support. We here discuss a number of categories which can be distinguished.

Dividing tasks based on the four quadrants is fundamental when selecting the most appropriate visual approach, with techniques found in different research areas: general visual techniques (Q1); static graph visualisation (Q2); temporal visualisation (Q3); temporal graph visualisation (Q4). All of these tasks - and therefore visual techniques from all of these areas - are involved when exploring temporal graph data.

Static graph tasks appear in the rows of the matrices involving a single time point. While Lee et al.'s [9] taxonomy for static tasks is comprehensive, our taxonomy additionally offers a detailed specification of the possible permutations of these tasks, and considers the relationship between attribute values and graph structure through the Andrienko notion of behaviours.

Graph comparison tasks can be clearly identified as the tasks which fall under Q1 and Q2 in the matrices. While *lookup*, *comparison*, and *relation seeking* tasks are all relevant, only elementary tasks and those involving graph structure (i.e. not trends over time) are applicable.

The rows and columns of the matrices neatly capture sets of tasks with same/different graph/time components and additional constraints. These tasks may warrant different visual approaches (e.g. comparison at the same time vs comparison at different times). In conjunction with these task variations, for multivariate graphs we also capture the possibility of comparison between different attributes. Tasks involving a specified pattern or attribute value are noted in

the matrices; in this case we may need some way to visually represent the specified pattern.

A further distinction which emerges is between what we term *evolutionary* and *contextual tasks*. Evolution - the notion of change in some object over time, be it graph structure, the attribute value of an individual node, or the distribution of attribute values over the graph - is often of interest when investigating temporal graph data, as reflected in the predominance of evolutionary tasks found in the literature. The task matrix structures easily distinguish - but do not limit us to consideration of - evolutionary tasks, which involve a combination of the same graph element or subset at different time points or over different intervals. Contextual tasks consider an object in the context of other objects, which may be at the same or different times. The range of such tasks identified in the taxonomy reminds us not to neglect the questions which enable us to situate our findings and bring perspective to our observations, perhaps in turn bringing deeper meaning to our study of evolutionary changes.

An important distinction when selecting a visual representation is task search space. Farrugia and Quigley [23] consider *temporal* search space, distinguishing between local (focussing on a specified time period), and global (searching across the entire time period). We extend this notion to consider the *graph search space*, giving four variations of task search space: no search (time and graph components are specified); graph search (time is specified, graph component is not - requires searching the entire graph); temporal search (graph component is specified, time is not - requires searching the entire time period); graph and temporal search (neither component is specified - requires searching the whole graph at all time points). Search space is independent of the *elementary/synoptic* distinction: even in *elementary* tasks involving a single element at a single time point, where both components are unspecified, our search space extends to the entire graph over the entire time period. The task search space is clearly identified in the task matrices by the columns and rows indicating the specified (constraint) and unspecified (target) time and graph components.

Our final consideration regards the distinction between structural and attribute based tasks. In our taxonomy we introduced an additional category of purely structural graph tasks. While this category sits separately from the attribute based tasks, the picture is more complex than indicated by the existing structural vs attribute distinction made in the task literature: for example, Lee et al. [9] separate their topology and attribute based tasks into distinct categories; Ahn et al.'s [1] 'property' dimension distinguishes structural attributes and domain properties. However, partitioning tasks into those purely involving structure and those purely involving attributes is not helpful when considering visual approaches, as it ignores the middle ground in which many graph based tasks reside i.e. tasks considering attribute values in the context of graph structure. We therefore suggest three categories of tasks:

*Structural* (no attributes involved): solely consider the structure of the graph, without reference to attributes. These are the graph structural tasks identified in the taxonomy. Visualisations supporting such tasks would focus on representing the graph structure.

*Attribute based in a structural context*: consider patterns of attributes over the graph structure and the position in the graph of the occurrence of attribute values. These tasks are captured in the attribute based tasks of the framework using the Andrienko behaviours. Visualisation approaches supporting these tasks require representation of the attribute values in the graph structural context.

*Attribute based*: consider attribute values in isolation from the graph structure. We may only be interested in attribute values associated with a graph in their temporal context e.g. the temporal trends in attribute values of individual elements (behaviour A2). We may also be interested in the frequency distribution (rather than the structural distribution) of the attribute values of all graph elements at a given time point, and how this distribution changes over time. Visualisation approaches which do not involve the graph structure e.g. [30] are appropriate in this case. This category also covers changes in structural metrics, which in themselves capture the structure of the graph, hence do not require an explicit structural representation when visualising them.

