A Decomposition Approach for Solving Critical Clique Detection Problems*

Jose L. Walteros and Panos M. Pardalos

Center for Applied Optimization,
Department of Industrial and Systems Engineering, University of Florida,
303 Weil Hall, Gainesville, FL, USA
jwalteros@ufl.edu, pardalos@ise.ufl.edu
http://www.ise.ufl.edu/cao

Abstract. The problem of detecting critical elements in a network involves the identification of a subset of elements (nodes, arcs, paths, cliques, etc.) whose deletion minimizes a connectivity measure over the induced network. This problem has attracted significant attention in recent years because of its applications in several fields such as telecommunications, social network analysis, and epidemic control. In this paper we examine the problem of detecting critical cliques (CCP). We first introduce a mathematical formulation for the CCP as an integer linear program. Additionally, we propose a two-stage decomposition strategy that first identifies a candidate clique partition and then uses this partition to reformulate and solve the problem as a generalized critical node problem (GCNP). To generate candidate clique partitions we test two heuristic approaches and solve the resulting (GCNP) using a commercial optimizer. We test our approach in a testbed of 13 instances ranging from 25 to 100 nodes.

Keywords: Critical element detection, critical clique detection, clique partitioning.

1 Introduction

The problem of detecting *critical elements* (nodes, arcs, paths, clusters, cliques, etc.) in a network has recently become a major endeavor. Identifying these elements can be crucial for studying many structural characteristics of a network such as connectivity, centrality, robustness, and vulnerability, as well as for identifying dominant clusters and/or partitions.

There is a wide variety of applications for which the detection of critical elements may be of great value. For example, analyzing beforehand how well a network would perform under certain disruptive events plays a vital role in the design and the operation such network. In order to detect vulnerability issues, it is particularly important to analyze how well connected a network remains after a disruptive event takes place destroying or impairing a set of elements in the

* This research is partially supported by NSF, DTRA and DURIP grants.

R. Klasing (Ed.): SEA 2012, LNCS 7276, pp. 393–404, 2012.

network. The main strategy is to identify which is the set of critical elements that must be protected or reinforced in order to mitigate the negative impact that the absence of such elements may produce in the network. Applications of this kind arise in many different contexts and fields such as in social networks analysis [4], homeland security [12], evacuation planning [16], immunization strategies [21], transportation [15], and power grid construction [19], among others.

In general, most of the critical element detection problems fall into the following definition. Given a connected undirected network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ are the set of nodes and edges, respectively, the critical element detection problem involves finding a subset of elements $\mathcal{A} = \{1, \ldots, k\}$ $(k < |\mathcal{V}|)$ such that its deletion minimizes a given connectivity measure over $\mathcal{G}$.

Several measures have been used to assess the level of disconnection of the residual network. There are mainly two classes in which these measures can be categorized. The measures from the first class can be associated mostly with network flow problems (e.g., shortest path problems and maximum flow problems) [5,10,16,22]. For these cases, the critical elements are the ones whose deletion results in the maximum increase of the shortest path, or consequently, the maximum decrease of the flow capacity between two predefined nodes s and t. This kind of measures are commonly used in the context of network interdiction, and are generally designed to tackle arc interdiction problems (detecting critical arcs).

On the other hand, the measures of the second class are associated with topological characteristics of the network. For example, one can account for the total number of pairwise connections (i.e., the total number of node pairs that are connected in the network by at least one path) [2,7], the total cost of pairwise connectivity (i.e., a weighted sum of the pairwise connections) [2,7], the size of the largest connected component (i.e., the number of nodes that belong to the largest maximal connected subgraph of $\mathcal{G}$) [17,20], and the total number of connected components [1,20]. The measures of this class are the ones that we will consider in this work.

Among all the critical element detection problems, the one of detecting critical nodes (CNP) is the one that has attracted more attention. From the complexity point of view, the CNP is proven to be $\mathcal{NP}$-hard on general networks for most of the connectivity measures described above [2,7,8]. There are few cases, though, for which the CNP is solvable in polynomial time (see, [7,20]). Existing methodologies for solving the CNP include heuristics (and metaheuristics) [1,2,4], mathematical programing [2,5,10,16,17,22], dynamic programing [7,20], approximated algorithms [8], and simulation approaches [14].

A simple heuristic approach regarding the CNP was explored by Albert et al. [1]. This work aims at analyzing the tolerance of complex networks with respect to strategic node deletions. Instead of finding the collection of nodes that must be removed to impair the connectivity of the network, the authors analyze the resulting consequences over the network when (i) a set of randomly chosen nodes is removed and (ii) when the nodes with large degree are removed.

