Discovering Approximate Gene Clusters as a Minimum-Weighted Clique Problem

Geoffrey Solano[1,2], Bianca Camille Silmaro[1], Sun Arthur Ojeda[1], Jaime DL Caro[2]
[1]University of the Philippines Manila, Philippines
[2]University of the Philippines Diliman, Philippines

ABSTRACT

Given a complete weighted undirected graph G with both vertex and edge weights, the Weighted Clique Problem (WCP) is the problem of finding a clique in G of order m with extremal weight. A variant of WCP is the Minimum Edge-Weighted Clique Problem (Min-EWCP) where the vertex weights are all 0 but edge weights ≥ 0 and the problem is that of finding the clique in G of order m with the least weight. In this paper, the algorithm presented by Eremin et. al. for Min-EWCP was reviewed and another algorithm was presented along with its performance guarantee for the metric and the ultrametric set of inputs. A way to determine the performance guarantee as the strictness of the metricity is varied through a factor σ was also shown. Furthermore, this study presents the problem of identifying sets of commonly existing putative co-regulated, co-expressed genes, called gene clusters, as a clique-finding problem. A gene distance matrix was constructed from a genome database by obtaining the distances of each pair of genes in each genome. Finally, experimental results are shown where approximate gene clusters are obtained from putative orthologous genes of the genomes $E.colistrK-12$ and $B.subtilis$.

CCS CONCEPTS

•Mathematics of computing → Trees; Graph algorithms; Approximation algorithms; •Applied computing → *Biological networks;*

KEYWORDS

graphs, approximation algorithm, cliques, star, genes, genomes, gene cluster discovery

1 INTRODUCTION

In this contribution, we study optimization problems related to finding cliques in complete graphs. Cliques have been thoroughly studied in the area of graph theory (*e.g.*, [14, 6, 3, 8, 23, 12, 13, 11, 18, 2, 7, 15, 16, 22]). Among these was a study [10] that presented a general combinatorial problem called the Weighted Clique Problem (WCP). Given a complete weighted undirected graph G with both vertex and edge weights, the Weighted Clique Problem is the problem of finding a clique of the graph G of order m with the smallest (largest) weight. It was shown in the said study that both minimum and maximum WCPs are not approximable in the general case. The edge-weighted clique problem (Min-EWCP), a variant of the WCP, was introduced in the said study, where the weights of all vertices are assigned 0 and the basis of determining minimum-weighted clique are the edge weights. A fast 2-approximation algorithm was also presented in [10], however, for two important cases of the problem, in which vertex weights are nonnegative and edge weights either satisfy the triangle inequality (the metric WCP) or are squared pairwise distances for a point configuration in Euclidean space (the quadratic Euclidean WCP).

Here we consider a slightly different version of the problem motivated by the challenge of identifying sets of commonly existing putative co-regulated, co-expressed genes, called gene clusters. In comparative genomics, which is the field in biological research wherein genomic sequences are analyzed to understand the genetic elements defining the commonality and uniqueness among different organisms [9], one of the fundamental problems is that of defining relationships between species. Among the methods used to achieve this is gene clusters discovery. A gene cluster is a set of closely-related genes which are arranged in close proximity with each other, even after genome sequences have evolved in multiple events such as gene duplication and gene loss. Genomic regions with relatively similar gene content is a result of duplication and divergence [17]. Gene clusters appear in two or more genomes and perform related functions. Organisms containing similar gene clusters with other species tend to share common traits. This was formulated as the Approximate Gene Cluster Discovery Problem (AGCDP) and was formulated as an integer-linear programming problem in [20]. This was later represented as a graph in [1].

In this paper, we present the problem of finding approximate gene clusters as a clique-finding problem. In Section 2, a review of the results in the study of Eremin et. al. [10] is presented as well as another proof approach which shows that the algorithm presented in the study returns at most twice the cost of the optimal solution for the metric case. Results on performance guarantees for two input classes for the said algorithm, as well as another algorithm for clique-finding problem as also presented in Section 3. In Section 4, we show the formulation of Approximate Gene Cluster Discovery as a weighted clique problem, specifically Min-EWCP. Experimental results are shown in Section 5, where approximate gene clusters are obtained from putative orthologous genes of the genomes $E.colistrK-12$ and $B.subtilis$.

2 PRELIMINARIES

2.1 The Weighted Clique Problem (WCP)

The Weighted Clique Problem (WCP) was presented in [10] as a general combinatorial problem. Given a complete simple weighted undirected graph $G = (V, E, a, c)$ and weight functions $a : V \to Q$ and $c : E \to Q$, which define the vertex weights and the edge weights respectively, the weight of the graph G is expressed as the sum $\Sigma_{v \in V} a_v + \Sigma_{e \in E} c_e$. Formally, it is defined as follows :

Definition 2.1 (Weighted Clique Problem (WCP) [10]*).* Given a complete weighted undirected graph $G = (V, E, a, c)$, where $a : V \rightarrow Q$ and $c : E \rightarrow Q$, and a positive integer m, find a complete subgraph (clique) of the graph G of order m with the smallest (largest) weight.

A variant of WCP was further defined in [10] as Min-EWCP, which is the problem of finding the clique of order m in the weighted complete graph $G = (V, E, a, c)$ with the smallest weight, where $a_i = 0$ and $c_{ij} \geq 0$. It was shown that this problem is not in APX, or the class of NP optimization problems that allow polynomial-time approximation algorithms with approximation ratio bounded by a constant.

2.2 Row's Subset of Symmetric Matrix Problem

Eremin et. al. [10] further presented the row's subset of symmetric matrix problem (RSSM) as a polynomial time equivalent formulation of the minimum edge-weighted clique problem (Min-EWCP) in the form of property verification problem.

