Clustering with Complex Constraints — Algorithms and Applications

**Weifeng Zhi,[⋆] Xiang Wang,[⋆] Buyue Qian,[⋆]
Patrick Butler,[†] Naren Ramakrishnan,[†] Ian Davidson[⋆]**
[⋆]Dept. of Computer Science, University of California, Davis, CA 95616
{wzhi, xiang, byqian, indavidson}@ucdavis.edu
[†]Dept. of Computer Science, Virginia Tech, Blacksburg, VA 24061
{pabutler, naren}@vt.edu

Abstract

Clustering with constraints is an important and developing area. However, most work is confined to conjunctions of simple together and apart constraints which limit their usability. In this paper, we propose a new formulation of constrained clustering that is able to incorporate not only existing types of constraints but also more complex logical combinations beyond conjunctions. We first show how any statement in conjunctive normal form (CNF) can be represented as a linear inequality. Since existing clustering formulations such as spectral clustering cannot easily incorporate these linear inequalities, we propose a quadratic programming (QP) clustering formulation to accommodate them. This new formulation allows us to have much more complex guidance in clustering. We demonstrate the effectiveness of our approach in two applications on text and personal information management. We also compare our algorithm against existing constrained spectral clustering algorithm to show its efficiency in computational time.

Introduction

A recent trend in clustering is to incorporate user guidance, or side information, into the clustering process in the form of constraints. Typical user information includes the MUST-LINK (the pair of instances must be assigned into the same cluster) and CANNOT-LINK (the pair of instances cannot be assigned into the same cluster). These types of constraints have been added to many different clustering algorithms, such as K-means clustering (Wagstaff et al. 2001), EM (Basu, Davidson, and Wagstaff 2008) and spectral clustering (Kamvar, Klein, and Manning 2003; Coleman, Saunderson, and Wirth 2008; Wang and Davidson 2010). However, in all of this work, the constraints are typically simple conjunctions of MUST-LINK (ML) and CANNOT-LINK (CL) constraints. In an edited book on the topic (Basu, Davidson, and Wagstaff 2008), all of the constraints were of this form. To our knowledge in the constrained clustering literature combining any type of constraints using a logical language has never been considered but offers the promise of more complex expression of knowledge. This is most useful for including conditional guidance, guidance encoded as complex data structures, such as hierarchies, and when dealing with heterogenous collections of objects the ability to encode relationships between different object types. We specifically tackle this in our experiments by exploring clustering images, their tags and locations, and also documents where a known hierarchical structure exists.

Incorporating extensive amounts of logical combinations of constraints is challenging since many clustering algorithms have mathematical programming formulations that can not be easily extended to incorporate large amounts of guidance. For example, previous work (Wang and Davidson 2010) on constrained spectral clustering could only incorporate a single set of guidance. In this paper, we show how to incorporate logical combinations of constraints as a collection of linear inequalities and equalities so that they can be added to any continuous optimization formulations. As compared to existing work, the quadratic programming formulation we choose is not only capable of encoding more complex constraints but also far more efficient compared to existing constrained spectral clustering algorithms.

Our contributions in this paper are:

- We incorporate logical combinations of constraints into clustering for the first time.
- We show how to express logical combinations of constraints as linear equalities and inequalities so that they can be incorporated into various mathematical programming formulations for clustering.
- We present one such formulation using quadratic programming that retains the spectral clustering objective.
- We show two innovative applications for using complex combinations of constraints, namely encoding user preferences for Personal Information Management and encoding hierarchies in text documents.
- We show that our algorithm is far more efficient (up to 1000 times faster) than the state-of-the-art constrained spectral clustering technique in terms of computational time when using conjunctions of constraints. It is approximately 10 times faster when using both conjunctions and disjunctions of constraints.

Related Work

There are three main bodies of related work. *Quadratic Programming Formulations of Clustering.* There is, to our knowledge, no general purpose formulation of quadratic programming for clustering, though there exist specific formulations for topics such as time series clustering (Chaovalitwongse 2008). The main reason for this is that the spectral clustering objective function is inherently quadratic and the given spectral clustering problem reduces to an eigenvalue problem which is easily solved in linear time for sparse matrices. Hence, the only advantage of the QP formulation has over the spectral formulation is that it allows specifying many constraints which has not been needed up to this point.