Recent endeavors using mathematical programing techniques can be found in [2] and [17]. In their work, Aurslevan et al. [2] provide a prove of the $\mathcal{NP}$-hardness

of the CNP for the pairwise connectivity measure. They also introduce a linear integer formulation and a fast constructive heuristic. An alternative formulation was presented in [17]. In this paper, the authors provide a detailed polyhedral analysis for different valid inequalities as well as a branch-and-cut framework.

The use of dynamic programming has been studied by Shen and Smith [20] and Di Summa et al. [7]. In both studies, the authors provide a detailed complexity analysis of the CNP over trees and other structures. They also prove that the cardinality version of some CNP variations over trees are polynomially solvable via dynamic programing.

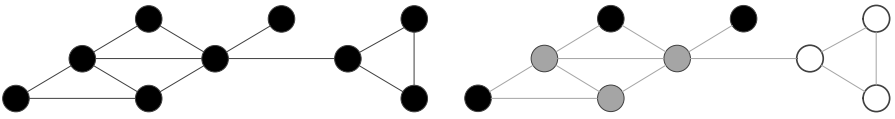
From the approximation algorithms perspective, a variation of the CNP problem is presented in [8]. In this work, the authors propose a reformulation for the CNP where the objective function is set to minimize the number of nodes (or edges) that must be removed in order to achieve a certain degradation (disruption) in the connectivity of the network. In addition to these reformulations, a thorough complexity and inapproximability analysis is presented as well as a pseudo-approximation scheme.

The main purpose of this paper is to extend the scope of previous works related with the CNP, and analyze the problem of detecting critical cliques on networks. We organize this paper as follows. In Section 2, we introduce the critical clique detection problem (CCP) including a complexity analysis regarding the $\mathcal{NP}$-completeness of the CCP. We also introduce a integer linear formulation and its respective variations for two of the connectivity measures described above. In Section 3, we present a decomposition approach for solving the CCP. The proposed approach is based on a reduction from the CCP to a generalized critical node problem (GCNP) by means of a clique partitioning problem. We also present two algorithms that can be used to obtain candidate clique partitions, as well as a formulation for the GCNP that is used to solve the resulting problem. In Section 4, we present our computational results, and finally, in Section 5 we provide conclusions an further directions for subsequent projects.

2 The Critical Clique Detection Problem (CCP)

Given a connected undirected network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ and $\mathcal{E}$ are the set of nodes and edges, respectively, and an integer k, the CCP involves finding a set of k disjoint cliques such that its deletion results in the maximum network disconnection. Additional constraints regarding the structure of the cliques can also be imposed, for instance, an upper bound on the size of the critical cliques. Notice that the CCP can be seen as a generalization of the CNP, where the objective is to find cliques instead of nodes. The CNP is then the case where the size of the cliques is limited to be one. Figure 1 presents an example of the CCP over a 9-node graph, where $k = 2$. Figure 1(a) displays the original network, and Figure 1(b) the optimal solution where the cliques selected are colored in gray and white.

Among the different connectivity measures described above which can be used as objective functions we discuss two: the total number of pairwise connections and the size of the largest component. A description of these objectives follows:



