A Clustering Approach for Scalable Network Design

Matthias Scheffel, Moritz Kiese,* *Munich University of Technology,*
Arcisstr. 21, 80333 Munich, {matthias.scheffel, moritz.kiese}@tum.de
Thomas Stidsen, *Technical University of Denmark,*
Richard Petersen Plad, DTU - Bygning 321, 2800 Lyngby, tks@imm.dtu.dk

Abstract

Scalability is an important issue when planning transport networks. In order to model architectural and technological constraints and cope with high computational complexity, it is essential to adapt the planning process. In this article we present a clustering approach that decomposes the overall transport network into appropriate partitions to which further design strategies can be applied individually. We formulate models based on integer linear programming, column generation, simulated annealing, and perform a case study.

Keywords: network planning, clustering.

1. Introduction

Today's transport networks consist of a mixture of various protocols such as *Synchronous Digital Hierarchy / Synchronous Optical Network* (SDH / SONET), *Asynchronous Transfer Mode* (ATM), *Multiprotocol Label Switching* (MPLS), *Internet Protocol* (IP), ... each having different functionality and data formats. This multilayer structure continues when considering network architectures. Depending on the distances between nodes, the network is partitioned into small-scale *Local Area Networks* (LANs), *Metropolitan Area Networks* (MANs), and *Wide Area Networks* (WANs) with ultra long transmission range. The resulting hierarchy facilitates network operation and management.
From a network provider's point of view, the consolidation of the telecommunication market enforces the need for efficient network planning. Simply overprovisioning the network by excessive capacity is not suitable due to high costs. Instead, future traffic characteristics should be analyzed and modeled carefully. In order to route the demands, sufficient capacity must be allocated at the network nodes and spans. Additionally, physical transmission limitations are relevant to be able to correctly detect the signal at the destination. Besides that, *Service Level Agreements* (SLAs) require network providers to deliver a certain degree of *Quality of Service* (QoS) to their customers. In order to guarantee reliable services, redundant resources are scheduled to offer an alternative routing in case of failures of network elements. An ideal network design approach has to consider all these details of the planning process simultaneously to yield optimal results. However, this is usually not possible due to the computational complexity. Mathematical programming is a promising approach to determine efficient network configurations, but the applicability is often limited to small networks in the range of 20 nodes depending on the number of spans. In this work we propose a clustering algorithm which can efficiently reduce the problem size for large-scale networks.

The idea to decompose the network into smaller clusters is motivated by a number of reasons. Firstly, it is intuitive that the number of variables and constraints will be kept in an acceptable order of magnitude if the network size is limited. Additionally, the clusters can inherently reflect the natural hierarchical structure of a transport network in terms of technologies used for different network sizes (e. g. *Coarse* vs. *Dense Wavelength Division Multiplex* (CWDM / DWDM)) as well as operational units such as *Autonomous Systems* (ASs). Regarding the network technology, the required bitrate and transmission distance dictate the choice with respect to minimal cost. The existence of administrative network entities eases network operation and is an important principle of the Internet as a "net of networks".

*The underlying work is part of a project supported by the German Ministry of Education and Research under the Project ID 01BP551. The authors are solely responsible for the contents of this publication.

The clustering concept has been studied in a number of different contexts [1] and plays an important role for example in data clustering [2] or ad hoc networks [3], resulting in a variety of proposed formulations. Our *Integer Linear Programming* (ILP) approach is related to work by Grötschel and Wakabayashi [4] modeling a *clique partitioning problem*. However, our aim is to determine groups of network nodes with respect to network properties reflecting the above mentioned structure using the intra-cluster node-distances as metrics. Apart from that, other criteria such as (minimizing) inter-cluster traffic are also conceivable and can be easily incorporated in the design.
Due to the computational complexity of the "classic" ILP formulation, we additionally implement a column generation approach (based on previous work by Vanderbeck [5, pp. 43]) and a simulated annealing heuristic and compare their respective performance in terms of solution quality and running times.