Definition 2.2 (Row's subset of symmetric matrix problem (RSSM) [10]*).* Given a symmetric n x n matrix $W = (w_{ij})$ with nonnegative entries and $w_{ii} = 0$, a positive integer m and a positive number D, determine whether the set of rows of W contains a subset C of cardinality m such that

$$F(C) = \frac{1}{2} \sum_{i \in C} \sum_{j \in C} w_{ij} \leq D$$

The following polynomial-time algorithm was presented in the said study as a solution to RSSM, and consequently to Min-EWCP, which we will refer to here as Algorithm A :

Algorithm A
Step 1. For each $j = 1, ..., n$, find a set B_j that consists of indices of m smallest entries in the jth row of W including the index j itself. Define $S(B_j) = \sum_{i \in B_j} w_{ij}$.
Step 2. Denote by k^* the value j for which $S(B_j)$ takes the minimum value $S^* = S(B^*) = \sum_{i \in B^*} w_{ik^*}$.
Take $C = B^*$ as an approximate solution of RSSM.

Each approximation algorithm has an inherent weakness. In the case of Algorithm A, one could note that it makes use of only $m-1$ edges in its prediction of the $\binom{m}{2}$ edges in an m-clique. In the process it actually ignores $\binom{m-1}{2}$ edges - edges which may have the least weights.

If we let W be the adjacency matrix representation of a complete graph G, then it is a symmetric n x n matrix $W = (w_{ij})$ with nonnegative entries and $w_{ii} = 0$. For a given vertex v_i, the row corresponding to it in W contains entries that represent weights of edges incident to v_i, with ith entry having the value 0. These entries are therefore the weights of the star having v_i as the hub.

Given a positive integer m, the $m-1$ vertices corresponding to the $m-1$ entries in the ith row with the lowest non-zero values make up the minimum weighted $(m-1)$-star, i.e. the $(m-1)$-star whose edges have the least total weight and having v_i as the hub.

Step 2 of Algorithm A clearly approximates minimum-weighted m-clique as it determines the minimum-weighted $(m-1)$-star by selecting the vertices B_j corresponding to the jth row, of which the sum of the m least entries, $S(B_j)$, are also the least across all rows.

3 AN ALGORITHM FOR MIN-EWCP

In this section we present Algorithm B, an alternative algorithm for Min-EWCP.

3.1 Algorithm Definition

It was shown in the previous section that Algorithm A clearly approximates minimum-weighted m-clique by determining the minimum-weighted $m-1$-star. However, the point of interest of Algorithm B is not in finding the jth row for which $S(B_j)$ is minimum. This algorithm is more concerned that the set of vertices corresponding to the lowest entries in each row appear not just in one row but most frequently appearing as the lowest entries across all rows. This algorithm takes such a case as an indicator of an optimal solution.

Given an m-clique, each of the vertices $v_0, v_1, v_2, ..., v_{m-1}$ is actually a hub of a $m-1$-star. The set of m vertices that most frequently corresponds to the m entries in a row with the least value across all rows are most probably the same set of vertices that make up the minimum-weighted m-clique.

We now therefore present Algorithm B as a polynomial-time algorithm solution to Min-EWCP:

Algorithm B
Step 1. For each $j = 1, ..., n$, find a set B_j that consists of indices of m smallest entries in the jth row of W including the index j itself. Define $S(B_j) = \sum_{i \in B_j} w_{ij}$.
Step 2. Denote by B^* the set(s) of indices (vertices) B_j which appear(s) with the highest frequency r, across all the n rows. In case more than one set of vertices appear with the highest frequency, the one with the smallest total weight is selected.
Take $C = B^*$ as an approximate solution.

The more number of times a set of vertices corresponds to the minimum entries across the rows, the more likely it is to be the vertex set of the minimum-weighted m-clique. If the same set of vertices B^* appears across all rows, then for sure B^* is the optimal clique. Now in the case that all the B_j's returned per row all have the frequency 1, that is, the set of indices for the minimum entries per row are unique, then the one with the minimum weight is chosen (similar to that of Algorithm A).

In Algorithms B, if a vertex set B^* correspond to the minimum weighted $(m-1)$-star in r rows, then $\sum_{i=1}^{r} (m-i)$ are considered and only $\binom{m-r}{2}$ are not considered.

3.2 Performance guarantee for the Metric Case

We now show that Algorithm B has a performance guarantee of returning less than twice the optimal result for the metric set of inputs.

Theorem 3.1. *Algorithm B is a* $2 - O(\frac{1}{m})$*-approximation algorithm for the metric case of* Min-EWCP *that runs in* $O(n^2)$

Proof. Since W is an n x n-matrix, which the algorithm traverses only once, it is easy to see that the algorithm runs in $O(n^2)$. Checking the vertex set with the maximum frequency also takes $O(n^2)$. Therefore the whole algorithm is still in $O(n^2)$. Furthermore, we recall that given a complete graph G with positive edge weights, we define $cost(G)$ as the sum of all the edges in G. We define Q as the optimal clique or the m-clique in G of minimum weight and we let Q' be the m-clique returned by Algorithm B. We define Q^* as the $(m-1)$-star in G with minimum weight or cost. We further note that Algorithm B in its approximation of the minimum weighted m-clique, selects the vertex sets corresponding to the m smallest entries in each row (entries on the diagonal of the adjacency matrix are 0 and are included), and chooses the set that has the highest frequency among the chosen vertices. Thus, $cost(Q') \leq \beta \cdot cost(Q)$, where β is the approximation ratio for Algorithm B.

