Biclustering with Background Knowledge using Formal Concept Analysis

Faris Alqadah* Joel S. Bader † Rajul Anand ‡ Chandan K. Reddy §

Abstract: Biclustering methods have proven to be critical tools in the exploratory analysis of high-dimensional data including information networks, microarray experiments, and bag of words data. However, most biclustering methods fail to answer specific questions of interest and do not incorporate background knowledge and expertise from the user. To this end, query-based biclustering algorithms have been recently developed in the context of microarray data; these algorithms utilize a set of seed genes provided by the user to prune the search space and guide the biclustering algorithm. In this paper, a novel **Q**uery-**B**ased **B**i-**C**lustering algorithm, *QBBC*, is proposed via a new formulation that combines the advantages of low-variance biclustering techniques and Formal Concept Analysis. We prove that statistical dispersion measures that are order-preserving induce an ordering on the set of biclusters in the data. In turn, this ordering is exploited to form query-based biclusters in an efficient manner. Our novel approach provides a mechanism to generalize query-based biclustering to sparse high-dimensional data such as information networks and bag of words. Moreover, the proposed framework performs a local approach to query-based biclustering as opposed to the global approaches that previous algorithms have employed. Experimental results indicate that this local approach often produces higher quality and precise biclusters compared to the state-of-the-art query-based methods. In addition, a performance evaluation illustrates the efficiency and scalability of QBBC compared to full biclustering approaches and other existing query-based approaches.

Keywords: Biclustering; Formal Concept Analysis;Background Knowledge; Query based Clustering.

1 Introduction

The abundance of high-dimensional data in applications ranging from text mining to bioinformatics prompted the development of biclustering, co-clustering and subspace clustering algorithms [11, 20]. All of these approaches attempt to identify clusters of objects in conjunction with the subset of features in high-dimensional datasets to avoid the curse of dimensionality. Although these methods have proven to be useful tools in exploratory analysis, most of these methods do not answer specific questions of interest and fail to incorporate prior knowledge and expertise from the user. For example, biologists often know that specific sets of genes are related to shared biological functions or pathways. Based on this prior knowledge, experts may want to enlist additional genes involved in that function in microarray data [13]. Obtaining Patterns that take advantage of the background knowledge provided by the user/expert is potentially more useful and reliable for further analysis. Exploiting background knowledge while extracting clusters has gained significant attention [5, 6, 32, 7, 3]. In some cases, user input is utilized to generate constraints; most widely these are instance-based constraints, namely must-link and cannot-link. The state-of-the-art methods utilize probabilistic relation models [35] or Bayesian methods [13] where the expert knowledge is provided through prior distributions. However, generally users do not have this level of expertise and can only provide intuitive keywords or an input seed set. To overcome these limitations,

*Johns Hopkins University, Baltimore, MD. Email: faris.alqadah@jhu.edu
†Johns Hopkins University, Baltimore, MD. Email: joel.bader@jhu.edu
‡Wayne State University, Detroit, MI. Email: rajulanand@wayne.edu
§Wayne State University, Detroit, MI. Email: reddy@cs.wayne.edu

query-based biclustering algorithms provide an ideal solution in which background knowledge is treated as a guide for obtaining clusters while simultaneously checking the relevance of the user input seed set.

Query-based biclustering algorithms have been originally developed in the bioinformatics community [18, 33, 23, 8, 35, 13] specifically targeting microarray data. These algorithms utilize a set of seed genes provided by the user with the assumption that these seeds are tightly co-expressed or functionally related. In turn, the seed set is employed to prune the search space and guide the biclustering algorithm. Query-based biclustering algorithms characteristically attempt to keep biclusters centered around the seed set. For example, consider an information network linking key terms with research papers; a database researcher may wish to uncover which authors have been performing research in the field of 'collaborative filtering'. Seeding an ideal query based biclustering algorithm with the seed terms *collaborative* and *filtering* would not only unveil the authors who have addressed this topic but also additional key terms that are related to collaborative filtering.

However, the query-based biclustering algorithms must also be robust and recognize an incoherent or partially incoherent seed set. Bio-inspired algorithms tend to adhere to these requirements, but are highly expensive and do not generalize well to large-scale datasets such as information networks, social networks and bag of words. As the data complexity grows, there is a need for an efficient query-based biclustering algorithm which can identify biclusters in complex data. To solve this problem, we propose a novel formulation of query-based biclustering in this paper. Our approach generalizes previous approaches to sparse and very high-dimensional data. Combining low-variance biclustering techniques and Formal Concept Analysis (FCA), the `QBBC` algorithm is developed in our work. We prove that the statistical dispersion measures that are order-preserving induce an ordering on the set of biclusters; consequently, this ordering is exploited to mine query-based biclusters in an efficient manner. Additionally, we capitalize on this ordering to identify neighboring biclusters that admit minimal noise when joining the clusters. In this manner, biclusters may be combined to enhance query results while still centering on the seed set.

1.1 Contributions

The main contributions of our work are summarized as follows:

- Novel formulation of query-based biclustering (and biclustering in general) through a combination of low-variance biclustering and Formal Concept Analysis. In this formulation, we prove that order-preserving statistical measures of dispersion can induce an ordering that permits efficient mining.
- Development of the `QBBC` algorithm to efficiently mine query-based biclusters and approximate their ordering relation. The `QBBC` algorithm extends the seminal `CHARM` [34] algorithm (traditionally utilized to mine closed itemsets) by making use of an original operator termed "range intersection".
- Formulation of two data-driven evaluation measures that capture the notion of coherence in biclusters.
- Experimental study with six real-world datasets from a wide range of real-world applications and performance comparison of the results with existing state-of-the-art approaches.

Following a review of existing works in Section 2, the theoretical clustering formulation is described in Section 3. Some additional mathematical background is provided in Section 4. Section 5 describes the `QBBC` algorithm, while Section 6 presents the results of our performance experiments. Finally, Section 7 offers concluding remarks, shortcomings, and avenues for future work.

2 Related Work

The clustering process must satisfy the constraints. The instance-based constraints have also been used for biclustering of text documents [27, 30]. Existing works that are most relevant to the proposed approach fall into three categories: (i) bio-inspired query-based biclustering, (ii) semi-supervised clustering, and (iii) pattern-based biclustering. In this section, prior works in each of these categories are succinctly summarized.

As mentioned in the introduction, query-based biclustering methods have been developed in the bioinformatics community. QDB [13] and ProBic [35] are the two algorithms that are relevant to our proposed work. QDB encompasses a Bayesian framework in which a conditional maximization is utilized for model estimation. Intuitively, biclusters are defined as sub-matrices of the original matrix whose expression values are modeled by a bicluster distribution as opposed to a "background" distribution (the rest of the data). Domain knowledge is encoded in the form of prior probability distributions. Finally, a resolution sweep method determines the ideal resolution that biclusters should be displayed at. Determining the ideal biclustering resolution still remains subjective. We address this issue by making use of a bicluster ordering mechanism. This ordering is similar to having a dendrogram in hierarchical clustering which allows us to view clusters according to the user needs. The ProBic method, which is a follow up to QDB, is conceptually similar but adapts a probabilistic relation model as an extension to the Bayesian framework. Hard assignment of biclusters is assigned with the Expectation Maximization (EM) algorithm used to learn the model. Another method in this category is an earlier approach, namely, the Iterative Signature Algorithm (ISA) [8] which utilizes the mean expression profile of the seed set to initialize the biclustering. Biclusters are then defined as fixed points with significant over or under expression. ISA does not deal with missing values, making it highly unlikely to be effective with sparse data. In addition, the algorithm is not purely query-based; there is no guarantee that a bicluster does not completely drift away from the original seed set. GeneRecommender [23] primarily focuses on prioritizing genes, and hence requires additional post-processing steps to convert to a biclustering approach. QDB was shown to outperform both GeneRecommender and ISA on synthetic data, while producing biologically relevant results in real data. More recently, ProBic was shown to be more effective compared to ISA and QDB. These existing query-based biclustering algorithms pose a tedious post-processing of partially redundant biclustering results. These algorithms are required to do such processing for each query seed set and test them with multiple parameter settings need to be tested in order to detect the 'most optimal bicluster size' adds to the redundancy problem [28]. Using the bicluster ordering, determining the optimal bicluster size embraced the closed pattern mining approach, widely used for mining maximal closed patterns.

Semi-supervised clustering has mainly been characterized by constraint-based one-dimensional clustering [7] which is a well-investigated research topic. In these works, pairwise constraints on objects, namely "cannot-link" and "must-link", are utilized. The must-link constraint indicate that two objects should belong to the same class while cannot-link constraint indicate that two objects should belong to different class. Respecting these constraints can allow us to incorporate domain knowledge into the clustering process and further improve the quality of clustering solution. Few methods have extended such constraint-based formulations to even biclustering settings [25, 29, 22, 27]. However, this problem is substantially different from the problem that we are addressing in this paper. *The query seed set provided as input does not impose an explicit constraint, rather, it represents a user preference which may infact be ignored by the algorithm if such a set is determined to be incoherent.* In a departure from the constraint based approaches, the proposed work incorporates background knowledge into the biclustering process by using it only as a "guidance" and systematically allows for an approximation of this knowledge to be obeyed during the search process.

Recently, some preliminary efforts have been made to extend closed pattern and association rule analysis to biclustering and multi-way clustering of real-valued data [10, 24, 19]. The advantage of these methods is the ability to exhaustively search the set of biclusters and locate smaller and finer grain clusters often missed or masked by other methods. The primary disadvantage of pattern based biclustering algorithms is their potentially prohibitive computational cost. To address this issue, both [24] and [19] apply order preserving dispersion measures allowing efficient pruning of the search space. In this work, we build upon and extend the theoretical foundation of pattern based biclustering. The notion of an ordering preserving statistic is explicitly introduced and is proven that in addition to permitting effective pruning, such statistics impose an ordering upon the set of biclusters. As real-world data is typically dominated by a small number of very strong biclusters, this ordering is critical in facilitating the identification of neighboring biclusters to a seed set that enhance the final result.

