



Defence Research and
Development Canada

Recherche et développement
pour la défense Canada



A survey of data mining of graphs using spectral graph theory

Reginald Sawilla

Defence R&D Canada – Ottawa
Technical Memorandum
DRDC Ottawa TM 2008-317
December 2008

Canada

A survey of data mining of graphs using spectral graph theory

Reginald Sawilla
Defence R&D Canada – Ottawa

Defence R&D Canada – Ottawa

Technical Memorandum

DRDC Ottawa TM 2008-317

December 2008

Principal Author

Original signed by Reginald Sawilla

Reginald Sawilla

Approved by

Original signed by Julie Lefebvre

Julie Lefebvre
Head/NIO Section

Approved for release by

Original signed by Pierre Lavoie

Pierre Lavoie
Head/Document Review Panel

© Her Majesty the Queen in Right of Canada as represented by the Minister of National Defence, 2008

© Sa Majesté la Reine (en droit du Canada), telle que représentée par le ministre de la Défense nationale, 2008

Abstract

Relational data may be represented by four classes of graphs: undirected graphs, directed graphs, hypergraphs, and directed hypergraphs. We present examples of data that is well suited to each type and describe how linear algebra has been used in the literature to cluster like objects, rank objects according to their importance to the system the graph represents, and predict relationships that likely exist in the system but have not yet been identified. In particular, we associate affinity and Laplacian matrices with a graph and apply spectral graph theory — the study of the eigenvalues, eigenvectors, and characteristic polynomial of graph matrices — to mine hidden patterns in the graphs. Shortcomings of existing techniques and open problems for future work are identified.

Résumé

On peut représenter des données relationnelles à l'aide de quatre classes de graphes : les graphes non orientés, les graphes orientés, les hypergraphes et les hypergraphes orientés. Nous présentons des exemples de données qui conviennent bien à chaque type et nous décrivons comment on a utilisé l'algèbre linéaire dans la documentation pour grouper des objets semblables, trier les objets selon leur importance au système que le graphe représente et prédire les relations qui existent probablement dans le système, mais qui n'ont pas encore été identifiées. Particulièrement, nous associons des matrices des affinités et de Laplace avec un graphe, et nous appliquons la théorie des graphes spectraux — l'étude des valeurs propres, des vecteurs propres et la polynomiale caractéristique des matrices graphiques — pour découvrir des tendances cachées dans les graphiques. On identifie les failles des techniques existantes et des problèmes ouverts qui feront l'objet de travaux futurs.

This page intentionally left blank.

Executive summary

A survey of data mining of graphs using spectral graph theory

Reginald Sawilla; DRDC Ottawa TM 2008-317; Defence R&D Canada – Ottawa; December 2008.

Background: Some information in data is obvious merely by viewing the data or conducting a simple analysis but deeper information is also present which may be discovered through data mining techniques. Data mining is the science of discovering interesting and unknown relationships and patterns in data. Discovery of this hidden information contributes in significant ways to many domains including image processing, web searching, anti-terrorism, computer security, and biology. Data mining makes sophisticated connections between data elements to find hidden information and patterns.

Principal results: Relational data (for example, networks or attack steps) may be represented by four classes of graphs: undirected graphs, directed graphs, hypergraphs, and directed hypergraphs. We present examples of data that is well suited to each type and describe how linear algebra may be used to cluster like objects, rank objects according to their importance to the system the graph represents, and predict relationships that likely exist in the system but have not yet been identified. In particular, we associate affinity and Laplacian matrices with a graph and provide a survey of the application of spectral graph theory — the study of the eigenvalues, eigenvectors, and characteristic polynomial of graph matrices — to mine hidden patterns in the graphs. Shortcomings of existing techniques and open problems for future work are identified.

Significance of results: Data mining is applicable to data sets of all sizes from a few dozen records to billions. The results of data mining are used in diverse areas including by law enforcement agencies to understand terrorist networks, by scientists to find patterns in proteins, and by businesses to understand and predict their customers' purchasing patterns and thus make them more profitable. This paper assists users to choose the correct graph representation for their type of data and identifies state-of-the-art techniques to perform data mining for each graph type as well as identify shortcomings of the art.

Future work: A collection of open problems and future work is identified in Section 5. In particular, data representation and parameter selection should be better justified and made rigorous; work on directed graphs and directed hypergraphs is significantly less mature than the area of undirected graphs; and techniques from social network analysis may be profitable in general graph data mining.

Sommaire

A survey of data mining of graphs using spectral graph theory

Reginald Sawilla ; DRDC Ottawa TM 2008-317 ; R & D pour la défense Canada – Ottawa ; décembre 2008.

Contexte : Une partie de l'information contenue dans des données est évidente : il suffit de les regarder ou de faire une analyse simple, mais on y retrouve aussi de l'information enfouie plus profondément dans les données. Il faut découvrir cette information à l'aide de techniques d'exploration des données. L'exploration des données est la science de la découverte de relations et de tendances intéressantes et inconnues dans des données. La découverte de cette information cachée contribue considérablement à de nombreux domaines, comme le traitement des images, la recherche dans le Web, la lutte contre le terrorisme, la sécurité informatique et la biologie. L'exploration des données permet de créer des liaisons sophistiquées entre des éléments de données pour trouver de l'information et des tendances cachées.

Principaux résultats : On peut représenter des données relationnelles (par exemple des réseaux ou les étapes d'une attaque) à l'aide de quatre classes de graphes : les graphes non orientés, les graphes orientés, les hypergraphes et les hypergraphes orientés. Nous présentons des exemples de données qui conviennent bien à chaque type et nous décrivons comment on a utilisé l'algèbre linéaire dans la documentation pour grouper des objets semblables, trier les objets selon leur importance au système que le graphe représente et prédire les relations qui existent probablement dans le système, mais qui n'ont pas encore été identifiées. Particulièrement, nous associons des matrices des affinités et de Laplace avec un graphe, et nous appliquons la théorie des graphes spectraux — l'étude des valeurs propres, des vecteurs propres et la polynomiale caractéristique des matrices graphiques — pour découvrir des tendances cachées dans les graphiques. On identifie les failles des techniques existantes et des problèmes ouverts qui feront l'objet de travaux futurs.

Importance des résultats : L'exploration des données s'applique à des ensembles de données de toutes tailles, allant de quelques douzaines à des milliards d'enregistrements. Les résultats de l'exploration des données sont utilisés dans différents domaines, y compris par les corps policiers, pour comprendre les réseaux terroristes, par des scientifiques, pour trouver des répétitions dans des protéines, et par des entreprises commerciales pour comprendre et prédire les comportements d'achat de leurs clients et réaliser ainsi de plus gros profits. Dans ce document, on aide les utilisateurs à choisir les représentations graphiques qui conviennent à leur type de données, on identifie les techniques de pointe d'exploration des données pour chaque type de graphe et on identifie les failles de cet art.

Travaux futurs : On identifie une collection de problèmes ouverts et de travaux futurs dans la section 5. Particulièrement, la représentation des données et la sélection des paramètres devraient être mieux justifiées et rendues plus rigoureuses ; les travaux sur les graphes et les hypergraphes orientés sont beaucoup moins avancés que ceux dans le domaine des graphes non orientés ; et des techniques d'analyse des réseaux sociaux peuvent être profitables dans l'exploration générale des données de graphes.

This page intentionally left blank.

Table of contents

Abstract	i
Résumé	i
Executive summary	iii
Sommaire	iv
Table of contents	vii
List of figures	ix
List of tables	x
1 Introduction	1
2 Graphs	1
2.1 Undirected graphs	1
2.2 Directed graphs	2
2.3 Hypergraphs	3
2.4 Directed hypergraphs	5
2.5 AND and OR vertices	6
3 Matrix representations	7
3.1 Affinity matrix	7
3.2 Laplacian matrix	10
4 Spectral graph theory	13
4.1 Clustering	14
4.2 Vertex ranking	19
4.3 Link prediction	21
4.4 Importance of matrix representation	21
4.5 Problems	22

5	Future work	23
6	Conclusion	25
	References	26

List of figures

- Figure 1: Example of an undirected graph — social network 2
- Figure 2: Example of a directed graph — web community 3
- Figure 3: Example of a hypergraph — travellers 4
- Figure 4: Clique expansion undirected graph of Figure 3 4
- Figure 5: Undirected graph equivalent to Figure 3 5
- Figure 6: Example of a directed hypergraph — attack graph 5
- Figure 7: Example of AND/OR graph — attack graph 7

List of tables

Table 1:	Graph matrices	8
Table 2:	Hypergraph clustering is equivalent to undirected graph clustering	12

1 Introduction

Data mining is the science of discovering interesting and unknown relationships and patterns in data. Discovery of this hidden information contributes in significant ways to many domains including image processing, web searching, anti-terrorism, computer security, and biology.

While some information is obvious merely by viewing the data or conducting a simple analysis, data mining makes sophisticated connections between data elements to find hidden information and patterns. It is applicable to data sets of all sizes from a few dozen records to billions. The results of data mining are used in diverse areas including: by law enforcement agencies to understand terrorist networks; by scientists to find patterns in proteins; and by businesses to understand and predict their customers' purchasing patterns and thus make them more profitable.

Data exists with varying degrees of structure. Unstructured data refers to data that is not easily readable or readily processed by a machine. For example, audio, video, and natural language text are unstructured data sources. Other data sources have a high degree of structure and the data is well suited to a graph representation. For example, relational databases and spreadsheets represent information in rows and columns; this data is well structured and is readily represented by a graph.

This paper discusses the four types of graphs and the techniques that are used to cluster data elements into groups, predict new data connections, and rank vertices by their prominence in the system the data represents. Graphs can be mined using several methods of matrix decompositions [1]; however, the focus of this paper is limited to the mining of graphs using spectral methods.