It has already been established before that

$$\frac{1}{m-1} \cdot \binom{m}{2} \cdot \sum_{i=1}^{m-1} w'_i \leq cost(Q),$$
$$\frac{m}{2} \cdot cost(Q^*) \leq cost(Q).$$

Without loss of generality we assume that the vertices involved with Q are also the set of vertices which represent the minimum entries for the said rows in Min-EWCP, making it indeed the optimal clique. We note that of the $(m-1)$-stars in Q, of which there are m, none would have a cost lower than that of Q^*. Therefore, at best there will be m minimum weighted $(m-1)$-stars in G and all of them are in Q, that is, each of the m vertices in Q is a hub of a minimum weighted $(m-1)$-star in G.

Now, for the upper-bound on $cost(Q')$, we note that the cost of Q' is actually the sum of the cost of Q^* and the cost of the remaining edges in Q' but not in Q^* is expressed as:

$$cost(Q') = \sum_{i=1}^{e} w'_i = \sum_{i=1}^{m-1} w'_i + \sum_{i=m}^{e} w'_i,$$

where $e = \binom{m}{2}$ and $m \geq 3$.

Here, in the usual case, $cost(Q^*)$ is taken from the cost of the optimal $(m-1)$-star Q^*, while cost of each of the remaining edges $E(Q' \setminus Q^*)$ is at most double of that of an edge in Q^* because of the metricity property. However, since the solution returned by the algorithm is the set of vertices $V(Q^*)$ which correspond to the minimum entries in r row(s), if a set of index vertices appears once, i.e. r=1, then we have $m-1$ optimal edges, and then we predict at worst double cost for $\binom{m-1}{2}$ edges. If r=2, then $m-2$ is added to the optimal edges, since one of the edges is already shared with the first set, and thus we have $m-1+m-2$ optimal edges. This makes the sum of the weights of the optimal edges equal to $\frac{cost(Q^*)}{m-1} \cdot \sum_{i=1}^{r}(m-i)$, for a vertex set with frequency equal to r. For the remaining $\binom{m-r}{2}$ edges in Q', each one is assigned at most double of that of an edge in Q^*. Thus, if a set vertices appears with the highest frequency r across all the rows, then $cost(Q') = \frac{cost(Q^*)}{m-1} \cdot \left[\sum_{i=1}^{r}(m-i) + 2 \cdot \binom{m-r}{2}\right]$.

Therefore,

$$\begin{aligned}
\beta &\geq \frac{cost(Q')}{cost(Q)} = \frac{\frac{cost(Q^*)}{m-1} \cdot \left[\sum_{i=1}^{r}(m-i) + 2 \cdot \binom{m-r}{2}\right]}{\frac{m}{2} cost(Q^*)} \\
&\geq \frac{2}{m(m-1)} \cdot \left[rm - \frac{r(r+1)}{2} + \frac{2[(m-r)^2 - (m-r)]}{2}\right] \\
&\geq 2 - \frac{2r}{m-1} + \frac{r(r+1)}{m(m-1)} \\
&\geq 2 - O(\tfrac{1}{m})
\end{aligned}$$

Thus, Algorithm B is a $2 - O(\frac{1}{m})$-approximation algorithm for the metric case of Min-EWCP.

□

We further note that when $r = m$, then $\beta = 1$. This means that if all the m vertices of an m-clique are the hubs of m minimum weighted $(m-1)$-stars, then that m-clique is the minimum weighted m-clique in G.

3.3 Performance guarantee for the Ultrametric Case

We now show performance guarantee of Algorithm B in another case of inputs, specifically, the ultrametric case of inputs.

Theorem 3.2. *Algorithm B is a* $1+O(\frac{1}{t})$*-approximation algorithm for ultrametric case of* Min-EWCP.

Proof. When considering ultrametric weight function w (i.e., for any distinct vertices $a, b, c \in V'$, $w(a,b) \leq max(w(b,c), w(c,a))$), we note the following:

In obtaining the performance guarantee of Algorithm B for the ultrametric case of inputs, we wish to obtain α for which $cost(Q') \leq \alpha \cdot cost(Q)$. Intuitively, necessary for this is determining and describing the instance when difference between the costs of Q' and Q is the greatest.

To describe such instance, we note that $cost(Q') \leq cost(Q^*) + \sum_{i=m}^{e} w'_i$, and $cost(Q) \leq cost(Q^A) + \sum_{i=m}^{e} w_i$, where, by definition, Q^A is a $(m-1)$-star, such that $cost(Q^*) < cost(Q^A)$, even if $cost(Q) \leq cost(Q')$. For Q^* to be chosen as the $(m-1)$-star in G of minimum weight, it is because though it has $(m-2)$ edges that have relatively larger weight, say b, it has an edge that has a very small weight, say a, making the $cost(Q^*)$ still the least among the $(m-1)$-stars. The same, however, cannot be said about Q'.

We note that $\sum_{i=m}^{e} w'_i$ makes use of edges that are part of Q^*, as is shown in Figure 2, but because of the ultrametric property, each of the edges in Q' but not in Q^*, i.e. $E(Q' \setminus Q^*) = w'_5, w'_6, ..., w'_9, w'_{10}$, will assume the weight value of the higher-weighted edge between the two edges it forms a triangle with. Since each of the edges in $E(Q' \setminus Q^*)$, i.e. the non-dashed edges, would have to be from between the dashed edges (i.e. edges in Q^*), then $\sum_{i=m}^{e} w'_i = \binom{m-r}{2} \cdot b$.