2. Network Clustering

The authors view network clustering as a first step when planning large-scale transport networks. It is essential for green-field planning, i.e. building up a network from scratch. Additionally, it facilitates the upgrade of an existing network configuration by means of additional equipment. We assume that only the locations of the network nodes are given. The respective decision process may be based on strategic considerations as well as the existence of major traffic sources and sinks. Clustering the network into smaller parts should result in node groups of adequate size to be able to apply further planning strategies such as the allocation of network spans (topological planning), the routing of traffic demands, and the dimensioning of network capacity. For this purpose clusters are considered as individual networks to reduce the problem size.

The interconnection of the clusters can be performed in a subsequent step. For example, a second level of hierarchy can be defined which interconnects the network partitions. This backbone layer may be determined by a second clustering algorithm that aggregates metro clusters into one backbone node. Alternatively, a designated backbone partition can be found in the first clustering step by adding corresponding constraints.

The crucial point is the criterion to obtain suitable domains. Our aim is to keep nodes that are assigned to the same partition close to each other. We want to minimize the distance between all intra-cluster nodes to create subnetworks that naturally expand into all directions according to the distance to neighboring nodes. Thus, limitations of transmission distances due to technological constraints can easily be incorporated in our design. Furthermore, this strategy supports topological planning as network spans are typically allocated towards neighboring nodes.

Integer Linear Program

Let N be the set of nodes of a network which have to be assigned to a cluster set C with a given number of clusters $|C|$. The physical distance between two nodes $n_1, n_2 \in N$ is given by $d_{n_1,n_2} \in \mathbb{R}^+$ and parameter $d^{\max} \in \mathbb{R}^+$ represents a maximum allowed distance between intra-cluster nodes. We denote the decision variables that determine whether a node $n \in N$ is contained in cluster $c \in C$ by $\mathcal{X}_{c,n} \in \{0,1\}$. The variables $\mathcal{Y}_{c,n_1,n_2} \in \mathbb{R}_0^+$ reveal whether nodes $n_1, n_2 \in N$ are situated in the same cluster $c \in C$. The mathematical model is then given by the following equations.

$$\text{Minimize} \sum_{\substack{c \in C \\ \{n_1,n_2\} \subset N}} \mathcal{Y}_{c,n_1,n_2} \cdot d_{n_1,n_2} \tag{1}$$

subject to

$$\sum_{c \in C} \mathcal{X}_{c,n} = 1 \qquad \forall n \in N \tag{2}$$

$$\mathcal{Y}_{c,n_1,n_2} \geq \mathcal{X}_{c,n_1} + \mathcal{X}_{c,n_2} - 1 \qquad \forall c \in C, \quad \forall \{n_1, n_2\} \subset N \tag{3}$$

$$\mathcal{Y}_{c,n_1,n_2} = 0 \qquad \forall c \in C, \quad \forall \{n_1, n_2\} \subset N : d_{n_1,n_2} > d^{\max} \tag{4}$$

$$\mathcal{X}_{c,n} \in \{0,1\}\,,\ \mathcal{Y}_{c,n_1,n_2} \in \mathbb{R}_0^+\,. \tag{5}$$

The Objective (1) minimizes the total sum of the distances between all nodes that are included in the same cluster c. For this we consider all node pairs $\{n_1, n_2\} \subset N$, i.e. all unsorted node subsets consisting of two distinct nodes $n_1, n_2 \in N$. In doing so, the distance between two nodes will be counted only once as opposed to twice if dealing with ordered 2-tupels. Constraints (2) guarantee that every node n is included in exactly one cluster c. In order to be able to distinguish node pairs that are assigned to the same cluster, Constraints (3) provide a lower bound based on the cluster allocation variables for each node. The right side of the equation will equal one, only if both considered nodes are in the identical network cluster. Equations (4) are optional constraints which reduce the solution space and are motivated by the goal of the clustering strategy at the same time. An efficient configuration integrates nodes in one cluster in case they are situated close to each other to reduce the sum of the node distances. Thus, it is very likely that nodes far away from each other are put into distinct clusters. The equation prohibits nodes to be in the same network partition as soon as their distance exceeds a maximum value $d^{\max}$.

