Design of Hierarchical Ring networks Using Branch-and-Price

Tommy Thomadsen* Thomas Stidsen†

Informatics and Mathematical Modelling
Technical University of Denmark
DK-2800 Kongens Lyngby, Denmark

May 21, 2004

Abstract

We consider the problem of designing hierarchical two layer ring networks. The top layer consists of a federal-ring which establishes connection between a number of node disjoint metro-rings in a bottom layer. The objective is to minimize the costs of links in the network, taking both the fixed link establishment costs and the link capacity costs into account.

The hierarchical two layer ring network design problem is solved in two stages: First the bottom layer, i.e. the metro-rings are designed, implicitly taking into account the capacity cost of the federal-ring. Then the federal-ring is designed connecting the metro-rings, minimizing fixed link establishment costs of the federal-ring. A branch-and-price algorithm is presented for the design of the bottom layer and it is suggested that existing methods are used for the design of the federal-ring. Computational results are given for networks with up to 36 nodes.

Keywords: Ring network design, Hierarchical network design, Branch-and-Price.

1 Introduction

Design of survivable communication networks is important for at least two reasons. First of all there is a growing reliance on electronic communication in society. Sec-

*Email: tt@imm.dtu.dk
†Email: tks@imm.dtu.dk

ondly failures (e.g. a link failure) may have a large impact, given the high capacity of links.

Self Healing Rings (or rings for short) have been widely used to ensure survivable communication for several reasons. First of all, the rings are pre-configured such that the only nodes that need to do re-routing in case of a link failure are the two endpoint nodes of the failed link. Thus no communication with other nodes is necessary making ring protection fast. Furthermore the node equipment is cheap to build and protection does not require the involvement of an expensive network management system.

Larger networks consist of several interconnected rings, since it is neither possible nor beneficial to restrict the entire network topology to a single ring. One possible way to interconnect the rings is in a hierarchy. Hierarchical networks have existed for decades and were introduced because of the limited switching capabilities in the telephone systems. Hierarchies are still used since they divide the network in subnetworks which can to some extend be treated independently, easing maintenance and upgrade.

In this paper we consider the design of hierarchical ring networks (HRNs), i.e. hierarchical networks where subnetworks are rings. We assume that communication demands are given and determine a HRN which satisfies the communication demands as cheaply as possible. In reality this includes both design *and* routing, but by modifying the problem, routing is implicitly considered as we shall see. We present models and algorithms for two layers only, but both models and algorithms can be generalized to more layers. We denote the ring in the top layer the federal-ring, and the (node disjoint) rings in the bottom layer, the metro-rings. See Figure 1 for an example of a HRN. We consider single homing, i.e. exactly one node from each metro-ring is in the federal-ring.

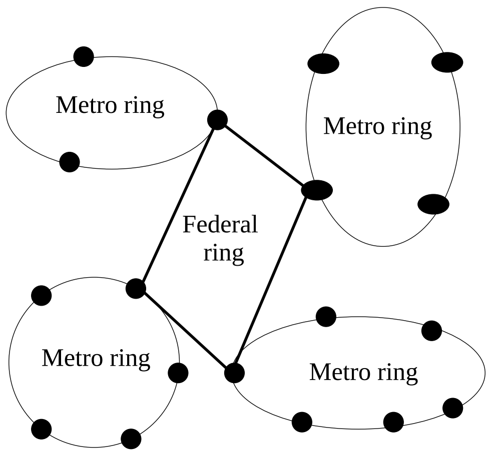

Figure 1: A two layer hierarchical ring network

The main contribution of the paper is the implementation of a branch-and-price algorithm which can be used to solve the modified problem of designing HRNs to optimality. To that end, we discuss our problem modification and point out under what circumstances an optimal solution for the modified problem is optimal for the original combined design and routing problem. The problem modification has previously been put forward and used for implementing heuristics but has in not been analysed in detail. Optimal solutions have previously been obtained for networks with up to 12 nodes and used for comparison with heuristic values. Our branch-and-price algorithm can in general solve instances with 20 nodes and for problems with special structure up to 36 nodes.

The outline of the paper is as follows. In Section 2, we discuss related papers. In Section 3 we consider the modification of the combined design and routing problem to a pure design problem. In Section 4, we give a mathematical formulation of the modified problem and in Section 5 we describe how the modified problem can be solved using branch-and-price. In Section 6 we give some computational results and finally we give some concluding remarks and some directions for future research in Section 7.

