Survivable Network Design using Polyhedral Approaches

Yogesh K. Agarwal
Indian Institute of Management
Prabandh Nagar, Off Sitapur Road
Lucknow, INDIA – 226 013
yka@iiml.ac.in

Abstract— We consider the problem of designing a survivable telecommunication network using facilities of a fixed capacity. Given a graph \( G = (V,E) \), the traffic demand among the nodes, and the cost of installing facilities on the edges of \( G \), we wish to design the minimum cost network, so that under any single edge failure, the network permits the flow of all traffic using the remaining capacity. The problem is modeled as a mixed integer program, which can be converted into a pure integer program by applying the well-known Japanese Theorem on multi-commodity flows. Using a key theorem that characterizes the facet inequalities of this integer program, we derive several families of 3- and 4-partition facets, which help to achieve extremely tight lower bounds on the problem. Using these bounds, problems of up to 20 nodes and 40 edges have been solved optimally in a previous work. Using heuristic approaches based on this framework, we solve problems of up to 40 nodes and 80 edges to obtain solutions that are approximately within 5% of optimal solutions.

Keywords— telecommunications, multi-commodity flow, network design, survivability, integer programming, polyhedral structure, facet inequalities, k-partition.

I. INTRODUCTION

The standard network design problem (NDP) is to find the minimum cost installation of capacities on the edges of a graph that will permit a feasible multi-commodity flow of a given set of traffic demands. In the survivable network design problem, the capacities must be installed so that a feasible flow of all traffic will be possible under any single edge failure (hereafter referred to as fault).

We consider the single-facility version of the problem where the traffic demands as well as the edges are undirected. We are given an undirected graph \( G = (V,E) \) with \( n = |V| \) and \( m = |E| \), called the supply graph, and \( K \), a set of commodities. For each \( k \in K \), let \( d_k^i \) be the demand of commodity \( k \) at node \( i \). Note that we define all traffic with a common source node as a single commodity, with \( d_k^i \) being negative for the source node and non-negative for all other nodes. This method of defining commodities, called aggregate commodity definition, is more efficient compared to defining each origin-destination pair as a separate commodity. The former requires a maximum of \( n(n-1)/2 \) commodities as compared to \( n(n-1)/2 \) in case of latter. By rescaling the demands, if necessary, we assume that each facility has unit capacity, and \( c_e \) is the cost of installing each unit of capacity on edge \( e \in E \). Let \( x_e \) be the number of facilities installed on edge \( e \in E \), and \( f_{ij}^k \) the (directed) flow of commodity \( k \) on edge \( e=(i,j) \) under fault \( t \), i.e. when edge \( t \in E \) has failed. Then the problem can be formulated as the following mixed integer program.

\[
\begin{align*}
\text{Minimize} & \quad \sum_{e \in E} c_e x_e \\
\text{subject to} & \quad \sum_{i \in N} f_{ij}^k - \sum_{i \in N} f_{ji}^k = d_k^i \quad \forall j \in N, \forall k \in K, \forall t \in E \quad (1) \text{(SNDP-M)} \\
& \quad \sum_{k \in K} (f_{ij}^k + f_{ji}^k) \leq x_e \quad \forall e=(i,j) \in E, \forall t \in E \quad (2) \\
& \quad x_e \geq 0, \quad x_e \geq 0, \text{ integer}
\end{align*}
\]

Constraints (1) are the flow conservation constraints at each node for each commodity under each fault, and (2) are capacity constraints. The number of constraints in set (1) is \( n.m^2 \) and that in (2) is \( m \). The problem has \( m \) capacity variables, and \( (n-1)m^2 \) flow variables, i.e. \( (n-1)m \) flow variable for each of the \( m \) faults. Clearly, the size of this formulation is much larger than the standard NDP because the flow variables and flow conservation constraints are replicated for each fault.

