A Case Study on Financial Ratios via Cross-Graph Quasi-Bicliques

Kelvin Sim[a,b], Guimei Liu[c], Vivekanand Gopalkrishnan[b], Jinyan Li[b]

[a]*Institute for Infocomm Research, A*STAR, Singapore.*
[b]*School of Computer Engineering, Nanyang Technological University, Singapore.*
[c]*School of Computing, National University of Singapore.*

Abstract

Stocks with similar financial ratio values across years have similar price movements. We investigate this hypothesis by clustering groups of stocks that exhibit homogeneous financial ratio values across years, and then study their price movements. We propose using *Cross-Graph Quasi-Biclique (CGQB) subgraphs* to cluster stocks, as they can define the three dimensional (3D) subspaces of financial ratios that the stocks are homogeneous in across the years, and they can also handle missing values that are rampant in the stock data. Furthermore, investors can easily analyze these 3D subspaces to explore the relations between the stocks and financial ratios. We develop a novel algorithm, $CGQBminer$, which mines the complete set of CGQB subgraphs from the stock data. Through experimental analysis, we show that the hypothesis is valid. Furthermore, we demonstrate that having an investment strategy which uses groups of stocks mined by CGQB subgraphs have higher returns than one that does not. We also conducted an extensive performance analysis on $CGQBminer$, and show that it is efficient across different 3D datasets and parameter settings.

Keywords: Financial data mining, Financial ratios analysis, Quasi-bicliques, 3D subspace clustering

Email addresses: shsim@i2r.a-star.edu.sg (Kelvin Sim), liugm@comp.nus.edu.sg (Guimei Liu), asvivek@ntu.edu.sg (Vivekanand Gopalkrishnan), jyli@ntu.edu.sg (Jinyan Li)

1. Introduction

In 1934, Graham and Dodd introduced the concept of *value investing* [12], which involves analyzing financial ratios to pick stocks. Today, value investors such as Warren Buffett, have been outperforming the market indices [34]. Financial ratios reflect the core 'health' status of a stock. For example, *Return on Equity* ratio (ROE) measures the efficiency of the stock in using its assets to produce profit, while *Debt-Equity* ratio (D/E) measures how much assets of the stock are debts[1]. It is believed that financial ratio values are crucial indicators of how future stock prices move in the market [11, 12], and if this claim is valid, investors can utilize this knowledge to make better investment choices. For example, if investors know which particular financial ratio values will lead to rising stock price, they can buy stocks having these financial ratio values to make profits. Note that the concept of value investing is different from *modern portfolio theory* [30]. The former advocates the buying of stocks with 'good' financial ratios, while the latter advocates the buying of stocks with good historical price performance, but with diverse price movements (to minimize the risk).

Previous works [5, 25] have studied the effect of an individual financial ratio on stock price movements across years, but no work has investigated how financial ratios collectively affect stock price movements across years. Different financial ratios quantify different financial aspects of a stock, so it will be interesting to study the collective influence of financial ratios on the stock price movements.

There is another school of thought which believes that the stock market is efficient and the stock prices reflect all known information [10]. That is, the current price of a stock fully reflects its 'health' status, and studying its financial ratios will not unearth hidden information, such as the stock is currently undervalued or overvalued.

Given these two contrasting views on financial ratios, we study the effect of financial ratios on price movements using the following hypothesis:

Hypothesis 1. *Stocks with similar financial ratio values across years have similar price movements.*

We conduct a systematic investigation of Hypothesis 1, and our *research design* consists of three main phases, namely, *data preparation, data mining* and *data analysis*. The data preparation phase prepares the stock data of financial

[1]The definitions of these financial ratios are given in Appendix A.

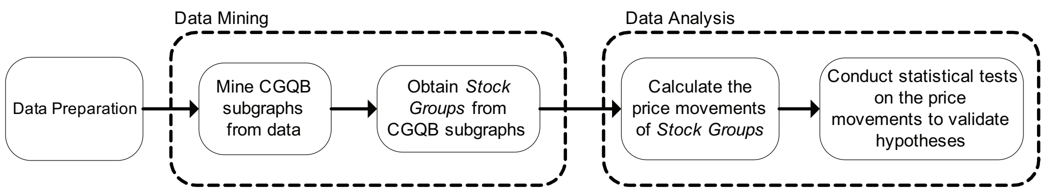

Data Mining
Data Analysis
Data Preparation
Mine CGQB subgraphs from data
Obtain *Stock Groups* from CGQB subgraphs
Calculate the price movements of *Stock Groups*
Conduct statistical tests on the price movements to validate hypotheses

Figure 1: The research design structure of this work.

ratios, the data mining phase clusters stocks with similar financial ratios values across years, and the data analysis phase evaluates the price movements of these stocks to validate Hypothesis 1. The structure of the research design is presented in Figure 1, and details of the research design is in Section 2.

The data preparation phase is an engineering task of converting the financial figures into financial ratios, while the data analysis phase uses standard statistical tests to validate the significance of Hypothesis 1. The challenge is in the data mining phase, as existing clustering techniques lack the ability to handle the highly complex stock data. Hence, in this paper, besides investigating Hypothesis 1, we also focus on developing a novel and efficient clustering approach to handle the stock data.

We first give a description of the characteristics of the stock data and the weakness of existing techniques in handling it, before presenting our proposed clustering approach. The stock data can be represented as a set of yearly transaction tables, with rows corresponding to stocks and columns corresponding to the financial ratios. An example is the set of tables shown in the middle of Figure 3.

Firstly, the stock data has a large number of features (financial ratios). If traditional clustering approaches (e.g. k-means) are used, they are bound to suffer from the curse of dimensionality [23]. Applying feature selection or transformation techniques such as PCA can reduce the number of features, but the stocks are clustered based on the global transformed space, which means that the financial ratios that a cluster of stocks are similar in is unknown. Therefore, subspace clustering approaches [23] are more suitable for the stock data, as they can find clusters of stocks in different subspaces of financial ratios. Secondly, the stock data is a three dimensional (3D) data (stocks $\times$ financial ratios $\times$ years) and hence 3D subspace clustering algorithm that handles 3D data is required. Thirdly, the stock data, similar to any real-world data, has high percentage of missing values (in fact, 26% of the stock data used in this paper are missing values). Clustering without considering the missing values naturally degrades the quality of the clus-

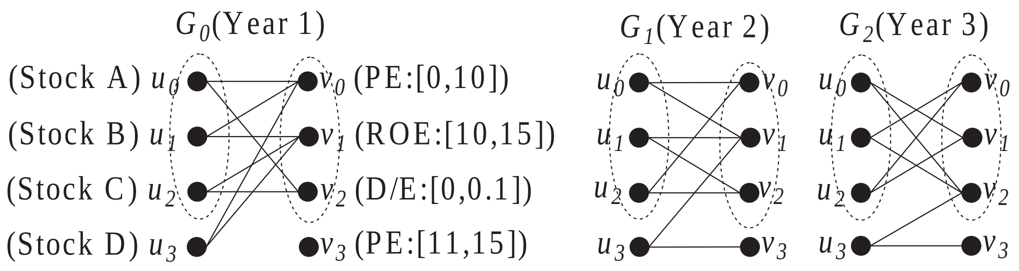


Figure 2: Example of a stock data modeled into a set of bipartite graphs. The encircled vertex sets correspond to a CGQB subgraph.

tering results. Existing 3D subspace clustering algorithms [6, 19, 50] do not handle errors, and it is not straightforward to modify them to handle missing values. Some existing 2D subspace clustering algorithms can handle missing values, but it is computationally expensive to first apply 2D subspace clustering algorithms on each year's data, and then combine the 2D subspace clusters mined from each year to form 3D subspace clusters.

1.1. Overview of Our Approach

We propose a graph-based 3D subspace clustering approach, which handles all three characteristics of the stock data, whereby

1. stocks are clustered based on subspaces of financial ratios that they are *similar* in.
2. the subspace clusters of stocks are of 3D.
3. stocks can still be clustered in the presence of missing financial ratio values.

We achieve this by first modeling the stock data into a set of bipartite graphs $D = \{G_1, \ldots, G_n\}$, as shown in Figure 2. Each bipartite graph represents stocks' data for a year, with vertices on one side of the graph representing stocks, and vertices on the other side representing partitions of financial ratio values. Let us assume that the partitions are arbitrary created for illustration purpose. The *Price-Earnings* (PE) ratio measures the price of the stock in relative to the earnings of the stock. For example, the vertex with PE:[0,10] represents the partition of PE from 0 to 10. If the value of a financial ratio of stock s falls in partition f of the corresponding ratio, an edge connects s and f.

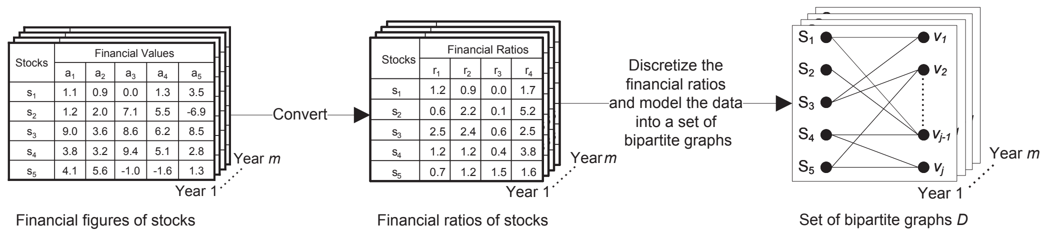
We then mine all *Cross-Graph Quasi-Biclique (CGQB)* subgraphs (depicted by pairs of encircled vertex sets in Figure 2) from D, where a CGQB subgraph corresponds to a group of stocks (denoted as *stock group*) and a corresponding group of partitions on financial ratios, for *most* of which they share similar values across several years. More specifically, a CGQB subgraph g consisting of two disjoint sets of vertices $\{U', V'\}$, occurs in D if every vertex in U' is connected to at least $|V'| - \epsilon$ vertices in V', and vice versa. ϵ is the *quasi* threshold which controls the strictness of the connectivities between U' and V'. Therefore, by mapping this financial problem into a 'quasi' graph-based problem, stocks can be clustered in the presence of missing financial ratio values. Note that u_3 is not part of the CGQB subgraph in Figure 2, as it is only densely connected to vertex set $\{v_0, v_1, v_2\}$ in year 1 but not in year 2 and 3.