(a) Original graph (b) Optimal solution

Fig. 1. Example for a 9-node graph

2.1 Objective Functions

Before presenting the objectives, we need to introduce the following definitions. For any subset $\mathcal{V}' \subseteq \mathcal{V}$ we define $\mathcal{E}(\mathcal{V}') \subseteq \mathcal{E}$ as the set of edges such that, for each edge $e \in \mathcal{E}(\mathcal{V}')$, both endpoints of e belong to $\mathcal{V}'$. We also define the induced graph $\mathcal{G}(\mathcal{V}')$ as the graph comprised by the set of nodes $\mathcal{V}'$, and the set of edges $\mathcal{E}(\mathcal{V}')$. We assume that two nodes $i, j \in \mathcal{V}$ are connected over $\mathcal{G}$ if there exist at least one path that connects i with j in $\mathcal{G}$. Let $\mathcal{Q}$ be the set of maximal connected components of $\mathcal{G}$. We define a maximal connected component $\mathcal{C}_q \in \mathcal{Q}$ as a subset $\mathcal{C}_q \subseteq \mathcal{V}$ of nodes such that every pair of nodes $i, j \in \mathcal{C}_q$ is connected over $\mathcal{G}(\mathcal{C}_q)$, and such that, for every node $l \in \mathcal{V} \setminus \mathcal{C}_q$, there is no edge connecting l with any node $i \in \mathcal{C}_q$. From now on we will refer to the maximal connected components only as components unless otherwise stated. Let $\sigma_q = |\mathcal{C}_q|$ be the number of nodes of component $\mathcal{C}_q \in \mathcal{Q}$. We define the number of pairwise connections of component $\mathcal{C}_q \in \mathcal{Q}$ as $\binom{\sigma_q}{2} = \sigma_q(\sigma_q - 1)/2$. Let $\mathcal{T} = \{\mathcal{K}_1, \ldots, \mathcal{K}_k\}$ be the set of k critical cliques of $\mathcal{G}$, $\mathcal{V}(\mathcal{K}_t) \subseteq \mathcal{V}$ be the subset of nodes that comprise clique $\mathcal{K}_t \in \mathcal{T}$, and $\mathcal{V}(\mathcal{T}) \subseteq \mathcal{V}$ be the set of all the nodes that belong to the critical cliques. Finally, let $\mathcal{G}^{\mathcal{T}} = (\mathcal{V} \setminus \mathcal{V}(\mathcal{T}), \mathcal{E}(\mathcal{V} \setminus \mathcal{V}(\mathcal{T})))$ be the resulting network after the deletion of the critical cliques, and $\mathcal{Q}^{\mathcal{T}}$ the corresponding set of remaining components. The definitions of the two objectives used follow:

Minimize the Total Pairwise Connectivity (TPW): Given a network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and an integer k, we try to find a collection of cliques $\mathcal{T}$, of size $|\mathcal{T}| \leq k$, such that the sum of the pairwise connections of all the components left is minimized:

$$\min \sum_{q \in \mathcal{Q}^{\mathcal{T}}} \sigma_q(\sigma_q - 1)/2 \tag{1}$$

Minimize the Size of the Largest Component (MinMS): Given a network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and an integer k, we try to find a collection of cliques $\mathcal{T}$, of size $|\mathcal{T}| \leq k$, such that the size of the largest component is minimized:

$$\min \max_{q \in \mathcal{Q}^{\mathcal{T}}} \{\sigma_q\} \tag{2}$$

2.2 $\mathcal{NP}$-Completeness of the CCP

We now prove that the decision version of the CCP problem is $\mathcal{NP}$-complete. The decision version of the CCP, defined as the α–CCP can be stated as follows.

Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a positive integer k, is there a set of disjoint cliques $\mathcal{T} = \{\mathcal{K}_1, \mathcal{K}_2, \ldots, \mathcal{K}_t\}, t \leq k$ such that the size of the largest component of $\mathcal{G}^{\mathcal{T}}$ is at most α? Note that in this case we use expression (2) as the connectivity measure, although, we could easily adapt this result for (1) as well.

Clearly, the α–CCP belongs to the class $\mathcal{NP}$ for any given $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Notice that given a collection of cliques $\mathcal{T}$ in $\mathcal{G}$, identifying the size of each component of $\mathcal{G}^{\mathcal{T}}$ can be done in polynomial time by means of a breadth first search algorithm [13].

To prove that the α–CCP belongs to the $\mathcal{NP}$–complete class, we propose the following reduction from the *clique partitioning problem* known to be $\mathcal{NP}$–complete [9]. The clique partitioning problem is defined as follows: Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a positive integer k, is it possible to partition set $\mathcal{V}$ into $t \leq k$ disjoint cliques $\mathcal{K}_1, \mathcal{K}_2, \ldots, \mathcal{K}_t$?

It can be easily argued that the α–CCP generalizes the partition into cliques problem. Note that for $\alpha = 0$, there is a collection of at most k cliques $\mathcal{K}_1, \ldots, \mathcal{K}_k$ such that every component left in the network has size zero (i.e., $|\mathcal{Q}^{\mathcal{T}}| = 0$) if and only if, every vertex in $\mathcal{V}$ belongs to one of the following subsets induced by the critical cliques $\mathcal{V}(\mathcal{K}_1), \mathcal{V}(\mathcal{K}_2), \ldots, \mathcal{V}(\mathcal{K}_k)$. Thereby, the α–CCP is $\mathcal{NP}$-complete.

