Massimo Ancona, Walter Cazzola, Enrico Martinuzzi, Paolo Raffo, and Ioan Bogdan Vasian. Clustering Algorithms for the Optimization of Communication Graphs. In Proceedings of the Fourth Conference Italo-Latino American of Industrial and Applied Mathematics, *pages 328–334, Havana, Cuba, March 2001. ISBN: 959-7056-13-5.*

Clustering Algorithms for the Optimization of Communication Graphs

Massimo Ancona[1], **Walter Cazzola**[1], **Enrico Martinuzzi**[2], **Paolo Raffo**[1], and **Ioan Bogdan Vasian**[3]

[1] DISI - Dipartimento di Informatica e Scienze dell'Informazione
Università di Genova

[2] Motorola ltd.

[3] Rumanian Ministry of Defense

Abstract

One of the main goals in optimizing communication networks is to enhance performances by minimizing the number of message hops, i.e. the number of graph nodes traversed by a message. Most of the optimization techniques are based on clustering, i.e., the network layout is reconfigured in sub-networks. Network clustering has been largely studied in the literature but most of the available algorithms are application dependent.
In this paper we restrict our attention to algorithms based on the location of the median points, in order to build clusters with a balanced number of elements and to minimize communication time. We present two algorithms and relative experimental results about the quality of the computed clusterizations, in terms of the minimum number of computed hops. One algorithm is based on the well-known multi-median heuristic algorithm, while the other adopts a greedy approach, i.e., at each step the algorithm computes clusters farther and farther from each central node.
To the achieved clusterization we apply a further step, which consists in finding a virtual path layout according to Gerstel's (VPPL) algorithm. The adopted criterium for our experimental comparisons is the optimality, in terms of the number of signal hops, of the achieved virtual path layout. The experiments are carried out upon a set of networks representing real environments.

Introduction

The management of signal routing in a large communication network could become a very hard task due to the complexity of the treated networks. The communication cost between a pair of nodes depends on several factors, such as distance between them, link reliability, traffic demands, and so on. To reduce signal latency between a pair of nodes, it is obviously necessary to reduce the number of nodes which the signal has to traverse. This fact requires to design the physical topology of the network minimizing this number. The physical topology of a network is the set of routing nodes and the links connecting them; the logical topology consists of a new set of connections among each pair of physical nodes.
The network topology may be either hierarchical or non-hierarchical. In non-hierarchical networks all nodes are considered to be identical from the point of view of routing, i.e., a path between two nodes can include any other intermediate node. In hierarchical networks nodes are classified in levels, usually two or three, according to their geographical position, number of users, and so on. Usually, a virtual graph is often constructed, where each node represents a sub-graph of the original graph and edges represent relationship among these sets of nodes. The problem consists then in finding a good method to organize the nodes of the network so that the number of hops performed by the signal is minimal. The choice of the logical topology is an important task in network planning and optimization because if the network is large (or it is becoming large in planning section) a completely non-hierarchical topology often becomes difficult to administer due to the large size and complexity of routing tables. The hierarchical topologies tries to simplify the administration task by organizing nodes into different levels. However pure hierarchical topologies may not be desirable because they lead to higher link costs [7].
Several approaches have been proposed in literature, most of them based on the creation of a set of directed *virtual paths* (VP) satisfying some constraints in terms of link load (the number of virtual paths sharing the link) and hop count (the number of VP necessary to establish a connection). In particular, the sum of the capacities of virtual paths that share a physical link is the load of the link; obviously the load cannot exceed the link capacity. Moreover the maximum number of VP to establish a connection cannot grow excessively because the transmission times could increase as well. In general terms the virtual path layout optimization is a problem in which, given a communication demand between pairs of nodes and some constraints on the maximum load and hop count it is required to minimize some function of the load and hop count.
To improve the management of a communication network we should optimize the connections among all possible pair of nodes, which it means to find the minimum number of hops needed to connect a source to a

destination.

1 The Model and Some Definitions

A communication network, consisting in switches and weighted links, is modeled by a connected undirected weighted graph $G = (V, E, c)$, where:

- V is the set of vertices of G,
- E is the set of edges of $G \quad (e \in E \equiv (u, v) \in V \times V)$,
- $c : E \to \mathbb{N}$ associates a capacity to each edge of G.