The paper is organized as follows. Section 2 introduces the four types of graphs. Section 3 discusses the matrices associated with graphs. Section 4 introduces spectral graph theory. Section 5 lists open problems and future work, and Section 6 concludes.

2 Graphs

There are four types of graphs: undirected graphs, directed graphs, undirected hypergraphs, and directed hypergraphs. Data is variously best suited to each of the different types and we give examples of each.

2.1 Undirected graphs

An undirected graph $G = (V, E)$ is composed of a set of vertices V and a set of edges E connecting the vertices. The elements of the edge set $E = \{\{u, v\} | u, v \in V\}$ are sets

representing that an edge exists between the vertices u and v .

Both graph vertices and edges may have attributes associated with them. Common attributes are labels, colours, and weights. Edge weights are assigned by a function called a similarity or affinity function. The weights might reflect the number of connections or the strength of affinities between the incident vertices while vertex weights could reflect the preference for the vertex as a starting point in the graph or a production cost. In this paper, we define weight functions for graphs as follows.

Definition 1. A vertex weight function $f : V \Rightarrow \mathbb{R}^+$ assigns positive real number weights to vertices. The weights may be assembled into a vector for convenience.

Definition 2. An edge weight function $w : E \Rightarrow \mathbb{R}^+$ assigns positive real number weights to edges. For undirected and directed graphs the weight function is often specified in terms of the vertices¹ and the weights are assembled into a matrix W that is indexed by the vertices.

An example of data well suited to an undirected graph representation is the friendships in a social network. In this case the vertices represent people and the edges represent friendships between them. The data is well suited to an undirected graph because friendships are one-on-one and bidirectional by their nature. Figure 1 gives an example of an undirected graph for a social network. For the figure, people are represented by the set of vertices

$$V = \{a, b, c, d, e, f\}$$

and their friendships are represented by the set of edges

$$E = \{\{a, b\}, \{a, c\}, \{a, d\}, \{a, f\}, \{b, c\}, \{c, e\}, \{d, e\}\} .$$

Figure 1: Example of an undirected graph — social network

2.2 Directed graphs

Directed graphs generalize undirected graphs by orienting the edges. The edges are called directed edges or arcs and the edge set is defined as $E = \{(u, v) | u, v \in V\}$. An example of data well suited to a directed graph representation is the World Wide Web. The data forms a “web graph” with web pages represented by vertices and links from one web page to another represented by arcs. The data is well suited to a directed graph representation because a hyperlink on page A to the page B does not imply that there will be a reciprocal

1. That is, the function $w(u, v)$ gives the weight of the edge connecting vertices u and v .

link from page B to page A. Figure 2 gives an example of a web community represented by a directed graph. For the figure, pages are represented by the set of vertices

$$V = \{A, B, C, D, E, F\}$$

and hyperlinks between pages are represented by the set of ordered pairs

$$E = \{(A, C), (B, C), (C, D), (C, E), (C, F), (D, E), (F, C)\} .$$

Figure 2: Example of a directed graph — web community

2.3 Hypergraphs

Hypergraphs are a natural generalization of undirected graphs. While they do not represent more complicated sets of data than undirected graphs, they represent the data in a more compact fashion when the relations between data elements are not inherently one-to-one.

Hypergraphs connect multiple vertices with a single undirected edge called a hyperedge. A hypergraph $G = (V, E)$ is composed of a set of vertices V and a set of edges E connecting the vertices. An element $e \in E$ is a set $e = \{v_1, v_2, \dots, v_k\}$ representing that all of the vertices in e are connected with a single edge. If all of the edges contain exactly k vertices, the hypergraph is said to be k -uniform. A 2-uniform hypergraph is exactly an undirected graph.

An example of data well suited to a hypergraph representation is a set of travellers and the countries they have visited. Travellers are represented by vertices and countries are represented by hyperedges. Figure 3 gives an example of a collection of travellers represented by a hypergraph. For the figure, travellers are represented by the set of vertices

$$V = \{a, b, c, d, e\}$$

and countries are represented by the set of hyperedges

$$E = \{\{a, e\}, \{a, b, d, e\}, \{c, d\}\} .$$

Agarwal *et al.* [2] present two methods used to convert a hypergraph into an undirected graph. The first is a clique expansion algorithm. If $G = (V, E)$ is a hypergraph, the clique expansion algorithm creates a graph $G^c = (V^c, E^c)$ where $V^c = V$ and $E^c = \{\{u, v\} | u, v \in e, e \in E\}$. That is, the vertex set is unchanged and a complete subgraph is

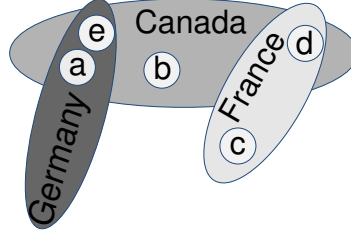


Figure 3: Example of a hypergraph — travellers

added for each hyperedge. If the hyperedges are weighted, the edges may be weighted by the averaging function

$$w^c(u, v) = \arg \min_{w^c(u, v)} \sum_{\substack{e \in E \\ \{u, v\} \in E^c}} (w^c(u, v) - w(e))^2 . \quad (1)$$

The operator *textrmarg* min in the above expression assigns a real number to $w^c(u, v)$ such that the summation is minimized.

Figure 4: Clique expansion undirected graph of Figure 3

Figure 4 is a clique expansion undirected graph representation of the information in Figure 3. Note that information is lost with this representation and it is not possible to reconstruct Figure 3 from Figure 4. For example, it is impossible to know the countries the people have visited and while a and e both went to the same two countries, there is no stronger affinity between a and e than there is between a and b who have only visited a single common country.

Compare this with the star expansion algorithm which creates a bipartite graph $G^* = (V^*, E^*)$ by adding a new vertex to the set of vertices for every hyperedge, and adding edges from the new vertex to the vertices that were part of the hyperedge. Hyperedge weights are evenly distributed among all vertices in the edge. Formally,

$$V^* = V \cup E \quad (2)$$

$$E^* = \{\{u, e\} | u \in e, e \in E\} \quad (3)$$

$$w^*(u, e) = w(e)/d(v) \quad (4)$$

$$d(v) = |\{e | v \in e, e \in E\}| \quad (5)$$

With this representation, the hypergraph can be reconstructed from the undirected graph. Figure 5 is a star expansion undirected graph representation of the information in Figure 3. Notice that no information is lost with the star expansion undirected graph but the representation is more complex.

Figure 5: Undirected graph equivalent to Figure 3

2.4 Directed hypergraphs

The fourth type of graph, a directed hypergraph generalizes all of the preceding graphs. A directed hypergraph $G = (V, E)$ is composed of a set of vertices V and a set of edges E connecting the vertices. An element $e \in E$ is an ordered pair of vertex sets $e = (V^-, V^+)$ where the vertices in the set V^- are the sources of the directed hyperedge and the vertices in the set V^+ are the targets [3]. Edge weights are generalized so that an edge weight function $w : V \times E \Rightarrow \mathbb{R}$ assigns negative real numbers for each of the source vertices of an edge and positive real numbers for each of the target vertices of an edge.

Since a single edge may have multiple targets, directed hyperedges can represent either a unified dependence upon multiple vertices (AND dependence) or a dependence upon a choice of vertices (OR dependence). An example of data well suited to a directed hypergraph representation is an attack against computer networks. Attacker privileges are represented by vertices and the dependence upon preconditions that enable an attacker privilege are represented by directed hyperedges. Figure 6 gives an example of an attack graph represented by a directed hypergraph. For the figure, attacker privileges are represented by the set of vertices

$$V = \{a, b, c, d, e, g\}$$

and dependencies are represented by the set of directed hyperedges

$$E = \{(\{g\}, \{a\}), (\{g\}, \{b, c, d\}), (\{g\}, \{e\})\} .$$

The attack graph states that the goal is satisfied by obtaining a ; or e ; or all of b , c , and d .

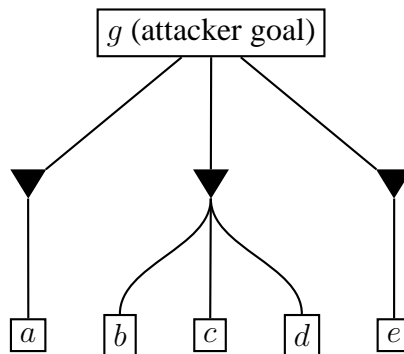


Figure 6: Example of a directed hypergraph — attack graph

Another common application of directed hypergraphs is to Very Large Scale Integration (VLSI) circuits [4, 5]. VLSI circuits contain many propositional logic formulas which

translate to directed hyperedges with multiple source and target vertices. Following are two logic clauses:

$$a \wedge b \Rightarrow c \wedge d \quad (6)$$

$$a \vee b \Rightarrow e \wedge f \quad (7)$$

Those two logic clauses are encoded in the following directed hypergraph.

$$G = (V, E) \quad (8)$$

$$V = \{a, b, c, d, e, f\} \quad (9)$$

$$E = \{e_1, e_2, e_3\} = \{(\{a, b\}, \{c, d\}), (\{a\}, \{e, f\}), (\{b\}, \{e, f\})\} \quad (10)$$

In summary, the four types of graphs become progressively more general. Undirected graphs represent the simplest relationships. Hypergraphs do not represent more complex information than undirected graphs but they represent multi-way relationships in a more compact fashion. Directed graphs generalize both undirected graphs and hypergraphs, and directed hypergraphs are the most general representation. In symbols: undirected graphs \equiv hypergraphs \subset directed graphs \subset directed hypergraphs.