With respect to adding relational information into clustering there are two main bodies of work: relational clustering and constrained clustering. *Relational Clustering.* In this field (Long, Zhang, and Yu 2010) the data to cluster is only described by relational information. This is not the topic of this paper since in our setting we have objects (and their description) and in addition relational information on the points. *Constrained Clustering.* This work is closest to our own because it allows clustering data using both the object descriptions and relations between the points as additional side information. The most flexible form of this work (Wang and Davidson 2010) allows degrees of belief in constraints (as our work does) which we now describe. Given a graph $\mathcal{G} = (V, E)$ with N nodes and its affinity matrix W, the degree matrix D is diagonal and $D_{ii} = \sum_{j=1}^{N} W_{ij}$. Let $L = D - W$ be the unnormalized graph Laplacian and let $Q = (Q_{ij})$ be a $N \times N$ constrained matrix, which incorporates pairwise constraints: MUST-LINK (+1) and CANNOT-LINK (-1). Then finding a constrained cut on the graph so that proportional to β of the constraints are satisfied involves optimizing the function below:

$$\arg\min_{v \in \mathbb{R}^n} v^T \bar{L} v \quad \text{s.t. } v^T \bar{Q} v \geq \beta,\ v^T v = 1, v \perp D^{1/2} e, \tag{1}$$

See (Wang and Davidson 2010) for the solution of (1) which involves solving a generalized eigenvalue problem. As the authors themselves mention, the complexity of their constrained spectral clustering formulation **is much greater than** regular spectral clustering since one cannot just return the top k eigenvectors. Rather all N eigenvectors are found and the one that most satisfies the constraints is chosen. This result is significant since though our quadratic programming formulation of clustering is slower than regular spectral clustering it is significantly faster than constrained spectral clustering as we shall later see.

Quadratic Programming Formulation of Clustering with Linear Constraints

The above formulation in equation (1) has the limitation that it cannot be generalized to many constraints since the notion of a generalized eigenvalue problem is limited to two matrices. In this section, we present our novel QP formulation that can have many constraints but has the challenge of preventing a trivial solution since we cannot use a quadratic constraint as in equation (1). We present the formulation for a 2-way partition first but it can be trivially extended to a multi-way partition. Our approach has the benefit of retaining the spectral clustering objective function whilst allowing complex constraints.

In the above spectral formulation, the nonlinear constraint $v^T v = 1$ normalizes and prevents a trivial solution by requiring the solution to lie on a hypersphere. However, such constraints are not allowed in QP formulations. To replace it, we incorporate the constraint $e_j^T v = 1$ for some j, where e_j represents the vector with j-th entry being one and other entries being zeros. This changes the requirement that the solution not lie on the hypersphere but rather on a hyperplane which does not run through the origin. We can choose any j here since $e_j^T v = 1$ means that node j is in one cluster. For the objective function, we still use $v^T \bar{L} v$, which represents the cost of the cut. Then we have the following QP problem:

$$\arg\min_{v \in \mathbb{R}^N} v^T \bar{L} v \quad \text{s.t. } e^T D^{1/2} v = 0,\ e_j^T v = 1, \text{for some } j. \tag{2}$$

The constraint $e^T D^{1/2} v = 0$ is to balance the cut of the graph and the constraint $e_j^T v = 1$ for some j is to prevent the trivial cut of the graph. Both of these constraints are linear equalities. We can easily solve this problem using Projective Preconditioned Conjugate Gradient method (PPCG) in MATLAB. Let $v^{(1)}$ be the optimal solution for equation (2). Then $D^{-1/2} v^{(1)}$ is the relaxed cluster indicator vector for 2-way partition.

K-way partition

In many applications, we are interested in not only a 2-way partition, but also a K-way partition for $K > 2$. To achieve this we need to obtain K vectors indicating cluster membership and cluster them as is the practice with spectral clustering. This can be achieved in an iterative manner by adding linear constraints so that the second and subsequent indicator vectors are orthogonal to those given. Thus after obtaining the first indicator vector $v^{(1)}$ we solve for the additional vector v as follows:

$$\arg\min_{v \in \mathbb{R}^N} v^T \bar{L} v \quad \text{s.t. } e^T D^{1/2} v = 0,\ e_j^T v = 1,\ v^T v^{(1)} = 0. \tag{3}$$

Converting Logical Statements into Linear Constraints

Conversion of the propositional logic constraints into linear inequalities has been considered in integer linear programming, but incorporating and using propositional logic constraints into clustering is novel. In this section, we show how to convert propositional logic statements into linear (in)equalities which can be easily added to a QP as linear constraints. We call these constraints *logical constraints*.