2.3 CCP formulations

When studying combinatorial problems, using a linear integer formulation is in general a natural starting point. Despite the inherent difficulty of these problems, techniques such as branch and bound, branch and cut, and others are proven to be very efficient approaches to obtain solutions for instances of manageable size. We now present an integer linear formulation for the CCP as well as the respective modifications to tackle the two objectives proposed above.

Let $\mathcal{V}(e)$ be the set of endpoints of edge $e \in \mathcal{E}$ and $\mathcal{T}$ be the set of critical cliques such that $|\mathcal{T}| = k$. Let x_i^t be a binary variable that takes the value of one if node i is assigned to clique $\mathcal{K}_t \in \mathcal{T}$, and zero otherwise. Let y_{ij} be a binary variable that takes the value of one if nodes i and j, belong to the same component in the residual graph, and zero otherwise. Let z_i be an auxiliary binary variable that takes the value of one if node i does not belong to a critical clique, and zero otherwise. The formulation for the CCP for the TPW objective is as follows:

$$\min \sum_{i,j \in \mathcal{V}} y_{ij} \tag{3}$$

$$\text{s.t. } x_i^t + x_j^t \leq 1 \qquad e \in \mathcal{V} \times \mathcal{V} \setminus \mathcal{E}, i, j \in \mathcal{V}(e), t \in \mathcal{T} \tag{4}$$

$$z_i + \sum_{t \in \mathcal{T}} x_i^t = 1 \qquad i \in \mathcal{V} \tag{5}$$

$$y_{ij} \geq z_i + z_j - 1 \qquad e \in \mathcal{E}, i, j \in \mathcal{V}(e) \tag{6}$$

$$y_{ij} + y_{jl} - y_{il} \leq 1 \qquad i, j, l \in \mathcal{V} \tag{7}$$

$$y_{ij} - y_{jl} + y_{il} \leq 1 \qquad i, j, l \in \mathcal{V} \tag{8}$$

$$-y_{ij} + y_{jl} + y_{il} \leq 1 \qquad i, j, l \in \mathcal{V} \tag{9}$$

$$x_i^t \in \{0,1\} \qquad i \in \mathcal{V}, t \in \mathcal{T} \tag{10}$$

$$z_i \in \{0,1\} \qquad i \in \mathcal{V} \tag{11}$$

$$y_{ij} \in \{0,1\} \qquad i,j \in \mathcal{V} \tag{12}$$

where the objective function (3) minimizes the sum of pairwise connections. Note that since y_{ij} is equal to 1 if nodes i and j belong to the same component, $\sum_{i,j \in V} y_{ij}$ is equivalent to $\sum_{q \in \mathcal{Q}^\tau} \sigma_q(\sigma_q - 1)$. Constraint (4) ensures that if there is no edge $e \in \mathcal{E}$ between nodes i and j (i.e., $e \in \mathcal{V} \times \mathcal{V} \setminus \mathcal{E}$), both nodes cannot be assigned to the same clique. Constraint (5) ensures that if node i is not assigned to a clique, its corresponding variable z_i must be equal to one. Constraints (6) define the relationship between y variables and z variables. Constraints (7–9) define the triangular relationship of y variables (i.e., if in the residual network node i is connected to node j and node j is connected to node k, then node i must also be connected to node k). And finally constraints (10–12) define the domain of the variables used. We will refer to this problem as CCP–TPW.

To use the MinMS as the objective, we can adapt the proposed model by introducing a new variable β defined as the size of the largest component. Then the model can be formulated as follows:

$$\min \quad \beta \tag{13}$$

$$\text{s.t.} \quad (4-12)$$

$$\sum_{i \in V} y_{ij} \leq \beta \qquad i \in \mathcal{V} \tag{14}$$

where objective function (13) combined with constraints (14) enforces the minimization of the size of the largest component. We will refer to this problem as CCP–MinMS.

Formulations CCP–TPW and CCP–MinMS are relatively large in size with respect to the size of the network (they require $O(|\mathcal{V}|^2)$ variables and $O(|\mathcal{V}|^3)$ constraints). To solve these formulations, it is common to use a cutting plane generation scheme that sequentially includes constraints (7–9) as needed. Moreover, it is easy to see that we can strengthen these formulations using some valid inequalities originally designed for similar problems [11,17], as well as symmetry breaking constraints.

As an alternative solution approach, we also provide a decomposition strategy for the CCP.