2.5 AND and OR vertices

As discussed in Section 2.4, directed hypergraphs allow for a distinction between edges that transition from a vertex to *one* of its out-neighbours and a transition from a vertex to *all* of its out-neighbours. In directed graphs, this distinction is also sometimes required and it is made by typing the vertex itself. That is, a vertex where a transition may be made to any *one* of its out-neighbours is typed as an *OR* vertex and a vertex where a transition is made to *all* of its out-neighbours is typed as an *AND* vertex. Even if a vertex has no out-neighbours (a sink vertex) it still may be typed an OR or AND vertex since the type is irrelevant. In some applications, sink vertices are treated differently from non-sink vertices so it may be preferable to explicitly label them as a sink.

Definition 3. A vertex typing function $h : V \Rightarrow \{AND, OR, SINK\}$ assigns a vertex type to vertices.

Exploit-dependency attack graphs are examples of an AND/OR directed graph. Figure 7 shows such an attack graph. OR vertices are represented with a diamond shape, AND vertices are represented with an ellipse shape and SINK vertices are represented with a box shape. The graph shows that the ability to execute arbitrary code on a PC can be obtained by a choice of two different means and hence the associated vertex is an OR vertex. One way to gain the ability is through a local login and the other is through a remote exploit. Both of the Local Login and Remote Exploit vertices are AND vertices since both of their

out-neighbours are required in order to fulfill the attack. Note that directed hypergraphs are still more general than directed graphs with vertex types since they allow multi-head, multi-tail, and both AND and OR edges incident to a vertex.

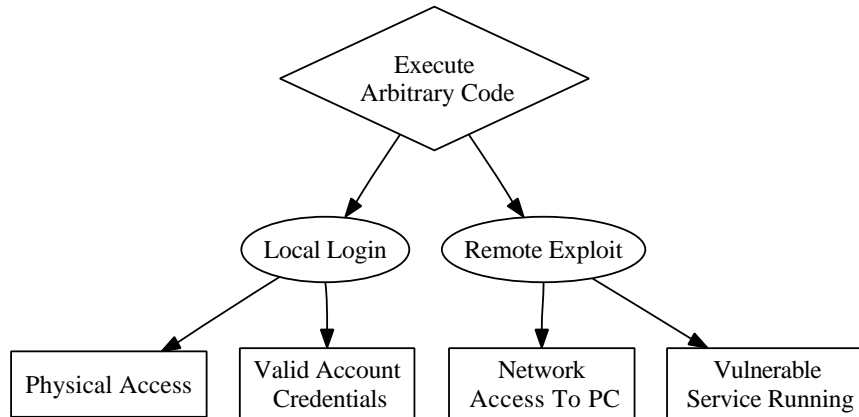


Figure 7: Example of AND/OR graph — attack graph

3 Matrix representations

Spectral graph theory applies the power of linear algebra to matrix representations of graphs and it is the study of their spectrum. The spectrum of a matrix is technically defined as the set of eigenvalues of a matrix but in practice, the spectrum also often refers to the eigenvalues, eigenvectors, and the characteristic polynomial. Graphs have two primary matrix representations, the affinity matrix and the Laplacian. The two representations may be derived from each other and two normalizations of the representations are common. The matrix used for the spectrum may be the affinity matrix or Laplacian matrix (and their normalizations). Since there is no standard choice, the matrix used should be explicitly stated. In this section we introduce the matrices associated with a graph and they are summarized in Table 1.

3.1 Affinity matrix

The affinity matrix W of an undirected or directed graph is a $|V| \times |V|$ matrix and is created by setting W_{uv} to $w(u, v)$ for each edge $(u, v) \in E$ and 0 otherwise. An adjacency matrix is a simplified affinity matrix that gives only connectivity information by setting W_{uv} to 1 if there is an edge from u to v , and 0 otherwise. For an undirected graph W is symmetric, that is, $W = W^T$.

Name	Definition	Applicable to
Incidence/Boundary	B	All
Edge Weight Matrix	$W_e = \text{diag}(w(e_1), \dots, w(e_{ E }))$	Hypergraph
Affinity/Weight	$W_{ij} = w(i, j)$	Undirected, Directed
Affinity/Weight	$W = BW_e$	Hypergraph
Affinity/Weight	$W_{ve} = w(v, e)$	Directed Hypergraph
Probability Transition	$P = D^{-1}W$	Undirected, Directed
(Out) Degree	$D = \text{diag}(d_1, \dots, d_{ V }), d_i = \sum_{j=1}^{ V } w_{ij}$	Undirected, Directed
In Degree	$D^- = \text{diag}(d_i^-), d_i^- = \sum_{j=1}^{ V } w_{ji}$	Directed
Vertex Degree	$D = \text{diag}(d_1, \dots, d_{ V }), d_i = \sum_{e_j, v_i \in e_j} w(e_j)$	Hypergraph
Out Degree	$D^+ = \text{diag}(d_i^+), d_i^+ = \sum_{\substack{e_j, v_i \in e_j \\ (V^-, V^+) \in e_j}} w(i, j)$	Directed Hypergraph
In Degree	$D^- = \text{diag}(d_i^-), d_i^- = \sum_{\substack{e_j, v_i \in e_j \\ (V^-, V^+) \in e_j}} w(i, j)$	Directed Hypergraph
Combinatorial Laplacian	$L = D - W = BB^T$	Undirected
Laplacian (Random Walk)	$L_{rw} = D^{-1}L = I - D^{-1}W$	Undirected
Laplacian (Symmetric)	$L_{sym} = D^{-1/2}LD^{-1/2} = I - D^{-1/2}WD^{-1/2}$	Undirected
Combinatorial Laplacian	$L = \Pi - \frac{\Pi P + P^T \Pi}{2}$	Directed
Laplacian (Symmetric)	$L_{sym} = I - \frac{\Phi^{1/2} P \Phi^{-1/2} + \Phi^{-1/2} P \Phi^{1/2}}{2}$	Directed

Table 1: Graph matrices

Example 4. The affinity matrix associated with the graph in Figure 1 is

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (11)$$

Example 5. The affinity matrix associated with the graph in Figure 2 is

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \quad (12)$$

The incidence matrix B , is a $|V| \times |E|$ matrix where the rows correspond to vertices and the columns correspond to edges. The entries of B for undirected and directed graphs are (for undirected graphs, the edges are considered to be oriented but the choice of orientation does not matter):

$$B_{ij} = \begin{cases} -1 & \text{if } e_j = (v_i, w) \in E, \text{ for some } w \\ 1 & \text{if } e_j = (u, v_i) \in E, \text{ for some } u \\ 0 & \text{otherwise .} \end{cases} \quad (13)$$

For hypergraphs, $B_{ij} = 1$ if $v_i \in e_j$. In the case of directed hypergraphs, for each $v_i \in V$ and $e_j = (V_j^-, V_j^+) \in E$ the entries of the matrix are formed by

$$B_{ij} = \begin{cases} -1 & \text{if } v_i \in V_j^- \\ 1 & \text{if } v_i \in V_j^+ \\ 0 & \text{otherwise .} \end{cases} \quad (14)$$

The edge weights of a hypergraph may be assembled into a diagonal $|E| \times |E|$ weight matrix W_e . The affinity matrix of a hypergraph is given by $W = BW_e$.

Example 6. The affinity matrix associated with the hypergraph of travellers in Figure 3 is

$$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix} . \quad (15)$$

For comparison, the affinity matrix of the equivalent star-expansion undirected graph presented as Figure 5 follows.

Example 7. The affinity matrix associated with the equivalent undirected bipartite graph in Figure 5 is

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} . \quad (16)$$

The matrix in Example 7 is a good illustration of the extra dimensions that are present in many graph matrices. The graph is represented in 3 dimensions by the matrix in Equation (15) while the matrix in Equation (16) uses 8 dimensions to represent the same information.

Example 8. The affinity matrix for the directed hypergraph of the VLSI circuit in Equation (8) is

$$\begin{bmatrix} -1 & -1 & 0 \\ -1 & 0 & -1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}. \quad (17)$$

Other quantities we will need when working with graphs are vertex degrees and graph volume.

Definition 9. The out-degree of a vertex, denoted d_v^+ for a vertex v , is the sum of the weights on its outgoing edges; the in-degree is defined analogously. Formulas to calculate vertex degrees may be found in Table 1.

Definition 10. The volume of a graph $G = (V, E)$ is the sum of its vertex degrees. That is, $\text{vol } G = \sum_{i=1}^{|V|} d_i$.

3.2 Laplacian matrix

The Laplacian matrix is another matrix representation for graphs and for many problems it is the more natural representation to work with. The combinatorial Laplacian of an undirected graph is given by $L = D - W$ where D is the degree matrix and W is the affinity matrix. It may also be computed from the incidence matrix, $L = BB^T$.

There are two normalizations applied to affinity and Laplacian matrices, computed by multiplying by the inverse of the degree matrix. The first, the “random walk” normalization, creates a probability transition matrix P or a random walk Laplacian L_{rw} .

$$W_{rw} = P = D^{-1}W \quad L_{rw} = D^{-1}L \quad (18)$$

The second normalization preserves symmetry (if it exists) by multiplying both column i and row i of the unnormalized matrix by $1/\sqrt{d_i}$.

$$W_{sym} = D^{-1/2}WD^{-1/2} \quad L_{sym} = D^{-1/2}LD^{-1/2} \quad (19)$$

The normalized Laplacian of a directed graph is defined by Chung [6] to be

$$L_{sym} = I - \frac{\Pi^{\frac{1}{2}} P \Pi^{-\frac{1}{2}} + \Pi^{-\frac{1}{2}} P \Pi^{\frac{1}{2}}}{2} \quad (20)$$

where I is the identity matrix, P is a probability transition matrix for the graph, Π is the diagonal matrix formed from the components of the vector π where π is the positive principal left eigenvector of P and π is normalized to sum to 1. This vector is in fact the unique stationary distribution of the matrix P . Chung's definition assumes that P has a unique positive principal eigenvector. Perron's theorem (see Meyer [7], for example) guarantees the existence of such a vector when P is irreducible and positive (equivalent to G being a strongly connected graph). A common strategy with web graphs is to force a strongly connected graph by setting all 0 entries in P to a small number $\epsilon > 0$.