The problem of graph clustering consists in finding a proper partition of the set of vertices V in subsets $V_1, \ldots, V_n$, called *clusters* of G, such that: $\bigcup_{k=1}^{n} V_k = V$, $V_k \bigcap V_j = \emptyset \quad$ for $i \neq j$.

Definition 1.1 $d : \wp(G) \times V \times V \to \mathbb{N}, \quad d(p, u, v) = \#\{x_i \in V \mid p = (u, x_1, \ldots, x_n = v)\}$ *is a function measuring the number of intermediate vertices along a given path p, between a pair of vertices.*

A path between a pair of nodes is a set of one or more concatenated links connecting the nodes.
For a general path this function is not a distance because the triangular inequality fails; but if we consider as p the shortest path between u and v we obtain a metric distance.

Definition 1.2 *Let $\wp(G)$ the set of all paths in $G \equiv (V, E)$. Given $\overline{E} = V \times V - E$, a virtual path layout is a pair (G', I), where $G' = (V, E')$ is a virtual graph where $E' = E \bigcup \overline{E'}$, with $\overline{E'} \subseteq \overline{E}$ and $I : E' \to \wp(G')$ is a function mapping a virtual edge to its corresponding route in G.*

The above definition can be extended to paths $p \in \wp(G'),\ p = (p_1, p_2, \ldots, p_n), p_i \in E', \forall i = 1, \ldots, n : I(p) = \bigcup_{i=1}^{n} I(p_i)$ obtained by concatenating the induced paths $I(p_i)$. Definition 2. Let $\mu \in \mathbb{R}$ be a real number called *the stretch factor*. The *hop count* $H : V \times V \to \mathbb{R}$ for a pair of vertices (u, v) is the minimum k such that:

- $\exists p = (p_1, p_2, \ldots, p_n) \in \wp(G')$, s.t. $p_i \in E', \forall i = 1, \ldots, n$
- $\exists x, y \in V$, s.t. $p_1 = (u, x), p_n = (y, v)$ and
- $I(p)$ is at most μ times longer than the shortest path between u and v in G.

The number of hops in a path is the number of concatenated links that constitute the path. Usually a path is required to be loopless , i.e. a node cannot be visited twice in a path. A route between a pair of nodes is a set of path, not necessarily disjoint, connecting the two nodes.

Definition 1.3 *The function $\mathcal{N} : V \to \mathbb{N}$, $\mathcal{N}(v) = \#\{p \in E' \mid v \in I(p)\}$ is the Load of a node v, while*
$\mathcal{L} : E \to \mathbb{N}$, $\mathcal{L}(e) = \#\{p \in E' \mid e \in I(p)\}$ is the Load of a physical edge e.

Definition 1.4 *A diameter D of a connected graph G is the maximum length of the shortest paths connecting two arbitrary vertices $u, v \in V$.*

Definition 1.5 *A median of a graph is a vertex such that the sum of its distance from all other vertices is minimized;*

a k-set of medians W, is a set of k vertices of V such that the sum of distances between all other vertices and W is the minimum among all the possible sets of k vertices of G.

2 Clusterization Algorithms

The problem of finding an optimal connection among all possible pairs of nodes of a network has been proven to be NP-complete [4]. So we consider a simpler case: the optimization of the connection between each node with a single one. Following the method described in [5] we try to partition the network in a proper number of subnetworks, also called clusters; then we connect each of them to the nearest and to the global center of the network. Then, the new problem consists in connecting each pair of nodes belonging to each cluster to k hops at most. Hence the maximum number of hops needed to connect two generic nodes is:

$$h = 2k + n,$$

where n is the number of clusters we have to traverse in order to connect the cluster containing the source node to the cluster containing the target node. We have experimentally determined that a good value for n is 2.
If we want to use the same parameter k for all subnetworks, it is important to have clusters containing, more or less, the same number of nodes. In fact in a cluster with many nodes it might be difficult to connect each of them with k hops, while in a cluster with few nodes k hops could be excessive.
Unfortunately there are no universally accepted criteria to define what is a good cluster, but only some intuitive understandings: the intuitive idea behind clustering consists in condensing a subgraph into a single node, where the choice of the cluster is application-dependent. The problem of finding a proper set of nodes or a set of edges partitioning the network into clusters has been studied in literature and several approaches have been proposed [3, 6, 8].
In the sequel of this section, we will introduce two algorithms for clusterization. The former is based on the well-known median approach whereas the latter is based on a greedy approach. These algorithms have been used to carry out our experiments.