Therefore, $cost(Q') = a + b \cdot \left[\sum_{i=1}^{r} (m-i) - 1 \right] + b \cdot \binom{m-1}{2}$

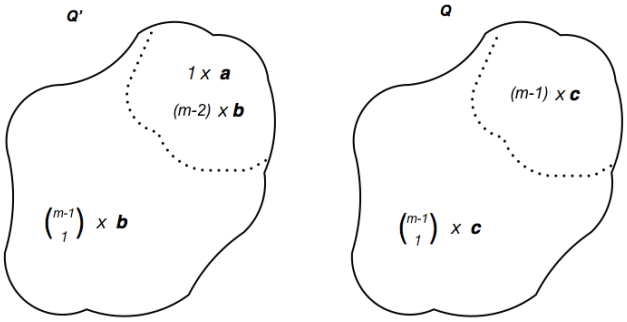

Figure 1: The instance where α is greatest where $cost(Q') \leq \alpha \cdot cost(Q)$

On the other hand, in describing such instance for Q, since $cost(Q) \leq cost(Q^A) + \sum_{i=m}^{e} w_i$, where $cost(Q^*) < cost(Q^A)$, then given that a is a very small positive value, we can let $cost(Q^*) = cost(Q^A) + a$. For the edges that are not in Q^A, without loss of generality, if we let $w_1 \leq w_2 \leq w_3 \leq ... \leq w_{m-1}$, where $w_i \in E(Q^A)$, then because of the ultrametric property, $\sum_{i=m}^{e} w_i = w_2(1) + w_3(2) + w_4(3) + ...+ \leq w_{m-1}(m-2)$.

Since larger weighted edges would be chosen more often to be in $E(Q \setminus Q^A)$, i.e. w_i for $t \leq i \leq e$, consequently making $cost(Q)$ larger, then clearly the instance when $cost(Q)$ is minimum is when all the edges in Q are all of equal weight, say c. Thus, $cost(Q) = \binom{m}{2} \cdot c$ as shown in Figure 1 , where $c = \frac{cost(Q^A)+a}{m-1} = \frac{2a+(m-2)\cdot b}{m-1}$.

Therefore,

$$
\begin{aligned}
cost(Q') &= \frac{a+b\cdot[\sum_{i=1}^{r}(m-i)-1]+b\cdot\binom{m-1}{2}}{\frac{2a+(m-2)\cdot b}{m-1}\cdot\binom{m}{2}} \cdot cost(Q) \\
&= \frac{a+b\cdot[\binom{m}{2}-1]}{2a+b\cdot(m-2)\cdot\frac{m}{2}} \cdot cost(Q) \\
&= (1+\epsilon)\cdot cost(Q)
\end{aligned}
$$

where $\epsilon = \frac{b(m-2)-2a(m-1)}{bm(m-2)+2am}$

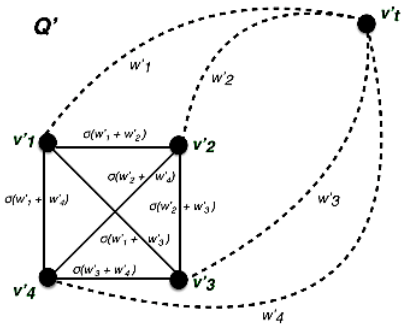

Figure 2: The weights of the edges of Q^* are used to provide variable upper bounds on the remaining edges of the solution Q' for $t = 5$

We further note that $\epsilon \approx \frac{1}{m}$ as $a \to 0$.

Therefore, Algorithm B is a $1+O(\frac{1}{m})$-approximation algorithm for the ultrametric case of Min-EWCP.

□

3.4 Varying the strictness of the metricity through a given factor σ

We introduce a variable σ in analyzing Algorithm B as a means of tightening or relaxing the metric property.

Theorem 3.3. *If for any distinct vertices $a, b, c, \in V$, $w(a,b) \leq \sigma(w(b,c) + w(c,a))$, then Algorithm B is a $\sigma + O(\frac{1}{m})$-approximation algorithm for Min-EWCP.*

Proof. The triangle inequality states that for any distinct vertices $a, b, c, \in V$, $w(a,b) \leq w(b,c) + w(c,a)$. We observe the effect if $w(a,b) \leq \sigma(w(b,c) + w(c,a))$. It has already been established that

$$
\begin{aligned}
cost(Q) &\leq cost(Q') \\
\sum_{i=1}^{e} w_i &\leq \sum_{i=1}^{e} w'_i \\
&\leq \sum_{i=1}^{t-1} w'_i + \sum_{i=t}^{e} w'_i \\
&\leq cost(Q^*) + \sum_{i=t}^{e} w'_i
\end{aligned}
$$

and that what is being approximated here is $\sum_{i=t}^{e} w'_i$. Applying the proposed bounds on the third side of the triangle, the weights of each of the $\binom{t-1}{2}$ remaining edges may be computed as follows for $1 \leq p < q \leq t-1$

$$
\begin{aligned}
cost(v'_p, v'_q) &\leq \sigma(cost(v'_t, v'_p) + cost(v'_t, v'_q)) \\
&\leq \sigma(w'_p + w'_q)
\end{aligned}
$$

As has already been shown, each of the vertices $v'_1, v'_2, v'_3, ..., v'_{t-1}$ in the $(t-1)$-clique, by definition, is adjacent to $t-2$ vertices in the $(t-1)$-clique as is illustrated in Figure 2 for $t = 5$.

Hence
$cost(Q') = \frac{cost(Q^*)}{m-1} \cdot [\sum_{i=1}^{r}(m-i) + \sigma \cdot \binom{m-r}{2}]$.