Table 1: Notations used in this paper

Notation	*Brief Description*
g or m	object
G or M	An **object-set** which is subset from $\mathbf{G}$ or $\mathbf{M}$
$\mathbf{G}$ or $\mathbf{M}$	sets of objects
$\mathbf{K}$	a $\|\mathbf{G}\|$ by $\|\mathbf{M}\|$ matrix
$\mathbb{K}$	a triple of $(\mathbf{G}, \mathbf{M}, \mathbf{K})$
$\Gamma(g)$ or $\Gamma(m)$	the set of adjacent vertices to g or m
M^q alias Q	**query set**
d	statistical measure of dispersion
f	**consistency function**
α	user selected parameter
$\psi_f^\alpha(M^q)$	**supporting set** of the query set
(G, M)	**subspace** or a sub-matrix $\mathbf{K}[G, M]$
α**-concept**	**bicluster**

Finally, [24] and [19] measure coherence of biclusters through comparison of the range to a constant user-selected threshold. In this work, we introduce two original data-dependent measures for the evaluation of coherence.

3 Preliminaries

In this section, we first introduce the notations used in this paper for query-based biclustering. Finally, we give a brief overview of Formal Concept Analysis (FCA) along with some lemmas that show how to modify the ideas from FCA to extract query-based biclusters from real-valued matrices.

A context $\mathbb{K} = (\mathbf{G}, \mathbf{M}, \mathbf{K})$ is a triple where $\mathbf{G}$ and $\mathbf{M}$ are sets of objects and $\mathbf{K}$ is a $|\mathbf{G}|$ by $|\mathbf{M}|$ matrix relating the objects of $\mathbf{G}$ and $\mathbf{M}$. $\mathbf{K}$ may also be thought of as the adjacency matrix of a bipartite graph with vertex sets $\mathbf{G}$, $\mathbf{M}$, and edge set $\{(g, m) | \mathbf{K}[g, m] \neq -\infty\}$ with the edge weighting function $w(g, m) = \mathbf{K}[g, m]$. $\mathbf{K}[g, m] = -\infty$ denotes an object $g \in \mathbf{G}$ is not related to object $m \in \mathbf{M}$. $\Gamma(g)$ denotes the set of adjacent vertices to g (dually $\Gamma(m)$). An **object-set** G or M is a subset of objects from $\mathbf{G}$ or $\mathbf{M}$. A **subspace** is any pair of object-sets (G, M) which also maybe thought of as a sub-matrix $\mathbf{K}[G, M]$. A **query set** is any object-set M^q (dually G^q) that is input by a user querying $\mathbb{K}$.

Given M^q, our goal is to identify a subspace (G, M), where $M \supseteq M^q$, that exhibit consistent values across the rows (columns) of $\mathbf{K}[G, M]$. In the terminology of biclustering [20], the desired result is to produce constant value biclusters in terms of rows or columns with the given constraint of a user query set. In order to quantify consistency of values in a subspace, statistical measures of dispersion such as standard deviation, inter-quantile range, range, and mean difference are utilized.

For a given query set M^q, the **supporting set** of M^q are those objects in $\mathbf{G}$ that are jointly adjacent to M^q and exhibit consistent values. Formally, define d as a dispersion measure that maps a subspace $(G, M) \mapsto \mathrm{R}$. Moreover, define a **consistency function**, f, that serves as a standard on what constitutes a consistent subspace; $f : (\mathbb{K}, g, M, \alpha) \mapsto \mathrm{R}$, where α is a user-selected parameter. Figure 2 displays several dispersion and consistency functions.

DEFINITION 1. *Given $\mathbb{K} = (\mathbf{G}, \mathbf{M}, \mathbf{K})$, query set M^q, and user selected parameter α, the **supporting set** of M^q, denoted as $\psi_f^\alpha(M^q)$, is defined as*

$$\{g \in \mathbf{G} \quad | \quad \Gamma(g) \subseteq M^q \wedge d(\mathbf{K}[g, M^q]) \leq f(\mathbb{K}, g, M, \alpha)\}$$

*where f is a **consistency function** and d is a statistical measure of dispersion.*

A general definition for a constant valued bicluster follows naturally from the supporting set formulation.

DEFINITION 2. *Given $\mathbb{K} = (\mathbf{G}, \mathbf{M}, \mathbf{K})$, consistency function f, dispersion measure d, and parameter α, a **bicluster** or **α-concept** of $\mathbb{K}$ is a subspace (G, M) such that*

1. $\psi_f^\alpha(M) = G$
2. ***Closure**: There does not exist $m \in \mathbf{M} \setminus M$ such that $\psi_f^\alpha(M \cup m) = G$.*

The **Closure** condition ensures that the maximum number of rows (columns) have been included in the bicluster without violating the consistency conditions.

DEFINITION 3. *Given $\mathbb{K} = (\mathbf{G}, \mathbf{M}, \mathbf{K})$, consistency function f, dispersion measure d, and query set $Q \subset \mathbf{M}$ (dually $Q \subset \mathbf{G}$), then an **ideal query-based bicluster** of $\mathbb{K}$ is an α-concept (G, Q'), where $Q' \supseteq Q$.*

4 Theoretical Background

In real datasets, we expect the structure and nature of query-based biclusters to be highly sensitive to the choices of α and f. Non-stringent measures cause biclustering algorithms to mask or miss small but relevant biclusters [24]; on the other hand, too stringent parameter settings may cause the algorithm to conclude that a query set is incoherent and thus will not return any bicluster. This resolution problem makes it difficult to locate an ideal query-based bicluster given a user query. We propose a computationally efficient scheme to account for the parameter selection problem that does include varying parameter settings. We advocate setting stringent parameter settings and utilizing several small, localized and similar α-concepts to construct larger query-based biclusters that still center around the query. We show in the sequel that utilizing order preserving dispersion measures induce an ordering on the set of α-concepts in the data. In turn, this ordering is exploited to identify α-concept neighborhoods that consist of similar α-concepts centered around the query set.

4.1 Formal Concept Analysis As applied to binary relations, Formal Concept Analysis (FCA) defines concepts as the maximal rectangles of 1s under any suitable permutations of the rows and columns of $\mathbf{K}$ [16]. Using this definition, FCA further stipulates a complete mathematical framework for reasoning about relations between concepts. This framework relies upon the fundamental theorem of FCA: concepts are ordered by the **hierarchical order** and under this ordering, the concepts form a complete lattice. Under this formulation, identifying neighborhoods of closely related concepts is a well-defined task and several efficient algorithms exist to identify these neighborhoods [9].

In this section, we prove that the **hierarchical order** also applies to α-concepts given that the dispersion measures and consistency functions adhere to certain ordering properties. As a result, we are able to leverage the FCA framework to identify combinations of α-concepts that are best centered around the query set.

DEFINITION 4. *Given α-concepts (G_1, M_1) and (G_2, M_2), then $(G_1, M_1) \leqslant (G_2, M_2)$ if and only if $G_1 \subseteq G_2$ and $M_1 \supseteq M_2$. This ordering relation is referred to as the **hierarchical** ordering.*

Undoubtedly, the selected dispersion measure and consistency function determine if the set of α-concepts are in fact ordered by the hierarchical ordering.

DEFINITION 5. *d is order-preserving if $d(\mathbf{K}[g, M]) \leq d(\mathbf{K}[g, M \cup m])$, where $m \in \mathbf{M} \setminus M$*

DEFINITION 6. *f is anti-monotone if $f(\mathbb{K}, g, M, \alpha) \geq f(\mathbb{K}, g, M \cup m, \alpha)$, where $m \in \mathbf{M} \setminus M$*

If d and f are order-preserving and anti-monotone, then $\psi^\alpha(M^q)$ is also anti-monotone; this in turn implies that α-concepts defined in terms of an order-preserving dispersion statistics and anti-monotone consistency functions are ordered by the hierarchical order.

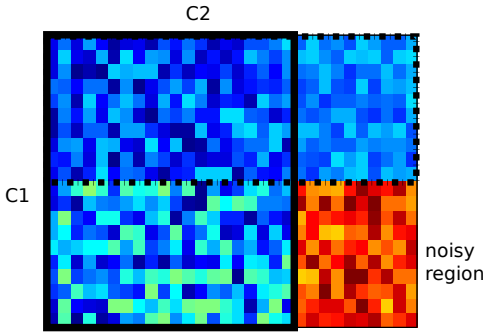


Figure 1: Noisy region induced by combining two neighboring concepts

THEOREM 4.1. *Given a context $\mathbb{K}$, α, d, and f, then if d and f are order-preserving and anti-monotone respectively the α-concepts of $\mathbb{K}$ are ordered by the hierarchical order.*

Proof. See Appendix.

FCA stipulates that concepts ordered by the hierarchical order form a complete lattice; therefore we conclude that the set of α-concepts also form a complete lattice. The α-concept lattice forms the basis for defining α-concept neighborhoods.