Literals for logic statements usually take on the binary values of 1 (true) or 0 (false). However, for mathematical convenience the literals will take on values of $+1$ (true) and -1 (false). Here the statement $v_i = \texttt{TRUE}$ means that it is true that instance i is in cluster 1.

Observation 1 *Let $v_i = -1$ when the i-th literal is false and $v_j = 1$ when the j-th literal is true. The $ML(i,j)$ and $CL(i,j)$ constraints can be encoded as the linear equalities $v_i = v_j$ and $v_i = -v_j$ respectively with ϵ-degree of belief variations encoded as the inequalities $|v_i - v_j| \le \epsilon$ and $|v_i + v_j| \le \epsilon$.*

We now go onto show how this encoding scheme with linear inequalities can be used to represent any statement in propositional logic. We begin by showing how a disjunction and conjunction can be represented as an inequality in Lemma 1 and 2 respectively and then how any statement can be represented in conjunctive normal form in Theorem 1.

Lemma 1 *Any disjunction of literals $v_1 \dots v_m$ can be represented by a linear inequality.*

Proof. Let $v_i = -1$ when the i-th literal is false and $v_j = 1$ when the j-th literal is true. Then the disjunction $v_1 \vee v_2 \vee \dots \vee v_m$ is represented by the predicate $\sum_{i=1}^{m} v_i \ge (-m+2)$. $\square$

Lemma 2 *Any conjunction of literals $v_1 \dots v_m$ can be represented by a linear equality.*

Proof. Let $v_i = -1$ when the i-th literal is false and $v_j = 1$ when the j-th literal is true. Then the conjunction $v_1 \wedge v_2 \wedge \dots \wedge v_m$ is represented by the predicate $\sum_{i=1}^{m} v_i = m$. $\square$

It is well known that any logical statement in propositional logic can be converted into conjunctive normal form (CNF). Therefore, if we can show that any statement in CNF can be defined as a set of inequalities we can then represent any constraint in propositional logic in our framework.

Theorem 1 *Given a set of literals $v_1 \dots v_m$, any set of clauses using those literals in conjunctive normal form can be represented by a linear inequality. Let the CNF formulae be of the form: $\mathcal{X} = C_1 \wedge C_2 \wedge \dots \wedge C_r$ where the i-th clause is defined as $C_i = v_{i,1} \vee \dots \vee v_{i,|C_i|}$ where $v_{i,j}$* ***references*** *the j-th literal of the i-th clause.*

Proof. Combining Lemmas 1 and 2 the inequalities to represent $\mathcal{X} = C_1 \wedge C_2 \wedge \dots \wedge C_r$ is simply:

$$\left(\sum_{j=1}^{|C_i|} v_{i,j} \right) \ge (-|C_i| + 2) \text{ for } i = 1, \dots, r,$$

where $|C_i|$ is the number of literals in clause C_i and r is the number of clauses or logical statements. $\square$

The strength of our work is that it can encode many new types of constraints as shown in Table 1 and used later in our experiments.

Encoding User Preferences in Personal Information Management Data

In this section, we consider the clustering of personal information management data set (PIM). We believe this is an ideal application of our work to allow complex guidance since how the underlying data is organized is highly dependent on a complex set of user preferences. All data and code used to produce these results will be made freely available. In our initial experiments, we shall explore encoding relations where the ground truth is known, hence we can measure the performance at recovering these relations. In our later experiments, we encode relations which indicate a user's preference and, though there is no ground truth comparison, we do show the results are not only meaningful but would not have been produced if no guidance was provided.