The naïve way to mine all CGQB subgraphs from D can be based on the two mining stages of Algorithm TRICLUSTER [50], which mines 3D subspace clusters. In the first stage, we mine quasi-biclique subgraphs from each bipartite graph $G \in D$, and then in the second stage, we generate CGQB subgraphs from these quasi-biclique subgraphs. This approach is computationally expensive, as we show in Section 6.3. To overcome this problem, we develop a novel algorithm, $CGQBminer$, to efficiently mine the complete set of CGQB subgraphs. $CGQBminer$ uses an effective search space strategy, *tri-extensions*, and incorporates several pruning techniques to efficiently mine the cross-graph quasi-bicliques. In our experiments, we show the efficacy of $CGQBminer$ on datasets of different dimensions and densities.

Besides being competent in solving this financial data mining problem, the CGQB model is in fact generic, and can be used for clustering any noisy 3D datasets in other domains, such as gene $\times$ sample $\times$ time microarray datasets in biology, item $\times$ transaction $\times$ store datasets in consumer analysis, etc.

1.2. Layout of the paper

The rest of the paper is organized as follows. Section 2 presents the research design used to investigate Hypothesis 1. Section 3 discusses the related work. Section 4.1 presents the formal definition, problem statement of CGQB subgraphs. Section 4.2 presents the algorithm to mine CGQB subgraphs, $CGQBminer$. Section 5 describes the methodology to evaluate the price movements of stock groups. Section 6 presents the experimental results. Section 7 discusses the key findings of the experimentations and the limitations of our approach. Section 8 concludes the paper.



Stocks	Financial Values				
	a_1	a_2	a_3	a_4	a_5
s_1	1.1	0.9	0.0	1.3	3.5
s_2	1.2	2.0	7.1	5.5	-6.9
s_3	9.0	3.6	8.6	6.2	8.5
s_4	3.8	3.2	9.4	5.1	2.8
s_5	4.1	5.6	-1.0	-1.6	1.3

Stocks	Financial Ratios			
	r_1	r_2	r_3	r_4
s_1	1.2	0.9	0.0	1.7
s_2	0.6	2.2	0.1	5.2
s_3	2.5	2.4	0.6	2.5
s_4	1.2	1.2	0.4	3.8
s_5	0.7	1.2	1.5	1.6

Year m
Year 1
Financial figures of stocks
Convert
Year m
Year 1
Financial ratios of stocks
Discretize the financial ratios and model the data into a set of bipartite graphs
S1 S2 S3 S4 S5
v1 v2 vj-1 vj
Year m
Year 1
Set of bipartite graphs D

Figure 3: The data preparation phase.

2. Research Design

As mentioned in Section 1, our research design consists of three main phases: data preparation, data mining and data analysis. Figure 1 presents the structure of our research design.

2.1. Data Preparation Phase

Figure 3 shows the general overview of the preparation phase. In the data preparation phase, financial figures of stocks across years are obtained and converted into a dataset of financial ratios, which can be represented as a set of yearly transaction tables, with rows corresponding to stocks and columns corresponding to the financial ratios. The set of tables shown in the left of Figure 3 is an example of a set of financial figures. For example, a_1 and a_2 can be financial figures *Stock Price per Share* and *Earnings per Share* respectively. The set of tables shown in the middle of Figure 3 is an example of a set of financial ratios. For example, r_1 can be the *Price-Earnings* (PE) ratio which is obtained by $\frac{\textit{Stock Price per Share}}{\textit{Earnings per Share}}$. This dataset of financial ratios is then discretized and modeled into a set of bipartite graphs D, with its rational explained earlier in Section 1.1. Details of the data preparation phase is given in Section 6.1.

2.2. Data Mining Phase

As we described in Section 1, the stock data is highly complex and existing clustering techniques have limitations in handling it. Hence, we propose mining CGQB subgraphs, which is a novel clustering approach that can handle the stock data. CGQB subgraphs are mined from the set of bipartite graphs D, and each CGQB subgraph corresponds to a set of stocks (denoted as *stock group*) that have similar financial ratio values across years. The formal definition and problem statement of CGQB subgraphs are presented in Section 4.1, and the algorithm to mine CGQB subgraphs is explained in Section 4.2. Section 6.2 explains how the parameters are set in mining CGQB subgraphs.

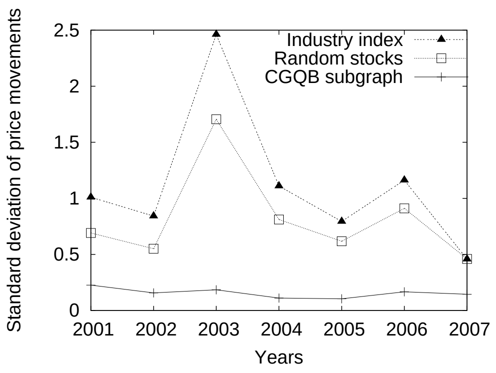


Figure 4: Standard deviation of price movements of different groups of stocks.

2.3. Data Analysis Phase

We calculate the price movements of the stocks in the stock group of each CGQB subgraph, and measure the standard deviations of their price movements, which is a standard way of measuring the dispersion of their price movements [8]. Statistical tests are then conducted to investigate if the standard deviations of price movements of stock groups are significantly lower than those of randomly picked stocks and industry indices. If so, we cannot reject Hypothesis 1 and conclude that it is valid. Details of the methodology to investigate Hypothesis 1 is presented in Section 5, and Section 6.2 presents the statistical results. Figure 4 shows a sample result of different groups of stocks from the Chemical industry. In this example, the standard deviation of the industry index is based on 35 stocks, the standard deviation of random stocks is based on an average of 10 groups of 5 random stocks, and the standard deviation of the CGQB subgraph is based on 5 stocks that have similar financial ratios across years. From our experiments, we show that stock groups obtained by CGQB subgraphs are statistically proven to have lower standard deviations than those of randomly picked stocks and industry index[2], thus confirming the validity of Hypothesis 1. Moreover, we show that more statistically significant stock groups can be discovered using CGQB subgraphs than using existing 3D subspace clusters, as CGQB subgraphs can tolerate high percentage of missing values in the stock data.

[2]The industry index is an aggregate representation of the prices of stocks from an industry.

3. Related Work

Jiang *et al.* [20, 32, 33] proposed *cross-graph quasi-clique* (CGQC), which is a set of graphs with vertex set g, and in each graph, each vertex connects at least $\gamma.(|g|-1)$ other vertices in g. Hence, CGQCs are groups of 'intra-connected' vertices, while CGQBs are pairs of groups of 'inter-connected' vertices.

Another related work to cross-graph mining is frequent subgraph mining [15, 16, 24, 47, 48], where subgraphs that occur frequently in a set of graphs are mined. There is no constraint that the structure of the subgraph must be clique or biclique, but the edges of the graph dataset must be labeled. Thus, frequent subgraph mining is not suitable for our problem.

Triclusters [50], closed 3-sets [6] and frequent closed cubes [19] are 3D subspace clustering algorithms which can mine *cross-graph biclique* (CGB) subgraphs from a set of bipartite graphs (represented as a set of binary matrices). CGB subgraphs are CGQB subgraphs without the quasi threshold, so they do not tolerate missing values. In our experiments, we show that it is harder to discover stock groups by CGB subgraphs.

Mining biclique subgraphs from a single graph is a well-established research problem [2, 9, 27, 29]. A biclique is a disjoint pair of vertex sets where there is full connection between them. Due to this strictness of connectivities, it is not suitable for dataset that has missing values.

There are several works in mining quasi-biclique (QB) subgraphs from a single graph [1, 4, 28, 31, 38, 39, 46], and not from a set of bipartite graphs. The QBs defined by [1, 31, 46] are inclined to have skewed distribution of missing edges in their QBs, as the missing edges allowed in each vertex of their QBs are not restricted. QBs defined by [4, 28, 38, 39] have this missing edges restrictions on each vertex of their quasi-biclique. Due to the high computation costs of mining QB subgraphs, only [28, 38, 39, 46] mine the complete set of results, while [1, 4, 31] do not. Sim *et al.* [38] proposed clustering stocks with similar financial ratio values, but their clustering technique is limited only to a year, and they did not study the relation between financial ratios and price movements of stocks.

In formal concept, Besson *et al.* [3] introduced fault tolerance into bi-sets by proposing DR-bi-sets, which correspond to QBs. However, Besson *et al.* mine the complete set of DR-bi-sets from a single binary matrix (which represents a bipartite graph), and not a set of binary matrices.

There are a wide range of discretization techniques [35, 41, 43] which can be applied on the financial ratios of the stock data, thus we leave the choice of the discretization technique to the users. In this paper, we use the agglomerative hierarchical clustering with $CDbw$ index [3] [14, 45], which maximizes the inter distances between partitions of a financial ratio and minimizes the intra distances within partitions of the financial ratio. As this technique is parameter-free, we avoid the study of the impact of discretization parameters on the clustering results. For a fair comparison in our experiments, CGQB subgraphs and CGB subgraphs are mined from the same set of bipartite graphs which represents the discretized stock data.

Kovalerchuk and Vityaev [22] gave an excellent introduction on using data mining techniques for financial applications. Similar to our problem, they discussed on mining patterns from stocks to identify good stocks to purchase. However, they focused on different types of stock data, namely the technical data (e.g. stock price) and macro-economic data (e.g. industrial production indices and currency exchange rates).