4.1.1 Neighboring Concepts Consider concepts (G_1, M_1) and (G_2, M_2) such that $(G_1, M_1) \geqslant (G_2, M_2)$. If there is no concept (G_3, M_3) fulfilling $(G_1, M_1) \geqslant (G_3, M_3) \geqslant (G_2, M_2)$ then (G_1, M_1) is an **upper neighbor** of (G_2, M_2); dually (G_2, M_2) is a **lower neighbor** of (G_1, M_1). For any concept C_1, its set of upper neighbors is denoted as $\Upsilon(C_1)$. Intuitively, neighboring concepts are assumed to be similar. This intuition has been formalized and exploited to extract knowledge in binary contexts [2, 21]. In terms of query-based biclustering, neighboring α-concepts provide a mechanism to explore and combine closely related α-concepts in order to enhance or broaden a seed bicluster. *We formally show that combining α-concept $C_1 = (G_1, M_1)$ and any upper (lower) neighbor $C_2 = (G_2, M_2)$ will result in the minimal degree of inconsistency among all possible pairings of larger α-concepts.*

Figure 1 depicts the result of combining α-concept C_1 and an upper neighbor C_2. By definition, the values encompassed by C_1 and C_2 are consistent.However, the values induced by the difference between C_1 and C_2 may not be consistent and can form a noisy region. The critical criterion here is to determine whether a combination of α-concept C_1 and an upper neighbor is suitable or not. We can solve this problem by introducing a way to measure the consistency of the noisy region. One possible measure of dissimilarity between two α-concepts is the degree of inconsistency introduced by the noisy region when combining the two α-concepts. Given below, the $dist$ score assesses the dissimilarity by computing the ratio of the consistency function as measured in the original concept to the noisy region.

$$dist((G_1, M_1), (G_2, M_2)) = \frac{1}{|G_1 \setminus G_2|} \sum_{g \in G_1 \setminus G_2} \frac{d(\mathbf{K}[g, M_2 \setminus M_1])}{d(\mathbf{K}[g, M_2])} \times s_g \tag{4.1}$$

where

$$s_g = 1 + |(\Gamma(g) \cap M_2) \setminus M_1|$$

Name	Computation	Order Preserving?
Range	$r(\mathbf{K}[m, G]) = \max \mathbf{K}[g, M] - \min \mathbf{K}[g, M]$	yes
Standard Deviation	$\sigma(\mathbf{K}[m, G])$	no
Inter-quantile Range	$Q_3(\mathbf{K}[m, G]) - Q_1(\mathbf{K}[m, G])$	no
Mean difference	$\frac{1}{\lvert M\rvert(\lvert M\rvert-1)} \sum_{i=1}^{\lvert M\rvert} \sum_{i=1}^{\lvert M\rvert} \lvert\mathbf{K}[g, m_i] - \mathbf{K}[g, m_j]\rvert$	no
Coefficient of variation	$\frac{\sigma(\mathbf{K}[g,M])}{\mu(\mathbf{K}[g,M])}$	no
Quartile coefficient	$\frac{Q_3(\mathbf{K}[g,M]) - Q_1(\mathbf{K}[g,M])}{Q_3(\mathbf{K}[g,M]) + Q_1(\mathbf{K}[g,M])}$	no

(a) Dispersion Functions

Name	Computation	Anti-monotone ?
Constant threshold	$f(\mathbb{K}, g, M, \alpha) = c$	yes
Min range	$f(\mathbb{K}, g, M, \alpha) = \min \mathbf{K}[g, \mathbf{M}]$	yes

(b) Consistency Functions

Figure 2: Dispersion and consistency functions

In Equation (4.1), the fact that d is order preserving bounds the left hand side ratio inside the summation to 1. The right hand side term, s_g, is introduced to account for sparse data. In the case that $\mathbf{K}$ is full ($\mathbb{K}$ is a complete bi-partite graph) then s_g evaluates to 1 and the ratios are simply summed up. On the other hand, if the row $\mathbf{K}[g, M_2 \setminus M_1]$ contains missing values, then a penalty term is proportionally imposed on the ratio. Finally, the consistency ratios are averaged over the total number of rows in the join of the concepts.

THEOREM 4.2. *For any α-concept (G_1, M_1), then*

$$\underset{(G_2,M_2)\geqslant(G_1,M_1)}{\operatorname{argmin}} \; dist((G_1, M_1), (G_2, M_2)) \in \Upsilon(G_1, M_1)$$

Proof. See appendix.

Theorem 4.2 provides a theoretical basis for combining neighboring α-concepts to form a query-centered bicluster.

4.2 Dispersion and Consistency Functions Figure 2 depicts several standard statistical measures of dispersion along with some select consistency functions that have been utilized recently in biclustering algorithms [24, 19]. The only dispersion measure that is order-preserving is range which justifies its use previously. Unfortunately, range is not a robust statistic; nevertheless, its use as a dispersion measure is justified because it is order-preserving and it bounds the standard deviation as follows:

$$2 \times \sigma(\mathbf{K}[g, M]) \leq r(\mathbf{K}[m, G]) \tag{4.2}$$

Furthermore, relatively few consistency functions exist; previous methodologies have simply imposed hard thresholds and thus adding another parameter whose optimal value can only be obtained experimentally. We solved this problem by developing two novel data-dependent, anti-monotone consistency functions.

4.2.1 α-sigma Consistency Function Given $\mathbb{K} = (\mathbf{G}, \mathbf{M}, \mathbf{K})$, assume that the rows and columns of $\mathbf{K}$ are i.i.d and normally distributed. That is, $\forall\, g \in \mathbf{G}$, $\mathbf{K}[g, \mathbf{M}]$ is normally distributed with $\sigma(\mathbf{K}[g, \mathbf{M}])$ and $\mu(\mathbf{K}[g, \mathbf{M}])$ respectively. The 3-sigma rule states that about 68.26 % of the values for each $\mathbf{K}[g, \mathbf{M}]$ lie within a single standard deviation of the mean while 95.44 % of the values lie within two standard deviations from the mean. Hence, randomly selecting query object-set M, the majority of values in the subspace $\mathbf{K}[g, M]$ are also expected

to lie within the $\mu(\mathbf{K}[g,\mathbf{M}]) \pm 2\sigma(\mathbf{K}[g,\mathbf{M}])$ with mean $\mu(\mathbf{K}[g,\mathbf{M}])$ and standard deviation $\sigma(\mathbf{K}[g,\mathbf{M}])$. A subspace $\mathbf{K}[g,M]$ where $\sigma(\mathbf{K}[g,M]) < \sigma(\mathbf{K}[g,\mathbf{M}])$ indicates that the values in the subspace are more consistent than expected. This can be generalized to subspaces where $\alpha \times \sigma(\mathbf{K}[g,M]) < \sigma(\mathbf{K}[g,\mathbf{M}])$, where $\alpha \geq 1$; then the α-sigma consistency function is formulated as

$$\begin{aligned} \alpha \times \sigma(\mathbf{K}[g,M]) &< \sigma(\mathbf{K}[g,\mathbf{M}]) \\ \alpha\frac{r_g}{2} &< \sigma(\mathbf{K}[g,\mathbf{M}]) \qquad (4.3) \\ r_g &< \frac{2\sigma(\mathbf{K}[g,\mathbf{M}])}{\alpha} \qquad (4.4) \end{aligned}$$

where r_g denotes $r(\mathbf{K}[m,G])$ and equation (4.3) follows from the stringent assumption imposed by inequality (4.2).

DEFINITION 7. *For context $\mathbb{K} = (\mathbf{G},\mathbf{M},\mathbf{K})$ and given subspace $\mathbf{K}[g,M]$, the α-**sigma** consistency function is defined as*

$$f(\mathbb{K},g,M,\alpha) = \frac{2\sigma(\mathbf{K}[g,\mathbf{M}])}{\alpha} \qquad (4.5)$$

The α-sigma consistency function is clearly anti-monotone.

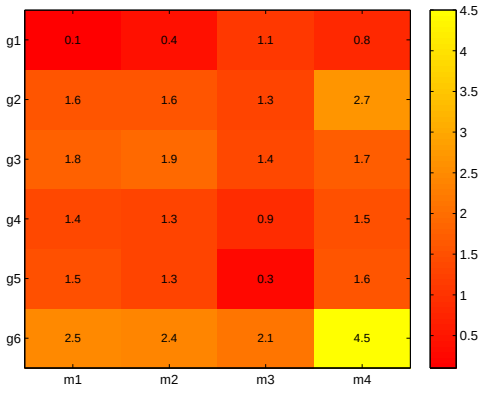
4.2.2 Maximum Spacing Uniform Estimator A less stringent consistency function may seek subspaces whose range is α times smaller than the range of a uniformly distributed random variable. Assume that the rows and columns of $\mathbf{K}$ are i.i.d and uniformly distributed with end points $\max(\mathbf{K}[\mathbf{g},\mathbf{M}])$ and $\min(\mathbf{K}[\mathbf{g},\mathbf{M}])$. Let $\{x_1,\ldots,x_n\}$ be an ordered sample from uniform distribution $U(a,b)$ with unknown endpoints a and b. A known uniformly minimum variance unbiased estimator of the end points is the maximum spacing uniform estimator given as follows:

$$\begin{aligned} \hat{a} &= \frac{nx_1 - x_n}{n-1} \qquad (4.6) \\ \hat{b} &= \frac{nx_n - x_1}{n-1} \qquad (4.7) \end{aligned}$$

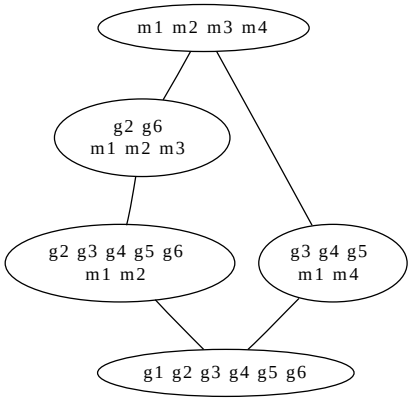
The end points of the distribution can be estimated utilizing only the end points of the sample. Hence, the range of the distribution can also be estimated by $\hat{b} - \hat{a}$. The actual range can then be compared to an estimated range as follows:

$$\begin{aligned} \hat{b} - \hat{a} &< \alpha(b-a) \\ \frac{nx_n - x_1}{n-1} - \frac{nx_1 - x_n}{n-1} &< \alpha(b-a) \\ x_n - x_1 + n(x_n - x_1) &< \alpha(n-1)(b-a) \\ x_n - x_1 &< \alpha\frac{(n-1)(b-a)}{(n+1)} \end{aligned} \qquad (4.8)$$

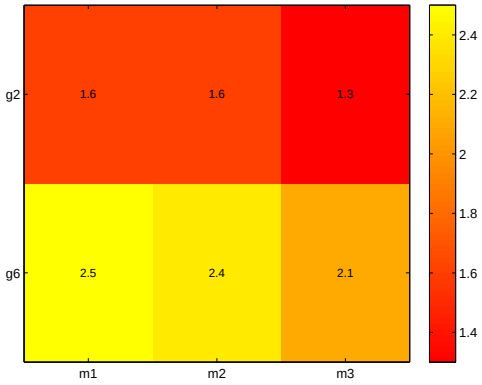
By the above reasoning, a subspace $\mathbf{K}[\mathbf{g},M]$ is considered consistent if its maximum space estimated range is α times smaller than the range of the distribution.