Our data set consists of $n = 500$ images taken at $y = 99$ different locations (described by longitude, latitude) around Asheville, North Carolina and named with one or more of $z = 590$ possible tags such as Biltmore, Rosegarden, Holiday, etc. Clustering this data is equivalent to clustering a heterogeneous graph as others have done (Long et al. 2006) except we now can provide complex guidance. After we cluster the data, we hope that every image has its location and tags in the same cluster. The criterion for the success of the algorithms for PIM data set is as follows. Assume image i is `TakenAt` location ℓ and `TaggedWith` k_i tags. Then there are $k_i + 1$ pieces of side information for image i (1 location and k_i tags). After clustering, we consider the success rate for image i is $r_i = s_i/(k_i+1)$ if image i has $s_i \le (k_i + 1)$ pieces of side information in the same cluster as image i. The success rate of all n images is the mean of the success rates of all n images (success rate $= (\sum_{i=1}^{n} r_i)/n$).

Baseline Method: Augmented Laplacian. Both the spectral and QP formulations of clustering require a Laplacian matrix for **all** objects. Let W_I, W_L and W_T be the affinity matrices of the images, locations and tags information, respectively. We can construct a big affinity matrix for all objects, which has a block diagonal form.

$$\begin{pmatrix} W_I & 0 & 0 \\ 0 & W_L & 0 \\ 0 & 0 & W_T \end{pmatrix}. \tag{4}$$

Clustering this data will yield no useful information apart from clustering all images together, all tags together and all locations together. We need to encode side information into the formulation to cluster the PIM data set into a useful form.

The baseline comparison we shall use is to augment the affinity matrix in equation (4). If image i is `TakenAt` ℓ, set $W(i, n+\ell) = 1$. If image i is `TaggedWith` t, set $W(i, n+y+t) = 1$. This augments the affinity matrix W and we can construct an augmented unormalized Laplacian L from W, $L = D - W$. We use the standard normalized min-cut formulation (von Luxburg 2007) to solve this graph partition problem and report our results in the first line of experiments in Table 2.

QP Formulation. Our work uses our QP formulation and, instead of using the side/relational information to augment the affinity matrix, we add logical constraints to for-

Logic Statements	Use
$(v_{festival} \wedge v_{LexingtonAve}) \Rightarrow \neg v_{mountains}$	Encode conditional exclusion of a node from a cluster.
$(Level_1 \wedge \neg Level_2 \wedge \neg Level_3) \vee (\neg Level_1 \wedge Level_2 \wedge \neg Level_3) \vee (\neg Level_1 \wedge \neg Level_2 \wedge Level_3)$	Encode one level (1,2 or 3) of guidance contained in a hierarchy.
$\exists im_1, im_2, tag_1, \ldots, tag_m, l_1, l_2$: `TakenAt`$(im_1, l_1) \wedge$ `TakenAt`$(im_2, l_2) \wedge$ `TaggedWith`$(im_1, tag_1) \wedge$ `TaggedWith`$(im_2, tag_1) \wedge \ldots \wedge$ `TaggedWith`$(im_1, tag_m) \wedge$ `TaggedWith`(im_2, tag_m) $\Rightarrow CL(l_1, l_2)$	For clustering of heterogenous collection of images, locations and tags. If two images are taken at different locations with many similar tags, then do not place the locations together in the same cluster.

Table 1: Examples of types of complex constraints that can be encoded and are used in later experiments.

Success Rate		
Algorithms	2-way partition	3-way partition
Augment Laplacian	0.5950	0.8229
QP formulation	0.7285	0.9120

Table 2: Comparison of success rates between the QP formulation and augmenting the graph Laplacian using all available relational data.

mulate a QP problem. As mentioned earlier, our underlying language can represent relational information and here we can relate image and location objects and image and tag objects respectfully with the relations `TakenAt` and `TaggedWith`. We set the constraints $v_i \Leftrightarrow v_{n+\ell}$ and $v_i \Leftrightarrow v_{n+y+t}$ if image i is `TakenAt` ℓ and image i is `TaggedWith` t. We can make these relations "soft" by relaxing the constraint so that if image i is `TakenAt` ℓ, we add the constraint $|v_i - v_{n+\ell}| \leq \alpha$. If image i is `TaggedWith` t, we add the constraint $|v_i - v_{n+y+t}| \leq \alpha$ where $0 < \alpha < 1$ is a parameter. Adding these constraints, we have a QP problem as follows.

$$\begin{aligned} &\arg\min_{v} v^T \bar{L} v \qquad (5)\\ &\text{s.t. } e^T D^{1/2} v = 0, e_j^T v = 1, \text{ for some } j,\\ &\forall (i,\ell) \in \texttt{TakenAt}\ |v_i - v_{n+\ell}| \leq \alpha\\ &\forall (i,t) \in \texttt{TaggedWith}\ |v_i - v_{n+y+t}| \leq \alpha. \end{aligned}$$

Our experimental results ($\alpha = 0.25$) in Table 2 show that our approach is more successful at recovering the idealized ground truth than the baseline technique, due to lack of space we did not show the result that the results improve monotonically depending on the amount of side information provided.