## 7 Conclusion

This paper describes the construction of a task taxonomy for temporal graph data through the extension and application of an existing formal task framework. Using matrix structures to combine task dimensions, we outline a task design space which specifies the possible task variants. We also demonstrate the usefulness of these structures in 'slicing and dicing' the tasks into a number of overlapping categories, which require distinct visual approaches for their support.

The use of a formal approach gives us confidence in the completeness of task coverage: we demonstrated this when comparing our tasks with those of existing taxonomies. It allows unambiguous specification of the nuanced distinctions between permutations of a full range of potential tasks, and reveals hidden tasks and corner cases, which may otherwise have been neglected from consideration. Our taxonomy is intentionally domain independent to be of use across any discipline calling for graph visualisation. It not only considers tasks for temporal graphs, but provides tasks for static graphs, multivariate graphs, and graph comparison.

The taxonomy, design space, and additional task categorisations are intended to support the design

and evaluation processes. In our future work, we plan to carry out user-centred studies to establish the tasks most important in real-world scenarios. We also plan to map existing visual techniques to the task categories which they support. Such a mapping will be of further use to designers and evaluators, and may reveal categories of tasks which are currently unsupported by existing techniques, which could provide interesting avenues for future research.

## ACKNOWLEDGMENTS

We thank Natalia Andrienko for her detailed and constructive feedback on an early draft of the taxonomy, and Martin Graham for his input and comments on the manuscript during the review process.

## REFERENCES

[1] J.-w. Ahn, C. Plaisant, and B. Shneiderman, "A Task Taxonomy for Network Evolution Analysis," *IEEE TVCG*, vol. 20, no. 3, pp. 365–376, 2014.

[2] J. S. Yi, N. Elmqvist, and S. Lee, "TimeMatrix: Analyzing Temporal Social Networks Using Interactive Matrix-Based Visualizations," *International Journal of Human-Computer Interaction*, vol. 26, no. 11-12, pp. 1031–1051, 2010.

[3] B. Bach, E. Pietriga, and J.-D. Fekete, "GraphDiaries: Animated Transitions and Temporal Navigation for Dynamic Networks." *IEEE TVCG*, vol. 20, no. 5, pp. 740–754, 2014.

[4] R. Amar, J. Eagan, and J. Stasko, "Low-level components of analytic activity in information visualization," *IEEE InfoVis*, pp. 111–117, 2005.

[5] B. Shneiderman, "The eyes have it: a task by data type taxonomy for information visualizations," in *Visual Languages, 1996. Proceedings.*,, 1996, pp. 336–343.

[6] S. Wehrend and C. Lewis, "A problem-oriented classification of visualization techniques," *Proc. IEEE Visualization '90*, pp. 139–143,, 1990.

[7] M. Brehmer and T. Munzner, "A multi-level typology of abstract visualization tasks." *IEEE TVCG*, vol. 19, no. 12, pp. 2376–2385, 2013.

[8] H.-J. Schulz, T. Nocke, M. Heitzler, and H. Schumann, "A design space of visualization tasks." *IEEE TVCG*, vol. 19, no. 12, pp. 2366–75, 2013.

[9] B. Lee, C. Plaisant, C. S. Parr, J.-D. Fekete, and N. Henry, "Task taxonomy for graph visualization," in *Proc. AVI BELIV '06*. New York, NY, USA: ACM, 2006, pp. 1–5.

[10] C. Plaisant, "The challenge of information visualization evaluation," *Proc. AVI '04*, pp. 109–116, 2004.

[11] N. Andrienko and G. Andrienko, *Exploratory analysis of spatial and temporal data: a systematic approach*. New York: Springer, 2006.

[12] J.-D. Fekete, J. van Wijk, J. Stasko, and C. North, "The value of information visualization," in *Information Visualization*, ser. Lecture Notes in Computer Science, A. Kerren, J. Stasko, J.-D. Fekete, and C. North, Eds. Springer Berlin Heidelberg, 2008, vol. 4950, pp. 1–18.

[13] N. Kerracher, J. Kennedy, and K. Chalmers, "The Design Space of Temporal Graph Visualisation," in *Proc. EuroVis'14*. Swansea: Eurographics Assocation, 2014.

[14] P. Saraiya, P. Lee, and C. North, "Visualization of Graphs with Associated Timeseries Data," in *IEEE InfoVis*. Minneapolis, MN, USA: IEEE, 2005, pp. 225–232.