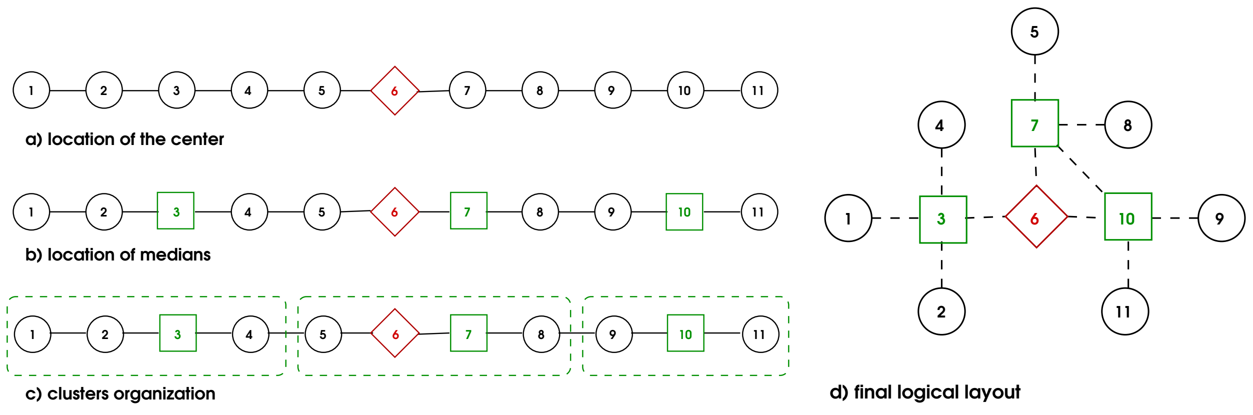

Figure 1: Processing a Test Network through the Median-Based Algorithm.

2.1 The Median-Based Algorithm

Our goal consists in partitioning a graph G into a set of subgraphs such that the size of each subset does not overcome a given lower bound, i.e. each cluster has to contain k nodes at least (k is fixed before the execution), and the maximum difference between subset sizes is lower than a given threshold ε.
The first step consists in finding the center of G; to this goal we build the minimum diameter spanning tree of G and locate its root as the center of G (in figure 1.a the center is represented by a red rhombus). A greedy algorithm, described in [9] computes the requested tree in polynomial time.
The next step consists in locating the set of medians (in figure 1.b medians are represented by green squares). The algorithm is a variation of the multi-median heuristic algorithm proposed in [2]. In particular we consider that each node has unitary weight and that the cost of an edge is defined by

$$c' : E \to \mathbb{N}, \quad c'(e) = M - c(e) + 1,$$

where $M = \max_{e \in E} c(e)$. In such way, when we evaluate the shortest path between two nodes we can take into account the edge capacities and to give more relevance to edges with larger capacity.
To satisfy the constraint on the maximum difference between cluster sizes, it is necessary to iterate the multi-median algorithm, increasing the number of medians to search, until the condition is satisfied.
Once the set of medians is located we construct the clusters by assigning each node to the closest median (as in figure 1.c). If a node can be assigned to two or more medians, it is labeled as ambiguous and not processed during this step. Ambiguous nodes are then assigned to a cluster, among the clusters related to its closest medians whose size is the smallest. If a cluster has less than k elements, it is deleted and its nodes are assigned to other clusters as we did in previous steps. Once clusters are made, their center is computed again in order to minimize the diameter. The final step consists in designing the final logical topology (as in figure 1.d): we connect each cluster center with the nearest ones and with the graph center. The resulting structure should be a ring with a center. Inside each cluster we connect elements to the center using the VPL^{1-m} algorithm described in [4]. Now we give a description of the algorithm in pseudocode.

```
INPUT:     A graph G
           h: Maximum number of hops to join each vertex with the center
           b: Lower bound on cluster size

OUTPUT:    G' ⊇ G : the logical graph
Compute ε:maximum difference among cluster sizes;
Find the minimum diameter spanning tree T of G;
Center of the graph  C := Root of T;
repeat
    Setup the set of medians (by computing or by reading them);
    Compute clusters;
    Assign ambiguous vertices to each cluster;
until maximum difference between cluster sizes is ≤ ε;

Recompute the center of clusters whose size changed;
Connect medians in pairs and medians with C;
Perform local optimization inside each cluster;
Evaluate the quality of clusterization.
```