Therefore,

$$\beta \geq \frac{cost(Q')}{cost(Q)} = \frac{\frac{cost(Q^*)}{m-1} \cdot [\sum_{i=1}^{r}(m-i) + \sigma \cdot \binom{m-r}{2}]}{\frac{m}{2} cost(Q^*)}$$

$$\geq \frac{2}{m(m-1)} \cdot [rm - \frac{r(r+1)}{2} + \frac{\sigma[(m-r)^2-(m-r)]}{2}]$$

$$\geq \frac{\sigma \cdot m(m-1)}{m(m-1)} + \frac{\sigma(r^2-2mr+r)-(r^2-2mr+r)}{m(m-1)}$$

$$\geq \sigma + \frac{r(\sigma-1)(r-2m-1)}{m(m-1)}$$

$$\geq \sigma + O(\frac{1}{m})$$

Therefore, applying the property for any distinct vertices $a, b, c, \in V$, $w(a,b) \leq \sigma(w(b,c)+w(c,a))$ will make Algorithm B a $\sigma+O(\frac{1}{m})$-approximation algorithm for Min-EWCP.

□

3.5 On Returning a Set of Cliques

Though a significant part of this study is concerned with extending the results on Min-EWCP, it was also mentioned at the onset that this study is motivated by the problem of identifying approximate gene clusters. Correspondence can be made between genes and nodes, while co-expression value between two genes can be represented by weights placed on edges joining a pair of vertices. Thus, the greater the association between a pair vertices, the smaller the distance between them, the smaller the weight of the edge joining them. A data structure containing the values of the associations across all possible pairs of genes would be an adjacency matrix of a complete undirected graph.

The output of Min-EWCP would be therefore be a clique representing a set of highly identical gene groups across a set of genomes which will be a candidate gene cluster. In practice, however, experts are more concerned with obtaining not just one candidate gene cluster but a set of candidate gene clusters, which can then be validated by experiments.

For this, a modified version of Algorithm B can be made - one that does not return only one solution but returns a list of candidate solutions that satisfy a metric. The algorithm can be modified to allow the selection of m-cliques whose total edge-weights, as approximated using the weight of the $(m-1)$-star, does not exceed a given threshold r. A slight modification can thus be made on Algorithm B that would select all those vertex sets satisfying the above condition. The outputs will be the candidate gene clusters. This would be the preference of many bioinformaticians and systems biologists since the candidate gene clusters returned would still have be verified by biochemists and molecular biologists.

With this, the modified Algorithm B would provide a list of approximate gene clusters. Since Algorithm B runs in $O(n^2)$, then this modified version will run in $O(n^2) + n = O(n^2)$.

4 FORMULATION OF APPROXIMATE GENE CLUSTER DISCOVERY AS MIN-EWCP

At this point we construct a complete graph G_K from the set of genomes GD. We first identify the vertex set. Since G_K, is complete, we will then only need to describe the edge weights. The gene distance matrix that is constructed will be the adjacency matrix representation of G_K. In a previous study [21], approximate gene clusters were obtained using minimum weight t-partite cliques. In this study, they are identified by determining the minimum edge-weighted cliques (Min-EWCP).

4.1 The Vertices of G_K

Let GD be the set of d genomes, that is, $GD = \{G^1, G^2, ..., G^d\}$. Let $U = \{g_1, g_2, ..., g_n\}$ be the set of all homologous genes in GD. The vertex set of $G_K = V(G_K) = U$. Since G_K is a complete graph, every pair of nodes (g_i, g_j) in U, where $i \neq j$, is an edge in G_K. Thus $|E(G_K)| = \binom{n}{2}$.

4.2 The Weights of the Edges in G_K

We note that in a given genome G^i, the position of a given vertex g_j is denoted by $p^i(g_j)$. We also note that in gene cluster models that do not allow the duplication of genes in a genome, then $|p^i(g_p) - p^i(g_q)|$ simply means the distance between the genes g_p and g_q in the genome G^i. However, for gene cluster models that allow the multiple occurrences of genes in a genome, the distance $|p^i(g_p) - p^i(g_q)|$ means the shortest distance between any occurrence of g_p and g_q in the genome G^i.

The weights of the edges in G_K are obtained by the following algorithm:

Algorithm 1 Building the distance matrix of the complete weighted graph G_K

Input: The set of genomes GD, the set of homologous genes U and the complete graph G_K
Output: The distance matrix of G_K

```
0: wt(g_p, g_q) ← 0, for every edge (g_p, g_q) ∈ G_K
0: for each genome G^i ∈ G where 1 ≤ i ≤ m do
0:   for each pair (g_p, g_q) ∈ U, s.t. 1 ≤ p < q ≤ n do
0:     if g_p and g_q are both ∈ G^i then
0:       wt(g_p, g_q) ← wt(g_p, g_q) + |p^i(g_p) − p^i(g_q)|
0:       wt(g_q, g_p) ← wt(g_q, g_p) + |p^i(g_p) − p^i(g_q)|
0:     else
0:       if either g_p or g_q ∈ G^i then
0:         wt(g_p, g_q) ← length(G^i) − 1
0:         wt(g_q, g_p) ← length(G^i) − 1
0:       else
0:         wt(g_p, g_q) ← length(G^i)
0:         wt(g_q, g_p) ← length(G^i)
  =0
```

The resulting complete graph G_K now stores the cumulative distances between each pair of genes across all genomes. This summary data provides a good basis where different algorithms

con be performed to obtain gene clusters. For example, hierarchical agglomerative clustering can be performed to form a dendrogram and then find gene groups. Hierarchical agglomerative clustering was performed using https://github.com/lbehnke/hierarchical-clustering-java. The resulting dendrogram, as well as the resulting gene clusters, can be found here : https://drive.google.com/drive/u/0/folders/1ZnsCS7RwbIkI3KOnnxDow_epEmSXI5Rh.