3 Decomposition Approach for Solving the CCP

The decomposition strategy proposed in this paper is based on the following theorem:

Theorem 1. *The set of critical cliques of any feasible solution* $(\mathbf{x}, \mathbf{y}, \mathbf{z})$ *belongs to at least one clique partition of the original network* $\mathcal{G}$.

Proof. Let $\mathcal{T}$ be the set of critical cliques of solution $(\mathbf{x}, \mathbf{y}, \mathbf{z})$, and $\mathcal{V}(\mathcal{T})$ be the set of nodes comprising the cliques. Let $\bar{\mathcal{R}}$ be any clique partition of the residual network $\mathcal{G}(\mathcal{V} \setminus \mathcal{T})$. Note that $\mathcal{R} = \mathcal{T} \cup \bar{\mathcal{R}}$ is a clique partition of $\mathcal{G}$ as $\mathcal{T}$ and $\bar{\mathcal{R}}$ are two disjoint sets of cliques that cover all the nodes in $\mathcal{G}$. □

Since every set of critical cliques can be associated with a clique partition, we propose to solve the CCP by: (i) generating a clique partition, (ii) collapsing each clique of the given partition into a single node forming a network $\mathcal{H}$, and (iii) using an exact or heuristic method, for solving a generalized CNP over $\mathcal{H}$ (see Algorithm 1). We now analyze each of the steps of this approach.

Algorithm 1. `CCPCollapseAlgorithm`($\mathcal{G}$)

$\mathcal{R} \leftarrow$ generate a clique partition
$\mathcal{H} \leftarrow$ `collapse`($\mathcal{R}$)
$\mathcal{T}^* \leftarrow$ `SolveGeneralizedCNP`($\mathcal{H}$)
return $\mathcal{T}^*$

3.1 Constructing Clique Partitions

The main component of this approach is the way in which the clique partition is generated. This is because, in order to obtain a good solution, we would like to generate a clique partition containing the optimal set of critical cliques (or at least a good proxy). We propose to heuristically generate candidate clique partitions. The idea behind our approach is that if we want to greedily reduce the number of pairwise connections, we can either aim at eliminating a large clique, or a clique with a large degree (i.e., a clique with many edges emanating from it), we propose two different algorithms for partitioning the network following this analysis.

The first approach approach is to use as a clique partition the solution of a maximum edge clique partition problem (Max-ECP). The Max-ECP problem looks for a clique partition that maximizes the number of edges within the cliques. Even though the Max-ECP is proven to be $\mathcal{NP}$–hard, there are several approximation algorithms to solve this problem. We decided to use the 2-approximation algorithm proposed by [6] that we called `MaxECP` (see Algorithm 2). Since Algorithm 2 requires solving sequentially a maximum clique problem, we used the approximation algorithm proposed in [3]. Note that it is also possible to get both, the clique partitioning or/and the maximum clique, by solving the corresponding mathematical problems, or by means of any other technique (exact or heuristic).

For the second approach, we propose to use a clique partition based on the degree of the cliques. We use a heuristic that greedily finds a clique with a large degree in $\mathcal{G}$ (see Algorithm 3). Once we find this clique, we remove it from the network and continue following the same process until all the nodes are eliminated (see Algorithm 4).

Algorithm 2. `MaxECP`$(\mathcal{G})$ `[6]`

$\mathcal{T} \leftarrow \emptyset$
repeat
  Select the maximum clique $\bar{\mathcal{K}}$ in $\mathcal{G}(\mathcal{V} \setminus \mathcal{V}(\mathcal{T}))$.
  $\mathcal{T} \leftarrow \bar{\mathcal{K}} \cup \mathcal{T}$
  $\mathcal{G} \leftarrow \mathcal{G}(\mathcal{V} \setminus \mathcal{V}(\mathcal{T}))$
until $\mathcal{G} = \emptyset$
return $\mathcal{T}$

Algorithm 3. `GreedyGetClique`$(\mathcal{G})$

$\mathcal{K} \leftarrow \emptyset$
repeat
  Select vertex i with maximum degree in the subgraph induced by $\mathcal{G}$.
  $\mathcal{K} \leftarrow \mathcal{K} \cup \{i\}$
  $\mathcal{N}(i) \leftarrow$ neighbors of i
  $\mathcal{G} \leftarrow \mathcal{G} \cap \mathcal{N}(i)$
until $\mathcal{G} = \emptyset$
return $\mathcal{K}$