2.2 The Spreading Algorithm

We propose a new approach based on the construction of clusters of fixed ray. The goal of such a method consists in building clusters whose depth is fixed, so that the VPL^{1-m} algorithm could be carried out faster. As in the previous algorithm a lower bound on cluster sizes is given.
To decide the ray value we first evaluate the ray R of the graph G, by constructing the minimum diameter spanning tree and locating its root as the center of G. Then we fix the ray r for computing the remaining clusters by choosing the best pair (r,n) of integers such that the ray R can be rewritten as:

$$R = r + 2r + \cdots + 2r = r(1 + 2n).$$

Starting from the center of G (in figure 2.a the center is represented by a red rhombus), we collect in a single cluster the nodes whose distance is less than r and

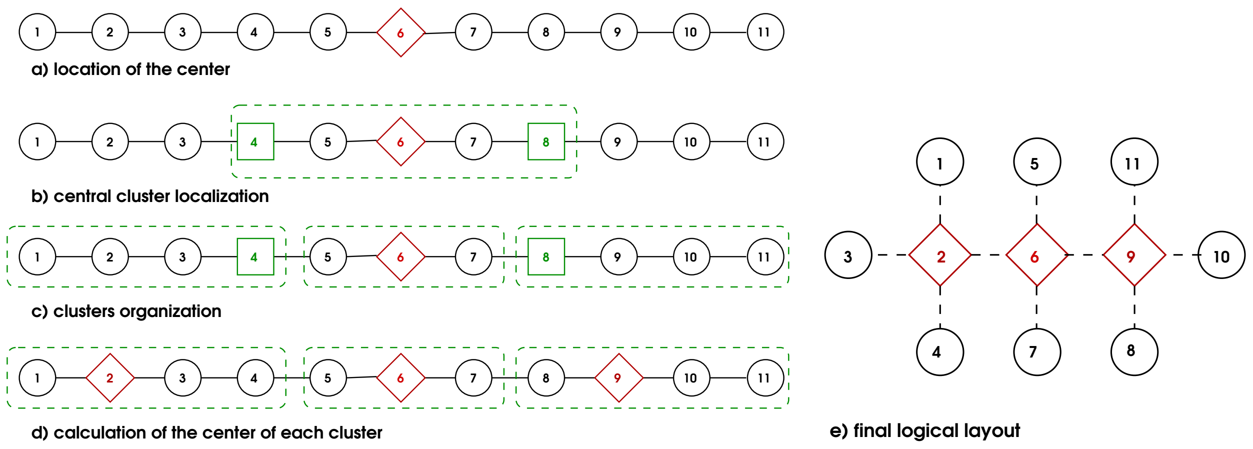
a) location of the center
b) central cluster localization
c) clusters organization
d) calculation of the center of each cluster
e) final logical layout

Figure 2: Processing a Test Network through the Spreading Algorithm.

create the set of leaves, i.e. the set of nodes whose distance from the center is equal to r (in figure 2.b, we consider $r = 2$, and leaves are represented by green squares). The method iterates starting from each leaf and finding nodes whose distance is less than $2r$. For each leaf a cluster and a new set of leaves are created (as in figure 2.c). If a node can be assigned to two or more leaves, it is labeled as ambiguous and processed later. Ambiguous nodes are then assigned to a cluster, among the clusters related to its closest leaves whose size is the smallest. Finally, the center of each cluster is computed (as in figure 2.d).

The reason for doubling the value of the ray r is that after the first step the starting nodes are expected to become diameter terminators in their, respective, clusters, so their expected distance from other cluster nodes is twice the ray.

The algorithm iterates starting from the new sets of leaves, until all nodes are assigned. If a cluster has too few elements it is deleted and its nodes are assigned to nearest clusters. The center of the modified clusters are then computed again. The final step is the connection of cluster centers in pairs and with the center of G; the resulting structure is n concentric rings with one center (see figure 2.e). Inside each cluster we connect the elements with the center via the VPL^{1-m} algorithm.