Encoding User Preferences. Next we explore encoding relations for which no ground truth exists, but rather a user preference is given. Therefore, we can verify our algorithm's performance not by accuracy rather only by visualizing the resulting the clustering.

Diversity. One clear guidance the user can specify is spatial diversity to provide useful results. Consider if two superficially similar locations share many similar tags then they may be placed in the same location. We can prevent this by easily encoding a conditional CL constraint as shown in the last line of Table 1. Using this guidance to cluster our data, we see some sample images of each cluster shown in Figure 1 (a) and (b). If this guidance was not provided, then these images would all be put in the same cluster.

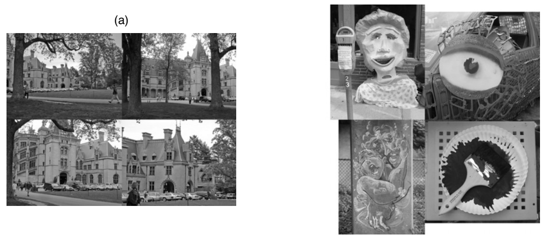

Figure 1: (a) and (b) are the sample images in two clusters when we use conditional guidance as shown in the last line in Table 1.

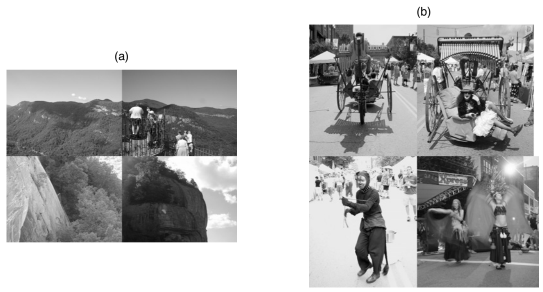

Figure 2: (a) and (b) are the sample images in two clusters when we add logical statement in the first row of Table 1.

Conditional Preference. Where the benefit of our work becomes evident is to encode more complex preferences. Consider in our data set the tags `mountains`, `festival` and `LexingtonAve`. We may know that there are Mountain Festivals and LexingtonAve Fesitvals and we do not want both to appear in the same cluster. We can encode this as $(v_{festival} \wedge v_{LexingtonAve}) \Rightarrow \neg v_{mountains}$, which is equivalent to $(\neg v_{festival}) \vee (\neg v_{LexingtonAve}) \vee (\neg v_{mountains})$ and can be encoded using Lemma 1.

We performed an experiment encoding this information with the result that the tag nodes (`festival`) (`LexingtonAve`) were both together in a cluster, and the tag node (`mountains`) in another. The images of the relevant clusters are shown in Figure 2 (a) and (b) and achieve our purpose. Such a clustering is not found if no guidance is provided.

Encoding Hierarchies in Newsgroup Data

In this section, we apply the QP formulation with logical constraints to the 20 Newsgroups data set (Lang 1995). The data set consists of 20,000 newsgroup documents from 20 different newsgroups. Each newsgroup corresponds to a different topic where some newsgroups are closely related sharing a common parent and others are not (Table 3).

It can clearly be seen that the hierarchy provides multiple sources of guidance. One source of guidance is to divide the data into six groups (Computers, Recreation, Sale, Science, Politics and Philosophy/Religion). Another is to divide the data into eleven groups using the next level in the hierarchy. Finally the third level of the hierarchy specifies twenty categories. If we had such a hierarchy as guidance, we would not wish to arbitrarily pick some level (as existing work must do) and constrain the clustering using it. Rather, we would like to encode all three levels and allow the algorithm to choose the one that is justified. This is precisely what our work allows. Consider three documents (D1,D2 and D3) belonging to: `rec.sports.basketball`, `rec.sports.baseball` and `rec.auto`. Then the constraints would be L3=*CL(D1,D2), CL(D1,D3), CL(D2,D3)* using the third level and L2=*ML(D1,D2), CL(D1,D3), CL(D2,D3)* using the second level and L1=*ML(D1,D2), ML(D1,D3), ML(D2,D3)* using the top level. The logical expression to specify use **exactly one of them** is not a simple disjunction but rather would be: $(L1 \wedge \neg L2 \wedge \neg L3) \vee (\neg L1 \wedge L2 \wedge \neg L3) \vee (\neg L1 \wedge \neg L2 \wedge L3)$.