The approach used in this study for finding gene clusters, however, was that of finding m-cliques, in the complete graph G_K .

THEOREM 4.1. *The problem of finding approximate gene clusters of size m in GD is equivalent to the MINIMUM EDGE-WEIGHTED CLIQUE PROBLEM*

PROOF. By definition, a gene cluster is a set of closely-related genes which are arranged in close proximity with each other across two or more genomes, even after genome sequences have evolved in multiple events such as gene duplication and gene loss [17].

With the above construction of the complete weighted graph G_K as having the vertex set $V(G_K) = U = \{g_1, g_2, ..., g_n\}$ as the set of all homologous genes in GD, and the edges weights of each pair of nodes in G_K as the sum of the distances between the corresponding genes across GD as obtained by Algorithm 1, finding the m-clique in the complete edge-weighted graph G_K of minimum weight will return the set of m genes that appear closest to each other across genomes, which is precisely how an approximate gene cluster has been defined.

Thus, the problem of finding approximate gene clusters of size m in GD is equivalent to the MINIMUM EDGE-WEIGHTED CLIQUE PROBLEM or the problem of finding the m-clique in the complete edge-weighted graph G_K of minimum weight

□

The time complexity of building the graph G_K is $O(n^2)$ if there are a total of n genes. On the other hand, the cost of assigning the weights to the edges of G_K is $O(dkn^2)$, where k is the average length of each genome.

5 EXPERIMENTAL RESULTS

5.1 Source of Data

The dataset is composed of 1406 putative orthologous genes of the genomes *E. coli str. K-12* and *B. subtilis*. The genes are named based on the gene symbols of *E. coli str. K-12*. Genes that are non-orthologs were labeled as 0 as they are not of interest for the problem.

The orthologous genes were determined using an automated pairwise genome comparison technique. The pairwise comparison of two genomes is modeled as a weighted bipartite graph-matching problem. The weights of the edges are identified using the Smith-Waterman algorithm and PAM120 matrix. The gene corresponding to the nodes of the best matching edges of the bipartite graphs are taken as orthologs, and are deleted from the further consideration [4]. The genomes of E. coli str K-12 and B. subtilis were extracted from Genbank in .gbk file format [5].

A gene without a known functionality has been referred to as Genome name: orf.index, where index is the ordering of the protein coding region.

5.2 Clusters shown in [4] from the genomes of Escherichia Coli and Bacillus Subtilis

Bansal defined a gene-group $< \gamma_{1I}, \gamma_{1J}, \gamma_{1K}, ... >$ as a cluster of genes of at least two distinct genes in close proximity with each other [5]. A gene-group $< \gamma_{1M}, \gamma_{1N}, \gamma_{1P}, ... >$ in genome 2 is ordered if it complies with the following conditions [5] upon mapping it to its corresponding gene group in genome 1: $[M < N < P$ when $I < J < K,$ or] and $[M > N > P$ when $I > J > K]$.

On the other hand, a gene-group in genome 1 is unordered if the following conditions hold [5]: $[(M > N$ when $I < J)$ or $(N > P$ when $J < K)]$ and $[(M < N$ when $I > J)$ or $(N < P$ when $J > K)]$ and $[(M \neq N$ and $I = J)$ or $(N \neq P$ and $J = K)]$.

The putative gene-groups shown in [4] were identified by finding the non-empty intersection set with more than one element between groups of neighboring genes from the genomes *E. coli str. K-12* and *B. subtilis* from the .gbk file extracted from Genbank [5].

Bansal presented 142 E. coli ordered gene-groups with gene cluster sizes equal to 2, 3, 4, 5, and 10. Out of the 142 gene-groups, 60 gene clusters contain non-orthologous genes. Only 82 ordered gene groups composed of orthologs are of interest in this study.

Bansal also presented 28 unordered gene-groups with gene cluster sizes equal to 3, 4, 5, 7, 8 and 9. For this study, we did not include the gene cluster = 27 as it is obviously way bigger the the other sizes . Out of these 28 gene-groups, 8 gene-groups contained non-orthologous genes as can be seen in Table 1. Only 20 unordered gene groups composed of orthologs are of interest in this study.

Bansal's Ordered and Unordered Gene Groups			
size	ordered	unordered	combined
2	118	3	121
3	17	15	32
4	5	5	10
5	1	1	2
6	0	1	1
7	0	1	1
8	0	1	1
9	0	1	1
10	1	0	1
Total	142	28	170
Non -orthologs	60	8	68
Orthologs only	82	20	**102**

Table 1: Bansal's Gene Groups from the genomes *E. coli str. K-12* and *B. subtilis*

These gene-groups were mapped on a standard template of metabolic pathway taken from Kegg database. The set of gene-groups acting as pathway seeds are summarized in [5]. A complete list of Bansal's gene-groups can be found in http://www.cs.kent.edu/~arvind/intellibio/database/gene-groups/ecoli-bsub_gr.html

5.3 Resulting Gene Clusters when Algorithm A and Algorithm B were applied on the Distance Matrix G_K from genomes of E. coli str K-12 and B. subtilis

Since those involved in gene cluster discovery are more concerned with obtaining not just one candidate gene cluster but a set of candidate gene clusters, implementations of both Algorithm A and Algorithm B in python were modified accordingly. Instead of returning the one clique(cluster), a sorted list of candidate clusters was provided by each algorithm. For Algorithm A, the list is sorted in ascending order according to the approximated weight of the candidate cluster. For Algorithm B, the list is sorted in descending order according to the frequency of a candidate cluster. Both lists were cross-checked with the list of candidate gene clusters of Bansal.