Here is a pseudocode description of the algorithm:

```
INPUT:    A graph G
          h: Maximum number of hops to join each vertex with the center
          R: Ray of each cluster
          $b$: Lower bound on cluster size

OUTPUT:   $G' \supseteq G$ : the logical graph

Find the minimum diameter spanning tree T of G;
r := R;
Center of the graph:    C := Root of T;

for each vertex v not yet assigned
   if $d(v,C) \leq r$
        Assign v to cluster(C)
Setup the set S of vertices w such that $d(w,C) = r$;
r := 2·R;
repeat
   for each vertex w in S do
      Find all vertices v such that $d(v,w) \leq r$;
      Setup the set X of vertices x such that $d(x,w) = r$;
   end for
   Assign ambiguous vertices to the nearest cluster whose size is the smallest;
   S := X;
until all vertices are assigned

Compute the center of each cluster;
Connect centers in pairs and centers with C;
Perform local optimization inside each cluster.
Evaluate the quality of clusterization.
```

3 Experimental Results

This research is part of a project relative to network optimization, which involves both the Marconi Comms and the Rumanian Ministry of Defense. The goal of the project consists in designing a logical topology over the physical topology of the network, enhancing its transmission performance. This result is achieved by reducing the number of nodes which a signal has to traverse, along the route from source to destination.

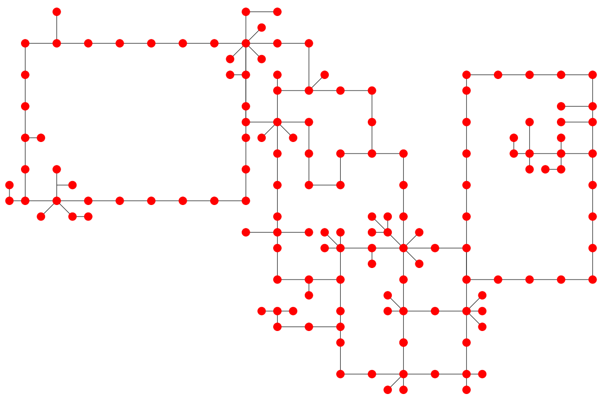

Figure 3: **Case Study Based on the Rumanian Network**

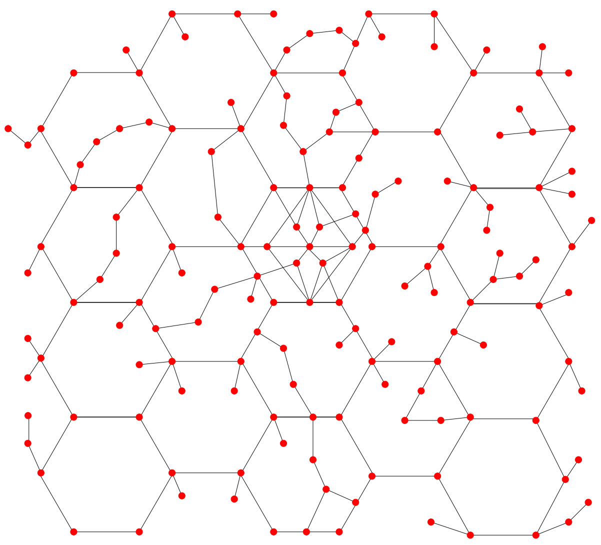
Figure 4: **Molecular Network**

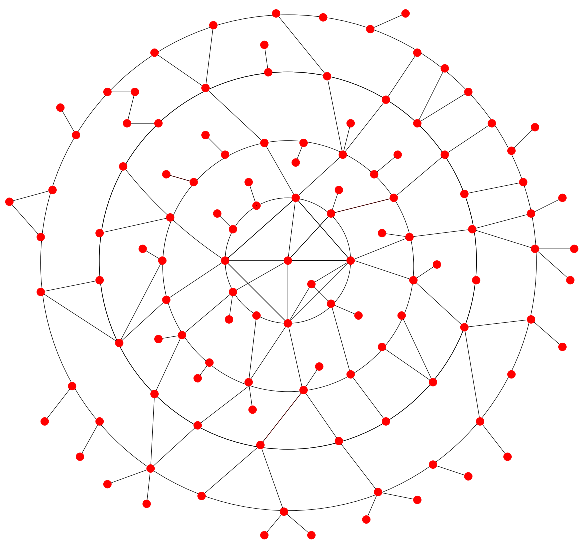
Figure 5: **Circular Network**

We performed several experiments, testing networks representing realistic environments. By the collaboration of an engineer of Rumanian army, we could also use, as case study, networks directly extracted from the existing Rumanian communication network, the one not covered by military secret.