II. THE PROJECTION POLYHEDRON

According to a well-known theorem on multi-commodity flows by Iri [??], and Onaga and Kakusho [??] (also see Lomonosov [??]), the flow conservation constraints in the formulation NDP-M can be replaced by constraints of the form \( \mu(c-d) \geq 0 \), where \( c \) and \( d \) are vectors of capacities and demands defined on the node-pairs of \( G \), and \( \mu \) is a metric.
The set of all metrics is infinitely large, and it forms a convex cone in $R^l$. If $M$ is the set of all primitive metrics, then SNDP-M can be reformulated as the following pure integer program:

$$\text{Minimize } \sum_{e \in E} c_e x_e$$
$$\text{s.t. } \sum_{e \in E(I)} \mu_e x_e \geq \sum \mu_{ij} d_{ij} \quad \forall \mu \in M, \forall e \in E \quad (1) \quad \text{SNDP}$$
$$x_e \geq 0, \text{ integer}$$

We note that solving SNDP directly using the above formulation is not practical, because $M$, the set of primitive metric is very large, and has not yet been fully characterized. However, facets of SNDP are clearly valid inequalities of SNDP-M. These facets, when added to SNDP-M, result in very strong bounds making it feasible to solve the problem by branch-and-cut methods for moderate size problems.

Although a complete description of $M$ is not available, it is well known that the cut-sets of $G$ are a very important subclass of primitive metrics. Consider a cut-set $(S,S')$ where $S \subseteq V$ and $S' = V \setminus S$, and define $\mu_i = 1$ if $i \in S$ and $j \in S'$, and $\mu_i = 0$ otherwise. Then it is easy to verify that $\mu$ as defined above for the cut-set $(S,S')$ is a metric, and the constraint $\mu(c-d) \geq 0$ reduces to the form $\sum_{e \in E(S,S')} \mu_{ij} d_{ij} \geq d(S,S')$, where $d(S,S')$ is the total demand of all commodities across the cut $(S,S')$. This is the familiar cut-inequality which has been extensively used in the network-design literature.

### III. THE K-PARTITION SUB-PROBLEM AND A KEY THEOREM

In Agarwal [2], the facets of the standard NDP was derived by partitioning the node-set $V$ into $k$ subsets: $V_1, V_2, \ldots, V_k$, then shrinking the nodes within each subset, and also the edges between the subsets, leading to a smaller $k$-node subproblem. According to Theorem 1 of [2], the facets of such $k$-node subproblems yield facets of the original problem if certain rather mild conditions are satisfied.

Here we extend the same approach for deriving the facets of SNDP, and present a similar theorem applicable to the subproblems resulting from $k$-partitions of the SNDP graph. An important difference is that while in case of NDP, all the edges between subsets $V_i$ and $V_j$ of the partition were shrunk into a single edge of the sub-problem, this cannot be done in case of SNDP. In case of SNDP the solution must permit a feasible routing of all traffic in the event of each edge failure. Therefore, a separate identity of each edge between distinct subsets must be maintained when defining the sub-problem resulting from the $k$-partition of $G$. In other words, while defining the sub-problem, we shrink the nodes within each subset $V_i$ into a single node, but the edges across the subsets are not shrunk, resulting in a $k$-node multi-graph, which may have multiple edges between its nodes. This shrinking process and the resulting $k$-node multi-graph are illustrated with an example in Figure 1.

Thus, a $k$-partition of $G$ leads to an SNDP defined on a multigraph of $k$ nodes. According to a theorem (see [2]), a facet of this $k$-node problem corresponds to a facet of the original problem.

Theorem 1: Given a facet-defining inequality $\alpha_x \geq \beta$ of polyhedron $P'$, inequality $\alpha_{x} \geq \beta$, with $\alpha$ constructed from $\alpha'$ as described above, is facet-defining for the polyhedron $P$ if the sub-graph $G_i$ of $G$ induced by each $V_i$ is 2-connected.

Our approach is to derive the facets of 3-node and 4-node multi-graphs, and translate them into facets of the larger problem by using this theorem.