There are also several works on applying computational techniques to analyze financial ratios, but they are solving problems different from ours. Wang and Lee [44] proposed to identify representative financial ratios by clustering the financial ratios and using a representative indicator to represent each cluster. Yen [49] proposed using adaptive resonance theory to predict accounting frauds using financial ratios. Self-organizing learning array [51], Gaussian case-based reasoning [26] and neural network-genetic programming hybrids [36] were used to predict the failure of companies using financial ratios.

4. Cross-Graph Quasi-Bicliques (CGQBs)

We present CGQBs, which are used in the data mining phase to mine stock groups that have similar financial ratio values across years. We first present the formal definition of CGQBs and the CGQB subgraphs mining problem, and then we present the algorithm to mine CGQB subgraphs.

4.1. Definitions and Problem Statement

An undirected graph G consists of a set of vertices denoted by $\mathcal{V}(G)$ and a set of edges denoted by $E(G) = \{\{u, v\} | u, v \in \mathcal{V}(G)\}$. An edge $\{u, v\}$ denotes that u and v are connected.

[3] $CDbw(c) = Intra\_den(c) \times Sep(c)$, where c is the set of clusters (partitions), $Intra\_den(c)$ measures the density of each cluster [14, 45] and $Sep(c)$ measures both the inter-cluster distances and inter-cluster densities.

Definition 1. (Neighborhood of v) *The neighborhood of v in graph G is denoted as $N_G(v) = \{u | \{v, u\} \in E(G) \wedge u \in \mathcal{V}(G)\}$. The neighborhood of v in V is denoted as $N_G^V(v) = \{u | \{v, u\} \in E(G) \wedge u \in V \subseteq \mathcal{V}(G)\}$, given that $v \in \mathcal{V}(G) \setminus V$.*

Graph G' is a subgraph of graph G, denoted as $G' \subseteq G$, if $\mathcal{V}(G') \subseteq \mathcal{V}(G)$ and $E(G') \subseteq E(G)$. Graph G' is a *proper* subgraph of G if G' is a subgraph of G, and $G' \neq G$. Graph G' is an *induced* subgraph of G if, for any pair of vertices u and v of G', edge $\{u, v\}$ is in $E(G')$ if and only if edge $\{u, v\}$ is in $E(G)$. The edge set of an induced subgraph is determined by its vertex set, so we also use $G(V')$ to represent the subgraph of G induced on vertex set $V' \subseteq \mathcal{V}(G)$.

Graph G is a bipartite graph if its vertex set consists of two disjoint sets of vertices U and V, and edges only exist between vertices in U and vertices in V. That is, $\mathcal{V}(G) = \{U, V\}$, and $E(G) = \{\{u, v\} | u \in U \wedge v \in V\}$. A bipartite graph G is complete if there is complete connections between vertices in U and vertices in V, that is $E(G) = \{\{u, v\} | \forall v \in V \wedge \forall u \in U\}$. For brevity, a complete bipartite graph (or subgraph) is called a biclique (or biclique subgraph).

The full connection requirement of bicliques is often too restrictive for real world data because real data often contain missing values. To solve this problem, we allow some missing edges between vertices in U and vertices in V, and we call such graphs *quasi-bicliques*. We give the formal definition below.

Definition 2. (ϵ-QB) *Let G be a bipartite graph with $\mathcal{V}(G) = \{U, V\}$, and ϵ be the quasi threshold. G is a ϵ-QB if and only if*

- $\forall u \in U$, $|N_G^V(u)| \geq |V| - \epsilon$
- $\forall v \in V$, $|N_G^U(v)| \geq |U| - \epsilon$.

Parameter ϵ controls the strictness of the connectivities between the vertex sets U and V of quasi-bicliques to prevent skewed distribution of missing edges. In Figure 2, at $\epsilon = 1$, there is a quasi-biclique subgraph $G_0(U', V')$ in G_0, with $U' = \{u_0, u_1, u_2\}$ and $V' = \{v_0, v_1, v_2\}$.

Lemma 1. *If a bipartite graph G is a ϵ-QB, then for any induced subgraph G' of G, G' is also a ϵ-QB.*

PROOF. Let $\mathcal{V}(G) = \{U, V\}$ and G' be induced on vertex set $\{U', V'\}$, where $U' \subseteq U$ and $V' \subseteq V$. For all $v \in V' \subseteq V$, we have $|U'| - |N_G^{U'}(v)| \leq |U| - |N_G^U(v)| \leq \epsilon$ because $U' \subseteq U$; and for all $u \in U' \subseteq U$, we have $|V'| - |N_G^{V'}(u)| \leq |V| - |N_G^V(u)| \leq \epsilon$ because $V' \subseteq V$.

We represent the stock data of one year as one bipartite graph, so groups of stocks having similar values on a group of financial ratios across several years correspond to quasi-biclique subgraphs that occur in several bipartite graphs. We use a minimum support threshold ms_g to constrain the minimum number of bipartite graphs containing a quasi-biclique subgraph.

Definition 3. (Cross-graph quasi-biclique (CGQB) subgraph) *Let $D = \{G_1, \ldots, G_n\}$ be a set of bipartite graphs with $\mathcal{V}(G_1) = \ldots = \mathcal{V}(G_n) = \{U, V\}$. Given a quasi threshold ϵ and a minimum support threshold ms_g, vertex set $\{U', V'\}$, where $U' \subseteq U, V' \subseteq V$, forms a CGQB subgraph if and only if*

- *$|\{G|G \in D \wedge G(U', V')\ is\ a\ \epsilon\text{-QB}\}| \geq ms_g$, where $G(U', V')$ is the subgraph of G induced on vertex set $\{U', V'\}$.*

We use $occ(U', V')$ to denote $\{G|G \in D \wedge G(U', V')\ is\ a\ \epsilon\text{-QB}\}$. Given $\epsilon = 1$ and ms_g=3, Figure 2 shows a CGQB subgraph with $U' = \{u_0, u_1, u_2\}$ and $V' = \{v_0, v_1, v_2\}$, and $occ(U', V') = \{G_1, G_2, G_3\}$.

Based on Lemma 1, if $\{U', V'\}$ forms a ϵ-QB in graph G, then for any $\{U'', V''\}$ such that $U'' \subseteq U'$ and $V'' \subseteq V'$, $\{U'', V''\}$ also forms a ϵ-QB in G. Therefore, if $\{U', V'\}$ is a CGQB, then $\{U'', V''\}$ must also be a CGQB, and we have $occ(U'', V'') \supseteq occ(U', V')$. In the case of $occ(U'', V'') = occ(U', V')$, $\{U'', V''\}$ becomes uninteresting because it provides no more information than $\{U', V'\}$. In this paper, we are interested in mining only maximal CGQB subgraphs.

Definition 4. (Maximal CGQB subgraph) *Let $D = \{G_1, \ldots, G_n\}$ be a set of bipartite graphs and $\{U', V'\}$ be a CGQB subgraph in D. If there does not exist another CGQB subgraph $\{U'', V''\}$ in D such that $\{U', V'\} \subset \{U'', V''\}$ and $occ(U', V') = occ(U'', V'')$, then $\{U', V'\}$ is called a maximal CGQB subgraph in D.*

CGQB subgraphs containing only a few vertices are not very interesting. A trivial case is quasi-biclique subgraphs containing only one stock and one financial ratio. To filter such small CGQB subgraphs, we use two thresholds ms_u and ms_v to constrain the minimum number of vertices in U' and minimum number of vertices in V' respectively.

Problem statement Given a set of graphs $D = \{G_1, \ldots, G_n\}$, the CGQB subgraphs mining problem is to mine the complete set of CGQB subgraphs from D, such that each CGQB subgraph is maximal and satisfies the quasi threshold ϵ, the minimum size thresholds ms_u, ms_v and minimum support threshold ms_g.

In the rest of the paper, we assume CGQB subgraphs to be maximal, unless it is explicitly stated not to be.

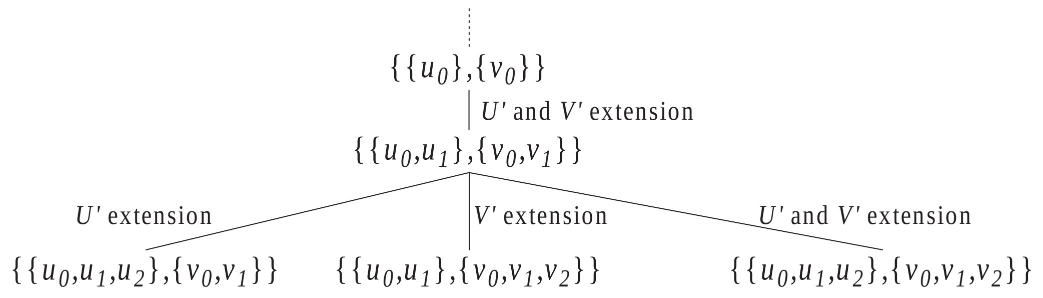


Figure 5: Partial set enumeration tree of the set of bipartite graphs in Figure 2. Each node of the tree is represented by a subset of U (denoted as $U' \subseteq U$) and a subset of V (denoted as $V' \subseteq V$).

4.2. $CGQBminer$: Mining Cross-Graph Quasi-Biclique (CGQB) Subgraphs

A naïve method to mine CGQB subgraphs can be based on the mining structure of Algorithm TRICLUSTER [50], which consists of two mining stages. In the first stage, we mine quasi-biclique (QB) subgraphs from each bipartite graph $G \in D$ using some existing QB mining algorithms [1, 4, 31, 38, 46], and then in the second stage, we generate CGQB subgraphs from these QB subgraphs. However, this approach is computationally slow as it is possible that a large number of QB subgraphs are generated, and many of them cannot form CGQB subgraphs. Therefore, there is a need to develop an efficient algorithm to mine CGQB subgraphs.

In this section, we present our CGQB subgraph mining algorithm, $CGQBminer$, which takes the set of bipartite graphs $D = \{G_1, \ldots, G_n\}$ with $\mathcal{V}(G_1) = \ldots = \mathcal{V}(G_n) = \{U, V\}$ as input.