2 Previous work

HRNs were introduced by Shi and Fonseka in [7] and were further developed in [8, 9, 10, 11]. The papers describe how the combined design and routing problem of HRNs can be modified such that a problem which is in essence a pure design problem is obtained. We use the same idea in this paper. The papers present an enumeration scheme and heuristics which solve the problem. The papers also consider extensions to the basic model, e.g. dual homing (i.e. two nodes from each metro-ring is in the federal-ring). The number of possible networks grows exponentially, making the enumerative scheme useless except for small and trivial instances (less than 10 nodes). The heuristics on the other hand make the model applicable to large networks, but give no guarantee on the quality of the solutions obtained.

In [5] an integer linear program is presented for the pure design problem obtained by Shi and Fonseka. Optimal solutions can in some cases be obtained using the model, for networks with up to 12 nodes and a maximum of 4 nodes in the metro-rings. The model, however, inadvertently allows metro-"rings" to consist of more sub-rings and thus some of the nodes may be disconnected from the network. A sub-ring consists of at least 3 nodes, and thus a metro-ring will only consist of more sub-rings if it contains at least 6 nodes. Since the networks considered have a maximum of 4 nodes in each metro-ring, the obtained results are not affected. The focus of the paper is a "partition, construct and perturb" heuristic. This heuristic is compared with optimal solution when these can be obtained and with Shi and Fonsekas heuristics. It is concluded that better results than Shi and Fonseka are in general obtained.

In [3] problems where hub location and network design is considered simultaneously are reviewed. The problem of designing a HRN is a special case of the problems reviewed in [3].

3 Problem Modification

Let the network $G(V, E)$ where V is the set of nodes and E is the set of possible bidirectional links. Let D be the set of demands, let R^{met} be the set of possible metro-rings and R^{fed} the set of possible federal-rings. For $r \in R^{fed}$ or $r \in R^{met}$, $r \subset E$, i.e. r is a subset of links, and the links induce a ring. Let d'_{ij}, $ij \in D$ denote the demand for communication flow between node $i \in V$ and $j \in V$. Also let c_e be the fixed cost for establishing link e and correspondingly let the cost per capacity unit on link e be b_e.

The purpose of modifying the problem is to obtain a formulation which includes routing indirectly but is a pure design problem. The modification also allows a decomposition of the total cost into costs for each ring which can be measured independently.

The cost of a HRN is assumed to depend solely on the links used by the rings in the network and the capacity of these links, i.e. the *fixed cost* and the *capacity cost* respectively. Thus the cost of a HRN is as given in equation (1), where $r^{fed} \in R^{fed}$ is the federal-ring, $\overline{R}^{met} \subset R^{met}$ is the set of node disjoint metro-rings covering all nodes and finally CAP_r is the minimal capacity required on each link of ring r to service the traffic flow.

$$\sum_{e \in r^{fed}} c_e + CAP_{r^{fed}} \cdot \sum_{e \in r^{fed}} b_e + \sum_{r \in \overline{R}^{met}} (\sum_{e \in r} c_e + CAP_r \cdot \sum_{e \in r} b_e) \tag{1}$$

The fixed cost of the federal-ring is left as a separate optimization problem, i.e. the HRN-cost is initially approximated by the fixed cost of the metro rings and the capacity cost:

$$\sum_{r \in \overline{R}^{met}} \sum_{e \in r} c_e + CAP_{r^{fed}} \cdot \sum_{e \in r^{fed}} b_e + \sum_{r \in \overline{R}^{met}} CAP_r \cdot \sum_{e \in r} b_e \tag{2}$$

We consider unidirectional self-healing rings, for which it holds that communication flow in the ring takes up capacity in all links in the ring. Thus if a demand $ij \in D$ traverse a ring, it takes up capacity d'_{ij} in all links on the ring. Assume that RL is the average ring-length with respect to c_e. An estimate of the capacity cost for satisfying a given demand is RL if nodes are in the same metro-ring and $3RL$ if nodes are in different metro-rings, since three rings are in that case traversed (two metro-rings and the federal-ring). Also the capacity cost can be expressed as a worst case cost, K (corresponding to that all demands traverse three rings) minus a savings obtained by handling communication demands within metro-rings. Denote by $D_r^{met} \subset D$ the set of demands handled within metro-ring r. In that case the capacity cost can be estimated as follows.

$$CAP_{r^{fed}} \cdot \sum_{e \in r^{fed}} b_e + \sum_{r \in \overline{R}^{met}} CAP_r \cdot \sum_{e \in r} b_e \approx K - 2RL \sum_{r \in \overline{R}^{met}} \sum_{ij \in D_r^{met}} d'_{ij} \tag{3}$$