The entries of π for an undirected graph G are shown [6] to be

$$\pi_v = \frac{d_v}{\sum_{u \in V} d_u} = \frac{d_v}{\text{vol } G} . \quad (21)$$

By substituting $\Pi = \frac{1}{\text{vol } G} D$, L_{sym} simplifies to $I - D^{-1/2} W D^{-1/2}$ so the normalized directed graph Laplacian is consistent with the definition of the symmetrically normalized Laplacian for undirected graphs.

Chung defines a combinatorial Laplacian of a directed graph to be

$$L = \Pi^{1/2} L_{sym} \Pi^{1/2} = \Pi - \frac{\Pi P + P^T \Pi}{2} . \quad (22)$$

For an undirected graph, this simplifies to $L = \frac{1}{\text{vol } G} (D - W)$. This definition differs from the combinatorial Laplacian of an undirected graph by the scalar $1/\text{vol } G$ and it requires the existence of a unique principal eigenvector, which is restrictive.

Agarwal *et al.* [2] make a significant contribution by showing that all work done to date on the application of Laplacians to hypergraphs is in fact equivalent to a conversion of the hypergraph to an undirected graph followed by the application of a combinatorial or normalized Laplacian.

In Section 2 we showed how a hypergraph can be converted to an undirected graph through a clique expansion and star expansion. Table 2 is a summary of the previous work on hypergraphs and Agarwal *et al.*'s demonstration of how their work is equivalent to spectral analysis of an undirected graph.

Their conclusion is that while, on the surface, hypergraphs seem to capture more information about higher order relations, the same analysis can be handled using pairwise relations

Algorithm	Year	Hypergraph	
		Expansion	Analysis Matrix
Li	1996	Star	Adjacency
Gibson	1998	Clique	Adjacency
Bolla	1993	Clique	Combinatorial Laplacian
Rodriguez	2002, 2003	Clique	Combinatorial Laplacian
Zhou	2005	Star	Symmetrically Normalized Laplacian

Table 2: Hypergraph clustering is equivalent to undirected graph clustering

in undirected graphs. The conclusion is surprising because the higher order relations are not directly represented in undirected graphs but can only be detected with further analysis. However, using appropriate weighting functions, the semantics of the hyperedges are transferred to undirected graphs. The graph theoretic problems behind unsupervised and semi-supervised learning in hypergraphs can be analyzed using existing undirected graph techniques.

Agarwal *et al.* show that while Laplacians of higher order may be defined for hypergraphs, only the lowest order Laplacian, L_0 , can analyze vertices. The objects of L_0 are vertices and edges (termed 0-chains and 1-chains by algebraic topologists) but, for example, the higher order Laplacians, L_1 and L_2 , operate on edges, triangles, and tetrahedrons (1-chains, 2-chains, and 3-chains) and so do not yield vertex functions useful for clustering, ranking, and other machine learning problems.

At this time, the Laplacian of a directed hypergraph has not yet been discovered. Following are several examples of graph Laplacians.

Example 11. The Laplacian matrix associated with the social network graph in Figure 1 is

$$L = D - W = \begin{bmatrix} 4 & -1 & -1 & -1 & 0 & -1 \\ -1 & 2 & -1 & 0 & 0 & 0 \\ -1 & -1 & 3 & 0 & -1 & 0 \\ -1 & 0 & 0 & 2 & -1 & 0 \\ 0 & 0 & -1 & -1 & 2 & 0 \\ -1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (23)$$

Example 12. To compute the normalized Laplacian of the directed graph in Figure 2, the weight $\epsilon = 0.001$ was added to the edge weights so that the graph is strongly connected. This guarantees that a (unique) positive eigenvector exists — something that is required by Chung’s definition.

$$L_{sym} = I - \frac{\Pi^{\frac{1}{2}}P\Pi^{-\frac{1}{2}} + \Pi^{-\frac{1}{2}}P\Pi^{\frac{1}{2}}}{2} = \begin{bmatrix} 0.999 & -0.001 & -1. & -0.001 & -0.001 & -0.001 \\ -0.001 & 0.999 & -1. & -0.001 & -0.001 & -0.001 \\ -0.001 & -0.001 & 0.999 & -1. & -1. & -1. \\ -0.001 & -0.001 & -0.001 & 0.999 & -1. & -0.001 \\ -0.001 & -0.001 & -0.001 & -0.001 & 0.999 & -0.001 \\ -0.001 & -0.001 & -1. & -0.001 & -0.001 & 0.999 \end{bmatrix} \quad (24)$$

Example 13. We now consider the Laplacian matrix associated with the hypergraph in Figure 3. Bolla defines a Laplacian for unweighted hypergraphs using a degree matrix, edge matrix and incidence matrix of the graph [2]. Bolla's Laplacian is defined to be:

$$L = D_v - BW_e^{-1}B^T \quad (25)$$

which for this example is:

$$L = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix} - \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1/2 & 0 & 0 \\ 0 & 1/4 & 0 \\ 0 & 0 & 1/2 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} \quad (26)$$

$$= \begin{bmatrix} 5/4 & -1/4 & 0 & -1/4 & -3/4 \\ -1/4 & 3/4 & 0 & -1/4 & -1/4 \\ 0 & 0 & 1/2 & -1/2 & 0 \\ -1/4 & -1/4 & -1/2 & 1/4 & -1/4 \\ -3/4 & -1/4 & 0 & -1/4 & 5/4 \end{bmatrix} \quad (27)$$

At this time, the Laplacian of a directed hypergraph has not been defined so it is not possible to present the Laplacian matrix for the graph in Figure 6.

4 Spectral graph theory

The spectrum of a matrix consists of its characteristic polynomial, eigenvalues, and eigenvectors. These are invariant for isomorphic graphs; however, two non-isomorphic graphs may still have an identical spectrum.

Definition 14. If the rank of a square matrix A is n , there are up to n eigenvalue-eigenvector pairs. The eigenvalues are the roots of the characteristic polynomial (the determinant of $A - \lambda I$) and they are paired with non-zero eigenvectors so that $\lambda_i X_i = AX_i$ and $\lambda_i \bar{X}_i = \bar{X}_i A$. X_i and \bar{X}_i are the right-eigenvector and left-eigenvector (respectively) corresponding to λ_i .

The two most developed applications of data mining using spectral graph theory techniques are vertex ranking and clustering. Vertex ranking assigns a quantitative measure of the prominence of objects in a system. For example, web search engines rank web pages using spectral methods so that user searches are very likely to return the most relevant pages. Clustering groups objects together with other objects that they are closely related to. For example, photo software uses spectral methods to detect subjects in images.

4.1 Clustering

Spectral graph theory has been successfully applied to clustering in all but directed hypergraphs. The goal of clustering is to partition vertices into groups, based upon their similarities. Good partitions are those that have low edge weights between the clusters and high edge weights within clusters.

Both the adjacency matrix and the Laplacian matrix may be used for clustering and research articles use both. For the adjacency matrix, the eigenvectors of large eigenvalues correspond to areas of high connectivity and close semantic proximity. The eigenvectors of small eigenvalues correspond to noise or local characteristics. For the spectrum of the Laplacian, the opposite is true. The eigenvector associated with the smallest non-trivial eigenvalue gives a clustering of the data based on high connectivity and close proximity.

Partitions that are balanced in size are often desirable. The partitions may be balanced by the number of vertices in each cluster or the weight of the edges for the vertices in each cluster. These two cuts are referred to as the RatioCut [8] and NCut [9] (respectively).

Computing an optimal balanced cut is an NP-hard problem [10]. However, von Luxburg [11] gives an excellent explanation of how this discrete optimization problem can be relaxed into a real valued optimization problem. The relaxed real valued optimization problem is immediately solved by the k eigenvectors of the Laplacian matrix corresponding to the k smallest eigenvalues (where k is the number of clusters in the data). The real valued solution must be converted back to a discrete solution and a common technique used for this purpose is k -means clustering [12].

Clustering using k -means partitions the vertices into k clusters $\mathcal{C} = \{C_1, C_2, \dots, C_k\}$. Clusters are chosen so that the distance to the cluster centroid c_i is minimized.

$$\sum_{i=1}^k \sum_{v_j \in C_i} \|v_j - c_i\|^2 \quad (28)$$

Unfortunately, clusters obtained by the Laplacian + k -means method do not necessarily bear any resemblance to the optimal solution. Guattery and Miller [13] show how clustering of some types of graphs always leads to a poor solution. In one case (cockroach graphs),

the optimal solution is a constant 2 cuts whereas Laplacian clustering suggests $O(n)$ cuts where n is the number of vertices in the graph. Von Luxburg states that the popularity of clustering with this method is not due to the good results it provides, but rather that it transforms an NP-hard problem into a relatively easy linear algebra problem.