### IV. REVIEW OF 2-PARTITION INEQUALITIES

This section is based on the results already reported by Beinstock and Muratore [??], Balakrishnan et. al. [??] and Magnanti and Wang [??]. Consider a 2-partition of $G$ and the resulting 2-node multi-graph with $m$ edges. Let $b$ be the total traffic demand across the partition. Then, according to Japanese Theorem, the following set of $m$ inequalities must clearly be satisfied by any feasible SNDP solution:

$$\sum_{e \in E(I)} x_e \geq b \quad \forall t \in E \quad (B_1)$$

Here $t$ is the index of the failed edge. We refer to these inequalities as B1-inequalities. These inequalities have been shown to be facet-defining by Beinstock and Muratore [?] for the polyhedron of 2-node SNDP with multiple edges, and hence they are also facet-defining for the original SNDP by virtue of Theorem 1.

Now consider adding all $m$ B1-inequalities, and divide both sides by $(m-1)$, and round up the RHS. This yields the following valid inequality:

$$\sum_{e \in E} x_e \geq b' \quad (B2)$$

Where $b' = \lceil b/(m-1) \rceil$. We refer to this inequality as B2-inequality. This inequality has also been shown to be facet-defining for 2-node SNDP under certain conditions (see Beinstock and Muratore [??]), and is therefore facet-defining for the original problem by virtue of Theorem 1.
In Sections V and VI below, we derive several classes of 3-partition and 4-partition inequalities by combining B1- and B2-inequalities described above. For convenience, we refer to B1- and B2- inequalities together as B-inequalities. The letter “B” in this labeling denotes a Bi-partition. Similarly, in the rest of this paper, inequalities derived from a 3-partition (or Tri-partition) are referred to a T-inequalities, and those from a 4-partition (tetRa-partition) as R-inequalities.

V. 3- AND 4-PARTITION INEQUALITIES

Consider a 3-node SNDP defined on a multi-graph \( G \). A 3-node SNDP has three 2-partitions: 1:23, 2:13 and 3:12. Let the traffic demands across these partitions be \( b_1, b_2, b_3 \), respectively. Let \( m_{ij} \) represent the number edges connecting nodes \( i \) and \( j \) in the multi-graph, and let \( E_{ij} \) represent the index set of edges between nodes \( i \) and \( j \).

The projection of this 3-node SNDP has the following constraint set comprising of B1-inequalities.

\[
\sum_{\forall e \in (E_{12} \cup E_{13} \setminus \{t\})} x_e \geq b_1 \quad \forall t \in E_{12} \cup E_{13} \quad (B1^t_1)
\]

\[
\sum_{\forall e \in (E_{12} \cup E_{23} \setminus \{t\})} x_e \geq b_2 \quad \forall t \in E_{12} \cup E_{23} \quad (B1^t_2)
\]

\[
\sum_{\forall e \in (E_{13} \cup E_{23} \setminus \{t\})} x_e \geq b_3 \quad \forall t \in E_{13} \cup E_{23} \quad (B1^t_3)
\]

Note that the superscript of the inequality denotes the specific 2-partition of the 3-node problem for which the inequality is applicable. This additional notation that precisely labels each inequality is necessary for further discussion, where we derive new valid inequalities using specific combinations of some of these inequalities. There are a total of \( 2(m_{12} + m_{13} + m_{23}) \) of these B1-inequalities in a 3-node problem, as each edge falls on two of the three 2-partitions.

In addition, we have the following three B2-inequalities, one for each 2-partition

\[
\sum_{\forall e \in (E_{12} \cup E_{13})} x_e \geq b_1' \quad (B2^1)
\]

\[
\sum_{\forall e \in (E_{12} \cup E_{23})} x_e \geq b_2' \quad (B2^2)
\]

\[
\sum_{\forall e \in (E_{13} \cup E_{23})} x_e \geq b_3' \quad (B2^3)
\]

Where \( b_1' = \lceil (m_{12} + m_{13})b_1/(m_{12} + m_{13} + 1) \rceil \) etc.