We took increasing amounts of documents and used only their labels to produce constraints that would encode only one of the three levels. We would expect that as more documents are used to generate the constraints, the more detailed segmentation is favored. Figure 3 shows this is the case.

- **Computers**
  - ○ Hardware
    - * comp.os.mswindows.misc
    - * comp.windows.x
    - * comp.graphics
  - ○ Software
    - * comp.sys.ibm.pc.hardware
    - * comp.sys.mac.hardware
- **Recreation**
  - ○ Automobiles
    - * rec.autos
    - * rec.motorcycles
  - ○ Sports
    - * rec.sport.baseball
    - * rec.sport.hockey
- **Sale**
    - * misc.forsale
- **Science**
    - * sci.med
  - ○ Technology
    - * sci.crypt
    - * sci.electronics
    - * sci.space
- **Politics**
    - * talk.politics.guns
  - ○ International
    - * talk.politics.mideast
    - * talk.politics.misc
- **Philosophy/Religion**
    - * alt.atheism
  - ○ Theism
    - * talk.religion.misc
    - * soc.religion.christian

Table 3: Hierarchical structure of the 20 Newsgroup data.

Space and Time Comparison and An Explanation

In this section, we compare our quadratic programming formulation containing equality constraints and inequality constraints to the spectral clustering (SC) algorithm and a popular constrained spectral clustering (CSC) algorithm (Wang and Davidson 2010) . These formulations address different settings, but we wish to answer two important questions:

- How similar in quality is the result of our QP formulation (without constraints) compared to the SC algorithm?

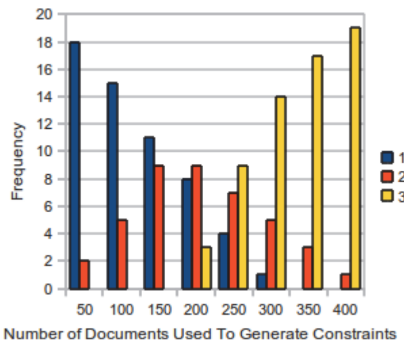

Figure 3: The frequency distribution of how many times each level of the hierarchy is discovered versus the number of documents to generate the constraints. As expected, with the number of constraints increasing the more complex segmentation is favored.

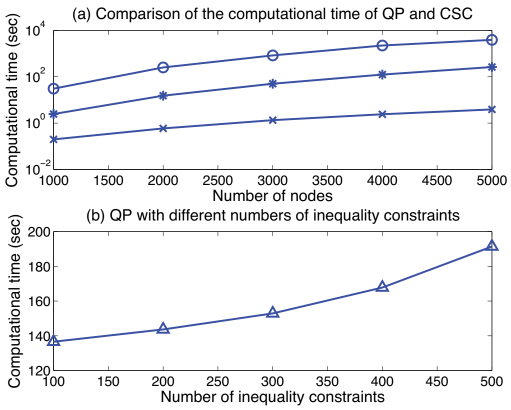

Figure 4: (a) The computational time of constrained spectral clustering (CSC) and QP formulation with different numbers of nodes. 'o-' represents CSC algorithm with 20 constraints, '$*$-' represents QP with both 10 linear equalities and 10 inequalities . '$\times$-' represents QP with 20 equalities. (b) The computational time of QP with different numbers of inequalities.

- How efficient in terms of space and time is the QP formulation (with constraints) as compared to the CSC algorithm?

For these two questions, we use a synthetic data set of varying size where the ground truth of the data is already known. We generate a graph of N nodes (N is even) with two connected components. Each of the connected components has $N/2$ nodes (cluster 1 is nodes 1 through $N/2$ and cluster 2 nodes $N/2+1$ through N) with the possibility of each pair of nodes in the same cluster having an edge between them being 50%. The weight of the edge, if chosen, is set to 1. To enforce a connected graph and make the problem more difficult, the weight of the edge between any two nodes in different connected components is a pseudo-random number uniformly distributed between 0 and 0.1.