4.2.1. Tri-extensions: Three dimensional search space traversal strategy

We project the three dimensional search space of the data into a modified *set enumeration tree* [37] (we shorten it to *tree* for conciseness) that represents the power sets of U and V. This modified tree is able to accommodate the two power sets, where each *node* of the tree, denoted as $\{U', V'\}$, represents a subset of U ($U' \subseteq U$) and a subset of V ($V' \subseteq V$). At each node $\{U', V'\}$, we also maintain $occ(U', V')$ which contains the set of graphs in which $\{U', V'\}$ forms a ϵ-QB.

We sort the vertices in U and V according to some order, so that only vertices that are after all the existing vertices in $U'(V')$ can be used to extend $U'(V')$. For example, vertex u_2 can be used to extend $U' = \{u_1\}$, but it cannot be used to extend $U' = \{u_3\}$. We use $last(U')$ to denote the last vertex in U' and use $last(V')$ to denote the last vertex in V'. Figure 5 shows a partial tree of the set of bipartite graphs in Figure 2.

We propose a strategy *tri-extensions*, to recursively traverse down the tree. Let us assume that $CGQBminer$ is at node $\{U', V'\}$ of the tree.

- U**-extension.** U' is extended with a vertex $u \in extCand(U')$, where $extCand(U') = \{u|u \in U \wedge u > last(U')\}$ is the set of extension candidates of U'. For a node reached by U-extension, only U-extensions are allowed for its subsequent extensions. Thus, U-extension is a series of traversal to nodes $\{U'', V''\}$, where $U'' \supset U'$.

- V**-extension.** V-extension is similar to U-extension, except that V' is extended with $v \in extCand(V')$, where $extCand(V') = \{v|v \in V \wedge v > last(V')\}$ is the set of extension candidates of V'. For a node reached by V-extension, only V-extensions are allowed for its subsequent extensions. Thus, V-extension is a series of traversal to nodes $\{U', V''\}$, where $V'' \supset V'$.

- (U, V)**-extension.** U' is extended with $u \in extCand(U')$ and V' is extended with $v \in extCand(V')$. For a node that is reached by (U, V)-extension, the three types of extension are still applicable in subsequent extensions.

Lemma 2. *Through tri-extensions, $CGQBminer$ recursively traverses all nodes of the tree that contain CGQB subgraphs, and every node is traversed only once.*

PROOF. We prove that every node $\{U', V'\}$ can be reached by $CGQBminer$ and every node is traversed at most once. Let $U' = \{u_1, u_2, \cdots, u_i\}$ and $V' = \{v_1, v_2, \cdots, v_j\}$. There are three cases:
(1) i=j. In this case, it is obvious that $\{U', V'\}$ can be reached by i (U, V)-extensions from the empty set. We need to prove that it is the only way to reach $\{U', V'\}$. Suppose there is another path to reach $\{U', V'\}$ through U-extension at node $\{U'', V''\}$, where $U'' = \{u_1, u_2, \cdots, u_k\}$, $V'' = \{v_1, v_2, \cdots, v_k\}$ and $k < i$. Since $\{U'', V''\}$ is extended by a U-extension, all of its subsequent extensions must be U-extensions, and $V'' \subset V'$ will not be extended further. Therefore, $\{U', V'\}$ can never be reached if there is a U-extension on the path. Similarly, we can prove that $\{U', V'\}$ cannot be reached if there is a V-extension on the path.
(2) $i > j$. In this case, $\{U', V'\}$ must be reached by j (U, V)-extensions followed by $(i - j)$ U-extensions. The proof is similar to (1).
(3) $i < j$. In this case, $\{U', V'\}$ must be reached by i (U, V)-extensions followed by $(j - i)$ V-extensions. The proof is similar to (1).

Algorithm 1 $CGQBminer(U', V', extCand(U'), extCand(V'))$

```
Require: $U'$ and $V'$ are subsets of $U$ and $V$ respectively
    $extCand(U')$ is the set of candidates to extend $U'$
    $extCand(V')$ is the set of candidates to extend $V'$
Ensure:
1: if $|occ(U', V')| < ms_g$ then return;
2: if $|U'| \geq ms_u \wedge |V'| \geq ms_v \wedge |occ(U', V')| \geq ms_g$ then
3:     Output CGQB $\{U', V'\}$ and $occ(U', V')$
4: end if
5: if $|U' \cup extCand(U')| < ms_u$ or $|V' \cup extCand(V')| < ms_v$ then return;

6: /* $U-$extension */
7: if $|V'| \geq ms_v$ then
8:     for all $u \in extCand(U')$ do
9:         $CGQBminer(U' \cup \{u\}, V', extCand(U' \cup \{u\}), \{\})$;
10:    end for
11: end if
12: /* $V-$extension */
13: if $|U'| \geq ms_u$ then
14:    for all $v \in extCand(V')$ do
15:        $CGQBminer(U', V' \cup \{v\}, \{\}, extCand(V' \cup \{v\}))$;
16:    end for
17: end if
18: /* $(U, V)-$extension */
19: for all $u \in extCand(U')$ do
20:    for all $v \in extCand(V')$ do
21:        $CGQBminer(U' \cup \{u\}, V' \cup \{v\}, extCand(U' \cup \{u\}), extCand(V' \cup \{v\}))$;
22:    end for
23: end for
```

The pseudo code of $CGQBminer$ is shown in Algorithm 1. When $CGQBminer$ is first called on the set of bipartite graphs D, U' is set to $\{\}$ and V' is set to $\{\}$. For U-extensions, vertex set V' will not be changed in the future. Therefore, we perform U-extension on $\{U', V'\}$ only if $|V'| \geq ms_v$. Similarly, we perform V-extension on $\{U', V'\}$ only if $|U'| \geq ms_u$.

4.2.2. Pruning techniques for the search space traversal

We now describe the set of pruning techniques which helps in minimizing the traversal of the search space. Before the set of bipartite graphs D is used as the input for $CGQBminer$, we pre-process it to reduce its size.

Pruning technique 1. (Pre-processing on the set of bipartite graphs D) *In each $G \in D$, vertices $u \in U$ whose neighborhood size $|N_G(u)|$ is less than $ms_v - \epsilon$*

are pruned, as they cannot form CQGB subgraphs. This pruning is also done on vertices in V. For each $G \in D$, if $|U| < ms_u - \epsilon$ or $|V| < ms_v - \epsilon$, then G is pruned, as no QB subgraph can be found in G.

Let us assume that D is pre-processed and $CGQBminer$ has traversed to node $\{U', V'\}$ in its tree. $CGQBminer$ proceeds to obtain the candidates to extend U' and V', which are pruned by the following technique.

Pruning technique 2. (Pruning extension candidates based on the minimum support constraint) *For a vertex $u \in U$ and $u > last(U')$, if $|\{G||N_G^{V'}(u)| \geq |V'| - \epsilon \wedge G \in D\}| < ms_g$, then u cannot form a CGQB subgraph with V', and it should be excluded from $extCand(U')$. Similarly, for a vertex $v \in V$ and $v > last(V')$, if $|\{G||N_G^{U'}(v)| \geq |U'| - \epsilon \wedge G \in D\}| < ms_g$, then v cannot from a CGQB subgraph with U', and it should be excluded from $extCand(V')$.*

Vertex sets $\{u\}$ and V' form a QB if $|N_G^{V'}(u)| \geq |V'| - \epsilon$. However, if there are less than ms_g number of graphs in which $\{u\}$ and V' form a QB, then no CGQB subgraph can be formed by u and V'. Thus, u is pruned. By adding pruning technique 2, the set of extension candidates of U' becomes $extCand(U') = \{u|u \in U \wedge u > last(U') \wedge |\{G||N_G^{V'}(u)| \geq |V'| - \epsilon \wedge G \in D\}| \geq ms_g\}$.

Example 1. We set $ms_u = ms_v = 2$, $ms_g = 3$ and $\epsilon = 1$. Let us assume that we are traversing the tree of the set of bipartite graphs of Figure 2 and is at node $\{\{u_0, u_1\}, \{v_0, v_1\}\}$. We have $last(\{u_0, u_1\}) = u_1$. For $u_2 \in U$, $u_2 > last(\{u_0, u_1\})$ and $|\{G||N_G^{\{v_0,v_1\}}(u_2)| \geq |\{v_0, v_1\}| - \epsilon \wedge G \in D| = |\{G_0, G_1, G_2\}| \geq ms_g$, so u_2 is an extension candidate of U'. For $u_3 \in U$, $u_3 > last(\{u_0, u_1\})$ and $|\{G||N_G^{\{v_0,v_1\}}(u_3)| \geq |\{v_0, v_1\}| - \epsilon \wedge G \in D| = |\{\}| < ms_g$, so u_3 is not an extension candidate of U'. Thus, $extCand(\{u_0, u_1\}) = \{u_2\}$.

Similarly the set of extension candidates of V' becomes $extCand(V') = \{v|v \in V \wedge v > last(V') \wedge |\{G||N_G^{U'}(v)| \geq |U'| - \epsilon \wedge G \in D\}| \geq ms_g\}$ after applying pruning technique 2. The sets of candidates for extension are then used to check if the current node $\{U', V'\}$ should be pruned.

Pruning technique 3. (Pruning nodes based on the minimum size constraint) *If $|U' \cup extCand(U')| < ms_u$ or $|V' \cup extCand(V')| < ms_v$, then current node $\{U', V'\}$ is pruned (Algorithm 1 line 5), as current node and subsequent nodes traversed from it do not contain CGQB subgraphs that fulfill the minimum size thresholds ms_u, ms_v. Since we reduce the sizes of $extCand(U')$ and $extCand(V')$ using pruning technique 2, more nodes can be pruned using this pruning technique.*

If current node $\{U', V'\}$ is not pruned, $CGQBminer$ proceeds to prune $occ(U', V')$, which contains the set of graphs in which $\{U', V'\}$ forms a ϵ-QB.