Column Generation

The integer linear programming relaxation of the clustering problem can be solved using a column generation algorithm. Applying this approach can help to determine optimal results even for large problem instances. We introduce again a set of nodes N and denote the distance between two nodes by d_{n_1,n_2}. A maximum quantity of clusters $q^{\max}$ and a maximal distance $d^{\max}$ for nodes inside the same cluster is given. We assume to have a set of potential clusters P already. The parameter $m_{p,n} \in \{0,1\}$ reveals whether cluster p contains node n. If the set P is complete, it will give us the optimal solution when solving the master problem. The relaxed binary variables $\mathcal{Z}_p$ decide whether a cluster should be used. The sum of distances between all node pairs situated in the same cluster p is denoted l_p.

$$\text{Minimize} \sum_{p \in P} \mathcal{Z}_p \cdot l_p \tag{6}$$

subject to

$$\sum_{p \in P} \mathcal{Z}_p \cdot m_{p,n} = 1 \quad \forall n \in N \tag{7}$$

$$\sum_{p \in P} \mathcal{Z}_p \leq q^{\max} \tag{8}$$

$$\mathcal{Z}_p \in \{0,1\}\,. \tag{9}$$

Solving the master problem given a non-complete set of eligible clusters P gives us a solution and a set of dual variables α_n from equation (7) and β from equation (8) of the master problem. New clusters that can improve the current solution are then found in the following sub-problem, where the binary variable $\mathcal{X}_n$ decides whether a node should be included into a new cluster. The fact that two nodes are in the same cluster is detected by variables $\mathcal{Y}_{n_1,n_2}$. If no clusters with negative reduced costs can be found, the current (relaxed) solution of the master problem will be optimal.

$$\text{Minimize} \sum_{\{n_1,n_2\}\subset N} \mathcal{Y}_{n_1,n_2} \cdot d_{n_1,n_2} - \sum_{n} \alpha_n \cdot \mathcal{X}_n - \beta \tag{10}$$

subject to

$$\mathcal{Y}_{n_1,n_2} \geq \mathcal{X}_{n_1} + \mathcal{X}_{n_2} - 1 \qquad \forall \{n_1, n_2\} \subset N \tag{11}$$

$$\mathcal{X}_{n_1} + \mathcal{X}_{n_2} \leq 1 \qquad \forall \{n_1, n_2\} \subset N : d_{n_1,n_2} > d^{\max} \tag{12}$$

$$\mathcal{X}_n \in \{0,1\}\,,\ \mathcal{Y}_{n_1,n_2} \in \mathbb{R}_0^+. \tag{13}$$

The sub-problem, solved by a standard (*Mixed Integer Linear Program*) MIP-solver (namely ILOG Cplex [6]), finds one new promising cluster for each iteration of the column generation algorithm. The parameter $d^{\max}$ is gradually increased, so that in the end it is set to ∞.

When the column generation algorithm terminates, the final master problem will be solved, but now with $\mathcal{Z}_p$ as a binary variable, i.e. $\mathcal{Z}_p \in \{0,1\}$. For all test samples where a column generation solution was found, a MIP solution is found of the same value, i.e. the gap is zero and a guaranteed optimal solution is available.

Simulated Annealing

As will be shown in the case study later on, all mathematical programming formulations have rather high running-times and/or can only operate efficiently on a reduced set of variables. Furthermore, the partitioning represents a preliminary step for a subsequent network planning process and thus does not have to be optimal by any means. Due to these reasons Simulated Annealing as implemented in the GRAPH-Library [7] is used for comparison.