It is important to note that since both are approximation algorithms, it would be possible that there are clusters discovered by Bansal that would not be returned by the algorithms. Furthermore, performance guarantees are not applicable since the weights between genes are not metric in nature. Intuitively, however, the gene clusters of Bansal are expected to be those on top of each list. It is interesting to note that the set of clusters of Bansal that are in the list returned by Algorithm A are also the same set in the list returned by Algorithm B. This is true for both ordered and unordered clusters.

Bansal's Ordered and Unordered Gene Groups				Bansal's Gene Groups in Algo A and Algo B		
ord	unord	comb	size	ord	unord	comb
66	2	68	**2**	44	1	45
11	12	23	**3**	8	7	15
4	4	8	**4**	4	3	7
0	1	1	**5**	0	1	1
0	1	1	**7**	0	1	1
1	0	1	**10**	1	0	1
82	20	**102**	Total	57	13	**70**

Table 2: Bansal's Gene Clusters and the outputs from Algorithms A and B

Out of the 102 gene clusters discovered by Bansal, 70 of these were returned by Algorithm A and Algorithm B as can be seen in Table 2. This is 68.63% of the gene clusters discovered by Bansal. Entries for gene cluster sizes equal to 6,8 and 9 were removed since such clusters contained non-orthologs. Both algorithms are approximation algorithms and their outputs are only approximates, that is for a gene cluster of size m, only $(m-1)$ weights between genes were considered of the $\binom{m}{2}$ possible weights that need to be considered. Just like any other approximation algorithm, this was done by the featured algorithms to avoid combinatorial computational explosion.

However when it comes to those clusters of Bansal that were also discovered by the algorithms, such clusters were on top of both lists as was expected. We would like to note that for each cluster size m and n, where $n > m$, there are $\binom{n}{m}$ possible clusters. In this case, where n, or the number of genes, is 1,369, Algorithm A and Algorithm B returned a maximum of 1,369 approximate gene clusters which were sorted in such a manner where the returned clusters with ideal scores were on top of the list.

Tables 3 and 4 show the results for Algorithm A and Algorithm B, respectively, where m = cluster size, A is list of clusters returned by Algorithm A, B is list of clusters returned by Algorithm B and Z is list of clusters returned by Bansal.

size	minimum weight(A)	maximum weight(A)	$\lvert A \cap Z \rvert$	average weight($A \cap Z$)
2	1	283	45	4.22
3	4	605	15	9.07
4	8	992	7	12.43
5	12	1,407	1	22.00
7	24	2,261	1	26.00
10	50	3,565	1	100.00

Table 3: Metrics from the Output of Algorithm A

Out of $\binom{1,369}{m}$ possible clusters, for each cluster size m, Algorithm A returned a list of 1,369 clusters sorted in ascending order according to weight. The basis is the m minimum entries for each row. The second and third columns of Table 3 show the weights of the gene clusters at the top and the bottom of that list, respectively. The space is not sufficient to list the actual clusters with their corresponding weights, but clearly the gene clusters of Bansal that are in these sorted lists are relatively on the top part of each list as the average weight of these clusters is relatively near the minimum weight, for each cluster size. For instance, for $m = 4$, the minimum weight is 8 and the maximum weight is 992, but the average weight of gene clusters of Bansal which are in the list is only 12.43.

size	minimum freq(B)	maximum freq(B)	$\lvert B \cap Z \rvert$	average freq($B \cap Z$)
2	1	2	45	1.98
3	1	3	15	2.93
4	1	4	7	4.00
5	1	5	1	5.00
7	1	7	1	7.00
10	1	10	1	10.00

Table 4: Metrics from the Output of Algorithm B

Instead of gene cluster weights, Algorithm B returned a list clusters sorted in descending order according to the frequency of each gene cluster in the list of 1,369 candidate clusters. As is seen in

Table 4, the maximum frequency of a gene cluster of size m is also m and the minimum is 1. Clearly, the average frequency of the gene clusters of Bansal for each cluster size m is practically equal to m.

These experimental results show that many of the gene clusters discovered by Bansal in his study are also the gene clusters corresponding to the approximate minimum edge-weighted cliques derived from the complete graph G_K constructed from accumulated gene-gene distances across genome.

One advantage of using Algorithm B, however, is that since frequency values are limited, they can be considered as a way of "clustering" the candidate gene clusters. Candidate gene clusters can be partitioned into m groups, since there are m possible frequency values, from $m, m-1, m-2, \ldots$ down to 1. In this particular study, the gene clusters of Bansal were generally part of the top cluster for each value of m. With the exception of the case when $m = 2$ and $m = 3$, where in both cases only 1 of the gene clusters fell into the second to the highest frequency, all of the gene clusters of Bansal were in groups with the highest frequency.

6 CONCLUSION

The Weighted Clique Problem (WCP) is the problem of finding a clique of the graph G of order m with the smallest (largest) weight. A variant of WCP is the Minimum Edge-Weighted Clique Problem (MIN-EWCP) where the vertex weights are all 0 and the problem is that of finding the clique of the graph G of order m with the least weight. In this study, we presented the problem of identifying sets of commonly existing putative co-regulated, co-expressed genes, called gene clusters, as a clique-finding problem, specifically MIN-EWCP. Initially, the algorithm used for finding gene clusters, which is clique-finding algorithm presented by Eremin et. al., was reviewed and results on performance guarantees for two input classes for the said algorithm was presented. Another algorithm was presented along with its approximation ratio for the metric case. The formulation of the problem as a gene cluster discovery problem was then presented with the construction of a gene distance matrix from a genome database by obtaining the distances of each pair of genes in each genome. It was then shown that the problem of finding approximate gene clusters of size m in GD is equivalent to the Minimum Edge-Weighted Clique Problem. Finally, experimental results were shown where approximate gene clusters are obtained from putative orthologous genes of the genomes $E.colistrK-12$ and $B.subtilis$. These results show that many of the gene clusters discovered by Bansal in his study are also the gene clusters corresponding to the approximate minimum edge-weighted cliques derived by the algorithms. It is hoped that implementations of the featured approximation algorithms would be helpful to biologists, biochemists and clinicians to narrow down the long list of possible gene clusters of interest which they are to further explore for biological significance.