Pruning technique 4. (Pruning $occ(U', V')$ based on the quasi threshold) *Given a graph $G \in D$, if $\exists u \in U', |N_G^{V'}(u)| < |V'| - \epsilon$ or $\exists v \in V', |N_G^{U'}(v)| < |U'| - \epsilon$, then U' and V' do not form a ϵ-QB in G. Thus G is pruned.*

After applying the above pruning technique, $occ(U', V') = \{G | G \in D \wedge \forall u \in U', |N_G^{V'}(u)| \geq |V'| - \epsilon \wedge \forall v \in V', |N_G^{U'}(v)| \geq |U'| - \epsilon\}$.

Example 2. Continuing from Example 1, assume that $CGQBminer$ has traversed to node $\{\{u_0, u_1, u_2\}, \{v_0, v_1, v_2\}\}$. We have $occ(U', V') = \{G | G \in D \wedge \forall u \in \{u_0, u_1, u_2\}, |N_G^{\{v_0, v_1, v_2\}}(u)| \geq |\{v_0, v_1, v_2\}| - \epsilon \wedge \forall v \in \{v_0, v_1, v_2\}, |N_G^{\{u_0, u_1, u_2\}}(v)| \geq |\{u_0, u_1, u_2\}| - \epsilon\} = \{G_0, G_1, G_2\}$.

The next pruning technique prunes the current node $\{U', V'\}$, based on the size of $occ(U', V')$.

Pruning technique 5. (Pruning nodes based on the minimum support constraint) *If the number of graphs in $occ(U', V')$ for node $\{U', V'\}$ is less than ms_g, then this node does not contain CGQB subgraphs that fulfill ms_g, and is pruned (Algorithm 1 line 1). Since we remove graphs in $occ(U', V')$ using pruning technique 4, more nodes can be pruned in this pruning technique.*

Based on Lemma 1, if $\{U', V'\}$ does not satisfy the minimum support constraint, then none of its supersets can satisfy the minimum support constraint. Therefore, there is no need to extend $\{U', V'\}$ further, and the whole sub-tree rooted at node $\{U', V'\}$ can be pruned.

As we are interested in only maximal CGQB subgraphs, we need to check whether every generated CGQB is maximal. One approach is to mine all CGQB subgraphs, store them and then check the results and return only those that are maximal. However, the stored results can be very large, which not only consumes lots of memory, but also slows down the checking operation. The better approach is to check directly if each CGQB is maximal, which is done by using two types of maximal checks on $\{U', V'\}$: (1) adding vertex $u \in U - U'$ to U'. If $\{U' \cup \{u\}, V'\}$ forms a QB in every graph in $occ(U', V')$, then $\{U', V'\}$ is not maximal. (2) adding vertex $v \in V - V'$ to V'. If $\{U', V' \cup \{v\}\}$ forms a QB in every graph in $occ(U', V')$, then $\{U', V'\}$ is not maximal.

4.2.3. Correctness and Time Complexity of $CGQBminer$

In summary, all nodes of the tree that contain CGQB subgraphs are traversed (shown in Lemma 2), and nodes that do not contain CGQB subgraphs are pruned by the pruning techniques described above.

Theorem 1. (Correctness of $CGQBminer$) *$CGQBminer$ generates the complete set of maximal CGQB subgraphs from a set of bipartite graphs D, with respect to the minimum size thresholds ms_u, ms_v, minimum support threshold ms_g and quasi threshold ϵ.*

The correctness and completeness of $CGQBminer$ is guaranteed by Lemma 1 and 2.

On the complexity analysis of $CGQBminer$, it is a well known fact that the worst case time complexity of algorithms that use set enumeration tree is exponential to the size of the input [42]. Thus, the worst case time complexity of $CGQBminer$ is in exponential. However in Section 6.3, we show that $CGQBminer$ generally performs well due to its efficient search space traversal strategy and effective pruning techniques.

4.2.4. Handling non-bipartite graphs

Besides mining from bipartite graphs, $CGQBminer$ can also mine CGQB subgraphs from a set of non-bipartite graphs. Assume $\mathcal{D} = \{H_1, \ldots, H_n\}$ is a set of non-bipartite graphs with $\mathcal{V}(H_1) = \ldots = \mathcal{V}(H_n) = U$. We first duplicate U and denote the duplicate as V. We then convert $\mathcal{D}$ into a set of bipartite graphs $D = \{G_1, \ldots, G_n\}$, with $V(G_i) = \{U, V\}$ and $E(G_i) = \{\{u, v\} | \{u, v\} \in E(H_i) \wedge u \in U \wedge v \in V\}, \forall i \in \{1, \ldots, n\}$. D will be used as the input dataset for $CGQBminer$. As $U = V$, Algorithm 1 is modified to prevent U' and V' containing the same vertex. Vertices in V' and U' are also prevented to be in $extCand(U')$ and $extCand(V')$ respectively.

5. Evaluating Price Movements of Stocks

In the data analysis phase, the price movements of the stock groups obtained by CGQB subgraphs are evaluated to check if their price movements are statistically similar. The following describes the methodology to evaluate the price movements.

1. CGQB subgraphs are mined from the set of bipartite graphs representing the stock data from an industry (refer to Section 1.1), and from each of them, we obtain its *stock group*, which we formally define as follows.

Definition 5 (Stock group). *Let $\{U', V'\}$ be a CGQB subgraph mined from a set of bipartite graphs representing a stock data. Let U' represents a set of stocks and V' represents a set of financial ratios' values which this set of stocks are similar in. U' is known as the stock group of this CGQB subgraph.*

For example, let $\{\{u_0, u_1, u_2\}, \{v_0, v_1, v_2\}\}$ be a CGQB subgraph from Figure 2, and $\{u_0, u_1, u_2\}$ is the stock group of this CGQB subgraph. In addition, we denote $\mathbb{S}$ as the set of stock groups obtained from a set of bipartite graphs.

2. **Calculating standard deviation of stock groups** In a stock group, we calculate the standard deviation of its stocks' price movements. Low standard deviation implies high similarity in price movements for these stocks. From each year y_i which the stocks of the stock group have similar financial ratio values, we calculate the standard deviation of these stocks for year y_{i+1}. This calculation will show how stocks having similar financial ratio values in current year will affect their next year price movements. The standard deviation is calculated per year basis since the financial ratios of stocks change every year.
3. **Conducting statistical hypothesis tests** Statistical tests are conducted to test if the standard deviations of stock groups are statistically significantly lower than those of randomly picked stocks from the industry, and the industry index. For the first comparison, we are comparing the price movements between stocks with similar financial ratio values and stocks without similar financial ratio values. For the second comparison, we are comparing the price movements of stocks with similar ratio values and the average price movements of all stocks. If the standard deviations of the stock groups are statistically significantly lower, then we cannot reject Hypothesis 1, and the claim that financial ratio values are indicators of future stock prices is valid.

5.1. Calculating standard deviation of stock groups

For a stock group $S = \{s_1, \ldots, s_j\}$ which are similar in a group of financial ratios across years $y_1, \ldots, y_m$, we track the price movements of S across its succeeding years, that is, $Y = \{y_2, \ldots, y_{m+1}\}$. Let $p(s, y)$ denote the closing price of the stock s for year y, and the price movement for year y is denoted as

$$d(s, y) = \frac{p(s, y+1) - p(s, y)}{p(s, y)}$$

We then denote the standard deviation of S for year y as

$$\sigma(S, y) = \sqrt{\frac{1}{|S|}\sum_{s\in S}(d(s, y) - \mu(s, y))^2}$$

where $\mu(S, y) = \frac{1}{|S|}\sum_{s\in S} d(s, y)$ is the mean price movements of S for year y.

We average $\sigma(S, y)$ over the set of years Y to calculate the standard deviation of S across the set of years,

$$\sigma(S) = \frac{1}{|Y|}\sum_{y\in Y}\sigma(S, y)$$

For the set of stock groups $\mathbb{S} = \{S_1, \ldots, S_k\}$, we calculate $\sigma(S_1), \ldots, \sigma(S_k)$, and obtain their average

$$\sigma(\mathbb{S}) = \frac{1}{|\mathbb{S}|}\sum_{S\in\mathbb{S}}\sigma(S)$$

We denote $\sigma(\mathbb{S})$ as *the standard deviation of the set of stock groups* $\mathbb{S}$. Low $\sigma(\mathbb{S})$ means on average, each stock group $S \in \mathbb{S}$ has highly similar price movements.

5.2. Conducting statistical hypothesis tests

We use t-test to conduct the following hypotheses as the population standard deviation of the stock data is unknown, and we assume that the standard deviation of the price movement of stocks follows a standard normal distribution.

Paired t-test is conducted to test the following hypothesis

Hypothesis 2. *The standard deviation of the set of stock groups is lower than the standard deviation of the randomly picked stocks.*

This is to compare if the standard deviation of the set of stock groups is lower than the standard deviation of stocks that are grouped not based on similarity in their financial ratios. Paired t-test is used since both the stock groups and randomly picked stocks are drawn from the same data. To have an unbiased paired t-test, each stock group is matched with a group of randomly picked stocks, and they are of the same size and industry.

Next, one sample t-test is conducted to test the following hypothesis, with the assumption that stocks in the set of stock groups are from the same industry.

Hypothesis 3. *The standard deviation of the set of stock groups is lower than the standard deviation of the industry index.*

This is to compare if the standard deviation of the stock groups is lower than the standard deviation of all stocks in the same industry. Stocks in same industries tend to have similar values in their financial ratios [13]. In a separate work, King [21] shows that price movements of stocks in the same industry are similar. Based on these two works, one may suspect that similar financial ratios' values are related to similar prices. If we can show that stocks in stock groups have more similar price movements than stocks in the same industry, then we are compounding the fact that stocks with high similarity in their financial ratios have similar price movements, as the stocks in stock groups have higher similarity in financial ratios than stocks in the same industry.