The total HRN cost is then estimated by the following.

$$K + \sum_{r \in \overline{R}^{met}} \sum_{e \in r} c_e - 2RL \sum_{r \in \overline{R}^{met}} \sum_{ij \in D_r^{met}} d'_{ij} \tag{4}$$

The intuition behind this rewrite is, that minimizing the capacity cost corresponds to maximizing the communication demand handled within metro-rings. Note that $2RL$ will have to be experimentally determined. Different values of $2RL$ will result in different cost structures, e.g. a low $2RL$ will correspond to the case where the capacity cost is higher in the federal-ring than in the metro-rings.

The ring length with respect to c_e may be far from constant (i.e. deviate considerably from RL). However if the capacity cost reflects a cost of node-equipment rather than a cost proportional to the distance between nodes, RL is thus proportional to the number of nodes in the rings. In that case it makes much more sense to have a known, fixed RL corresponding to a known fixed number of nodes in the rings, and in particular [7, 8, 9] study such networks. For HRNs where the ring length is not RL in all cases, optimal solutions for the modified problem may not be optimal in the original combined design and routing problem.

Note that the cost can now be decomposed into costs minus a reward for each metro-ring plus a constant K, which can be measured *independently*. Thus the cost for metro-ring $r \in R^{met}$ is:

$$c_r = \sum_{e \in r} c_e - 2RL \sum_{ij \in D_r^{met}} d'_{ij} \tag{5}$$

Consider the demand d'_{ij} where i and j are in different metro-rings, i is in the federal-ring and r is the metro-ring including i. In that case equation (4) includes a cost for routing d'_{ij} in r, but d'_{ij} need not be routed "from i via r to i" - there is no need to route it in r at all. Thus additional savings should be included if i is in the federal-ring. This saving is included as a reward on nodes when the federal-ring is designed. The node reward is the sum of all demands starting or ending in the node.

4 The Problems

Given the modification of the problem, the idea is now to select the lowest cost set of metro-rings, which includes nodes exactly once, i.e. a set-partitioning problem. However, since there are too many metro-rings to pregenerate all, we generate metro-rings when needed. Thus what we describe is actually a column generation algorithm or, since branching is needed to get integer solutions, an integer programming column generation algorithm, also known as branch-and-price [1, 13].

In this section we will describe the two problems we need to solve; the ring-partitioning problem (which is a set-partitioning problem) and the ring-generation problem. We will describe the branch-and-price algorithm in detail in Section 5.

When the metro-rings have been designed, the federal-ring is designed as the shortest ring, which includes exactly one node from each metro-ring and takes into account node rewards as described in the previous section. This is a Generalized Travelling Salesman Problem which can be solved using a branch-and-cut algorithm as done in [2]. This problem seems to be easier than the ring-generation problem which is solved many times, and thus the design of the federal-ring is not the bottleneck of the algorithm. We will not consider the design of the federal-ring any further in this paper.

4.1 The Ring-Partitioning Problem

Given a set of metro-rings $R \subset R^{met}$, the ring-partitioning problem is the problem of choosing the lowest cost subset of metro-rings in R, such that all nodes are covered exactly once. Define $p_{ir} = 1$ if node i is part of ring r, 0 otherwise. The variables u_r is 1 if ring r is selected, 0 otherwise. The ring-partitioning problem is then:

$$\min \quad \sum_{r \in R} c_r \cdot u_r \tag{6}$$

$$\text{s.t.} \quad \sum_{r \in R} p_{ir} \cdot u_r = 1 \qquad \forall i \in V \qquad (\pi_i) \tag{7}$$

$$u_r \in \{0, 1\} \tag{8}$$

The objective (6) is the total cost of selecting metro-rings, where c_r is defined in equation (5). Constraints (7) ensure that each node is in exactly one metro ring and constraints (8) are the integer domain constraints. Finally π_i are the dual variables for constraints (7). The problem obtained by relaxing constraint (8) is denoted the relaxed ring-partitioning problem. If branching is necessary, additional constraints are added, see Section 5.1. Rings are iteratively generated and added to R. The ring-generation problem is described in the following section.