(a) Heat map of data matrix



(b) Lattice of α-concepts, $\alpha = 3$



(c) An α-concept, $\alpha = 3$

Figure 3: Concepts of a sample data matrix

DEFINITION 8. *For context* $\mathbb{K} = (\mathbf{G}, \mathbf{M}, \mathbf{K})$ *and given subspace* $\mathbf{K}[g, M]$*, the* ***maximum spacing uniform estimator*** *consistency function is*

$$f(\mathbb{K}, g, M, \alpha) = \alpha(|\mathbf{M}| - 1)r(\mathbf{K}[g, \mathbf{M}]) - |\mathbf{M}|r(\mathbf{K}[g, M]) \tag{4.9}$$

The maximum spacing uniform estimator consistency function is also anti-monotone.

EXAMPLE 1. *Consider the data matrix in Figure 3(a). The* α*-concept lattice of the data is depicted in Figure 3(b) utilizing the* α*-sigma consistency function with* $\alpha = 3$*. Figure 3(c) displays a heat map of the concept* $(\{g2, g6\}, \{m1, m2, m3\})$*.*

5 QBBC Algorithm

5.1 Overview Our algorithm starts with a search for an ideal query-based bicluster for the given query set $Q \subset \mathbf{M}$ (steps 2-3) by enumerating all α-concepts in the sub-matrix $\mathbf{K}[\mathbf{G}, Q]$. Exact details of the search and enumeration procedure are presented in Section 5.2. If an ideal bicluster is located, then the closure of that bicluster is computed in the full matrix and the algorithm is terminated.

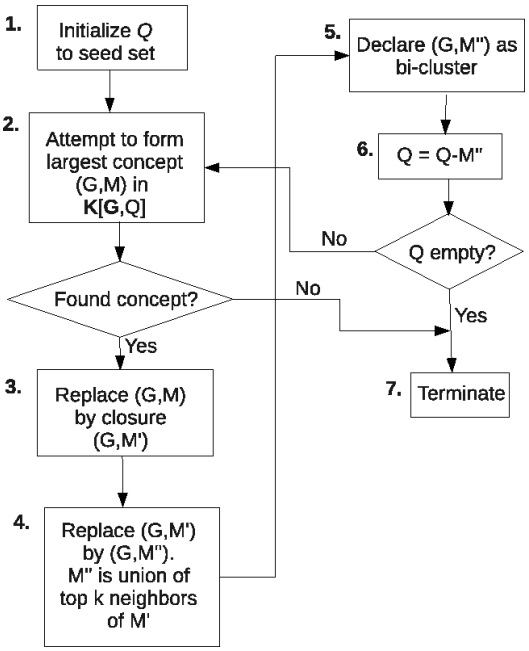


Figure 4: Overview of QBBC algorithm

Figure 4 describes the main steps of the QBBC algorithm in the more likely scenario that an ideal bicluster is not found, then QBBC retains the α-concept, (G, M), with most query objects. Next, QBBC assumes that Q is a coherent query set and an ideal bicluster was not located due to the resolution of the consistency function. Theorem 4.2 stipulates that if the query set Q was indeed coherent, then the objects in the query set not found in (G, M) would appear in α-concepts in the neighborhood of (G, M). Hence, guided by Theorem 4.2, the top k upper neighbors, ranked in ascending order by $dist$, are augmented to (G, M) to form the bicluster (G, M''). At this point, any objects in Q but not in M'' are considered incoherent with the current bicluster. Hence, a new query set is formed as $Q \setminus M''$ (step 6). The entire procedure is repeated with the new query set until Q is empty or no α-concepts are enumerated in step 2. In the sequel, we describe the algorithmic and implementation details for completing steps 2, 3 and 4 by taking advantage of the properties of order-preserving dispersion measures and building upon the seminal CHARM algorithm for closed itemset mining.

5.2 Identifying α-concepts Step 2 of QBBC calls for enumerating α-concepts in the sub-matrix $\mathbf{K}[\mathbf{G}, Q]$. The search space for this task is formulated as a prefix tree as shown in Figure 5. Recalling the dataset presented in example 1, Figure 5 depicts the search tree for query set $Q = \{m1, m2, m3\}$. *The tree utilizes the idea of prefix-based equivalence classes in order to break up the search tree into independent sub-problems.* Let us consider any object set $P \subseteq Q$ as a string, then two object-sets are in the same prefix-class if they share a common-length prefix. Each node of the tree, P, and the associated supporting set of P represents an object-set. Any node of the search tree with a non-empty supporting set is either an α-concept or a non-closed α-concept.

For a given node P, the next level of the search is generated by computing $\psi_f^\alpha(P \cup P_r)$ for nodes P_r located to the right of P under the same branch. For example, in Figure 5 consider node $\{m1, m2\}$ and notice that only node $\{m1, m3\}$ is located to the right of $\{m1, m2\}$ under the same branch. Hence, the next level of the search tree consists of node $\{m1, m2, m3\}$ and supporting set $\psi_f^\alpha(\{m1, m2, m3\})$. Moreover, *the order-preserving and anti-monotone properties of d and f ensure that the search tree may be pruned whenever an empty supporting set is encountered.* Computing $\psi_f^\alpha(P \cup P_r)$ requires identifying the supporting objects common to both P and P_r which contains consistent values. We fulfill this requirement by introducing a new operator: *range intersection*.

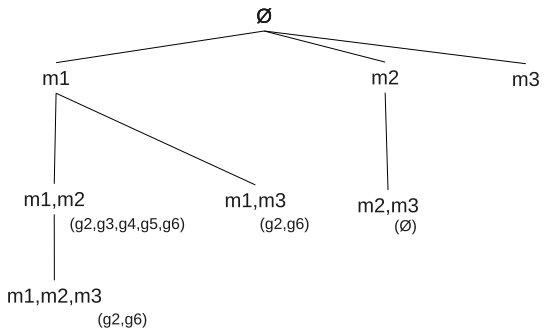


Figure 5: Prefix based search for concepts

Given two object sets P and P_r, coupled with their supporting sets S and S_r, the range intersection is defined as follows:

$$\begin{aligned}&\psi^{\alpha}(P \cup P_r) = S \sqcap S_r \\ &= \{s \in S \cap S_r | d(\mathbf{K}[s, P \cup P_r] \leq f(\mathbb{K}, s, P \cup P_r, \alpha)\}\end{aligned}$$

where d is the range dispersion function.

A naive strategy to identify α-concepts is to simply enumerate the entire prefix-based search tree. On the other hand, several closed itemset enumeration algorithms follow this strategy with additional pruning steps that vastly shrink the search space. We exploit the main theorem guiding the seminal closed item set enumeration algorithm, CHARM, which *offers further opportunity to prune the search tree by investigating the result* of $\psi_f^{\alpha}(P \cup P_r)$ when generating the next level of the search tree. **We prove that the** CHARM **theorem is in fact generalizable to α-concepts** by utilizing the α-sigma function (or the maximum spacing uniform estimator) and the range dispersion function given that f is defined to be a constant.

THEOREM 5.1. *Given the range dispersion function d and consistency function f that is either constant, the α-sigma function, or the maximum spacing uniform estimator then the following holds: Let PP_l and PP_r be two nodes under the same branch in the search tree with PP_r located to the right of PP_l. Moreover let S_l and S_r be the supporting sets of PP_l and PP_r respectively. Given that*

1. *For any prefix node PP_x under the same branch*
   $\forall s \in S_x \quad d(\mathbf{K}[s, PP_x]) \leq \frac{f(\mathbb{K}, s, PP_x, \alpha)}{2}$.
2. $SS = S_l \sqcap S_r \neq \emptyset$

then the following properties hold:

1. *If $|SS| = |S_l| \wedge |SS| = |S_r|$ then every occurrence of PP_l maybe replaced by $PP_l \cup PP_r$ and the node PP_r and all its children may be pruned from the search tree.*
2. *If $|SS| = |S_r|$ then every occurrence of PP_l may be replaced with $PP_l \cup PP_r$.*
3. *If $|SS| = |S_l|$ then every occurrence of PP_r and its children maybe pruned from search tree, but a new child node $PP_l \cup PP_r$ must be formed with the supporting set $\psi_f^{\alpha}(PP_l \cup PP_r)$.*

4. *If* $|SS| \neq |S_l| \wedge |SS| \neq |S_r|$ *then no condensation of the search tree is possible and new child node* $PP_l \cup PP_r$ *with supporting set* $\psi_f^{\alpha}(PP_l \cup PP_r)$ *must be formed.*

Proof. See Appendix.