- *Romania*. It is a case study based on the Rumanian communication network; Our sample consists of 184 nodes and 269 edges (Figure 3);
- *Molecular*. It is composed of adjacent hexagonal cells; such a network is usually employed in cellular communication. Our sample has 160 nodes and 240 edges (Figure 4);
- *Circular*. It is composed by four concentric rings, each of them connected with the adjacent ones. This topology is largely used in military communications. Our sample consists of 120 nodes and 214 edges (Figure 5);
- *Arborescent*. It has a structure similar to a tree, but it also contains some edges which connect each branch of the tree to its siblings. This kind of network is used for small environments, mostly a subsection of a larger network. Our sample has 81 nodes and 99 edges (Figure 6).

To compare the quality of the computed clusterization we used the following measures:

- *maximum path length*: we compare the length of the longest path connecting a pair of nodes in the original network with the longest one in the new logical layout. This measure describes how much the diameter of the network is reduced by reconfiguring its topology via clustering.
- *Number of clusters and their sizes*: we also adopt these measures because, in according to [4], a better clusterization is achieved if cluster sizes are balanced and if their number is not too elevate.
- *Percentage of shortest connections*: for all pairs of nodes we compare the length of their connection in the original network with the corresponding length in the clusterized networks then we calculate the percentage of paths whose length has been reduced with respect to the original network.

In table 1 we show the values of the longest paths in original and in computed networks; the longest path in the original network might not be the longest in clusterized ones, too. These values can only point out if clusterization gives a global reduction of the diameter of the network, without taking into account if the diameter terminators are the same both in original and in clusterized networks. As expected, in all cases the diameter has been reduced, besides, as larger the network is as more relevant the difference becomes.
In table 2 we compare the number of computed clusters and their sizes. As we can easily see the median-based algorithm generates clusters whose sizes are more balanced than that of the spreading algorithm. We can also

Method	*Romania*	*Molecular*	*Circular*	*Arborescent*
Original	35	17	12	12
Median	14	10	8	6
Ray = 2	23	10	8	6
Ray = 3	14	6	6	6
Ray = 4	14			
Ray = 5	10			

Table 1: Summary of the experiments related to networks 3, 4, 5, and 6. For each network is reported the diameter of the achieved graph, i.e., the length of the longest path in the new layout.

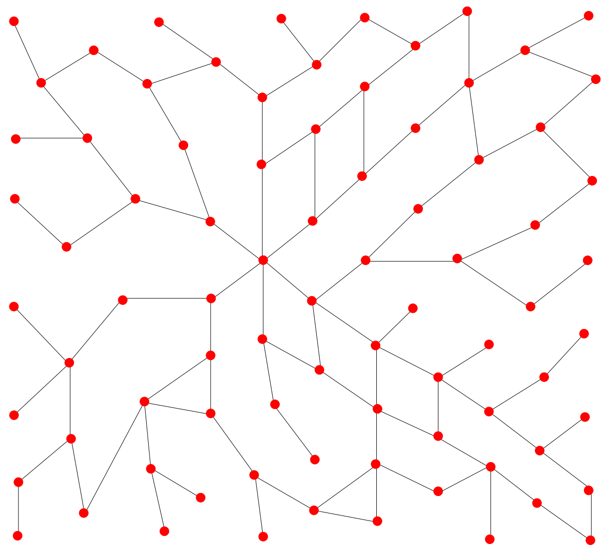
Figure 6: **Arborescent Network**

notice that if network size is very large and its structure is rectilinear (such as the network in figure 3) the spreading algorithm gives better balanced clusters but as a counterpart there is an high number of clusters which forces connections between distant nodes to still take a high number of hops.

Table 3 gives a comparison among the percentage of connections whose length has been reduced by clusterization, with respect to the original network.

The number of connections which are improved by clusterization is elevate; we can stress out that unbalanced clusters leads to worse situations. This is mostly due to the different value of hops required to connect the element of cluster to their center: if a cluster has a big size, there would need many hops to connect elements, so there will be more connections inside the cluster which will not be improved.