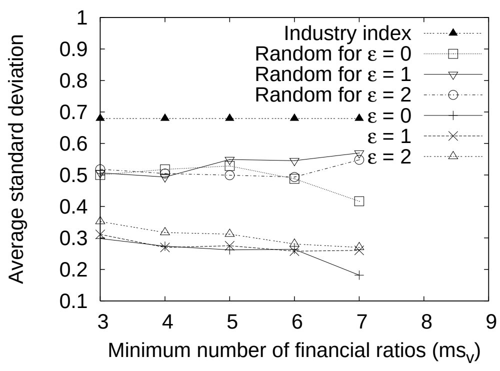
If Hypothesis 2 and 3 are not rejected by the tests, then we cannot reject Hypothesis 1 and we will conclude that it is valid.

6. Experimental Results

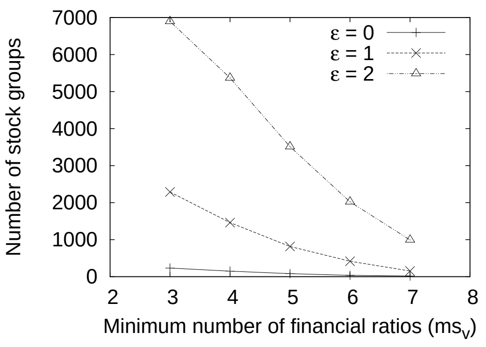
Our experiments were performed on Windows XP environment, using Intel 3.4Ghz and 2GB RAM. All algorithms were coded in C++. We present the statistical test results on the price movements of stock groups obtained by CGQB subgraphs, and then we show how stock groups mined by them can be used for stock investments. Lastly, we present the performance study of $CGQBminer$.

6.1. Data Preparation

We downloaded financial figures of stocks from from Compustat [40], for year 2000 to 2006. These financial figures of stocks are from 9 industries, namely Apparel $\&$ Other Textile Products, Chemicals $\&$ Allied Products, Food, Insurance $\&$ Real Estate, Oil $\&$ Gas Extraction, Paper $\&$ Allied Products, Printing $\&$ Publishing, Wholesale Trade and Metal Mining, and each industry consists of 28 to 36 stocks. We converted these financial figures to 32 major financial ratios, based on the ratios' formula from Investopedia [18]. The formulae of the popular financial ratios are presented in Appendix A for the interested reader. In total, we have 32 financial ratios of 234 stocks for 7 years, with 26% of the data being missing values. As the financial ratios are in continuous values, we discretized them into partitions using the agglomerative hierarchical clustering with $CDbw$ index [14, 45]. We then converted the stock data into sets of bipartite graphs, as described in Section 1.1. We obtained 9 sets of bipartite graphs $D_1, \ldots, D_9$, with each D representing the stock data of an industry.



(a)



(b)

Figure 6: (a) Average standard deviation of price movements of different stock groups from 9 industries. (b) Number of stock groups mined from 9 industries.

6.2. Study on the stock price movements

We mined CGQB subgraphs from each D to obtain its set of stock groups. Recall that a stock group has a group of similar financial ratio values across several years. Setting the appropriate threshold to mine CGQB subgraphs can be a tricky task. We attempted to study the effect of varying thresholds, but at the same time we want to obtain large CGQB subgraphs that do not contain one stock or financial ratio. We set a minimum number of 7 years ($ms_g = 7$), as we required the stock groups to exist across all years of the data. We fixed a minimum number of 5 stocks ($ms_u = 5$) to prevent the stock groups to be trivially small. We varied the minimum number of financial ratios from 3 to 7 (ms_v) to study the effect of stock groups having different number of similar financial ratio values. We set the upper bound to 7 as we found that very few or no CGQB subgraphs were mined if $ms_v > 7$, and we set the lower bound to 3 as we found that too many small CGQB subgraphs were mined if $ms_v < 3$. We kept the quasi threshold ϵ small and varied it from 0 to 2 as we want the mined stock groups to be highly similar in their financial ratios. Thus, we have 135 sets of stock groups (9 sets of data $\times$ 5 varying ms_v $\times$ 3 varying ϵ). Note that we are mining CGB subgraphs at $\epsilon = 0$, and they can be mined using the frequent closed cube [19], tricluster [50] or closed 3-sets [6] models. We adopted the frequent closed cube model and used the CubeMiner algorithm [19] to mine cross-graph biclique subgraphs. The CubeMiner algorithm was kindly obtained from the authors.

6.2.1. Standard deviation of the sets of stock groups

We calculated the standard deviation of each set of stock groups $\mathbb{S}$, $\sigma(\mathbb{S})$, and Figure 6(a) shows the average standard deviations of the sets of stock groups from the 9 industries. The sets of stock groups are obtained by varying the minimum number of financial ratios from 3 to 7 (ms_v). The average standard deviations of stock groups obtained by different quasi threshold ϵ are shown in the lines labeled with 'ϵ'. The lines labeled with 'Random for ϵ' are the average standard deviations of the groups of randomly picked stocks matched to each set of stock groups. The line labeled with 'Industry index' is the average standard deviations of all stocks in the 9 industries.

From Figure 6(a), we can see that the average standard deviations of all 135 sets of stock groups are distinctly lower than those of groups of randomly picked stocks and the average industry index. Besides this, we can also see that the average standard deviations decrease as ms_v increases. This means that stock groups which have a larger number of similar financial ratio values across several years have higher similarity in their price movements.

6.2.2. Number of stock groups mined

Figure 6(b) presents the number of stock groups obtained from CGQB subgraphs mined from $D_1, \ldots, D_9$. The minimum number of financial ratios ms_v is varied from 3 to 7. The lines labeled with different ϵ indicates the number of stock groups obtained by different quasi threshold ϵ settings. Figure 6(b) shows that the number of stock groups mined depends on two factors: the minimum number of financial ratios ms_v and the quasi threshold ϵ. Either decreasing ms_v or increasing ϵ results in increasing number of stock groups. Therefore, a prudent investor will select the appropriate ms_v and ϵ to obtain his/her desired number of stock groups.

Figure 6(b) also shows the advantages of having quasi threshold ϵ. At $\epsilon = 0$, very few cross-graph biclique subgraphs were mined to obtain the stock groups, even at low ms_v. This clearly shows that the high percentage (26%) of missing values in the stock data hinders the discovery of potential stock groups, and the strict connectivities requirement of cross-graph biclique subgraphs does not help in discovering these stock groups. However, with the introduction of quasi threshold ϵ, the connectivities in CGQB subgraphs can be relaxed and more stock groups can be discovered in the stock data, as shown in Figure 6(b).

6.2.3. Statistical hypothesis tests

We conducted the paired t-test for Hypothesis 2 and one sample t-test for Hypothesis 3 (refer to Section 5.2) on each of the 135 sets of stock groups. We

Table 1: Percentage of test outcomes whose p-value < 0.05, for 135 sets of stock groups.

	Paired t-test		One sample t-test	
ϵ	Available test outcomes	All test outcomes	Available test outcomes	All test outcomes
0	87	44.4	91.3	46.7
1	94.3	73.3	100	77.8
2	90	80	100	88.9

calculated the p-value for each test and categorized the outcome of a test into three categories: p-value < 0.05, p-value ≥ 0.05 and N.A.. p-value < 0.05 means that the average standard deviations of the set of stock groups are statistically significantly lower than the standard deviation of the groups of randomly picked stocks or the industry index, depending on which test is conducted. N.A. means that the test outcome is not available as the size of the set of stock groups is too small to be considered for the statistical test.

Table 1 presents the percentage of test outcomes whose p-value < 0.05. The test outcomes of sets of stock groups obtained with different ϵ settings are shown in each row. The column 'Available test outcomes' presents the percentage of test outcomes whose p-value < 0.05, with respect to all tests which have outcomes. The column 'All test outcomes' presents the percentage of test outcomes whose p-value < 0.05, with respect to all tests outcomes (which includes N.A.).

Almost all the paired t-tests and one sample t-tests outcomes have p-value < 0.05. In particular, when $\epsilon > 0$ on 'Available test outcomes', more than 90% of the test outcomes have p-value < 0.05. This shows that it is very rare that Hypothesis 2 and 3 were rejected. When $\epsilon = 0$, the percentage of the t-tests outcomes having p-value < 0.05 is lower than those of $\epsilon > 0$. Since less number of stock groups is obtained when $\epsilon = 0$, the statistical tests have less confidence of not rejecting Hypothesis 2 and 3.

There is also a low percentage of p-value < 0.05 on 'All test outcomes' when $\epsilon = 0$. Again, this is due to few or no cross-graph biclique subgraphs being mined to obtain sets of stock groups, resulting in a large percentage of test outcomes being N.A..

The results in Table 1 convey two important information. First, we cannot reject Hypothesis 1 due to the overwhelming percentage of tests failing to reject Hypothesis 2 and 3. This shows that stock groups that have groups of similar financial ratio values across several years have similar price movements. Second, the quasi relation of CGQB subgraphs is crucial in leading to discovery of stock groups that have more number of similar financial ratio values across several years, and these stock groups are statistically proven to have higher similarity in their price movements, as shown in the rows of Table 1 when $\epsilon = 1, 2$.

6.2.4. Utility of stock groups

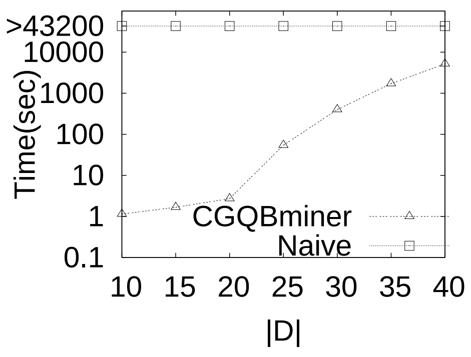
After proving that stock groups with similar financial ratio values have similar price movements, the natural question to ask is how do we utilize this valuable knowledge? Let us assume that we have a simple investment strategy which we denote as strategy A. Strategy A assumes that if the average return of a stock from year 2000 to 2005 is positive, then it predicts that the return of the stock for 2006 is positive too.