Algorithm 4. `MaxDegree`$(\mathcal{G} = (\mathcal{V}, \mathcal{E}))$

$i \leftarrow 1$
while $V \neq \emptyset$ **do**
  $\mathcal{K}_t \leftarrow$ `GreedyGetClique`$(\mathcal{V})$
  $\mathcal{V} \leftarrow \mathcal{V} \setminus \mathcal{K}_t$
  $t \leftarrow t + 1$
end while

3.2 Clique Collapsing

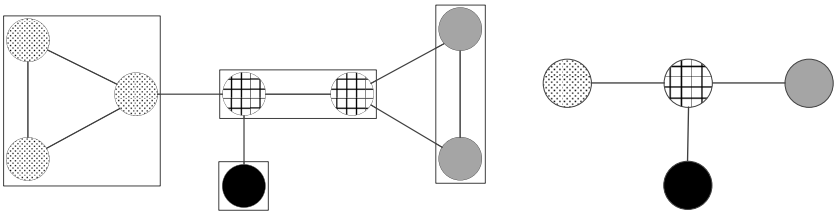
First, assume that we have a clique partition $\mathcal{R} = \{\mathcal{K}_1, \mathcal{K}_2, \ldots, \mathcal{K}_l\}$. We can collapse each of the cliques in $\mathcal{R}$ into a single node. Let $\mathcal{H}$ be a network comprised by these nodes. Let $\mathcal{V}^{\mathcal{R}}$ be the set of nodes representing the cliques and $\mathcal{E}^{\mathcal{R}}$ be the edges connecting the nodes in $\mathcal{V}^{\mathcal{R}}$. There exists an edge (i, j) in $\mathcal{E}^{R}$ if there exists at least one edge in $\mathcal{E}$ connecting a node in K_i with a node in $\mathcal{K}_j$. Let $\mathcal{H} = (\mathcal{V}^{\mathcal{R}}, \mathcal{E}^{\mathcal{R}})$ be the network induced by the collapsed nodes. Finally, let $s(\mathcal{K}_i)$ be the size of clique $\mathcal{K}_i$. Figure 2 provides an example of the clique collapsing, given a clique partition.

3.3 CNP Generalization for Solving the CCP

Assuming that we have a partition $\mathcal{R}$, once we have the collapsed network $\mathcal{H}$ we can obtain the solution of the CCP by solving a generalized CNP problem. We will discuss only the reformulation for the CCP-TPW case, although, this result can be trivially extended for the CCP-MinMS.

Notice that if we want to count the total number of pairwise connections in $\mathcal{H}$, we need to take into account the connections at the interior of each node in

$\mathcal{V}^{\mathcal{R}}$ (recall that at the point, each clique is now represented by a node), as well as the connection associated with each edge in $\mathcal{E}^{\mathcal{R}}$. For the sake of clarity, we abuse the notation in this formulation using i and j when referring to the collapsed nodes in $\mathcal{V}^{\mathcal{R}}$ and by defining x_i as a binary variable that takes the value of one if collapsed node i is removed and zero otherwise. Within each clique $\mathcal{K}_i \in \mathcal{R}$, the total number of connections is given by $p_i = \binom{s(\mathcal{K}_i)}{2} = (s(\mathcal{K}_i)(s(\mathcal{K}_i) - 1)/2)$. Moreover, note that if nodes i and j are connected in $\mathcal{H}$, the number of pairwise connections represented by edge $(i, j) \in \mathcal{E}^{\mathcal{R}}$ is given now by $t_{ij} = s(K_i)s(K_j)$. Thus, the generalized formulation for the CNP follows.



(a) Clique Partition (b) Clique Collapse

Fig. 2. Clique collapse

$$\min \sum_{i \in \mathcal{V}^{\mathcal{R}}} p_i(1 - x_i) + \sum_{i,j \in \mathcal{V}^{\mathcal{R}}} t_{ij} y_{ij} \tag{15}$$

$$\text{s.t. } y_{ij} + x_i + x_j \geq 1 \qquad \forall (i,j) \in \mathcal{E}^{\mathcal{R}} \tag{16}$$

$$y_{ij} + y_{jk} - y_{ki} \leq 1 \qquad \forall (i,j,k) \in \mathcal{V}^{\mathcal{R}} \tag{17}$$

$$y_{ij} - y_{jk} + y_{ki} \leq 1 \qquad \forall (i,j,k) \in \mathcal{V}^{\mathcal{R}} \tag{18}$$

$$-y_{ij} + y_{jk} + y_{ki} \leq 1 \qquad \forall (i,j,k) \in \mathcal{V}^{\mathcal{R}} \tag{19}$$

$$\sum_{i \in \mathcal{V}^{\mathcal{R}}} x_i \leq k \tag{20}$$

$$x_i \in \{0,1\} \qquad \forall i \in \mathcal{V}^{\mathcal{R}} \tag{21}$$

$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in \mathcal{V}^{\mathcal{R}} \tag{22}$$

where objective (15) accounts for the minimization of the total pairwise connections taking into account the connections at the interior of the cliques. Constraints (16–22) are defined exactly as in [2].