Initial Solution For the initial solution, all nodes are assigned randomly to one of $q^{\max}$ clusters so that the number of nodes in a cluster is equal to $\frac{|N|}{q^{\max}}$. If $\frac{|N|}{q^{\max}}$ is non-integer, all but one of the clusters will be populated with $\left\lceil \frac{|N|}{q^{\max}} \right\rceil$ nodes.

Iteration Step In every iteration step one cluster p_1 is randomly selected from those $p \in P$ which contain at least one node. From this cluster a randomly selected node n is assigned to a second randomly chosen cluster p_2 where $p_1 \neq p_2$ and removed from the first cluster p_1. For this newly generated solution the cost is calculated equivalently to (1).

Cooling Schedule Even the most simplistic cooling schedule namely *constant cooling* which results in a temperature of T_i in the iteration step i

$$T_i = T_{\text{start}} - \alpha \cdot i \tag{14}$$

was able to find very good solutions with a starting temperature $T_{\text{start}} = 10000$ and a cooling factor $\alpha = 10^{-4}$.

Transition Probability In the used implementation the 'classic' transition probabilities proposed by Kirkpatrick et Al. in [8] are used:

- At every iteration a better solution s is always accepted and
- a worse solution s is accepted accordingly with a probability of $\exp\left(\frac{-|s-s_{\text{best}}|}{T_i}\right)$.

Although computationally far from optimal, calculating the cost for a new solution s involves a re-computation of costs for the whole solution in the current implementation.

Stopping Criterion The heuristic will terminate if either the temperature T_i equals zero or a configurable running time has been exceeded.

3. Case Study

In a case study we investigate the network clustering concept and compare the performance of the implemented approaches. Figure 1 shows the node topology that is used for our investigation. The network comprises 90 large cities of Germany. The distance between two nodes is determined by the Haversine formula [9] taking into account the spherical surface of the earth. Latitude and longitude coordinates taken from [10] serve as input parameters.

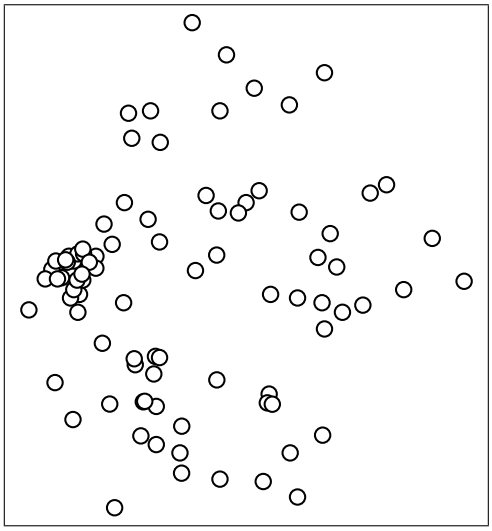

Figure 1: Geographical locations of the 90 node test network.

In order to be able to assess the previously presented algorithms with respect to scalability, subsets of the 90 nodes were generated by taking the $|N|$ first nodes (in steps of ten) in alphabetical order of the city names. Each of these samples is partitioned into $\frac{|N|}{10}$ sets, so that the average number of nodes per cluster is always 10, which is a suitable size for an exhaustive network optimization from our point of view. As it can be seen in Figure 2 optimal solutions could be calculated for the 20 and 30 node samples with column generation, but the running times became prohibitively high (more than 9 hours) for bigger samples. Instead of limiting the running times the number of column generation iterations was consequently restricted to 5000.

The linear programming model proves to be quite complex to solve for an increasing number of clusters. The computation time increases from fractions of a second for two clusters to about one hour for three partitions. For four or more clusters it is not possible to calculate optimal results within a few hours. The insertion of optional Constraints (4) significantly improves the obtained objective value by limiting the solution space via eligible clusters that restrict the maximum distance between two nodes. The reason for the computational complexity is the large number of equivalent solutions due to the fact that there exist $|C|!$ cluster variations for identical node subsets.