Moreover, we noticed that the spreading approach works fine with a low value for the ray, typically two or three, whereas it results quite inadequate when the value of the ray grows. Our test networks have too few nodes, hence with a big ray the spreading approach will find a big cluster and many very little clusters. Of course, if

Method	Romania		Molecular		Circular		Arborescent	
Median	10	10-26	8	15-21	13	5-10	9	5-10
Median	17	7-20	12	8-15	5	18-27	10	5-11
Ray = 2	34	5-9	9	5-34	11	5-16	8	6-12
Ray = 3	26	5-11	16	5-25	11	5-22	9	5-30
Ray = 4	9	5-28						
Ray = 5	9	5-37						

Table 2: Summary of the experiments related to networks 3, 4, 5, and 6. For each network are reported the number of achieved clusters, and their minimum and maximum size.

Method	Romania		Molecular		Circular		Arborescent	
Median	10	83	8	88	13	86	9	90
Median	17	86	12	86	5	82	10	90
Ray = 2	9	77	9	85	11	89	8	91
Ray = 3	9	81	16	79	11	82	9	84

Table 3: Summary of the experiments related to networks 3, 4, 5, and 6. For each network are reported the number of achieved clusters, and the percentage of improved paths.

applied to networks with a large number of nodes the results will become more significant also for rays larger than three. In the tables, we did not report values that we consider non interesting.

From a theoretical point of view it is possible to connect all the elements of a cluster to the center with the same number of hops, independently from the cluster size. This fact would lead to connect several vertices to one vertex; so there would be several edges insisting on the same vertex, strongly increasing the vertex load. In physical terms it is impossible to a node to support so many edges; then there is a physical upper bound on vertex load that avoid such a situation.

4 Conclusions and Future Works

We have presented two clustering algorithms and discussed their merits in terms of the reduction of path length between pairs of nodes of a network. Preliminary results related to this work have been presented in [1].

In order to evaluate the improvements due to the clusterization we have applied to each cluster a greedy iterative algorithm described in [4]. The methods build a static layout of VP for a network, in such a way that there is an efficient route, with low VP hop count, between any pair of switches. Some experiments have been computed and the results show that clusterization lead to a great improvement in both cases, reducing the length of routes between pairs of switches.

There is not a deep difference between the results obtained by the two methods. The spreading algorithm seems to be more efficient in circular networks while the median-based algorithm gives better results with linear networks. In arborescent networks both of them gives good improvements. Further experiments will be performed, both on small network with a particular structure and on larger networks taken from real environments.

Acknowledgments

Enrico Martinuzzi participated to this project when he worked for Marconi Comms. This project has been financially supported by Marconi Comms.

References

[1] Massimo Ancona, Walter Cazzola, Paolo Raffo, and Ioan Bogdan Vasian. Virtual Path Layout Design Via Network Clustering. In *Proceedings of International Conference Communications 2000*, pages 352–360, Bucharest, Romania, on 7th-9th of December 2000. IEEE.

[2] Joseph Cheriyan and Ram Ravi. Lecture Notes on Approximation Algorithms for Network Problems, September 1998. Available at `http://www.math.uwaterloo.ca/~jcheriya/lecnotes.html`.

[3] Sándor P. Fekete and Henk Meijer. On Minimum Stars, Minimum Steiner Stars and Minimum Matchings, 1999. ACM.

[4] Ornan Gerstel, Israel Cidon, and Shmuel Zaks. Optimal Virtual Path in ATM Networks with Shared Routing Table Switches. *Chicago Journal of Theoretical Computer Science*, October 1996.

[5] Ornan Gerstel and Shmuel Zaks. The Virtual Path Layout Problem in Fast Networks. In *Proceedings of the 13th ACM Symposium on Principles of Distributed Computing*, pages 235–243, Los Angeles, USA, August 1994.

[6] Tom Roxborough and Arunabha Sen. Graph Clustering Using Multiway Ratio Cut. In *Proceedings of Symposium on Graph Drawing (GD'97)*, pages 291–296, Berlin, 1997.

[7] Vikram R. Saksena. Topological Analysis of Packet Networks. *IEEE Journal on Sel. Areas in Communications*, 7(8), October 1989.

[8] Benno Stein and Oliver Niggemann. On the Nature of Structure and Its Identification. In , LNCS 1665, pages 122–134. Springer, 1999.

[9] S. Tricot. A Polynomial Time Algorithm for the Minimum Diameter Spanning Tree Problem. technical report, Institute Blaise Pascal, MASI laboratory, Université Paris VI, 1994.