Let us also assume that we have strategy B, which is using CGQB subgraphs with strategy A. In strategy B, the stocks are clustered as stock groups described in Section 6.2, and if the average return of a stock group from year 2000 to 20005 is positive, then it predicts that the return of this stock group is positive for year 2006. The purpose of this strategy is to study how using stock groups can improve an investment strategy.

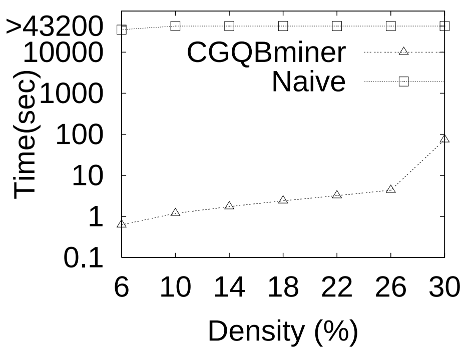
We obtained 346 stocks for strategy A and 1,328 stock groups for strategy B. For strategy A, 47.4% of the stocks have positive return for year 2006 while 83.8% of the stock groups have positive return for year 2006. We can see that the simple approach of strategy A to predict future returns is closed to random guessing. However, by incorporating the same strategy with CGQB subgraphs, the percentage of positive returns increases considerably. This shows that the usage of CGQB subgraphs can provide a margin of 'safety' for an investment strategy.

6.3. Study on the performance of $CGQBminer$

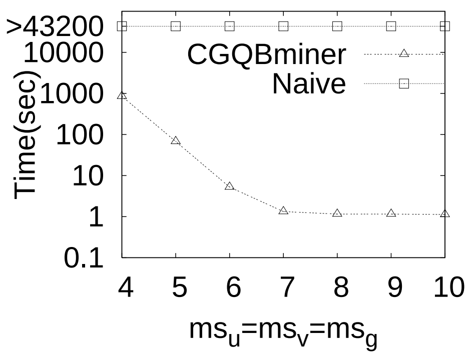
As there are no existing algorithms to mine CGQB subgraphs, we implemented the naïve approach that we described in Section 4.2 for performance comparison. This naïve approach consists of two mining stages, which follows the mining structure of algorithm TRICLUSTER [50]. Given a set of graphs $D = \{G_1, \ldots, G_n\}$, this naïve algorithm first mines quasi-biclique subgraphs from each $G \in D$. In the second stage, CGQB subgraphs are generated from them. For the first stage, we used the $CompleteQB$ algorithm [38] to mine quasi-biclique subgraphs.



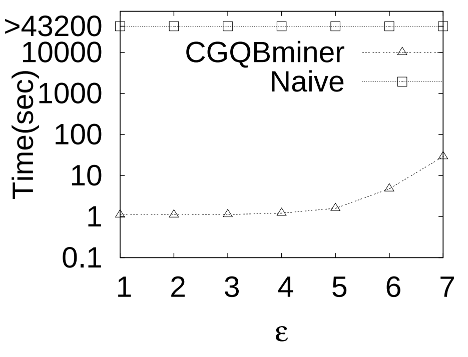
(a) Varying $|G|$



(b) Varying density



(c) Varying ms



(d) Varying ϵ

Figure 7: Performance study of $CGQBminer$ across different scenarios

We evaluated how the size and density of the datasets, and the parameter settings of $CGQBminer$ affect the running time of $CGQBminer$. For this performance study, we generated a synthetic set of bipartite graphs $D = \{G_1, \ldots, G_n\}$, using IBM Quest Market-Basket Synthetic Data Generator [17]. Each bipartite graph contains 2000 vertices, with 1000 in each of its vertex set.

6.3.1. Effect of the size and density of the dataset

We studied how the size and density of the dataset affect the running time of $CGQBminer$. To evaluate the effect of the size of the dataset, we varied the number of bipartite graphs in D from 10 to 40, and set the density of D to 10% [4].

Hence, the number of edges in D varies from 1 to 4 million. For the parameters, we set $ms_u = ms_v = ms_g = 10$, $\epsilon = 1$. Note that we fixed the parameters so that we can have a fair evaluation of how the number of bipartite graphs in D affects the running time of $CGQBminer$. Figure 7(a) presents the running time of $CGQBminer$ and the naïve algorithm across the different sizes of D. The naïve algorithm could not complete mining after 12 hours for all different sizes of D, while $CGQBminer$ completed all mining in less than 1.5 hours. The naïve algorithm is slow as its two mining stages potentially can generate large number of quasi-biclique subgraphs that are not cross-graph quasi-bicliques. On the contrary, $CGQBminer$ is fast as the *tri-extensions* traversal strategy is efficient in traversing the search space of D.

To evaluate the effect of the density of the dataset, the number of bipartite graphs in D is fixed at 10 and we varied the density of D from 6% to 30%. So, the number of edges in D varies from 0.6 to 3 million. Figure 7(b) presents the results. The naïve algorithm completed mining in about 9 hours when the density is 6%, but could not complete mining for the rest after more than 12 hours. On the other hand, $CGQBminer$ completed all mining in less than 100 seconds.

6.3.2. Effect of the parameter settings

We studied how the minimum sizes ms_u, ms_v, minimum support ms_g and quasi threshold ϵ affect the running time of $CGQBminer$. We used D containing 10 bipartite graphs with density 10%. To evaluate the effect of ms_u, ms_v, ms_g, we varied them from 4 to 10, and set $\epsilon = 1$. Figure 7(c) presents the running time of $CGQBminer$ and the naïve algorithm. The naïve algorithm could not complete mining after 12 hours for all settings, but $CGQBminer$ completed all mining in less than 1000 seconds. This shows that $CGQBminer$ is able to exploit ms_u, ms_v, ms_g to efficiently prune the search space, as the running time is constantly low across the thresholds. There is a noticeable increase in the running time when the threshold dropped from 5 to 4, which is due to a large increase in the number of CGQB subgraphs mined.

To evaluate the effect of ϵ, we set ms_u, ms_v, ms_g to 10 and varied ϵ from 1 to 7. Figure 7(d) presents the results. The naïve algorithm could not complete

[4]Density of D is calculated as $\frac{\prod_{G \in D} |E(G)|}{\prod_{\mathcal{V}(G)=\{U,V\}, G \in D} |U||V|}$, which is the number of edges of D divided by the total number of possible edges of D [7]. Note that the density of a graph is different from the density of a cluster explained in Footnote 3.

Table 2: Key findings of the study on the stock price movements in Section 6.2.

Experiment	Key Findings
(1) Standard deviation of stock groups	- Standard deviation of stock groups are distinctly lower than those of randomly picked stocks and industrial indexes - Stock groups having larger number of similar financial ratios have lower standard deviation
(2) Number of stock groups mined	- Decreasing ms_v or increasing ϵ results in increasing stock groups
(3) Statistical hypothesis test	- Hypothesis 1 is not rejected as Hypothesis 2 and 3 are not rejected - Stock groups with quasi relation are statistically proven to have higher similarity in their price movements than those without quasi relation
(4) Utility of stock group	The probability of making profit increases by 36.4% if an investment strategy utilizes CGQB subgraphs

Table 3: Key findings of the study on the performance of $CGQBminer$ in Section 6.3.

Experiment	Key Findings
(1) Effect of the size and density of the dataset	$CGQBminer$ is much scalable than the naïve algorithm for different sizes and densities of D
(2) Effect of the parameter settings	$CGQBminer$ is much efficient than the naïve algorithm across different thresholds settings.

mining after 12 hours across the varying ϵ while $CGQBminer$ completed all mining within 30 seconds. The running time of $CGQBminer$ decreases as ϵ decreases, which evinces the effectiveness of $CGQBminer$ in utilizing ϵ to prune the dataset.

7. Results Discussions

7.1. Key Findings from Experimentation

We summarize and present the key findings from our experimental results of Section 6. Table 2 presents the summary of the study on the stock price movement in Section 6.2 and Table 3 presents the summary of the study on the performance of $CGQBminer$ in Section 6.3.

7.2. Limitations of Proposed Approach

Although the results are promising, there are certain limitations of our proposed approach, which we discuss as follows:

1. There are no methods to rank the CGQB subgraphs in terms of their utility or usefulness. Ranking CGQB subgraphs will be useful in cases where a large number of them are mined and the user would like to select a subset of them.
2. The experiment results show that $CGQBminer$ is suitable for medium size dataset and may not be scalable for large dataset, which contains thousands of bipartite graphs or millions of vertices.
3. The quasi threshold of CGQB subgraphs is designed to be absolute based to allow efficient implementation of the pruning techniques of $CGQBminer$. However, a percentage based quasi threshold may be a more natural way of controlling the strictness of the connectivities in CGQB subgraphs.
4. Prudence must be exercised in setting the appropriate thresholds for $CGQBminer$. Setting a large quasi and small minimum size and support thresholds will likely to result in an exorbitant number of CGQB subgraphs being discovered, while setting a small quasi and large minimum size and support thresholds will likely to result in no CGQB subgraph being discovered.

8. Conclusion

We tackled an important hypothesis in financial data mining, and showed that indeed, stocks with similar financial ratio values across years do have similar price movements. This was achieved by mapping the problem to that of discovering CGQB subgraphs, which are essentially 3D co-clusters between stocks and financial ratios in multiple yearly graphs. CGQB subgraphs fit the bill, because they can handle the problem of missing data, which is quite rampant in financial data in general. Its *quasi* threshold allows users to control the degree of similarity between financial ratio values of stocks within a cluster. We developed a novel and efficient algorithm, $CGQBminer$, to mine the complete set of CGQB subgraphs from a set of bipartite graphs. In our experiments, we showed by statistical tests that groups of stocks mined by CGQB subgraphs have similar price movements, and they are statistically more significant than groups of stocks mined by existing 3D subspace clustering algorithm. We also demonstrated that the probability of making profit by using groups of stocks mined by CGQB subgraphs is substantially higher than by using an investment strategy based solely on historical prices.