Input: Prefix node PP_l, supporting set S_l
Result: Update PP_l by its closure
Result: Prune search space
Result: Compute children of PP_l at this branch

begin
  $C \leftarrow \emptyset$;
  `// children`
  $B \leftarrow \{(PP_r^1, S_r^1), \ldots, (PP_r^n, S_r^n)\}$;
  `// nodes to the right`
  $flg \leftarrow \forall (PP_r^i, S_r^i) \quad \forall s \in S_r^i \quad d(\mathbf{K}[s, PP_r^i]) \leq \frac{f(\mathbb{K}, s, PP_r^i, \alpha)}{2}$;
  for $(PP_r, S_r) \in B$ **do**
    $SS_r \leftarrow S_r \sqcap S$;
    if $|SS_r| = |S_r| \wedge |SS| = |S|$ **then**
      $P \leftarrow P \cup P_r$;
      if flg **then**
        Remove (PP_r, S_r) from tree ;
    else if $|SS_r| = |S_r|$ **then**
      $P \leftarrow PP_l \cup PP_r$;
    else if $|SS_r| = |S|$ **then**
      $C \leftarrow C \cup (PP_l \cup PP_r, SS_r)$;
      if flg **then**
        Remove (PP_r, SS_r) from tree ;
    else
      $C \leftarrow C \cup (PP_l \cup PP_r, SS_r)$;

Algorithm 1: A single search step corresponding to a single branch

The actualization of the above theorem yields three important outcomes: 1) computing the closure of PP_l, 2) several possible condensations of the search tree, and 3) enumerating the children of PP_l.

EXAMPLE 2. *Consider the branch under node* $\{m1\}$ *with nodes* $P_l = \{m1, m2\}$, $P_r = \{m1, m3\}$ *along with supporting sets* $S_l = \{g2, g3, g4, g5, g6\}$ *and* $S_r = \{g2, g6\}$ *as depicted in Figure 5. Condition 1 of Theorem 5.1 is satisfied due to the fact that no other prefix nodes exist under this branch. Let* f *be set to the* α*-sigma consistency function with* $\alpha = 3$ *then* $SS = S_l \sqcap S_r = \{g2, g6\}$*, hence condition 2 is also satisfied. In this case* $|SS| = |S_r|$*, hence case 2 of the theorem is applied and* P_r *is replaced by* $\{m1, m2, m3\}$*. In effect, the closure of the subspace* (S_r, P_r) *is computed and the need to form a new child in the search tree is eliminated resulting in computational savings.*

As pointed out in [34] checking for each of the four cases is a constant computational cost due to the fact that set intersection must be computed to generate the next level of the search in any case. Computing range

Name	Domain	Size	Density	Num.classes
mer	Bag of words	1,990 x 21,258	0.003	2
$allpc$	Bag of words	4,966 x 26,323	0.001	5
$allsci$	Bag of words	3,975 x 30,440	0.001	4
$papers$	Information network	28,564 x 16,891	0.0003	4
$emap$	Microarray	3,300 x 3,300	0.113	na
$hughes$	Microarray	4,684 x 300	1.0	na

Figure 6: Six real-world datasets used in our experiments.

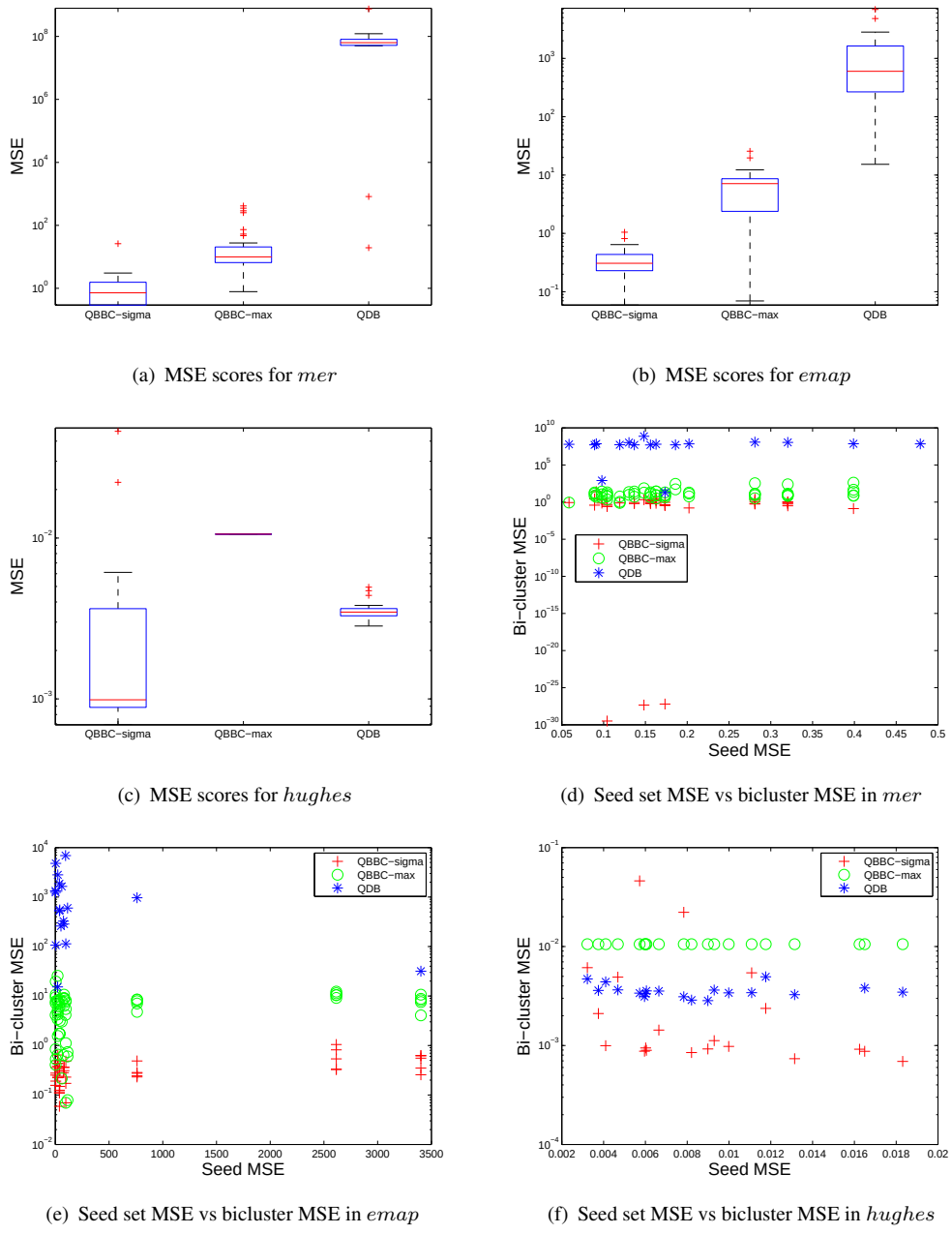
intersection of supporting sets S_l and S_r is performed in $O(|S|)$ time as it entails set intersection and a constant time operation to compute range (if the indices of maximum and minimum elements are maintained throughout the search). Augmenting Theorem 5.1 with the range intersection operation, a procedure to generate children prefix nodes and prune the tree at any branch of the search tree is given in Algorithm 1.The procedure is an exact implementation of Theorem 5.1.

5.2.1 Closure and Upper Neighbors Computing the closure of a subspace (G, M) (step 3 of QBBC) is accomplished by applying a single step of CHARM where M is the prefix node and objects $m_r \in \mathbf{M} \setminus M$ constitute nodes to the right of M. A similar strategy is utilized to determine the upper neighbors of (G, M') (step 4); in this case two rounds of CHARM are applied. Unfortunately, applying CHARM in this manner does not guarantee the exact set of upper neighbors in the α-concept lattice; rather, a superset of the upper neighbors may in fact be enumerated [34]. On the other hand, Theorem 4.2 guarantees that the top ranked concepts in the generated set are upper neighbors of (G, M').

5.3 Algorithm Complexity and Analysis As illustrated above, the main computational burden of QBBC revolves around identifying the initial candidate set of α-concepts, followed by computing the neighbors of the top candidate α-concept. We analyze the complexity of enumerating candidate α-concepts from the point of view of enumerating itemsets in a binary context. For a binary context $\mathbb{K}^b$ and non-binary context $\mathbb{K}$, let $\mathcal{B}^b$ and $\mathcal{B}$ be the sets of all itemsets and α-concepts respectfully. By definition of an α-concept, we have that $|\mathcal{B}| \leq |\mathcal{B}|$. This follows by the fact that in order for a subspace to qualify as an α-concept; it must first satisfy the definition of an itemset and then obey the coherence criterion. Hence we may analyze the worst case running-time complexity in terms of itemset mining. Extending the analysis in [1] to CHARM, it is straightforward that enumerating itemsets is performed in time linear to the number of itemsets. Hence, the running time complexity of the initial candidate enumeration phase is $O(|\mathcal{B}|)$. In the worst case $|\mathcal{B}| \in O(2^{min(|\mathbf{G}|,|\mathbf{M}|)})$; however, typically in sparse data $|\mathcal{B}| \in O(min(|\mathbf{G}|, |\mathbf{M}|)^2)$ [17]. Moreover, in our case we can expect the query set to be orders of magnitude smaller than $\min(|\mathbf{G}|, |\mathbf{M}|)$. Hence we may re-write this initial phase complexity for sparse data as $O(|M^q|^2)$ where M^q is the query set. Computing the upper neighbors of an α-concept (G, M) utilizing the CHARM-like approach requires computing the $\sqcap$ operator over the set $\mathbf{M} \setminus M$. By maintaining the maximum and minimum values of each row or column in the search, the $\sqcap$ operator reduces to set intersection and can be computed in linear time. Therefore the complexity of the entire operation is $O(|\mathbf{M}|^2)$. In practice, if the set intersection of all pairs in the dataset is pre-computed utilizing an upper triangular matrix method, the complexity of computing neighbors can be greatly reduced.