4.2 The Ring-Generation Problem

The objective of the ring-generation problem is based on the cost in equation (5). However this cost does not include any information on which other rings are in R, and thus it is possible that a node will never be included in any ring. The idea is to add a reward to the objective, which reflects how difficult a node is to cover in the ring-partitioning problem given the *current* set of rings R. A node is difficult to cover if e.g. a single ring $r \in R$ contains the node and thus r need to be selected regardless of the cost. If a node i is difficult to cover a high reward is put on including i in a ring. The reward used is the value of the dual variables in the optimal solution to the ring-partitioning problem, π_i.

Let $d_{ij} = RL \cdot d'_{ij}$, let $n(r) \subseteq V$ be the nodes in r and let $D_r \subset D$ be the set of demands which start *and* end in r. Formally, we generate the ring with most negative reduced cost, where the reduced cost is given by the following equation.

$$c_r - \sum_{i \in n(r)} \pi_i = \sum_{e \in r} c_e - \sum_{ij \in D_r} d_{ij} - \sum_{i \in n(r)} \pi_i \tag{9}$$

We assume an upper limit, m is given on the number of nodes in the ring. Define the following variables, $y_i = 1$ if node i is in the ring, 0 otherwise, $x_e = 1$ if link e is in the ring, 0 otherwise and $z_{ij} = 1$ if demand ij can be handled by the ring, otherwise 0. (Equivalently, $z_{ij} = 1$ if $y_i = 1$ and $y_j = 1$, otherwise 0.)

For $S \subset V$, let $\delta(S) \subset E$ denote the set of edges with an endpoint in S and an endpoint not in S. Then the ring-generation problem can be stated as follows.

$$\min \quad \sum_{e \in E} c_e \cdot x_e - \sum_{ij \in D} d_{ij} \cdot z_{ij} - \sum_{i \in V} \pi_i \cdot y_i \tag{10}$$

$$\text{s.t.} \quad \sum_{e \in \delta(\{i\})} x_e = 2y_i \qquad \forall i \in V \tag{11}$$

$$z_{ij} \leq y_i \qquad \forall ij \in D \tag{12}$$

$$z_{ij} \leq y_j \qquad \forall ij \in D \tag{13}$$

$$z_{ij} \geq y_i + y_j - 1 \qquad \forall ij \in D \tag{14}$$

$$\sum_{i \in V} y_i \leq m \tag{15}$$

$$\sum_{e \in \delta(S)} x_e \geq 2(y_k + y_l - 1) \qquad \forall S \subset V, 3 \leq |S| \leq n-3, k \in S, l \notin S \tag{16}$$

$$x_e \in \{0,1\}, y_i \in \{0,1\}, z_{ij} \in \{0,1\} \tag{17}$$

The objective (10) corresponds exactly to the reduced cost given in equation (9). If a node is selected ($y_i = 1$), two links should be incident to node i, which is ensured by constraint (11). If both nodes i and j are selected the variable $z_{ij} = 1$, which is ensured by the constraints (12), (13) and (14). The number of nodes in the rings is bounded by the hop constraint (15). Subtour elimination constraints (16) ensure that a single ring is generated and finally integer solutions are ensured by the domain constraints (17).

We solve the ring-generation problem by branch-and-cut as described in [12], where the subtour elimination constraints are generated as needed. Also [4] describes cuts which may improve the performance of the branch-and-cut algorithm. The ring-generation problem is a generalization of the (Selective) Travelling Salesman Problem and of the Quadratic Knapsack problem and thus we denote it the Quadratic Selective Travelling Salesman Problem.

If branching is necessary, additional terms are added to the objective function and addional constraints are added. These additions are described in Section 5.1.

5 The Branch-and-Price Algorithm

The branch-and-price algorithm is described in pseudo code in Figure 2. The main

```
INCUMBENT = Infinity.
BRANCH-NODES = {Initial Relaxed Ring-Partitioning problem}
while BRANCH-NODES ≠ ∅ do
  Select branch B ∈ BRANCH-NODES
  do
    Solve relaxed ring-partitioning problem B
    Solve ring-generation problem, based on dual variables of B
    if Reduced cost of optimal ring < 0 then
      Add optimal ring to B
  while Reduced cost of optimal ring < 0
  Let OBJ_VAL = Optimum of B
  if the solution to the relaxed ring-partitioning problem is feasible (integer)
      and OBJ_VAL < INCUMBENT then
    Update incumbent: INCUMBENT = OBJ_VAL
    Fathom branch
  else if OBJ_VAL ≥ INCUMBENT then
    Fathom branch
  else
    Branch: Add two branches to BRANCH-NODES
end while
```

Figure 2: *The Branch-and-Price algorithm*

idea in a branch-and-price algorithm is to perform the bounding in a branch-and-bound algorithm using column generation. The algorithm maintains an incumbent, i.e. the lowest cost feasible solution known, and a set of branch-nodes, i.e. a set of relaxed ring-partitioning problems. Initially the set of branch-nodes contains the ring-partitioning problem without any branching decisions. A branch-node corresponding to a relaxed ring-partitioning problem is solved using column generation in the inner while loop. It is resolved in each iteration of the inner while loop and a ring is generated by the ring-generation problem. If no ring exists with negative reduced cost the value of the ring-partitioning problem is a lower bound. This lower bound is used in the outer loop which is the branch-and-bound part of the algorithm.