Quality of Clustering Solution. Both the spectral and QP formulation share the same convex objective function, hence on the surface it would appear they would converge to the same result. However, since they use different constraints this need not be the case. First, we compare the clustering quality between spectral clustering and our QP formulation with different numbers of nodes $N = 1000, 2000, 3000, 4000$ and 5000 of the synthetic data set. For the QP formulation, we use equation (2) with $j = 1$. In our experiments both the normalized min-cut SC formulation and our QP formulation cluster the nodes of the graph into the exact same two clusters that match the ground truth. Understanding when the two algorithms converge to substantially different results is left to future work, but based on our experiments not described, the algorithms converge to similar results.

Computational Time Comparisons. The earlier CSC (Wang and Davidson 2010; Wang, Qian, and Davidson 2012) can only encode conjunctions of constraints hence we present two different sets of experimental results for our work: just using equality constraints to encode conjunctions of constraints (see Lemma 2) and both conjunctions and disjunctions of constraints (see Theorem 1). The algorithms are implemented in MATLAB using standard functions `eig` and `quadprog`.

The computational time of the CSC algorithm, the QP formulation with only equality constraints and also the QP formulation with both equality and inequality constraints are shown in Figure 4(a). We see that the QP solver is much faster than CSC when we use conjunctions of constraints (approximately 1000 times faster) and (10-15 times faster) when both conjunctions and disjunctions of constraints are used. This is so since with the CSC algorithm all N eigenvectors must be obtained and sorted through as per the author's own paper.

In Figure 4(b), the number of nodes is fixed as $N = 4000$ and the numbers of constraints are ranged from 100 to 500. Figure 4(b) shows that the computational time increases moderately when we have more inequality constraints. We can see that the QP formulation is still faster than the CSC algorithm even with several hundreds of constraints. We can conclude that the QP formulation using conjunctions and disjunctions constraints is faster than the CSC algorithm. If we have only conjunctions of constraints, the QP formulation is much more efficient in computation.

Memory Usage. Figure 5(a) shows the comparison of memory usages for CSC algorithm and our QP formulation. The experiments are exactly the same as Figure 4(a). We compare the memory usage and see in Figure 5(a) that solving the QP problem needs a limited amount of additional memory than running the CSC algorithm. The experiments of Figure 5(b) are the same as Figure 4(b). Figure 5(b) shows that solving the QP formulation needs more memory in computation if we have more inequality constraints.

An Explanation of Our Analysis. The CSC algorithm needs to solve a generalized eigenvalue (GEV) problem. The `eig` function of MATLAB uses the QZ method to solve this problem. The complexity of QZ method is approximately $\mathcal{O}(66N^3)$ (Golub and van Loan 1996) and space requirement is $\mathcal{O}(N^2)$. To solve the QP with equality constraints, MATLAB uses the Projective Preconditioned Conjugate Gradient method (PPCG) (Coleman 1994), whose complexity is $\mathcal{O}(pN^2)$ and space requirement is $\mathcal{O}(N^2)$,

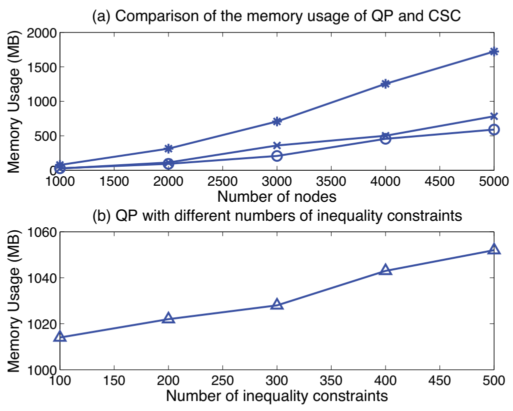

Figure 5: (a) The memory usage of constrained spectral clustering (CSC) and QP formulation with different number of nodes. 'o-' represents CSC algorithm with 20 constraints, '$*$-' represents QP with both 10 linear equalities and 10 inequalities . '$\times$-' represents QP with 20 equalities. (b) The memory usage of QP with different numbers of inequalities.

where p is the number of iterations in the optimization. Hence of the methods PPCG is much faster than QZ as per their complexity which is reflected in our experiments.