Meilă and Shi [14] make the connection between the eigenvalues of the random walk Laplacian and the random walk adjacency matrix (the probability transition matrix $P = D^{-1}W$). Recall that W is the weighted affinity matrix, D the degree matrix, and $L_{rw} = I - P$ the random walk Laplacian of a graph. The eigenvectors of L_{rw} are the solutions to $L_{rw}X = \lambda X$. The following equations show that L_{rw} and P share the same eigenvectors and, if λ is an eigenvalue of L_{rw} then $1 - \lambda$ is an eigenvalue of P .

$$L_{rw}X = \lambda X \quad (29)$$

$$\Rightarrow (I - P)X = \lambda X \quad (30)$$

$$\Rightarrow PX = (1 - \lambda)X \quad (31)$$

The same result is valid for the eigenvectors of the symmetrically normalized Laplacian and affinity matrices. These results are important because they show that after the normalization method is chosen, clustering using the Laplacian or the affinity matrix give the same result.

Although the eigenvectors of the normalized Laplacian and affinity matrices are the same, the clustering interpretation differs. The intuition of clustering the vertices of a random walk is to choose clusters that have a high probability of keeping the random walker transitioning between vertices in each cluster with a low probability of the random walker transitioning to another cluster.

Meilă and Shi [14] prove a formal equivalence between normalized cuts and probability transitions. A normalized cut (NCut) between a partition C and its complement \bar{C} is given by

$$NCut(C, \bar{C}) = \left(\frac{1}{\text{vol } C} + \frac{1}{\text{vol } \bar{C}} \right) \sum_{u \in C, v \in \bar{C}} w(u, v) \quad (32)$$

where $\text{vol } C = \sum_{v \in C} d_v$. Let $P(B|A)$ be the probability of a random walk transitioning from the set A to the set B in one step if the random walker is in A and the random walk begins in its stationary distribution. Meilă and Shi show that $NCut(C, \bar{C}) = P(\bar{C}|C) + P(C|\bar{C})$. In words, the normalized cut between C and \bar{C} is equivalent to the probability that a random walker transitions from C to \bar{C} or transitions from \bar{C} to C .

A third interpretation of spectral clustering is that the correlation between graph data points is encoded in the cosine angles between the points when they use an eigenbasis. Brand and

Huang [15] offer an interpretation in which the eigenvectors of the affinity matrix embed the graph data points onto the surface of a hypersphere. As the size of the eigenbasis used to represent the points shrinks (through the retention of only the most dominant eigenvectors), the angle between similar points decreases while the angle between dissimilar points increases.

They introduce a new clustering algorithm that alternates between two projections of the data points. The first projection amplifies the clusters by projecting onto a lower-dimensional space (the actual dimension is dependent upon the data and is automatically chosen) and the second projection normalizes the vectors so the points are again cast onto the surface of a hypersphere. After each double-projection the probability matrix is tested for reducibility (equivalent to having two or more unit eigenvalues). When the matrix is reducible, k indicator vectors are retrieved corresponding to k clusters in the data where the exact number of clusters depends upon the affinity matrix data. Note that their method produces clusters directly without the need for a post-processing step such as the k -means algorithm.

Their algorithm automatically selects the number of clusters for the data set. This is an advantage when the actual number of clusters is unknown, which is often the case. However, when the number of clusters is known, there is not a convenient way to have the algorithm assemble the correct number of clusters. The kernel used to construct a weighted affinity matrix is a Gaussian kernel $W_{ij} = \exp(-\|x_i - x_j\|^2/2\sigma^2)$. When the number of clusters produced by their algorithm was different from the known actual number, they were required to adjust the averaging parameter σ of the Gaussian kernel in order to produce the correct number of clusters. A shortcoming of the algorithm is that it requires a positive symmetric matrix. Thus, the technique is limited to the affinity matrix of an undirected graph.

Even so, the performance of their algorithm on challenge problems in the literature was perfect (when the Gaussian kernel was appropriately chosen). In addition, they demonstrated the algorithm on an experiment consisting of 2D data of three different faces. The data consisted of 500 frames of tracking data of the faces. The faces were combined into a single layer and the task was to detect the three faces — something the algorithm did with perfect success.

Recent work [16, 17, 18, 19] has applied classical wavelet theory to graphs and manifolds. In this new domain, the wavelets are called *diffusion wavelets* because they diffuse the data in different scales. In particular, Nadler *et al.* [19] show that the normalized Laplacian corresponds to a diffusion process that is well suited for spectral clustering if the data contains well separated clusters. The research thus far has only offered an alternative interpretation of Laplacian clustering and has not offered a wavelet diffusion with better performance.

The previous discussion relates to clustering of undirected graphs. Clustering using the spectrum of directed graphs is more complicated since the affinity matrix is not symmetric and so the spectrum very likely contains complex numbers. Only recently have definitions

for cuts in directed graphs been proposed [20, 21]. In order to obtain a real valued spectrum, a symmetric matrix is associated with the directed graph. Common transformations are $\frac{1}{2}(A+A^T)$, AA^T , and $A^T A$ to create a symmetric matrix, and in effect an undirected graph.

Zhou *et al.* [20] generalize the normalized cut for directed graphs in order to partition a directed graph into k partitions. Let P_k be a k -partition $\mathcal{C} = \{C_1, C_2, \dots, C_k\}$. A balanced k -partition is obtained by minimizing

$$NCut(P_k) = \sum_{1 \leq i \leq k} \frac{\text{vol } \partial C_i}{\text{vol } C_i} \quad (33)$$

where

$$\text{vol } \partial C = \sum_{(u,v) \in \partial C} \pi(u)p(u,v) \quad (34)$$

$$\partial C = \{(u,v) | u \in C, v \in \bar{C}\} \quad (35)$$

$$\text{vol } C = \sum_{v \in C} \pi(v) \quad (36)$$

The symbol $\pi(u)$ in the above equations represents the probability that that a random walker is at vertex u , and $p(u,v)$ represents the probability that a random walker located at vertex u will transition to vertex v . The eigenvectors of the k smallest eigenvalues of the directed Laplacian are computed as a continuous relaxation of the above discrete optimization problem and k -means clustering is used to produce the clusters.

Meilă and Pentney [21] define clustering in directed graphs to be a generalized weighted cut and they include Zhou *et al.*'s work as a special case. The graph is scaled so that $\sum_i d_i^+ = 1$, where d_i^+ is the out-degree of vertex i . Two weights are defined for vertices: T_i^V is a volume weight and T_i^R is a row weight. Cluster volumes are given by $T_{C_k}^R = \sum_{i \in C_k} T_i^R$. The generalized weighted cut (WCut) is then defined as

$$WCut(\mathcal{C}) = \sum_{m=1}^k \sum_{m' \neq m} \frac{Cut(C_m, C_{m'})}{T_{C_m}^R} \quad (37)$$

where

$$Cut(C_m, C_{m'}) = \sum_{i \in C_m} \sum_{j \in C_{m'}} T_i^R W_{ij} \quad (38)$$

Zhou *et al.*'s work corresponds to using Meilă and Pentney's algorithm with volume weights T_i^V being the stationary distribution of the teleporting random walk. The above $Cut(C_m, C_{m'})$ is a generalization of normalized cuts in undirected graphs. The goal is to minimize

$WCut(\mathcal{C})$ to produce a clustering with balanced clusters having low arc weights between clusters.

In their work, Meilă and Pentney take a similar linear programming approach as was done in the undirected graph case. Suppose X_i is an indicator vector of cluster C_i (i.e.: the j th component of X_i is 1 if $v_j \in C_i$ and 0 otherwise). Finding an optimum clustering is NP-hard but the linear programming approach is to relax the integrality constraint and find solution vectors Y_i whose elements are in the range $[0, 1]$. They show that if $B = (T^V)^{-1/2}(D - W)(T^V)^{-1/2}$ then the eigenvectors of the Hermitian matrix $H(B) = \frac{1}{2}(B + B^T)$ give a continuous solution approximation for the indicator vectors X_i . The approximation gives a lower bound on the sum of the edge weights that are cut to form the clusters. Thus, they make the small relaxation on integrality which allows the same clustering methods to be used as the undirected case (since $H(B)$ is symmetric). Previous authors had attempted this by using the co-citation matrix ($W^T W$), co-reference matrix ($W W^T$), or simply $\frac{1}{2}(W + W^T)$. Meilă and Pentney's method gave very low classification errors on experiments with known correct clusters and it performed much better than the previous, more naïve, methods.

Chen *et al.* [22] claim to generalize Zhou *et al.* and, in fact, they directly copy large portions of their paper [20]. It seems they were unaware that Zhou *et al.* suggested using a teleporting random walk and they cite it as one of the advantages of their paper. They use the combinatorial Laplacian of a directed graph where Zhou *et al.* use the symmetrically normalized Laplacian but are also clearly aware of the combinatorial Laplacian since they use it at one point to compute an approximation to their solution. When conducting comparative experiments, Chen *et al.* implement Zhou *et al.*'s method with incorrect parameters and they conduct a simpler version of the experiment in [20] so it is not possible from the information given to determine if using the combinatorial Laplacian is indeed advantageous.

Agarwal *et al.* [2] prove that previous work on spectral clustering of hypergraphs reduces to first expanding the hypergraph to an undirected graph (with an appropriate weighting function) and then applying spectral graph theory in that realm. In Table 2 of Section 3.2, we gave the equivalencies between previous proposals for hypergraph clustering and their undirected graph equivalents. In the case of clique expansions, where the vertex set of the hypergraph and its expansion are the same, the eigenvectors of the hypergraph matrix and expansion graph matrix are identical; thus, the clustering of the undirected graph is the clustering of the hypergraph. In the case of star expansions, the vertex set is enlarged to $V^* = V \cup E$. If $X_{V^*} = [X_V \ X_E]$ is an eigenvector of a Laplacian or adjacency matrix of the expansion graph, then X_V is an eigenvector of the related hypergraph matrix with only minor algebraic manipulation.