In the outer loop it is checked whether the optimal solution to the relaxed ring-partitioning problem solution is feasible, i.e. integer, or if it is a lower bound only. If the solution is integer and better than the current incumbent, the incumbent is updated and that branch is fathomed. If the solution is fractional, the lower bound is compared with the current incumbent and if it is worse, the branch is fathomed. If neither is the case, branching is performed.

5.1 Ryan-Foster Branching

Branching in a branch-and-price algorithm is more complicated than in a standard branch-and-bound algorithm. We use Ryan-Foster branching [6] to obtain integer solutions. This is possible since all coefficients of all constraints in the ring-partitioning problem are 0 or 1 and all right hand sides are 1, see constraint (7).

Consider constraint i. Since the right hand side is 1 and variables have to be integer, exactly one ring with $p_{ir} = 1$ has to be selected ($u_r = 1$). For all other selected rings, $p_{ir} = 0$. We say that "node i is covered by ring r". The idea is now to identify a set of rings $S \subset R^{met}$ and create two branches, 1) node i has to be covered by a ring in S and 2) node i has to be covered by a ring not in S. The question is now, how do we select i and S.

Assume node i is partially covered by more than one ring, and assume ring r is one of these rings (i.e. $0 < u_r < 1$). Usual variable branching corresponds to letting $S = \{r\}$, thus the branches will be $u_r = 1$ and $u_r = 0$. This sort of branching is not suitable in a column generation algorithm for several reasons all related to the vast amount of variables that exists (but are not explicitly known). First of all since we set $u_r = 0$ in the ring-partitioning problem, r usually has a negative reduced cost and hence when solving the ring-generation problem, r will be generated *again*. This can be handled by modifying the ring-generation problem to specifically exclude r. However, usually rings similar to r exists and thus these rings will be generated instead. This means that the bound of the $u_r = 0$ branch will not improve much when branching and we have an unbalanced branch-tree where the depth is considerable.

The idea is to let S contain several rings and in particular include rings which *have not yet been generated* (i.e. not in R). Thus in general $S \backslash R \neq \emptyset$. Identify a fractional ring ($0 < u_r < 1$) and two nodes i and j with $p_{ir} = 1$ and $p_{jr} = 1$. If the solution is fractional, such two nodes always exists. Let $S = \{r \in R^{met} | p_{ir} = 1 \wedge p_{jr} = 1\}$, that is the rings that cover both i and j. The two branches are thus, 1) i and j are covered by the same ring and 2) i and j are covered by different rings.

A branch decision is identified by a node-pair $\{i, j\}$ and whether i and j should be covered by the same ring or not. For a ring-partitioning problem, we have several such branch decisions of both types. Denote by $B^{SAME} \subset V^2$ the set of branching decisions where node-pairs should be covered by the same ring and correspondingly denote by $B^{DIFF} \subset V^2$ the set of branching decisions where node-pairs should be covered by different rings. Then we add the following constraints to the ring-partitioning problem which implement the actual branching.

$$\sum_{\{r \in R | p_{ir}=1 \wedge p_{jr}=1\}} u_r = 1 \qquad \forall \{i, j\} \in B^{SAME} \qquad (\gamma_{ij}) \tag{18}$$

$$\sum_{\{r \in R | p_{ir}=1 \wedge p_{jr}=1\}} u_r = 0 \qquad \forall \{i, j\} \in B^{DIFF} \qquad (\delta_{ij}) \tag{19}$$