4 Computational Experiments

We tested efficacy of our approach on 13 randomly generated networks ranging in size from 25 to 100 nodes. All the networks were generated using the algorithm

proposed by Palmer and Steffan [18] such that the degree of the nodes follows a power-law distribution. We solved the IP formulations and applied the decomposition strategy for the CCP-TWP and the CCP-MinMS for different values of k. We solved first the IP formulations using the commercial optimizer CPLEX 12.0, fixing a time limit of four hours (14,400 seconds). We also implemented both the `MaxDegree` and the `MaxECP` algorithms to generate the clique partitions and used CPLEX to solve the resulting GCNP formulations. The computational results are listed in Table 1. For the cases in which the optimizer fails to obtain an optimal solution within the time frame, we provide the best integer solution found.

Table 1. Computational results. The optimal solutions are listed in bold and the time is described in seconds. (*) indicates that the optimizer was not able to find an integer solution within the time limit

			CCP-TPW						CCP-MinMS					
			IP Form.		`MaxDeg`		`MaxECP`		IP Form.		`MaxDeg`		`MaxECP`	
$\|\mathcal{V}\|$	$\|\mathcal{E}\|$	k	Best	Time	Best	Time	Best	Time	Best	Time	Best	Time	Best	Time
25	75	3	**80**	381.03	105	0.59	120	0.16	**13**	658.33	15	0.55	16	0.20
25	75	5	**25**	96.52	36	0.17	39	0.08	**6**	184.51	9	0.25	9	0.14
25	75	7	**2**	14,400	4	0.05	5	0.05	**1**	8.23	3	0.14	3	0.08
50	100	5	**398**	9,730.05	497	126.48	535	89.94	**28**	7,125.01	32	1,482.40	33	450.59
50	100	10	**18**	3,470.09	26	1.16	76	9.94	**2**	6,256.77	4	7.01	9	34.58
50	150	5	502	14,400	561	47.69	561	17.77	49	14,400	34	247.77	34	62.86
50	150	10	**54**	12,701.60	76	3.73	114	6.32	10	14,400	10	16.22	13	20.67
75	150	10	1,545	14,400	836	465.68	947	270.75	52	14,400	41	14,153.12	43	12,981.10
75	150	15	423	14,400	50	13.37	132	640.41	4	14,400	6	461.36	11	13,984.49
75	200	20	50	14,400	7	3.35	39	31.74	**1**	13,7028.50	3	63.72	5	306.00
100	200	15	1,152	14,400	295	6,385.16	412	9,685.28	*	14,400	28	4,295.45	39	6,458.30
100	200	30	*	14,400	6	9.79	17	19.90	*	14,400	2	59.49	3	3,647.48
100	300	30	*	14,400	17	18.23	18	13.49	*	14,400	3	718.34	3	443.96

We were able to obtain optimal solutions for 6 instances out of 13 for both, the CCP-TPW and the CCP-MinMS. Furthermore, note that with the proposed approach, we obtained good solutions for most of the instances. Notice that for the CCP-TPW case, since the total number of pairwise connections grows quadratically with respect to the size of the remaining components, a near optimal solution having just a few additional nodes may have a significantly larger number of pairwise connections compared to the optimal solution.

In terms of the running times, the decomposition approach ran significantly faster than the IP formulation. Moreover, we found that the execution time for finding clique partitions is negligible (less that a second) compared with the time used by CPLEX to solve the GCNP. Note that the time required to solve the GCNP can be significantly reduced by using a simple variation of the heuristic proposed in [2]. Finally, we observe that the clique partitions obtained with MaxDegree yield better results for both objectives.

5 Concluding Remarks

This study was motivated by the increasing interest of solving critical element detection problems. We introduced the problem of identifying critical cliques

(CCP) over networks considering two connectivity measures: the total pairwise connectivity and the size of the largest component. To address this problem, we formulated it as an integer program. In addition, we proposed a decomposition strategy for solving large-scale instances that first generates a clique partition and then reformulates and solves the problem as a generalized critical node problem (GCNP). We introduced two heuristics for obtaining clique partition candidates. The resulting GCNP is then solved using a commercial optimizer. We evaluated the performance of our approach by solving 13 randomly generated instances ranging in size from 25 to 100 nodes.