6 Experimental Results

6.1 Datasets and evaluation criteria Six real-world datasets were used in our experimental study and are listed in Figure 6. The first three datasets came from the large 20Newsgroups dataset [4], $papers$ [31] is a subset of the DBLP database linking authors with paper titles, and $emap$ [26] and $hughes$ [15] are both microarray datasets.



(a) MSE scores for mer

(b) MSE scores for $emap$

(c) MSE scores for $hughes$

(d) Seed set MSE vs bicluster MSE in mer

(e) Seed set MSE vs bicluster MSE in $emap$

(f) Seed set MSE vs bicluster MSE in $hughes$

Figure 7: MSE distributions of query-based biclusters with QBBC and QDB

mer represents the news feeds from the Middle East politics and Religion forum, $allpc$ is a combination of all news feeds pertaining to computers, and $allsci$ is an aggregation of documents associated with science. In all text-based datasets, stop words were removed and TF-IDF weights were computed. For each dataset, two categories of seed sets were constructed. The first category of seed sets was manually assembled and generally contained 2-5 objects per query; *these objects were determined to be coherent*. For example, in mer query terms such as $\{israel, palestine\}$ formed a query while in $papers$ the terms $\{query, optimization\}$ were utilized. For $emap$ and $hughes$, query genes were assembled by consulting the Biological Process hierarchy of the Gene Ontology [14]. Sets that were annotated by the same functional class were retained for the query. The second category

of seed sets consisted of randomly selecting objects from each dataset; typically 10-50 objects were selected per query. A total of 10 manually created and 50 random queries were generated for each dataset. Evaluation of our experimental results was based on three criterion:

1. Evaluation using mean square error to measure the cluster quality.
2. Average purity of biclusters using class labels available in most of the datasets.
3. Visual assessment of cluster quality.

For comparison, we used the R implementation of QDB, available at http://homes.esat.kuleuven.be/~kmarchal, with the default parameter settings. The QBBC algorithm was implemented in C++ with both the α-sigma (QBBC-alpha) and maximum spacing uniform estimator consistency functions (QBBC-max) and is available at http://faris-alqadah.heroku.com.The α parameter was set to 3 on both QBBC-sigma and QBBC-max and both methods were set to augment initial query clusters with the top 20 neighbors in the α-concept lattice. At the time of this writing, no implementation was available for ProBic. Although QDB is designed to handle datasets with missing values, when attempting to find query based biclusters in the sparse datasets of this study (all but $hughes$), no biclusters were ever produced. This may be due to the extreme sparsity levels affecting the probabilistic model that QDB is based upon. As a result, when executing QDB, the sparse datasets were filled in with randomly generated values or zeros. In effect, this created a background distribution from which QDB should discern actual biclusters.

6.2 Cluster quality The first evaluation criterion utilized was mean square error (MSE) as given by Cheng and Church [12]:

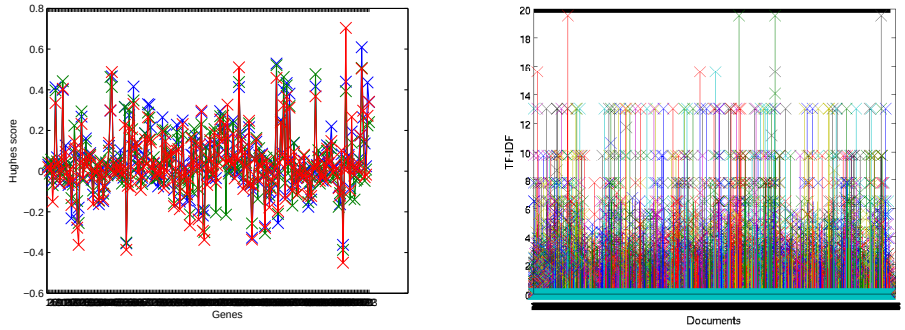
$$MSE(G, M) = \frac{1}{|G||M|} \sum_{g \in G, m \in M} (\mathbf{K}[g, m] - \mu_{g,M} - \mu_{G,m} + \mu_{G,M})^2$$

where $\mu_{g,M}$ is the mean of the g^{th} row under M columns, $\mu_{G,m}$ is the mean of the m^{th} column respectively under the G rows and $\mu_{G,M}$ is the overall mean of the subspace. Under this formulation the minimum value of MSE is 0 when all values in the subspace are equal. To accommodate sparse data, missing or non-edge values were either ignored or computed as usual with the missing values filled by the generated background distribution; the best results are reported here. Average purity of a set of biclusters $\mathtt{C} = \{(G_1, M_1), \ldots, (G_n, M_n)\}$ and set of class labels $\mathtt{L} = \{L_1, \ldots, L_m\}$ is defined as

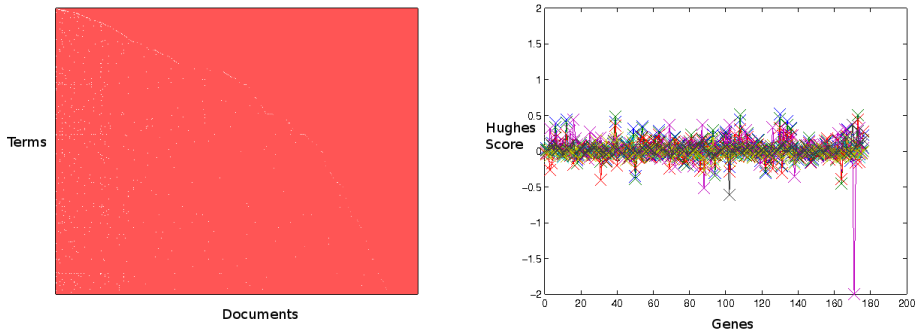
$$\frac{1}{|\mathtt{C}|} \sum_k \max_j \frac{|M_k \cap L_j|}{|M_k|}$$

In other words, each cluster is assigned to the class which is most frequent in the cluster and the purity measure is computed as the precision of the cluster with respect to this class; the average of the purity scores is then computed.

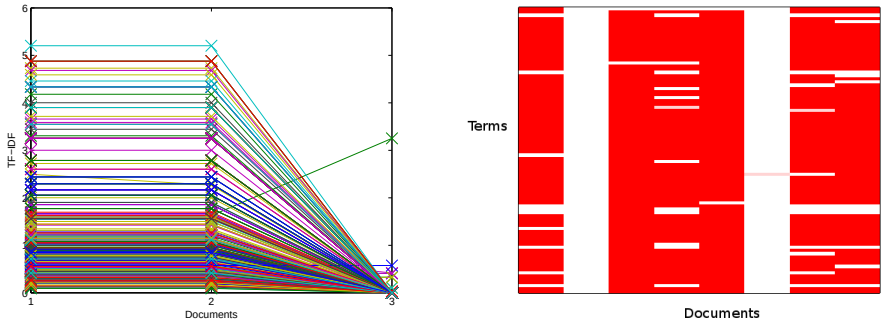
Figures 7(a)-7(c) displays the distribution of MSE scores for biclusters mined with both manually created and randomly generated query sets. In sparse data (Figures 7(a)-7(c)) the QDB cluster MSE scores were strikingly larger than those of both QBBC methods. This trend is explained by the fact that QDB tended to produce very large clusters even when a small seed set was input. For example, a seed set in mer contained only two terms, yet QDB returned a cluster containing 10,256 words and 1,600 documents. Clearly, the extreme sparsity levels of these datasets confound the QDB algorithm and it is unable to discriminate between the background distribution and high quality query centered biclusters. These facts **illustrate the clear advantage of** QBBC **in sparse high-dimensional data**. Nonetheless, both methods have similar performance on standard non-sparse microarray data



(a) QDB cluster in $hughes$ with manually selected query (b) QDB cluster in mer with manually selected query



(c) QDB cluster in $allpc$ with manually selected query (d) QBBC cluster in $hughes$ with manually selected query



(e) QBBC cluster in mer with manually selected query (f) QBBC cluster in $allpc$ with manually selected query

Figure 8: Sample cluster visuals

(Figure 7(c)). The lack of variance in MSE scores of clusters produced by `QBBC-max` in $hughes$ was attributed to the uniform distribution assumption. All queries resulted in similar biclusters indicates that `QBBC-max` *consistency function is more suited for sparse data*.