We denote the dual variables of the constraints by $\gamma_{\{i,j\}}$ and $\delta_{\{i,j\}}$ as indicated. The constraints added to the ring-partitioning problem affect the calculation of the reduced costs of rings, thus the objective of the ring-generation problem is changed. Note that $p_{ir} = 1 \wedge p_{jr} = 1$ exactly if $z_{ij} = 1$ in the ring-generation problem. Let $\gamma_{\{i,j\}} = 0$ if $\{i, j\} \notin B^{SAME}$ and $\delta_{\{i,j\}} = 0$ if $\{i, j\} \notin B^{DIFF}$, then the objective of the ring-generation problem (see equation (10)) becomes:

$$\sum_{e \in E} c_e \cdot x_e - \sum_{ij \in D} (d_{ij} + \gamma_{\{i,j\}} + \delta_{\{i,j\}}) \cdot z_{ij} - \sum_{i \in V} \pi_i \cdot y_i \tag{20}$$

When solving the ring-generation problem, it is furthermore necessary to ensure that only rings which fulfill the branching decisions are generated. This is ensured by the following constraints.

$$y_i - y_j = 0 \qquad \forall \{ij\} \in B^{SAME} \tag{21}$$

$$y_i + y_j \leq 1 \qquad \forall \{ij\} \in B^{DIFF} \tag{22}$$

Both constraints allows rings where both $y_i = 0$ and $y_j = 0$, but constraints (21) ensure that if node i is selected, then so is j and vice versa. On the other hand, constraints (22) ensure that rings generated include at most one of i and j.

6 Computational Results

To test the branch-and-price algorithm, problem instances with between 10 and 20 nodes are generated. The problem instances are generated similarly to what is done in [12]. The nodes are placed in a plane with the coordinates uniformly distributed between 0 and 100. The fixed costs (c_e) are determined as the Euclidean distance.

Rather than generating both capacity costs (b_e) and the demands (d'_{ij}) and compute an average ring-length to obtain d_{ij} (as discussed in Section 3), we generate d_{ij} only. The d_{ij} values are generated as uniformly distributed between 0 and an upper bound u.

Selecting a proper value of u is critical. If u is selected too small, then the optimal solution is a single federal-ring including all nodes and no metro-rings. Using the same value of u as in [12] proved sufficient. The upper bound u used is given in the following equation.

$$u \approx \frac{5}{\sqrt{|V|^3}} \tag{23}$$

The value of u arise by considering the tradeoff between total average demand and the shortest tour measured in fixed link costs for rings with $|V|/2$ nodes. We refer to [12] for an in-depth explanation. The important observation is, that a tradeoff exists between the fixed link cost and the savings obtained from demands. As we shall see, the hop constraint (15) is in most, but not all cases binding; thus a tradeoff exists. The tests were run on a 1200 Mhz SUN Fire 3800. We use CPLEX 9.0 to solve linear programming models.

For each of 10, 12, 14, 16, 18 and 20 nodes, 10 different random instances are generated. We report results as averages over 10 instances. We vary the maximal number of nodes in the metro-rings, m between 4 and $\min\{10, |V|-3\}$. In addition to this, we investigate networks with 25 and 36 nodes with m equal to 5 and 6 respectively. It turns out, that since $|V|/m$ is integer for these networks, they are easier to solve than networks for which this is not the case. The results are given in Table 1. The table shows the number of nodes, the maximum number of nodes in metro-rings, the number of branch-nodes, the total time spent in seconds and the percentage spent on the ring-partitioning problem and the ring-generation problem respectively. Finally the number of times that metro-rings are generated (this includes cases where no metro-rings are actually found) and the number of metro-rings in the optimal solution are listed.

For all problem instances with up to 20 nodes, the branch-and-price algorithm terminates in at most 3 hours (average worst case is 73 minutes). Since the design of HRNs are considered strategic problems, the computational time is acceptable. As it can be seen, the bottleneck in the algorithm is the generation of rings which consistently takes more than 90% of the running time. The gradually increasing running time for increasing $|V|$ may both be attributed to increased running time for each ring-generation problem solved and to the increasing number of metro-rings which are generated (second last column). The number of branch-nodes is limited, making memory issues negligible. However, each branch requires generation of a substantial number of metro-rings, causing substantially higher running time.