Future research may involve testing additional methodologies for obtaining clique partitions, as well as testing the performance of the proposed approach when additional constraints over the cliques are included.

References

1. Albert, R., Jeong, H., Barabasi, A.-L.: Error and attack tolerance of complex networks. Nature 406(6794), 378–382 (2000)
2. Arulselvan, A., Commander, C.W., Elefteriadou, L., Pardalos, P.M.: Detecting critical nodes in sparse graphs. Computers and Operations Research 36(7), 2193–2200 (2009)
3. Boppana, R., Halldorsson, M.: Approximating maximum independent sets by excluding subgraphs. BIT 32, 180–196 (1992)
4. Borgatti, S.P.: Identifying sets of key players in a social network. Comput. Math. Organ. Theory 12, 21–34 (2006)
5. Corley, H., Sha, D.Y.: Most vital links and nodes in weighted networks. Operations Research Letters 1(4), 157–160 (1982)
6. Dessmark, A., Jansson, J., Lingas, A., Lundell, E.-M., Persson, M.: On the approximability of maximum and minimum edge clique partition problems. International Journal of Foundations of Computer Science 18 (2006, 2007)
7. Di Summa, M., Grosso, A., Locatelli, M.: Complexity of the critical node problem over trees. Computers and Operations Research 38(12), 1766–1774 (2011)
8. Dinh, T.N., Xuan, Y., Thai, M.T., Pardalos, P.M., Znati, T.: On new approaches of assessing network vulnerability: Hardness and approximation. IEEE/ACM Transactions on Networking PP(99) (2011)
9. Garey, M.R., Johnson, D.S.: Computers and Intractability; A Guide to the Theory of NP-Completeness. W. H. Freeman & Co., New York (1990)
10. Grötschel, M., Monma, C., Stoer, M.: Design of survivable networks. In: Ball, C.M.M.O., Magnanti, T.L., Nemhauser, G. (eds.) Network Models. Handbooks in Operations Research and Management Science, vol. 7, pp. 617–672. Elsevier (1995)
11. Grötschel, M., Wakabayashi, Y.: Facets of the clique partitioning polytope. Mathematical Programming 47, 367–387 (1990)
12. Grubesic, T.H., Matisziw, T.C., Murray, A.T., Snediker, D.: Comparative approaches for assessing network vulnerability. International Regional Science Review 31(1), 88–112 (2008)
13. Hopcroft, J., Tarjan, R.: Algorithm 447: efficient algorithms for graph manipulation. Commun. ACM 16, 372–378 (1973)
14. Houck, D.J., Kim, E., O'Reilly, G.P., Picklesimer, D.D., Uzunalioglu, H.: A network survivability model for critical national infrastructures. Bell Labs Technical Journal 8(4), 153–172 (2004)

15. Jenelius, E., Petersen, T., Mattsson, L.-G.: Importance and exposure in road network vulnerability analysis. Transportation Research Part A: Policy and Practice 40(7), 537–560 (2006)
16. Matisziw, T.C., Murray, A.T.: Modeling s-t path availability to support disaster vulnerability assessment of network infrastructure. Comput. Oper. Res. 36, 16–26 (2009)
17. Oosten, M., Rutten, J.H.G.C., Spieksma, F.C.R.: Disconnecting graphs by removing vertices: a polyhedral approach. Statistica Neerlandica 61(1), 35–60 (2007)
18. Palmer, C., Steffan, J.: Generating network topologies that obey power laws. In: Global Telecommunications Conference, GLOBECOM 2000, vol. 1, pp. 434–438. IEEE (2000)
19. Salmeron, J., Wood, K., Baldick, R.: Analysis of electric grid security under terrorist threat. IEEE Transactions on Power Systems 19(2), 905–912 (2004)
20. Shen, S., Smith, J.C.: Polynomial-time algorithms for solving a class of critical node problems on trees and series-parallel graphs. Networks (2011)
21. Tao, Z., Zhongqian, F., Binghong, W.: Epidemic dynamics on complex networks. Progress in Natural Science 16(5) (2005)
22. Wollmer, R.: Removing arcs from a network. Operations Research 12(6), 934–940 (1964)