Although the simulated annealing algorithm does not guarantee any solution quality at all, it can find better solutions than non-optimal results from the restricted ILP and column generation approaches – quite obviously nodes which are further away are more and more important and thus more and more prohibitive to ignore in larger sets. Where optimal solutions are known they are found by simulated annealing. With running times below 15 seconds on a Linux-based Opteron 248 (search terminated because temperature T_i reached 0) compared to hours for the ILP and column generation approach using Cplex [6], the presented simulating annealing algorithm is a clear winner under the given circumstances.

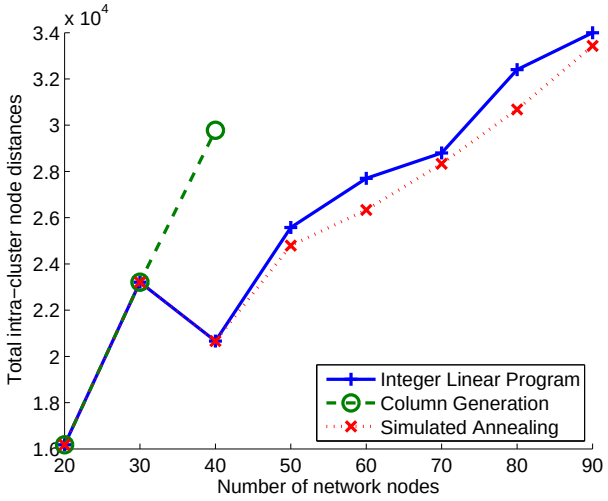

Figure 2: Total sum of distances between nodes of the same cluster over the number of network nodes $|N|$. The number of clusters is $\frac{|N|}{10}$ respectively.

4. Conclusion

This work investigates the clustering of large-scale transport networks. We divide a set of nodes into partitions where each node is included only once. Our aim is to minimize the physical distance between intra-cluster nodes to account for transmission limitations and create natural node accumulations. The clustering strategy facilitates scalable network design as the clusters help to reduce the problem complexity. They can be planned individually and the inter-cluster design may be done in a subsequent phase. We formulate an ILP, a column generation approach, and a simulated annealing strategy. Our results show that simulated annealing can outperform the mathematical programs due to the problem structure and additionally offers short simulation times. Future work will analyze appropriate strategies for intra- and inter-cluster network planning.

References

[1] S. Chopra and M. R. Rao, "The Partition Problem," *Mathematical Programming*, vol. 59, March 1993.

[2] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay, "Clustering in Large graphs via Singular Value Decomposition," *Journal of Machine Learning*, vol. 56, pp. 6–33, 2004.

[3] C. Bettstetter, "The Cluster Density of a Distributed Clustering Algorithm in Ad Hoc Networks," in *Proceedings of the IEEE International Conference on Communications 2004*, Paris, France, June 2004.

[4] M. Grötschel and Y. Wakabayashi, "A Cutting Plane Algorithm for a Clustering Problem," *Mathematical Programming*, vol. 45, 1989.

[5] François Vanderbeck, "Decomposition and Column Generation for Integer Programs," Ph.D. dissertation, Université Catholique de Louvain, Louvain La Neuve, September 1994.

[6] *ILOG CPLEX 9.0 User's Manual*, ILOG, S. A., 2003.

[7] *GRAPH Manual*, Layonics, 2006, http://www.layonics.com.

[8] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, "Optimization by Simulated Annealing," *Science*, vol. 220, no. 4598, pp. 671–680, May 1983.

[9] R. W. Sinnot, "Virtues of the Haversine," *Sky and Telescope*, vol. 68, no. 2, p. 159ff, 1984.

[10] M. Hoppe and T. Mack, *OpenGeoDB*, 2006, http://sourceforge.net/projects/opengeodb.