Figures 7(d)-7(f) display the relationship between MSE scores of the initial query sets and the final biclusters. The score of the query set was computed as the MSE of the subspace encompassing the query set in conjunction with all the rows (or columns) of the dataset. Random seeds that lead to no cluster being enumerated are not displayed. Figures 7(d)-7(f) depicts that both `QBBC` **algorithms' bicluster scores tend to be lower than the query set while** `QDB` **scores are higher**. As expected, the difference in seed scores and bicluster scores are much closer for manually created query sets. Interestingly, for random query-sets, the `QBBC` scores tended to be orders

Dataset	Seed set	bicluster
$papers$	**query optimization**	**databases semantic models based analysis dynamic evaluation systems information algorithms query design xml database distributed processing multiple queries efficient relational optimization temporal**
$allsci$	**cryptography hacker checksum algorithm cipher**	**algorithm cipher cryptography states rsa signed plaintext freedom scheme text number exists men recover selected application create united versions comments archive included documentation approved attempt licensed internet broad claimed recommended newsgroups**
mer	**israel palestine**	**arab palestine israelis israel israeli opposition zionist ground zionism international peace problem state wrote meant states human feel necessity statements creation guess occupation statement forms disregard refers minister racism fully intervention**

(a) Sample seed sets and corresponding bicluster

Dataset	Algorithm	Avg. Purity	Avg. num documents
mer	QBBC-sigma	**0.89**	28.125
	QBBC-max	**0.89**	38.75
	QDB	0.47	602.4
$allpc$	QBBC-sigma	**0.60**	51.4
	QBBC-max	0.54	66.2
	QDB	0.24	838
$allsci$	QBBC-sigma	**0.75**	19.4
	QBBC-max	**0.75**	26
	QDB	0.25	804.25
$papers$	QBBC-sigma	**0.72**	102.8
	QBBC-max	**0.72**	103.2
	QDB	0.31	1021.2

(b) Cluster purity

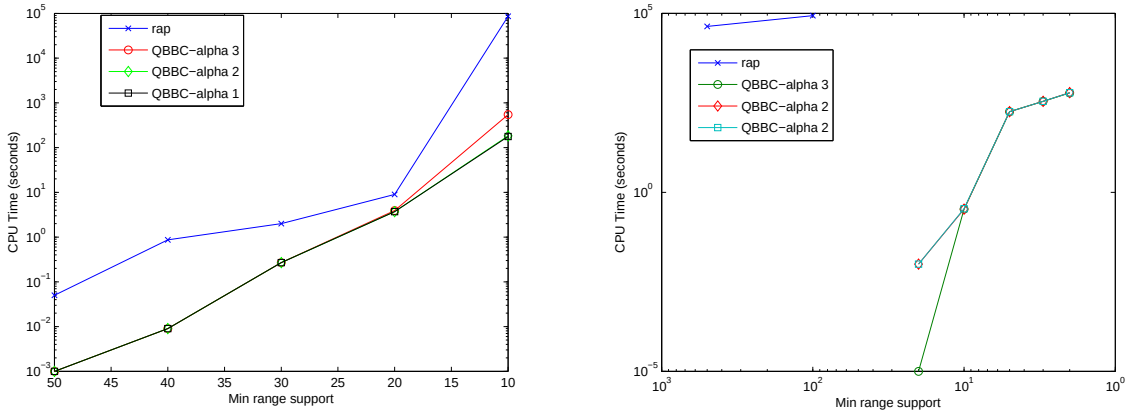
Figure 9: Sample clusters and precision-recall of clusters

of magnitude lower than those of the initial query set. In most cases, QBBC broke down the random query and produced several biclusters that were consistent in a small subspace of the initial query set, while QDB failed to do this. Once again, in all sparse datasets the performance of the QBBC algorithms is clearly superior to QDB while the results are comparable in hughes dataset.

Sample clusters mined by algorithms are depicted in Figures 8 and 9. Visually inspecting these Figures, biclusters mined by QDB and QBBC (Figures 8(a) and 8(d)) appear to be of similar quality with QBBC having less variance. On the other hand, once again, biclusters produced by QDB in sparse data (Figures 8(b) and 8(c)) are clearly less informative compared to those mined by QBBC (Figures 8(e) and 8(f)). These images manifest the ability of QBBC to filter out the background distribution and zoom in on query-centered biclusters in both sparse and dense datasets. Figure 9(a) illustrates a sampling of manually created query sets and the resulting word clusters. In general, the resulting biclusters contained most of the original query set and greatly expanded the cluster to include mostly pertinent terms with a few noisy terms. The most coherent bicluster came from $papers$; an argument can be made that all the terms in this bicluster are relevant, while the entire seed set was preserved. This was expected as $papers$ is extremely sparse and only contains paper titles. In the case of $allsci$, a much noisier dataset than $papers$, the terms $hacker$ and $checksum$ were dropped while several noisy terms were introduced.

Finally, the average purity of all biclusters resulting from manual query sets are presented in figure 9(b). Due to the inability of QDB to scale to the three larger datasets, $papers$, $allpc$, and $allsci$, the average purity scores were computed by repeating experiments on small subsets of these datasets with appropriate query terms and computing the average. Each experiment was repeated ten times on the sampled dataset with all algorithms and the average purity scores being reported. QBBC-sigma and QBBC-max produced clusters with approximately the same purity levels. Due to the large size of QBD clusters, the purity scores are near random. On the other hand, the QBBC scores were not induced by extreme cluster sizes as is demonstrated by the average number of

documents in each cluster.



(a) *hughes* (b) *papers*

Figure 10: Performance comparisons between QBBC and RAP algorithms.

6.3 Performance Tests Performance tests were conducted to evaluate the practical running time and scalability of the QBBC. All experiments were conducted on a 3.33 GHz Intel I7 quad core CPU with 16 GB of RAM. Due to its implementation in R, QDB scaled extremely poorly in the case of the three largest datasets and ran out of memory on a 16GB main memory machine in every instance. On the other hand, the memory requirements of QBBC never exceeded 10 MB. Moreover, running times on these datasets exceeded an hour even with query sets containing fewer than 5 objects. In order to conduct a more fair comparison, and evaluate the true effectiveness of the pruning rules introduced, we re-implemented QBBC specifying it to enumerate all α-concepts in an entire dataset; we compared the running times to the state-of-the-art pattern-based biclustering algorithm RAP on all datasets. An implementation of RAP in C++ was downloaded from http://vk.cs.umn.edu/gaurav/rap/. The performance tests are displayed in Figure 10. The min range support is a user defined parameter instructing the algorithms to only retain clusters with a minimum degree of supporting sets. Due to min range support also being anti-monotone it can also be utilized to prune the search space. As shown by these results, the additional pruning steps introduced by Theorem 5.1 clearly result in much improved performance. In non-sparse data (Figure 10(a)), we observed well over three orders of magnitude speed up at the lowest minimum support levels. On the largest sparse data set, *papers*, rap was unable to complete within 48 hours at the minimum support levels specified for QBBC (Figure 10(b)) while QBBC completed in under 700 seconds at the lowest support levels.

7 Conclusion

In this paper, a novel query-based biclustering algorithm, QBBC, was developed. It was shown that statistical dispersion measures that are order-preserving induce an ordering on the set of biclusters in the data; in turn this ordering is exploited to mine similar biclusters centered around a query seed set. Making use of an original operator, range intersection, it was further shown that the seminal CHARM algorithm for mining closed itemsets is generalizable to the computational framework for mining query-based biclustering. Experimental results unveiled that in high dimensional sparse data, QBBC has a clear advantage over the current state-of-the-art query-based biclustering methods while similar performance was observed on standard microarray datasets. Moreover, we illustrated that the pruning measures introduced in the QBBC algorithm resulted in significant computational savings compared to both QDB and RAP. Shortcomings of QBBC include the inability to compute exact neighbors

of a bicluster in the concept lattice. We believe that this fact leads to significantly higher computational cost especially in dense datasets. Moreover, no data-driven criterion was derived for determining if and when neighboring biclusters should be merged to enhance the seed bicluster. Finally, additional order-preserving dispersion methods and anti-monotone consistency functions should developed in future in order to accommodate some specific application demands.

Acknowledgements

This work is supported in part by the US National Science Foundation grants IIS-1242304 and IIS-1231742.

References

[1] F. Alqadah and R. Bhatnagar. An effective algorithm for mining 3-clusters in vertically partitioned data. In *CIKM '08*, 2008.

[2] F. Alqadah and R. Bhatnagar. Discovering substantial distinctions among incremental bi-clusters. In *SDM'09*, 2009.

[3] R. Anand and C. K. Reddy. Graph-based clustering with constraints. In *Advances in Knowledge Discovery and Data Mining - 15th Pacific-Asia Conference, PAKDD 2011, Shenzhen, China, May 24-27, 2011, Proceedings, Part II*, pages 51–62, 2011.

[4] A. Asuncion and D. Newman. UCI machine learning repository, 2007.

[5] S. Basu, A. Banerjee, and R. J. Mooney. Semi-supervised clustering by seeding. In *ICML '02*, pages 27–34, 2002.

[6] S. Basu, A. Banerjee, and R. J. Mooney. Active semi-supervision for pairwise constrained clustering. In *Proceedings of the Fourth SIAM International Conference on Data Mining, Lake Buena Vista, Florida, USA, April 22-24, 2004*, 2004.

[7] S. Basu, I. Davison, and K. Wagstaff. *Constrained Clustering: Advances in Algorithms and Theory*. Chapman & Hall / CRC, 2008.

[8] S. Bergmann, J. Ihmels, and N. Barkai. Iterative signature algorithm for the analysis of large-scale gene expression data. *Phys. Rev. E*, 67(3):031902, Mar 2003.

[9] A. Berry, J.-P. Bordat, and A. Sigayret. A local approach to concept generation. *Annals of Mathematics and Artificial Intelligence*, 49:117–136, 2007.

[10] J. Besson, C. Robardet, L. D. Raedt, and J.-F. Boulicaut. Mining bi-sets in numerical data. In *Knowledge Discovery in Inductive Databases, 5th International Workshop, KDID 2006, Berlin, Germany, September 18, 2006, Revised Selected and Invited Papers*, pages 11–23, 2006.

[11] S. Busygin, O. Prokopyev, and P. M. Pardalos. Biclustering in data mining. *Comput. Oper. Res.*, 35(9):2964–2987, 2008.

[12] Y. Cheng and G. Church. Biclustering of expression data. In *8th International Conference on Intelligent Systems for Molecular Biology*, 2000.

[13] T. Dhollander, Q. Sheng, K. Lemmens, B. D. Moor, K. Marchal, and Y. Moreau. Query-driven module discovery in microarray data. *Bioinformatics*, pages 2573–2580, 2007.