Instances where $|V|/m$ is integer are easier than instances where this is not the case. This is due to the increased number of branch-nodes which is caused by an

$\|V\|$	m	#Branch Nodes	Total Time (sec.)	Time Part.	Time Gene.	#Rings Gene.	#Metro Rings
10	4	8.0	4.0	6.7%	93.3%	40.2	3.0
10	5	1.0	2.5	4.6%	95.4%	18.8	2.0
10	6	11.2	7.8	4.7%	95.3%	64.4	2.0
10	7	7.2	4.9	5.7%	94.3%	42.1	2.0
12	4	4.0	3.7	4.9%	95.1%	23.0	3.0
12	5	13.0	16.4	3.9%	96.1%	83.2	3.0
12	6	1.0	7.3	3.1%	96.9%	30.1	2.0
12	7	15.0	24.5	3.5%	96.5%	105.5	2.0
12	8	19.2	27.6	3.7%	96.3%	123.4	2.0
12	9	9.8	14.1	3.6%	96.4%	65.4	2.0
14	4	3.2	5.7	4.7%	95.3%	27.2	4.0
14	5	6.4	18.5	2.6%	97.4%	51.1	3.0
14	6	44.4	105.1	2.7%	97.3%	277.3	3.0
14	7	1.0	29.8	1.4%	98.6%	46.8	2.0
14	8	32.6	110.2	2.9%	97.1%	275.8	2.0
14	9	50.6	118.5	3.2%	96.8%	327.5	2.0
14	10	36.4	85.6	3.4%	96.6%	251.5	2.0
16	4	5.8	11.9	4.2%	95.8%	39.6	4.4
16	5	29.4	87.9	2.6%	97.4%	176.3	4.0
16	6	34.4	158.4	2.1%	97.9%	251.1	3.0
16	7	68.8	359.3	2.3%	97.7%	539.8	3.0
16	8	1.8	84.2	0.9%	99.1%	68.7	2.0
16	9	44.2	324.9	1.9%	98.1%	405.2	2.0
16	10	60.6	383.0	2.5%	97.5%	570.7	2.0
18	4	16.4	32.3	3.6%	96.4%	74.2	5.0
18	5	2.6	32.4	1.8%	98.2%	40.3	4.0
18	6	6.0	89.5	1.2%	98.8%	75.5	3.2
18	7	33.4	446.5	1.3%	98.7%	325.6	3.0
18	8	124.0	1183.7	2.1%	97.9%	1116.0	3.0
18	9	1.0	217.3	0.6%	99.4%	89.9	2.0
18	10	30.2	737.2	1.3%	98.7%	440.2	2.0
20	4	6.8	27.0	3.9%	96.1%	50.6	5.7
20	5	8.6	91.8	1.5%	98.5%	71.7	4.7
20	6	24.2	356.4	1.1%	98.9%	190.3	4.0
20	7	12.8	407.0	0.7%	99.3%	143.4	3.2
20	8	53.4	1854.6	0.9%	99.1%	659.6	3.0
20	9	179.8	4344.2	1.4%	98.6%	2026.2	3.0
20	10	1.0	688.0	0.3%	99.7%	117.8	2.0
25	5	11.2	302.0	1.0%	99.0%	110.8	5.9
36	6	21.0	5457.8	0.4%	99.6%	245.9	7.0

Table 1: Computational Results. Averages over 10 instances.

increased amount of fractional variables. Especially when $|V|/m = 2$, the possibility of obtaining an integer solution without branching is high. The special case when all metro-rings and the federal-ring have the same number of nodes, i.e. $|V|/m = m$ and $|V|/m$ integer, is considered in [7, 8, 9]. Since $|V|/m$ is integer, as discussed above, such instances are easier to solve to optimality than instances where this is not the case. The last two rows in Table 1 gives results for instances with $|V| = 25$, $m = 5$ and $|V| = 36$, $m = 6$. The most difficult instances with 36 nodes are solved in less than 6 hours and on average over 10 instances in just above $1\frac{1}{2}$ hour.

For networks with up to 20 nodes, in most cases, the optimal solution contains exactly the minimum number of metro-rings needed, given m. Only in 22 cases out of 380 test runs in total, one more than the minimum number of metro-rings needed is in the optimal solution. In Table 1, this is the reason why the last column contains fractions. This indicates that the demand values are sufficiently high to make the metro-rings profitable and the hop constraint (15) thus binding. On the other hand, since some instances exists for which this is not the case, the demand values are not too high.

The results reported here involve the design of the metro-rings only. Recall that the problem of designing the federal-ring is actually a Generalized Travelling Salesman Problem and can thus be solved using methods from [2].

6.1 Future Research

In order to be able to handle larger instances in reasonable time, it is paramount to reduce the time spent on ring-generation. Note that for each branch-node, at least one ring-generation problem has to be solved (the one giving no rings) to ensure that the value obtained when solving the ring-partitioning problem is indeed a bound. Thus it is inevitable that the ring-generation problem has to be solved at least as many times as there are branch-nodes. The remaining number of times that rings are generated can be reduced, however, by e.g. pre-generating rings or by generating more rings at a time. The actual time spent on each ring-generation problem can be reduced by using heuristics.