ACKNOWLEDGEMENTS

G. Solano is supported by the Engineering Research and Development for Technology (ERDT) Scholarship Program of the Department of Science and Technology(DOST) and the Doctoral Sandwich Scholarship Program of the Commission on Higher Education (CHEd) Philippines.

REFERENCES

[1] J. A. Aborot, H. Adorna, J. B. Clemente, B. K. de Jesus and G. Solano. Search for a Star: Approximate Gene Cluster Discovery Problem (AGCDP) as a Graph Problem. Philippine Computing Journal, vol.7 no.2 (2012)

[2] B. Alidaee, F. Glover, G. K. H. W. Solving the maximum edge weight clique problem via unconstrained quadratic programming. Eur. J. Oper. Res. 181, 2 (2007), 592-597.

[3] J. G. Augustson, and J. Minker. An Analysis of Some Graph Theoretical Cluster Techniques. Journal of the ACM (JACM), vol. 17, no. 4, pp. 571-588, October (1970)

[4] A. K. Bansal, An automated comparative analysis of 17 complete microbial genomes, Bioinformatics, vol. 15, no.11. pp.900-908 (1999)

[5] A. K. Bansal, A framework of automated reconstruction of microbial metabolic pathways, in Bio-Informatics and Biomedical Engineering, 2000. Proceedings. IEEE International Symposium on, pp. 184-190, IEEE, (2000)

[6] E. Bierstone. Cliques and Generalized Cliques in a Finite Linear Graph. Unpublished Report, (1960s)

[7] I. Bomze, M. Budinich, P. Pardalos, and M. Pelillo. The maximum clique problem. In D.-Z. Du and P. M. Pardalos, editors, Handbook of Combinatorial Optimization, volume 4. Kluwer Academic Publishers, (1999)

[8] C. Bron and J. Kerbosch. Finding All Cliques in an Undirected Graph. Communications of the ACM, vol. 16, pp. 575-577(1973)

[9] J. R. Brown, Comparative genomics: basic and applied research, CRC Press (2007)

[10] I.I. Eremin, E. Kh. Gimadi, A. V. Kel'manov, A. V. Pyatkin, and M. Yu. Khachai, 2-Approximation algorithm for finding a clique with minimum weight of vertices and edges, Proc. Steklov Inst. Math. 284(Suppl. 1), S87-S95 (2014)

[11] E.K. Gimadi, A.V. Kel'manov, A. V. Pyatkin, and M. Yu Khachai.Efficient algorithms with performance guarantees for some problems of finding several cliques in a complete undirected weighted graph. Proc. Steklov Inst. Math. 289(Suppl 1) 88. (2015)

[12] O. Goldschmidt, D.S. Hochbaum, C. Hurkens, G. Yu. Approximation algorithms for the k-clique covering problem. SIAM Journal on Discrete Mathematics. v.9 n.3, p.492-509, (1996)

[13] R. Gupta, J. Walrand and O. Goldschmidt. Maximal cliques in unit disk graphs: Polynomial approximation. In Proceedings INOC (2005)

[14] F. Harary, and I. C. Ross. A Procedure for Clique Detection Using the Group Matrix. Sociometry, vol. 20, pp. 205-215. (1957)

[15] J. Hastad, Clique is hard to approximate within $n^{1-\epsilon}$, Acta Math. 182 (1), 105-142 (1999)

[16] D. Kumlander, A new exact algorithm for the maximum-weight clique problem based on a heuristic vertex-coloring and a backtrack search. In: Proc. 5th Int. Conf. on Modelling, Computation and Optimization in Information Systems and Management Sciences. pp. 202-208 (2004)

[17] J. Lawrence, Jeffrey, Selfish operons: the evolutionary impact of gene clustering in prokaryotes and eukaryotes, Current opinion in genetics & development, Elsevier, vol. 9 . no. 6, pp. 642–648, (1999)

[18] G. He, J. Liu and C. Zhao, Approximation algorithms for some graph partitioning problems, Journal of Graph Algorithms and Applications, vol. 4, no. 2, pp. 1-11 (2000)

[19] W. Pullan, Approximating the maximum vertex/edge weighted clique using local search, J. Heuristics, vol. 14, no. 2, pp. 117-134 (2008)

[20] S. Rahmann, G. Klau, Integer Linear Programming Techniques for Discovering Approximate Gene Clusters, Bioinformatics Algorithms, Techniques and Applications, Wiley-Interscience (2008)

[21] G. Solano, G. Blin, M. Raffinot, J. Caro, A Clique Finding Algorithm for the Approximate Gene Cluster Discovery Problem, Theory and Practice of Computation, pp. 72-88 (2018)

[22] M. Sorensen M, New facets and a branch-and-cut algorithm for the weighted clique problem. European J. Oper. Res., 154:57-70 (2004)

[23] S. Tsukiyama, M. Ide, H. Ariyoshi and I. Shirakawa.A New Algorithm for Generating all the Maximal Independent Sets. SIAM Journal of Computing, vol. 6, pp. 505-517 (1977)