To solve the QP with inequality constraints, MATLAB uses the interior-point method (Byrd, Hribar, and Nocedal 1999), whose complexity is $\mathcal{O}(qN^3)$ and space requirement is $\mathcal{O}(N^2)$, where q is the number of iterations in optimization using interior-point method. Though both the QZ method and interior-point method have complexity $O(N^3)$, the advantage of the interior-point method is that it converges in a few steps. Hence, solving the QP problem is somewhat faster than GEV problem in our experiments using disjunctions and conjunctions of constraints.

Conclusions

In this paper, we propose a quadratic programming formulation with logical combinations of constraints to cluster the data set. Our new formulation of constrained clustering has two main advantages. Firstly, it can encode basic conjunctions of MUST-LINK and CANNOT-LINK constraints as other algorithms can but also complex combinations of constraints including conjunctions and disjunctions. Secondly, due to more efficient solvers being available for the QP solvers than for the generalized eigenvalue problems, our method is substantially (1000 times) faster than a state-of-the-art constrained spectral clustering algorithm for conjunctions of constraints. For disjunctions and conjunctions of constraints, it is approximately ten times faster though no other methods can model disjunctions. Using PIM and 20 Newsgroup data sets, we show the benefits of complex constraints which include modeling conditional guidance as well as complex data structures such as hierarchies.

Acknowledgements

The authors gratefully acknowledge support of this research via ONR grants N00014-09-1-0712, N00014-11-1-0108 and NSF Grant NSF IIS-0801528.

References

Basu, S.; Davidson, I.; and Wagstaff, K., eds. 2008. *Constrained clustering: Advances in algorithms, theory, and applications*. Chapman & Hall/CRC.

Byrd, R. H.; Hribar, M. E.; and Nocedal, J. 1999. An interior point algorithm for large-scale nonlinear programming. *SIAM Journal on Optimization* 9(4):877–900.

Chaovalitwongse, W. 2008. Novel quadratic programming approach for time series clustering with biomedical application. *Journal of Combinatorial Optimization* 15(3):225–241.

Coleman, T.; Saunderson, J.; and Wirth, A. 2008. Spectral clustering with inconsistent advice. In *ICML 2008: Proc. of the 25th Intl. Conference on Machine Learning*, 152–159.

Coleman, T. F. 1994. Linearly constrained optimization and projected preconditioned conjugate gradients. In *Proceedings of the Fifth SIAM Conference on Applied Linear Algebra*, 118–122. Philadelphia, PA: SIAM.

Golub, G. H., and van Loan, C. F. 1996. *Matrix Computations*. Johns Hopkins University Press.

Kamvar, S.; Klein, D.; and Manning, C. 2003. Spectral learning. In *IJCAI 2003: Proc. of the 17th Intl. Joint Conference on Artificial Intelligence*, 561–566.

Lang, K. 1995. Newsweeder: Learning to filter netnews. In *ICML 1995: Proc. of the 12th Intl. Conference on Machine Learning*, 331–339.

Long, B.; Zhang, Z. M.; Wu, X.; and Yu, P. S. 2006. Spectral clustering for multi-type relational data. In *ICML 2006: Proc. of the 23rd Intl. Conference on Machine Learning*, 585–592.

Long, B.; Zhang, Z.; and Yu, P. 2010. *Relational Data Clustering: Models, Algorithms, and Applications*. Chapman & Hall/CRC.

von Luxburg, U. 2007. A tutorial on spectral clustering. *Statistics and Computing* 17(4):395–416.

Wagstaff, K.; Cardie, C.; Rogers, S.; and Schroedl, S. 2001. Constrained k-means clustering with background knowledge. In *ICML 2001: Proc. 18th Intl. Conf. on Machine Learning*, 577–584.

Wang, X., and Davidson, I. 2010. Flexible constrained spectral clustering. In *KDD 2010: Proc. 16th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, 563–572.

Wang, X.; Qian, B.; and Davidson, I. 2012. Improving document clustering using automated machine translation. In *CIKM 2012: Proc. 21st ACM Intl. Conf. on Information and Knowledge Management*, 645–653.