Very little work has been done to date on clustering in directed hypergraphs and no work uses spectral methods. VLSI circuits are a common application of directed hypergraphs

but partitioning the directed hypergraph corresponding to a circuit is commonly done by non-spectral methods or by reducing the directed hypergraph to an undirected or directed graph where better partitioning algorithms are available [23]. The vast majority of research involves the transformation of a VLSI circuit problem into a problem where the application of undirected graph partitioning techniques may be applied. The most recent advance in partitioning directed hypergraphs is Gallo *et al.* [24] where the minimum cardinality cut problem is reduced to finding a minimum set of hyperarcs covering the set of hyperpaths for the graph. They use a linear programming and divide-and-conquer approach and their experiments show that their technique has exponential complexity. Further, their work does not address directed hypergraphs in general but only those with hyperarcs including a single head vertex. The most recent work on cuts in general directed hypergraphs is by Liu *et al.* [4]. They give a gradient descent method of partitioning a directed hypergraph and their work does not involve spectral graph theory.

4.2 Vertex ranking

PageRank [25] and Hyperlink-Induced Topic Search (HITS) [26] are seminal algorithms that introduced vertex ranking for directed graphs in the context of web pages. Those algorithms, along with the Stochastic Approach for Link-Structure Analysis (SALSA) [27]; AuthRank, HubRank, and SymRank [28]; and AssetRank [29], are applications of spectral graph theory to directed graphs. The algorithms have been applied to ranking web pages, privileges in attack graphs, clustering autonomous systems on the Internet, and predicting areas of link stress.

All of these methods use only the eigenvectors corresponding to the principal eigenvalue of their matrix. Kleinberg [26] shows several examples illustrating that the secondary (non-principal) eigenvectors sometimes show ranks within sub-communities in a directed graph. Skillicorn [30] also gives several examples of how the secondary eigenvectors detect prominent vertices in sub-communities of an undirected graph. In his case, he uses the eigenvectors of $L_{rw}L_{rw}^T$ ² and plots the graph vertices using neighbouring eigenvectors which gives a visual representation that makes it easier to identify vertices which could be prominent in sub-communities.

It is not known how to predict which eigenvectors will contain information on sub-communities, nor why one sub-community will be found while another is not. Borodin *et al.* [31] show that it is easy to construct examples where sub-communities exist but the secondary eigenvectors contain no information about them. Intuitively, the eigenvectors each correspond to an orthogonal vector corresponding to highly-correlated vertices so one expects they would

2. The paper uses the middle columns of the U matrix (the left singular vectors) of a singular value decomposition of L_{rw} . The columns of U correspond to finding the eigenvectors of $L_{rw}L_{rw}^T = I - P - P^T + PP^T$. If P were symmetric (it is not, in general), the columns of U would be the eigenvectors of L_{rw} since the eigenvectors of L_{rw} and L_{rw}^2 are the same.

in fact give information about sub-communities. To date, no rigorous results exist in this area.

PageRank [25] computes the stationary distribution of a probability transition matrix. The original graph is first modified, into what is referred to as a teleporting random walk, with the introduction of a random jump from each vertex with probability α . The random jump forces a strongly connected graph which guarantees the principal eigenvector is real, unique, and positive by Perron's theorem. Bianchini *et al.* [32] give a rigorous explanation of the random walker and stochastic process models which describe the semantics of PageRank for vertex ranking.

It is known that the principal eigenvalue is always 1 for probability transition matrices. Haveliwala and Kamvar studied the second eigenvalue [33] and the condition number [34] of the PageRank affinity matrix. They give an upper-bound on the second eigenvalue which in turn gives the eigengap. The eigengap determines the speed of convergence when using an iterative approach to calculate the principal eigenvector. They show that the upper-bound is in fact $1 - \alpha$ and so is directly controllable by the algorithm user.

A matrix P is ill-conditioned if small perturbations produce relatively large changes in the matrix inverse. The condition number of P is defined to be the ratio of the norm of P over the norm of P^{-1} . Haveliwala and Kamvar show that the condition number of the PageRank matrix is $\frac{2-\alpha}{\alpha}$, showing that the PageRank matrix is stable under perturbations with the commonly used $\alpha = 0.15$ but becomes unstable as α approaches 0.

AssetRank [29] extends the PageRank algorithm to AND/OR directed graphs. Here, a modification of the probability transition matrix is required in order to handle the unique semantics of AND vertices. Arcs from OR vertices have their weights normalized to sum to 1 and correspond to the probability of a random walker moving to the vertex's out-neighbours. The weight of each arc originating at AND vertices is set to 1 and correspond to a random walker replicating itself and then simultaneously moving to all of the out-neighbours. Note that the affinity matrix is no longer stochastic. To force a strongly connected graph, and thus a unique principal eigenvector, a random jump factor is added, similar to the teleporting random walk. A closed form solution of the condition number and bound on the eigengap are future work.

For both PageRank and AssetRank, the principal eigenvector gives the expected probability for a random walker to be found at each vertex. Computing the stationary distribution using the power method gives an intuitive model of a random walker.

If the adjacency matrix of a graph is not irreducible, the principal eigenvector is not guaranteed to be unique [35, 36]. For example, HITS and SALSA are two graph mining algorithms where the representation matrix is not irreducible. Thus there will be as many principal eigenvectors as the multiplicity of the largest eigenvalue (in absolute value). These algorithms calculate their output using an iterative power method and so they will return

one of the principal eigenvectors but the actual vector computed is sensitive to the initialization vector's value.

4.3 Link prediction

In many networks, it is interesting to predict the future interactions that will occur. For citation networks this could predict which authors will co-author a paper together and in terrorist cells, link prediction could discover individuals who are interacting but have not been detected yet. Liben-Nowell and Kleinberg [37] obtained good results by applying the PageRank algorithm to a citation graph. They tested five data sets from the physics e-Print arXiv where their problem was to take the data sets from 1996 and try to predict the co-authorings that would occur. It is not possible to know all future collaborations until the individuals in the data set have ceased working but they did look at which co-authorings had occurred by 1999.

In all five data sets, PageRank exceeded the prediction performance of simple graph distance. In three of the five data sets, it exceeded the performance of common neighbours prediction. However, the performance greatly depended upon the random jump parameter of PageRank and was only better for some values. The values they tested were 0.01, 0.05, 0.15, 0.30, and 0.50. The optimal values across the five data sets essentially spanned the extreme values with optimal values ranging from 0.05 to 0.50. This highlights the importance of setting correct edge weights in the adjacency matrix. Research is needed to discover what weights are optimal under which conditions.

4.4 Importance of matrix representation

The various vertex ranking, clustering, and prediction algorithms start with the same graph but they each perform a unique transformation on it to create a matrix representation to be used in further analysis. Getoor and Diehl [38] discuss the problem of data representation in their survey but while research conducted on the topic itself has not been done, examples are given where the application of a known technique on a different representation was enough to produce significant improvements. Von Luxburg [11, Sec. 8.1] writes, "Constructing the similarity graph for spectral clustering is not a trivial task, and little is known on theoretical implications of the various constructions."

Another example is Chang *et al.* [39] where a change in the arc weights of the affinity matrix using gradient ascent personalizes the vertex ranking so that it reflects user priorities. Selecting a representative matrix and suitable normalizations is an open problem and it is unknown how rigorous the process can be made.

Useful research would be to catalog all matrix representation choices available and determine which one is best suited for the various classes of data. Almost certainly, different

representations will be chosen based upon the task (e.g.: clustering, prediction, ranking). We would like to know what information the full spectrum of each representation gives, particularly the secondary eigenvectors and any relations between the various spectra.

4.5 Problems

Farahat *et al.* [36] show examples of graphs for which HITS yields non-unique or non-intuitive rankings, and similarly for SALSA. The reason is that the matrix representations chosen by those algorithms can lead to a principal eigenvalue with a multiplicity greater than one. Thus, the eigenvector is not unique and many different eigenvectors are generated. Using the iterative approaches that the authors of HITS and SALSA used, a vertex ranking is indeed obtained; however, the ranking computed in these situations depends upon the initial vector used by the algorithm (normally the all-ones vector or a scalar of it). Perron's theorem gives conditions under which the principal eigenvector exists and is unique.

Farahat *et al.* prove that multiple principal eigenvectors exist for HITS and SALSA if and only if their authority graphs (whose corresponding matrices are $W^T W$ and $D_{\text{in}}^{-1} W^T D_{\text{out}}^{-1} W$, respectively) are disconnected. They suggest several perturbations to the original graphs that make them strongly connected but more research is needed to understand how the rankings from the perturbed matrices correspond to the original matrices.

Two applications of spectral methods we have seen are link prediction and ranking. Using spectral methods, these applications only model the *eventual* state of these indicators. It is also useful to include a temporal aspect where predictive trends for link addition and deletion as well as rank trends at points in time are required. O'Madadhain *et al.* [40] show how PageRank and HITS are poor in this regard and introduce the EventRank algorithm. They apply their algorithm to modelling influence between actors in a social network over time based upon email communications [41]. A weakness in their work is that the size of the representation matrix is constant. As a result, vertices that are added to the system at time $t > 0$ are included in the system at inception (time $t = 0$) with a non-zero rank. Such a system contains erroneous rank values and could not be used for a live dynamically-changing system since the final size of the matrix would not be known in advance.

As with any data-mining method, one must take care when applying spectral analysis techniques. Gkantsidis *et al.* [42] attempt to cluster autonomous systems (ASs) using spectral analysis with mixed results. Their goal was to rank the ASs in a hierarchical order and compare the rank with the 5-level ordering of Subramanian *et al.* [43]³ as well as to find the links that were likely congested. Gkantsidis *et al.* ignore the size of the ASs as well as the bandwidth of the links between ASs. That is, they represented the ASs in a graph without assigning vertex or edge weights.