[15] B. Shneiderman and A. Aris, "Network visualization by semantic substrates." *IEEE TVCG*, vol. 12, no. 5, pp. 733–40, 2006.

[16] C. Erten, P. J. Harding, S. G. Kobourov, K. Wampler, and G. Yee, "Exploring the Computing Literature Using Temporal Graph Visualization," *Proc. SPIE*, vol. 5295, pp. 45–56, 2004.

[17] P. A. Gloor and Y. Zhao, "Analyzing Actors and Their Discussion Topics by Semantic Social Network Analysis," *Tenth International Conference on Information Visualisation (IV'06)*, pp. 130–135, 2006.

[18] H. Kang, L. Getoor, and L. Singh, "Visual analysis of dynamic group membership in temporal social networks," *ACM SIGKDD Explorations Newsletter*, vol. 9, no. 2, pp. 13–21, 2007.

[19] T. Munzner, "A Nested Model for Visualization Design and Validation," *IEEE TVCG*, vol. 15, no. 6, pp. 921–928, 2009.

[20] M. Ghoniem, J.-D. Fekete, and P. Castagliola, "On the readability of graphs using node-link and matrix-based representations: a controlled experiment and statistical analysis," *Information Visualization*, vol. 4, no. 2, pp. 114–135, 2005.

[21] H. Purchase and A. Samra, "Extremes Are Better: Investigating Mental Map Preservation in Dynamic Graphs," in *Diagrammatic Representation and Inference*, G. Stapleton, J. Howse, and J. Lee, Eds. Berlin / Heidelberg: Springer, 2008, pp. 60–73.

[22] D. Archambault, H. Purchase, and B. Pinaud, "Animation, Small Multiples, and the Effect of Mental Map Preservation in Dynamic Graphs." *IEEE TVCG*, vol. 17, no. 4, pp. 539–552, 2011.

[23] M. Farrugia and A. Quigley, "Effective Temporal Graph Layout: A Comparative Study of Animation versus Static Display Methods," *Information Visualization*, vol. 10, no. 1, pp. 47–64, 2010.

[24] D. J. Peuquet, "It's About Time : A Conceptual Framework for the Representation of Temporal Dynamics in Geographic Information Systems," *Annals of the Association of American Geographers*, vol. 84, no. 3, pp. 441–461, 1994.

[25] F. Beck, M. Burch, and S. Diehl, "Matching application requirements with dynamic graph visualization profiles," in *Information Visualisation (IV), 2013 17th International Conference*, July 2013, pp. 11–18.

[26] W. Aigner, S. Miksch, H. Schumann, and C. Tominski, *Visualization of Time-Oriented Data*. London: Springer, 2011.

[27] J. Bertin, *Semiology of graphics: diagrams, networks, maps.* Madison: University of Wisconsin Press, 1983.

[28] M. Burch and D. Weiskopf, "Visualizing Dynamic Quantitative Data in Hierarchies - TimeEdgeTrees: Attaching Dynamic Weights to Tree Edges," in *GRAPP/IVAPP*, 2011, pp. 177–186.

[29] N. Christakis and J. Fowler, "The spread of obesity in a large social network over 32 years." *The New England Journal of Medicine*, vol. 357, no. 4, pp. 370–379, 2007.

[30] R. Gove, N. Gramsky, R. Kirby, E. Sefer, A. Sopan, C. Dunne, B. Shneiderman, and M. Taieb-Maimon, "NetVisia : Heat Map & Matrix Visualization of Dynamic Social Network Statistics & Content," in *Proc. IEEE Conference on Social Computing*. Boston, MA: IEEE, 2011, pp. 19–26.

**Natalie Kerracher** is a Ph.D. candidate in the Institute for Informatics and Digital Innovation at Edinburgh Napier University. She holds an MA (Hons.) degree in Mental Philosophy, and an MSc in Software Technology for the Web. Her research interests lie in the visualisation of temporal networks.

**Prof. Jessie Kennedy** is Dean of Research and Innovation at Edinburgh Napier University. She has published widely in information visualisation, has been programme chair, committee member and organiser of many international conferences including General Chair for BioVis 2012 and 2013, and currently leads a BBSRC Network in Biological Visualisation.

**Kevin Chalmers** is a Senior Lecturer working with the Institute for Informatics and Digital Innovation since 2009. Kevin's background is in concurrency and parallel systems, as well as formal modelling of systems. He was awarded a PhD from Edinburgh Napier University in 2009, and also holds a BEng (Hons) in Software Engineering.