As mentioned in Section 3, the optimal solution of the modified problem may not be optimal in the original combined design and routing problem. This is mainly for 2 reasons, 1) the metro-rings and the federal-ring is designed in separate (thus non-optimal) stages and 2) the modification assumes the capacity costs of rings are the same. It seems possible but nontrivial to cope with 1) by including the federal-ring design in the branch-and-price algorithm, but 2) is more difficult. However, one initial approach to take is to investigate how much the optimal solution for the modified problem deviates from the optimal solution to the original problem. This could be done either by investigating very small instances for which optimal solutions can be found or by finding a lower bound on the original problem cost.

Also it would be interesting to allow bidirectional instead of unidirectional self-healing rings. One possibility is to use the same problem modification, and thus approximate the bidirectional rings with unidirectional rings. However, in that cases there is even more reason to investigate 2).

7 Conclusion

In this paper we have considered the problem of designing HRNs. A problem modification has been presented which has previously been used to build heuristics for designing HRNs. A branch-and-price algorithm is described, implemented and tested. For the modified problem this algorithm finds provably optimal solutions to networks with up to 20 nodes in less than 3 hours. For problems with special structure, the algorithm finds provably optimal solutions with up to 36 nodes in less than 6 hours. The computational time depends heavily on the instances considered, and in particular it is possible to design considerably larger networks if the maximum number of nodes in metro-rings are small and/or if the maximum number of nodes divides the number of nodes in the network. Algorithmic improvements which could speed up the algorithm have been suggested and we also suggest an investigation of how much the optimal solution to the modified problem deviates from the solution to the original problem. In particular this investigation is important if bidirectional self healing rings are considered.

References

[1] C. Barnhart, E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsbergh, and P.H. Vance. Branch-and-price: column generation for solving huge integer programs. *Operations Research*, 46(3):316–29, 1998.

[2] M. Fischetti, J.J Salazar Gonzalez, and P. Toth. A branch-and-cut algorithm for the symmetric generalized traveling salesman problem. *Operations Research*, 45(3):378–94, 1997.

[3] John G. Klincewicz. Hub location in backbone/tributary network design: a review. *Location Science*, 6(1-4):307–335, 1998.

[4] V. Mak and T. Thomadsen. Polyhedral combinatorics of the cardinality constrained quadratic knapsack problem and the quadratic selective travelling salesman problem. *Submitted for publication*, 2004.

[5] A. Proestaki and M.C. Sinclair. Design and dimensioning of dual-homing hierarchical multi-ring networks. *IEE Proceedings-Communications*, 147(2):96–104, 2000.

[6] D.M. Ryan and B. Foster. An integer programming approach to scheduling. *Computer Scheduling of Public Transport. Urban Passenger Vehicle and Crew Scheduling. Proceedings of an International Workshop*, pages 269–80, 1981.

[7] Jianxu Shi and J.P. Fonseka. Dimensioning of self-healing rings and their interconnections. *Global Telecommunications Conference, 1993, including a Communications Theory Mini-Conference. Technical Program Conference Record, IEEE in Houston. GLOBECOM '93., IEEE*, pages 1579 –1583 vol.3, 1993.

[8] Jianxu Shi and J.P. Fonseka. Design of hierarchical self-healing ring networks. *Communications, 1994. ICC '94, SUPERCOMM/ICC '94, Conference Record, 'Serving Humanity Through Communications.' IEEE International Conference on*, pages 478–482 vol.1, 1994.

[9] Jianxu Shi and J.P. Fonseka. Hierarchical self-healing rings. *IEEE/ACM Transactions on Networking*, 3(6):690–697, 1995.

[10] J.J. Shi and J.P. Fonseka. Interconnection of self-healing rings. *1995 IEEE International Conference on Communications. Converging Technologies for Tomorrow's Applications. ICC '96. Conference Record (Cat. No.96CH35916)*, pages 1563–7 vol.3, 1996.

[11] J.J. Shi and J.P. Fonseka. Analysis and design of survivable telecommunications networks. *IEE Proceedings-Communications*, 144(5):322 –330, 1997.

[12] T. Thomadsen and T. Stidsen. A branch-and-cut algorithm for the quadratic selective tsp. *Submitted to Telecommunication Systems*, 2003.

[13] F. Vanderbeck and L.A. Wolsey. An exact algorithm for ip column generation. *Operations Research Letters*, 19:151–159, 1996.