3. The ordering in [43] was obtained by a straightforward connectivity analysis.

As a result, a link was regarded to be congested whenever a relatively few links connected two dense clusters. Ignoring the bandwidth of the links is an erroneous assumption since long distance links are likely to be few in number but high in bandwidth. For example, links between continents have this property.

They use the adjacency matrix of four different data representations interchangeably and alter the matrices by assigning half the weight to the diagonal without giving adequate justification. The eigenvectors of the $n \times n$ representation matrix are used to cluster the vertices using their own algorithm. When using the linear programming approach of a discrete to continuous relaxation, the first k eigenvectors of the matrix are used where k corresponds to the number of clusters. In contrast, they arbitrarily choose k to be $n/2$ and their algorithm is:

- 1: Order the eigenvalues so that $\lambda_0 \geq \lambda_1 \geq \dots \geq \lambda_{k-1}$
- 2: **for** i from 0 to $k - 1$ **do**
- 3: Order the entries of eigenvector X_i from largest to smallest
- 4: Form the cluster C_{2i} from the vertices corresponding to the largest 25% and the cluster C_{2i+1} from the smallest 25% of the eigenvector as long as the vertices are not already in a previous cluster.
- 5: **end for**

Clearly, the first two clusters will each contain 25% of the vertices. On the other hand, it is not clear that this algorithm will include every vertex in a cluster, nor that this method will produce good clusters.

They also rank the ASs but since they did not use vertex or edge weights, the rankings do not accurately reflect the actual flows. They claim the rankings corresponded well with those in [43] but in fact the correspondence there too is poor. For example, many of the nodes that are ranked by [43] as a Level 2 AS have very low rank values according to Gkantsidis *et al.*'s analysis.

5 Future work

Graph matrices are frequently a high dimensional representation of low dimensional data. Example 6, which shows the traveller's hypergraph affinity matrix, and Example 7, which shows the equivalent bipartite undirected graph, illustrate how data is often represented with many more dimensions than are required. For a graph with known properties we need to understand the minimum basis size required to represent every possible element. Can the basis components be chosen to reflect data of interest?

Systems are modelled in graphs by assigning objects to vertices and relationships to edges. A shortcoming of all work in this area is that parameters are assigned without an adequate scientific rationale. No systematic study exists which gives theoretical justification for the

assignment of the affinity weights and other parameters, nor the effect that those parameters have on the quality of ranking, clustering, and link prediction. Graph matrices most often capture only the local affinities of the vertices' immediate neighbours; however, some algorithms (for example, Exponentiated Input to HITS [36]) also capture global affinities with exponentially decaying weights.

Graph mining algorithms extend the local affinities to global ones and the representation choice determines how this extension is done. The three Laplacian matrices each extend the global affinities in a different manner and there is a question of which representation best suits various types of data. If the distribution of degrees varies broadly, then the results obtained from the three Laplacians will differ substantially. For ranking, Ding *et al.* [28] show how the ranking algorithms fit into a unified framework under the common normalizations; however, they predict that PageRank and HITS rankings will correlate highly with simple rankings by the in and out degrees whereas Langville and Meyer [35] found that not to be the case. Research is required to give theoretical justification for the application of each normalization.

The eigenvectors of the affinity and Laplacian matrices are used to approximate a solution to the clustering problem. Transforming the continuous solution back into a discrete solution is usually done with the k -means clustering algorithm. The k -means algorithm assumes the data is clustered around a centroid in a spherical shape which is an incorrect assumption, in general. Can the shape of the clusters of graphs with known properties be cataloged or can the shape be automatically detected?

Haveliwala and Kamvar [33] computed bounds on the second eigenvalue of the PageRank matrix. Their preliminary research suggests that the analysis of secondary eigenvectors identifies spamming communities. Other authors have also suggested that sub-communities might be detectable but no technique exists to identify these subgraphs in a rigorous manner.

Application of spectral methods to the mining of directed graphs has only been addressed recently and it is still in its infancy. Laplacians have only been defined for a small class of directed graphs, namely, those that are strongly connected. Research is needed to generalize Laplacians to all directed graphs, and to mixed type AND/OR graphs.

The field is wide open for the application of spectral methods to directed hypergraph research. A cut T_{st} of a directed hypergraph $G = (V, E)$ was defined in [3] to be a partition of V into two subsets so that $s \in V_s$ and $t \in V_t$. The cut set E_{st} is the set of hyperarcs $\{e = (V^-, V^+) | e \in E, V^- \subseteq V_s, V^+ \subseteq V_t\}$. Consider the following graph and cut.

$$V = \{1, 2, 3\} \tag{39}$$

$$E = \{(\{1\}, \{2, 3\})\} \tag{40}$$

$$V_1 = \{1, 2\}, V_3 = \{3\} \tag{41}$$

The above edge is not included in the cut set $E_{1,3}$ since the cut definition requires all tail vertices to be in cluster V_s and all head vertices to be in cluster V_t . It not clear that this definition is reasonable; experiments to rank and cluster various types of data will give justification for an appropriate cut definition. In addition, Laplacian matrices, ranking, clustering, and link prediction need to be generalized to this class of graph.

Chakrabarti and Faloutsos [44] identify research areas from social network analysis where nothing equivalent has been addressed by the graph mining community. These topics are those of social roles and the influence of vertices. In their excellent survey they identify how knowledge can be inferred about the connection of vertices in social networks just by knowing their role. For example, if someone's role is that of a husband, we know he possesses the attribute "male" and he is related to another vertex with the attribute "female". There is also a known probability that he is related to one or more child vertices. Vertex influence is the idea that vertices benefit differently from the same action, depending on their influence or prestige in the network. This influence may be captured by applying a weight to the vertices; however, very little research has been done that takes vertex weights into account.

Understanding the role and influence of vertices is important in solving problems including knowing what cluster of a fixed size will maximally disrupt the network or alternatively, what is the optimal cluster of fixed size to disseminate a commodity in the network. Future research could look at incorporating these concepts from social networks into other networks and to understand how spectral methods could utilize roles and influence in the computation of rankings and clusterings in other areas.

6 Conclusion

In this paper we reviewed the four types of graphs and show how they are used to represent structured data. Spectral graph theory of undirected graphs is the most mature and the techniques have been applied to a broad range of applications. Since hypergraphs are equivalent to undirected graphs, the tools available for undirected graphs may also be applied to them. Recent work in directed graphs has generalized the tools to that domain; however, directed hypergraphs are the most general type of graph and no spectral graph theory techniques exist for them.

Spectral graph theory primarily depends upon the affinity matrix. Obtaining an affinity matrix that accurately reflects the problem that is to be modelled is an area in which no rigorous methods exist. Nevertheless, graph data mining is applied in numerous applications daily with exceptional results. However, a better understanding of how to obtain accurate matrix representations and the effect of tuning parameters will give the graph mining results a higher degree of accuracy and applicability. Assuming an accurate affinity matrix is available, an understanding of how local affinities should map into global affinities in each

problem area, as well as the effect of each of the affinity and Laplacian matrices in this analysis, is critical to remove subjectivity and give well-founded semantics for the results.

References

- [1] Skillicorn, D. B. (2007), *Understanding Complex Datasets: Data Mining with Matrix Decompositions*, New York: Chapman & Hall/CRC.
- [2] Agarwal, S., Branson, K., and Belongie, S. (2006), Higher order learning with graphs, *Proceedings of the 23rd International Conference on Machine Learning*, pp. 17–24.
- [3] Gallo, G., Longo, G., Pallottino, S., and Nguyen, S. (1993), Directed hypergraphs and applications, *Discrete Applied Mathematics*, 42(2-3), 177–201.
- [4] Liu, L.-T., Kuo, M.-T., Huang, S.-C., and Cheng, C.-K. (1995), A Gradient Method on the Initial Partition of Fiduccia-Mattheyses Algorithm, In *Proceedings of the 1995 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, pp. 229–235, IEEE Computer Society.
- [5] Ausiello, G., Franciosa, P. G., and Frigioni, D. (2001), Directed Hypergraphs: Problems, Algorithmic Results, and a Novel Decremental Approach, *Lecture Notes in Computer Science*, pp. 312–327.
- [6] Chung, F. R. K. (2005), Laplacians and the Cheeger Inequality for Directed Graphs, *Annals of Combinatorics*, 9(1), 1–19.
- [7] Meyer, C. D. (2000), *Matrix analysis and applied linear algebra*, Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.
- [8] Hagen, L. and Kahng, A. B. (1992), New spectral methods for ratio cut partitioning and clustering, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 11(9), 1074–1085.
- [9] Shi, J. and Malik, J. (2000), Normalized cuts and image segmentation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8), 888–905.
- [10] Wagner, D. and Wagner, F. (1993), Between min cut and graph bisection, In *Proceedings of the 18th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, p. 744–750, Springer.
- [11] von Luxburg, U. (2006), A tutorial on spectral clustering, Technical Report Max Planck Institute for Biological Cybernetics.