[14] M. A. et al. Gene ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25–29, 2000.

[15] T. H. et. al. Functional discovery via a compendium of expression profile. *Cell*, 102(1):109–126, 2000.

[16] B. Gamter and R. Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer-Verlag, Berlin, 1999.

[17] F. Geerts, B. Goethals, and J. V. D. Bussche. Tight upper bounds on the number of candidate patterns. *ACM Trans. Database Syst.*, 30:333–363, 2005.

[18] M. A. Hibbs, D. C. Hess, C. L. Myers, C. Huttenhower, K. Li, and O. G. Troyanskaya. Exploring the functional landscape of gene expression: directed search of large microarray compendia. *Bioinformatics*, 23(20):2692–2699, 2007.

[19] Z. Hu and R. Bhatnagar. Algorithm for discovering low-variance 3-clusters from real-valued datasets. In *ICDM '10*, pages 236–245, 2010.

[20] S. C. Madeira and A. L. Oliveira. Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1 (1):24–45, 2004.

[21] M. O. Mohammed J. Zaki. Theoretical foundations of association rules. *3rd SIGMOD'98 Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD)*, 1998.

[22] O. Odibat, C. K. Reddy, and C. N. Giroux. Differential biclustering for gene expression analysis. In *BCB '10*, 2010.
[23] A. B. Owen, J. Stuart, K. Mach, A. M. Villeneuve, and S. Kim. A gene recommender algorithm to identify coexpressed genes in c. elegans. *Genome Research*, 13(8):1828–1837, 2003.
[24] G. Pandey, G. Atluri, M. Steinbach, C. L. Myers, and V. Kumar. An association analysis approach to biclustering. In *KDD '09*, pages 677–686, 2009.
[25] R. G. Pensa and B. Jean-Francois. Constrain co-clustering of gene expression data. In *SDM '08*, 2008.
[26] M. Schuldiner, S. Collins, J. Weissman, and N. Krogan. Quantitative genetic analysis in saccharomyces cerevisiae using epistatic miniarray profiles (e-maps) and its application to chromatin functions. *Methods*, 40(4):344–352, 2006.
[27] X. Shi, W. Fan, and P. S. Yu. Efficient semi-supervised spectral co-clustering with constraints. In *ICDM 2010, The 10th IEEE International Conference on Data Mining, Sydney, Australia, 14-17 December 2010*, pages 1043–1048, 2010.
[28] R. D. Smet and K. Marchal. An ensemble biclustering approach for querying gene expression compendia with experimental lists. *Bioinformatics*, 27(14):1948–1956, 2011.
[29] Y. Song, S. Pan, S. Liu, F. Wei, M. X. Zhou, and W. Qian. Constrained coclustering for textual documents. In *AAAI '10*, 2010.
[30] Y. Song, S. Pan, S. Liu, F. Wei, M. X. Zhou, and W. Qian. Constrained text co-clustering with supervised and unsupervised constraints. *IEEE Transactions on Knowledge and Data Engineering*, 99(PrePrints), 2012.
[31] Y. Sun, Y. Yu, and J. Han. Ranking-based clustering of heterogeneous information networks with star network schema. In *KDD'09*, 2009.
[32] P. Wang, C. Domeniconi, and J. Hu. Using wikipedia for co-clustering based cross-domain text classification. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy*, pages 1085–1090, 2008.
[33] C.-J. Wu and S. Kasif. Gems: a web server for biclustering analysis of expression data. *Nucleic Acids Research*, 33(suppl 2):W596–W599, 2005.
[34] M. J. Zaki and C.-J. Hsiao. Efficient algorithms for mining closed itemsets and their lattice structure. *IEEE Transactions on Knowledge and Data Engineering*, 17 (4):462–478, 2005.
[35] H. Zhao, L. Cloots, T. Van den Bulcke, Y. Wu, R. De Smet, V. Storms, P. Meysman, K. Engelen, and K. Marchal. Query-based biclustering of gene expression data using probabilistic relational models. *BMC Bioinformatics*, 12(Suppl 1):S37, 2011.

Appendix
Proof of Theorem 4.1

Proof. Let (G_1, M_1) and (G_2, M_2) be α-concepts of $\mathbb{K}$, such that $M_2 \supseteq M_1$. Assume that $G_2 \not\subseteq G_1$. Two cases arise:

1. $G_2 \supset G_1$. In this case, $\exists g \in G_2 \setminus G_1$ s.t.

$$\begin{aligned} d(K[g, M_2]) &\leq f(\mathbb{K}, g, M_2, \alpha) \\ d(K[g, M_2]) &\leq f(\mathbb{K}, g, M_1, \alpha) \end{aligned}$$

   however, this implies that (G_1, M_1) is not an α-concept which contradicts the original assumption.

2. $G_2 \cap G_1 = \emptyset \wedge |G_2| > 0$. Same argument as above by the fact that $M_2 \supseteq M_1$.

Hence, we conclude that $G_2 \subseteq G_1$ implying that $(G_2, M_2) \geqslant (G_1, M_1)$.

Proof of Theorem 4.2

Proof. By definition, for any $(G_3, M_3) \notin \Upsilon((G_1, M_1)) \wedge (G_3, M_3) \geqslant (G_1, M_1)$ there exists $(G_2, M_2) \in \Upsilon((G_1, M_1))$ such that $(G_3, M_3) \geqslant$

(G_2, M_2). This implies that :

$$dist((G_1, M_1), (G_3, M_3)) =$$

$$\frac{1}{|G_1 \setminus G_3|}\left(\sum_{g \in G_1 \setminus G_2} \frac{d(\mathbf{K}[g, M_3 \setminus M_1]) \times s_g}{d(\mathbf{K}[g, M_3])} + \sum_{g \in G_2 \setminus G_3} \frac{d(\mathbf{K}[g, M_3 \setminus M_1]) \times s_g}{d(\mathbf{K}[g, M_3])} \right) \tag{.1}$$

$$\sum_{g \in G_1 \setminus G_2} \frac{d(\mathbf{K}[g, M_3 \setminus M_1])}{d(\mathbf{K}[g, M_3])} > \sum_{g \in G_1 \setminus G_2} \frac{d(\mathbf{K}[g, M_2 \setminus M_1])}{d(\mathbf{K}[g, M_3])} \tag{.2}$$

$$\sum_{g \in G_2 \setminus G_3} \frac{d(\mathbf{K}[g, M_3 \setminus M_1])}{d(\mathbf{K}[g, M_3])} > \sum_{g \in G_1 \setminus G_2} \frac{d(\mathbf{K}[g, M_2 \setminus M_1])}{d(\mathbf{K}[g, M_3])} \tag{.3}$$

$$(1 + |(\Gamma(g) \cap M_3) \setminus M_1|) > (1 + |(\Gamma(g) \cap M_2) \setminus M_1|) \tag{.4}$$

where inequality (.2) follows from the order-preserving property of d, inequality (.3) follows from the order-preserving property of d, the anti-monotone property of f and the definition of a concept. Inequality (.4) follows by the properties of set difference. Hence, by the fact that $|G_1 \setminus G_3| = |G_1 \setminus G_2| + |G_2 \setminus G_3|$ and inequalities (.2), (.3) and (.4).

$$dist((G_1, M_1), (G_3, M_3)) > dist((G_1, M_1), (G_2, M_2))$$

Proof of Theorem 5.1

Proof. By the triangle inequality and definition of consistency we have

$$d(\mathbf{K}[s, PP_l \cup PP_r \cup PP_x]) \leq$$
$$d(\mathbf{K}[s, PP_l \cup PP_r]) + d(\mathbf{K}[s, PP_r \cup PP_x]) \tag{.5}$$
$$d(\mathbf{K}[s, PP_l \cup PP_r \cup PP_x]) \leq f(\mathbb{K}, s, PP_l \cup PP_r \cup PP_x, \alpha) \tag{.6}$$

for any prefix node PP_x and any $s \in S_l \cap S_r \cap S_x$, and d is the range dispersion statistic. Considering each case:

1. $|SS| = |S_l| \wedge |SS| = |S_r|$. In this case $SS = S_l$ and $SS = S_r$ and the subspace $(SS, PP_l \cup PP_r)$ is consistent by definition of range intersection. Hence to ensure closure, PP_l should be replaced by $PP_l \cup PP_r$. Moreover, by equation (.6) and the properties of set intersection

$$\begin{aligned} S_l \sqcap S_x &= S_r \sqcap S_x \\ &= S_l \cup S_r \sqcap S_x \\ &\quad implying \\ \psi_f^\alpha(PP_l \cup PP_x) &= \psi_f^\alpha(PP_r \cup PP_x) \\ &= \psi_f^\alpha(PP_l \cup PP_r \cup PP_x) \end{aligned}$$

   for any prefix node PP_x under the same branch.

2. $|SS| = |S_r|$. In this case $S_r \supset S_l$ and the subspace $(SS, PP_l \cup PP_r)$ is consistent by definition of range intersection. Hence, as shown above, it is possible to combine PP_l and PP_r into a single node. On the other hand, the formation of a subspace involving PP_r but not PP_l is possible hence both nodes $PP_l \cup PP_r$ and PP_r must be maintained.

3. $|SS| = |S_l|$. In this case $S_l \supset S_r$ and the subspace $(SS, PP_l \cup PP_r)$ is consistent by definition of range intersection. By the definition of closure and as shown above every consistent subspace involving PP_r will also involve PP_l, therefore PP_r and its children maybe pruned. On the other hand, the formation of a subspace involving PP_l but not PP_r is possible hence both nodes $PP_l \cup PP_r$ and PP_l must be maintained.

4. No pruning possible, hence new nodes must be formed.