Lastly, we showed that $CGQBminer$ is highly efficient across datasets of different dimensions and densities. Our future work consists of three parts. First, we will conduct in-depth analysis on which ranges of financial ratio values are important indicators of similar price movements, particularly rising price movements. Second, based on the domain application, we will propose some ranking system to rank the usefulness or utility of the CGQB subgraphs mined. Third, we will use the price information of stocks to guide the discretization of financial ratios, and incorporate it into the clustering process.

References

[1] J. Abello, M. G. C. Resende, and S. Sudarsky. Massive quasi-clique detection. In *Proceedings of the 5th Latin American Theoretical Informatics Symposium (LATIN)*, pages 598–612, 2002.

[2] G. Alexe, S. Alexe, Y. Crama, S. Foldes, P. L. Hammer, and B. Simeone. Consensus algorithms for the generation of all maximal bicliques. *Discrete Applied Mathematics*, 145(1):11–21, 2004.

[3] J. Besson, C. Robardet, and J.-F. Boulicaut. Mining a new fault-tolerant pattern type as an alternative to formal concept discovery. In *Proceedings of the 6th International Conference on Computational Science (ICCS)*, pages 144–157, 2006.

[4] D. Bu *et al.* Topological structure analysis of the protein-protein interaction network in budding yeast. *Nucleic Acids Research*, 31(9):2443–2450, 2003.

[5] J. Y. Campbell and R. J. Shiller. Valuation ratios and the long-run stock market outlook: An update. *Journal of Portfolio Management*, 24:11–26, 2001.

[6] L. Cerf, J. Besson, C. Robardet, and J.-F. Boulicaut. Data peeler: Contraint-based closed pattern mining in n-ary relations. In *Proceedings of the 8th SIAM International Conference on Data Mining (SDM)*, pages 37–48, 2008.

[7] R. Diestel. *Graph Theory*. Graduate Texts in Mathematics. Springer-Verlag Heidelberg, 3rd edition, 2005.

[8] E. J. Elton, M. J. Gruber, S. J. Brown, and W. N. Goetzmann. *Modern Portfolio Theory and Investment Analysis*. John Wiley & Sons, 2007.

[9] D. Eppstein. Arboricity and bipartite subgraph listing algorithms. *Information Processing Letters*, 51(4):207–211, 1994.

[10] E. F. Fama. Efficient capital markets: A review of theory and empirical work. *The Journal of Finance*, 25(2):383–417, 1970.

[11] B. Graham. *The Intelligent Investor: A Book of Practical Counsel*. Harper Collins Publishers, 1986.

[12] B. Graham and D. Dodd. *Security Analysis*. McGraw-Hill Professional, 1934.

[13] M. C. Gupta and R. J. Huefner. A cluster analysis study of financial ratios and industry characteristics. *Journal of Accounting Research*, 10(1):77–95, 1972.

[14] M. Halkidi and M. Vazirgiannis. Clustering validity assessment using multi representatives. In *Proceedings of the 2nd Hellenic Conference on Artificial Intelligence (SETN)*, 2002.

[15] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*, pages 535–555. Morgan Kaufmann, 2nd edition, 2005.

[16] J. Huan, W. Wang, and J. Prins. Efficient mining of frequent subgraphs in the presence of isomorphism. In *Proceedings of the 3rd International Conference on Data Mining (ICDM)*, page 549, 2003.

[17] IBM Quest Market-Basket Synthetic Data Generator. http://www.cs.umbc.edu/˜cgiannel/assoc_gen.html [Last accessed: 2009].

[18] Investopedia. http://www.investopedia.com/university/ratios/ [Last accessed: 2009].

[19] L. Ji, K.-L. Tan, and A. K. H. Tung. Mining frequent closed cubes in 3D datasets. In *Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB)*, pages 811–822, 2006.

[20] D. Jiang and J. Pei. Mining frequent cross-graph quasi-cliques. *ACM Transactions on Knowledge Discovery from Data*, 2(4):1–42, 2009.

[21] B. F. King. Market and industry factors in stock price behavior. *The Journal of Business*, 39(1):139–190, 1965.

[22] B. Kovalerchuk and E. Vityaev. Data mining for financial applications. In *The Data Mining and Knowledge Discovery Handbook*, pages 1203–1224. Springer, 2005.

[23] H.-P. Kriegel, P. Kröger, and A. Zimek. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Transactions on Knowledge Discovery from Data*, 3(1):1–58, 2009.

[24] M. Kuramochi and G. Karypis. Frequent subgraph discovery. In *Proceedings of the 1st International Conference on Data Mining (ICDM)*, pages 313–320, 2001.

[25] J. W. Lewellen. Predicting returns with financial ratios. *Journal of Financial Economics*, 74:209–235, 2004.

[26] H. Li and J. Sun. Gaussian case-based reasoning for business failure prediction with empirical data in China. *Information Sciences*, 179(1-2):89–108, 2009.

[27] J. Li, H. Li, D. Soh, and L. Wong. A correspondence between maximal complete bipartite subgraphs and closed patterns. In *Proceedings of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, pages 146–156, 2005.

[28] J. Li, K. Sim, G. Liu, and L. Wong. Maximal quasi-bicliques with balanced noise tolerance: Concepts and co-clustering applications. In *Proceedings of the 8th SIAM International Conference on Data Mining (SDM)*, pages 72–83, 2008.

[29] G. Liu, K. Sim, and J. Li. Efficient mining of large maximal bicliques. In *Proceedings of the 8th International Conference on Data Warehousing and Knowledge Discovery (DaWak)*, pages 437–448, 2006.

[30] H. Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–91, 1952.

[31] N. Mishra, D. Ron, and R. Swaminathan. A new conceptual clustering framework. *Machine Learning*, 56(1-3):115–151, 2005.

[32] J. Pei, D. Jiang, and A. Zhang. Mining cross-graph quasi-cliques in gene expression and protein interaction data. In *Proceedings of the 21st International Conference on Data Engineering (ICDE)*, pages 353–354, 2005.

[33] J. Pei, D. Jiang, and A. Zhang. On mining cross-graph quasi-cliques. In *Proceedings of the 11th ACM International Conference on Knowledge Discovery in Data Mining (SIGKDD)*, pages 228–238, 2005.

[34] J. Price and E. Kelly. Warren Buffett: Investment genius or statistical anomaly? In *Workshop on Intelligent Finance: A Convergence of Mathematical Finance with Technical and Fundamental Analysis*, 2004.

[35] S. Rabaséda, R. Rakotomalala, and M. Sebban. A comparison of some contextual discretization methods. *Information Sciences*, 92(1-4):137–157, 1996.

[36] P. Ravisankar, V. Ravi, and I. Bose. Failure prediction of dotcom companies using neural network-genetic programming hybrids. *Information Sciences*, 180(8):1257–1267, 2010.

[37] R. Rymon. Search through systematic set enumeration. In *Proceedings of the 8th International Conference on Principles and Knowledge Representation and Reasoning (KR)*, pages 539–550, 1992.

[38] K. Sim, J. Li, V. Gopalkrishnan, and G. Liu. Mining maximal quasi-bicliques to co-cluster stocks and financial ratios for value investment. In *Proceedings of the 6th International Conference on Data Mining (ICDM)*, pages 1059–1063, 2006.

[39] K. Sim, J. Li, V. Gopalkrishnan, and G. Liu. Mining maximal quasi-bicliques: Novel algorithm and applications in the stock market and protein networks. *Statistical Analysis and Data Mining*, 2(4):255–273, 2009.

[40] Standard & Poor’s Compustat. http://www.compustat.com [Last accessed: 2009].

[41] P. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison-Wesley, 2005.

[42] E. Tomita, A. Tanaka, and H. Takahashi. The worst-case time complexity for generating all maximal cliques. In *Proceedings of the 10th International Computing and Combinatorics Conference (COCOON)*, pages 161–170, 2004.

[43] C.-J. Tsai, C.-I. Lee, and W.-P. Yang. A discretization algorithm based on class-attribute contingency coefficient. *Information Sciences*, 178(3):714–731, 2008.

[44] Y.-J. Wang and H.-S. Lee. A clustering method to identify representative financial ratios. *Information Sciences*, 178(4):1087–1097, 2008.

[45] S. Wu and T. W. S. Chow. Self-organizing-map based clustering using a local clustering validity index. *Neural Processing Letters*, 17(3):253–271, 2003.

[46] C. Yan, J. G. Burleigh, and O. Eulenstein. Identifying optimal incomplete phylogenetic data sets from sequence databases. *Molecular Phylogenetics and Evolution*, 35:528–535, 2005.

[47] X. Yan and J. Han. gSpan: Graph-based substructure pattern mining. In *Proceedings of the 2nd International Conference on Data Mining (ICDM)*, page 721, 2002.

[48] X. Yan and J. Han. Closegraph: mining closed frequent graph patterns. In *Proceedings of the 9th ACM International Conference on Knowledge Discovery in Data Mining (SIGKDD)*, pages 286–295, 2003.

[49] E. C.-C. Yen. Warning signals for potential accounting frauds in blue chip companies - an application of adaptive resonance theory. *Information Sciences*, 177(20):4515–4525, 2007.

[50] L. Zhao and M. J. Zaki. TRICLUSTER: an effective algorithm for mining coherent clusters in 3D microarray data. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*, pages 694–705, 2005.

[51] Z. Zhu, H. He, J. A. Starzyk, and C. Tseng. Self-organizing learning array and its application to economic and financial problems. *Information Sciences*, 177(5):1180–1192, 2007.

Appendix A. Definitions of Abbreviations

Abbreviation	Definition
CGB	Cross-graph biclique
CGQB	Cross-graph quasi-biclique
CGQC	Cross-graph quasi-clique
QB	Quasi-biclique
D/E	*Debt-Equity* ratio = $\frac{\textit{Total Liabilities}}{\textit{Shareholders' Equity}}$
PE	*Price-Earnings* ratio = $\frac{\textit{Stock Price per Share}}{\textit{Earnings per Share}}$
ROE	*Return on Equity* ratio = $\frac{\textit{Net Income}}{\textit{Average Shareholders' Equity}}$