- [12] MacQueen, J. B. (1967), Some methods for classification and analysis of multivariate observations, In *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297, Berkeley: University of California Press.
- [13] Guattery, S. and Miller, G. L. (1998), On the quality of spectral separators, *SIAM Journal on Matrix Analysis and Applications*, 19(3), 701–719.
- [14] Meilă, M. and Shi, J. (2001), A Random Walks View of Spectral Segmentation, In *AI and STATISTICS (AISTATS) 2001*.
- [15] Brand, M. and Huang, K. (2003), A unifying theorem for spectral embedding and clustering, In *Proceedings of the 9th International Workshop on Artificial Intelligence and Statistics (AISTATS)*, pp. 1–8.
- [16] Maggioni, M. and Coifman, R. (2007), Multiscale analysis of data sets using diffusion wavelets, In *Proceedings of the Data Mining for Biomedical Informatics Workshop*.
- [17] Mahadevan, S. and Maggioni, M. (2006), Value Function Approximation with Diffusion Wavelets and Laplacian Eigenfunctions, *Advances in Neural Information Processing Systems (NIPS)*, 18, 843–850.
- [18] Lafon, S. and Lee, A. B. (2006), Diffusion Maps and Coarse-Graining: A Unified Framework for Dimensionality Reduction, Graph Partitioning, and Data Set Parameterization, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1393–1403.
- [19] Nadler, B., Lafon, S., Coifman, R. R., and Kevrekidis, I. G. (2006), Diffusion maps, spectral clustering and reaction coordinates of dynamical systems, *Applied and Computational Harmonic Analysis*, 21, 113–127.
- [20] Zhou, D., Huang, J., and Schoelkopf, B. (2005), Learning from labeled and unlabeled data on a directed graph, In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, pp. 1041–1048.
- [21] Meilă, M. and Pentney, W. (2007), Clustering by weighted cuts in directed graphs, In *Proceedings of the 7th SIAM International Conference on Data Mining*, pp. 135–144.
- [22] Chen, M., Yang, Q., and Tang, X. (2007), Directed Graph Embedding, In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 2707–2712.
- [23] Chen, S.-J. and Cheng, C.-K. (2000), Tutorial on VLSI partitioning, *VLSI Design*, 11(3), 175–218.

- [24] Gallo, G., Gentile, C., Pretolani, D., and Rago, G. (1998), Max Horn SAT and the minimum cut problem in directed hypergraphs, *Mathematical Programming*, 80(2), 213–237.
- [25] Page, L., Brin, S., Motwani, R., and Winograd, T. (1998), The PageRank Citation Ranking: Bringing Order to the Web, Technical Report Stanford University.
- [26] Kleinberg, J. M. (1999), Authoritative sources in a hyperlinked environment, *Journal of the ACM*, 46(5), 604–632.
- [27] Lempel, R. and Moran, S. (2001), SALSA: The Stochastic Approach for Link-Structure Analysis, *ACM Transactions on Information Systems*, 19(2), 131–160.
- [28] Ding, C., He, X., Husbands, P., Zha, H., and Simon, H. D. (2002), PageRank, HITS and a unified framework for link analysis, In *Proceedings of the 25th Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pp. 353–354, New York, NY, USA: ACM.
- [29] Sawilla, R. E. and Ou, X. (2008), Identifying Critical Attack Assets in Dependency Attack Graphs, In *Proceedings of the 13th European Symposium on Research in Computer Security (ESORICS)*, pp. 18–34.
- [30] Skillicorn, D. B. (2007), Detecting Anomalies in Graphs, In *Proceedings of the IEEE Intelligence and Security Informatics*, pp. 209–216.
- [31] Borodin, A., Roberts, G. O., Rosenthal, J. S., and Tsaparas, P. (2001), Finding authorities and hubs from link structures on the World Wide Web, In *WWW '01: Proceedings of the 10th International Conference on World Wide Web*, pp. 415–429, New York, NY, USA: ACM.
- [32] Bianchini, M., Gori, M., and Scarselli, F. (2005), Inside PageRank, *ACM Transactions on Internet Technology (TOIT)*, 5(1), 92–128.
- [33] Haveliwala, T. H. and Kamvar, S. D. (2003), The second eigenvalue of the Google matrix, *A Stanford University Technical Report*, 8090, 2003–2020.
- [34] Kamvar, S. D. and Haveliwala, T. H. (2003), The condition number of the PageRank problem, *A Stanford University Technical Report*, pp. 1–4.
- [35] Langville, A. N. and Meyer, C. D. (2005), A Survey of Eigenvector Methods for Web Information Retrieval, *SIAM Review*, 47(1), 135–161.
- [36] Farahat, A., LoFaro, T., Miller, J. C., Rae, G., and Ward, L. A. (2006), Authority Rankings from HITS, PageRank, and SALSA: Existence, Uniqueness, and Effect of Initialization, *SIAM Journal on Scientific Computing*, 27(4), 1181–1201.

- [37] Liben-Nowell, D. and Kleinberg, J. M. (2003), The link prediction problem for social networks, In *Proceedings of the 12th International Conference on Information and Knowledge Management (CIKM)*, pp. 556–559, New York, NY, USA: ACM.
- [38] Getoor, L. and Diehl, C. P. (2005), Link mining: a survey, *ACM SIGKDD Explorations Newsletter*, 7(2), 3–12.
- [39] Chang, H., Cohn, D., and McCallum, A. (2000), Creating customized authority lists, In *Proceedings of the 17th International Conference of Machine Learning*, pp. 49–54.
- [40] O’Madadhain, J. and Smyth, P. (2005), EventRank: a framework for ranking time-varying networks, In *Proceedings of the 3rd International Workshop on Link Discovery*, pp. 9–16, ACM Press New York, NY, USA.
- [41] O’Madadhain, J., Hutchins, J., and Smyth, P. (2005), Prediction and ranking algorithms for event-based network data, *ACM SIGKDD Explorations Newsletter*, 7(2), 23–30.
- [42] Gkantsidis, C., Mihail, M., and Zegura, E. (2003), Spectral analysis of Internet topologies, In *Proceedings of the 22nd Annual IEEE INFOCOM 2003*, Vol. 1.
- [43] Subramanian, L., Agarwal, S., Rexford, J., and Katz, R. H. (2002), Characterizing the Internet hierarchy from multiple vantage points, In *INFOCOM 2002. Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies*, Vol. 2.
- [44] Chakrabarti, D. and Faloutsos, C. (2006), Graph Mining: Laws, Generators, and Algorithms, *ACM Computing Surveys*, 38, 2–70.
- [45] Belkin, M. and Niyogi, P. (2002), Laplacian eigenmaps and spectral techniques for embedding and clustering, *Advances in Neural Information Processing Systems*, Vol. 14.
- [46] Chung, F. R. K. (2006), Spectral Graph Theory (revised and improved), Manuscript.
- [47] Henzinger, M. R. (2003), Algorithmic Challenges in Web Search Engines, *Internet Mathematics*, 1(1), 115–126.
- [48] Ng, A. Y., Jordan, M. I., and Weiss, Y. (2002), On spectral clustering: Analysis and an algorithm, In *Advances in Neural Information Processing Systems 14: Proceedings of the 2002 Conference*, pp. 849–856, MIT Press.
- [49] Zhou, D., Huang, J., and Schoelkopf, B. (2006), Learning with hypergraphs: Clustering, classification, and embedding, *Advances in Neural Information Processing Systems (NIPS)*, Vol. 19.

This page intentionally left blank.

DOCUMENT CONTROL DATA		
<small>(Security classification of title, body of abstract and indexing annotation must be entered when document is classified)</small>		
1. ORIGINATOR (The name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.) Defence R&D Canada – Ottawa 3701 Carling Avenue, Ottawa ON K1A 0Z4, Canada	2. SECURITY CLASSIFICATION (Overall security classification of the document including special warning terms if applicable.) UNCLASSIFIED	
3. TITLE (The complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title.) A survey of data mining of graphs using spectral graph theory		
4. AUTHORS (Last name, followed by initials – ranks, titles, etc. not to be used.) Sawilla, R.		
5. DATE OF PUBLICATION (Month and year of publication of document.) December 2008	6a. NO. OF PAGES (Total containing information. Include Annexes, Appendices, etc.) 44	6b. NO. OF REFS (Total cited in document.) 49
7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.) Technical Memorandum		
8. SPONSORING ACTIVITY (The name of the department project office or laboratory sponsoring the research and development – include address.) Defence R&D Canada – Ottawa 3701 Carling Avenue, Ottawa ON K1A 0Z4, Canada		
9a. PROJECT OR GRANT NO. (If appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant.) 15BO04	9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.)	
10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.) DRDC Ottawa TM 2008-317	10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.)	
11. DOCUMENT AVAILABILITY (Any limitations on further dissemination of the document, other than those imposed by security classification.) <input checked="" type="checkbox"/> Unlimited distribution <input type="checkbox"/> Defence departments and defence contractors; further distribution only as approved <input type="checkbox"/> Defence departments and Canadian defence contractors; further distribution only as approved <input type="checkbox"/> Government departments and agencies; further distribution only as approved <input type="checkbox"/> Defence departments; further distribution only as approved <input type="checkbox"/> Other (please specify):		
12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11)) is possible, a wider announcement audience may be selected.) Unlimited		

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

Relational data may be represented by four classes of graphs: undirected graphs, directed graphs, hypergraphs, and directed hypergraphs. We present examples of data that is well suited to each type and describe how linear algebra has been used in the literature to cluster like objects, rank objects according to their importance to the system the graph represents, and predict relationships that likely exist in the system but have not yet been identified. In particular, we associate affinity and Laplacian matrices with a graph and apply spectral graph theory — the study of the eigenvalues, eigenvectors, and characteristic polynomial of graph matrices — to mine hidden patterns in the graphs. Shortcomings of existing techniques and open problems for future work are identified.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus. e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

spectral graph theory, graph data mining, eigenvector, clustering, ranking, prediction

Defence R&D Canada

Canada's leader in Defence
and National Security
Science and Technology

R & D pour la défense Canada

Chef de file au Canada en matière
de science et de technologie pour
la défense et la sécurité nationale



www.drdc-rddc.gc.ca