For a given \( t \in E_{12} \), consider adding the inequalities \( B1^t_1, B1^t_2 \), and \( B2^3 \), and dividing the resulting inequality by 2. If \( b_1 + b_2 + b_3 \) is odd, then the resulting inequality can be strengthened by rounding up the RHS. This yields the following valid inequality:

\[
\sum_{\forall e \in E_1 \setminus \{t\}} x_e \geq \alpha_1^t (T1^t)
\]

where \( \alpha_1^t = \lceil (b_1'+b_2'+b_3')/2 \rceil \). If \( b_1+b_2+b_3 \) is odd, the resulting inequality is non-redundant. Note that the inequality contains all edges except edge \( t \in E_{12} \). We call edge \( t \) the excluded-edge, and note that many inequalities derived in the rest of this paper have one or more such excluded edges. The specific edge \( t \) which is excluded is indicated in the superscript of the label. It is possible to get such an inequality for any \( t \in E_{12}, t \in E_{13}, \) or \( t \in E_{23} \) by appropriately selecting the B-inequalities to combine. This inequality is shown to be facet-defining under appropriate conditions (see Agarwal [4]).

Now, consider adding the inequalities \( B2^1, B2^2 \) and \( B2^3 \), and dividing the resulting inequality by 2. If \( b_1'+b_2'+b_3' \) is odd, then the inequality can be strengthened by rounding up the RHS. This yields the following valid inequality:

\[
\sum_{\forall e \in E_2} x_e \geq \alpha_2 (T2)
\]

where \( \alpha_2 = \lceil (b_1+b_2+b_3)/2 \rceil \). If \( b_1+b_2+b_3 \) is odd, the resulting inequality is non-redundant. This inequality is also shown to be facet-defining under appropriate conditions.

Similarly, by combining various combinations of B1 and B2 inequalities, a large number of new valid inequalities can be derived for a 4-node multi-graph. For details the reader is referred to Agarwal [4].

Using appropriate separation heuristics described in Agarwal [4], these violated inequalities can be easily generated and added to the formulation SNDP-M in order to strengthen the same. This strengthening leads to a substantial improvement in the linear programming lower bounds, due to which the problems of up to 20 nodes can be solved optimally by branch and cut methods. In the next section we describe a heuristic algorithm based on this framework which has been successfully used to solve larger problems of up to 40 nodes and 80 edges for obtaining solutions which are shown to be within 0-5% of optimal solution.

VI. A POLYHEDRAL HEURISTIC FOR SURVIVABLE NETWORK DESIGN

The primary difficulty with formulation SNDP-M is the large number of variables and constraints in it. As the network size increases, the formulation size grows too fast to permit optimal solutions within reasonable time. For example, a
problem with 40 nodes, 80 edges, and 39 commodities has 249,680 variables and 256,080 constraints. On the other hand, the formulation SNDP contains only 80 variables, but it has an exponential number of constraints. Even if we confine ourselves to cut-metrics, the separation problem for generating them is NP-hard.

In view of these difficulties, we propose a two-phase heuristic approach for solving the problem. In Phase-I of the heuristic, we work with formulation SNDP. A simple heuristic described in Agarwal [4] is used to generate 2-, 3- and 4-partition inequalities described in Sections III and IV to solve the LP relaxation of a restricted version of SNDP. When no further inequalities can be generated, the problem is submitted to CPLEX IP solver to get the optimal integer solution. While solving this IP, we adopt a branch-and-cut approach, where at each node of the branch-and-bound tree we attempt to generate additional violated cuts, and add them to the problem. In our experience, CPLEX can solve this IP relatively quickly for moderate size problems because it has a small number of variables, and the most of the constraints added are facet inequalities. For solving larger problems, we impose a time limit on CPLEX solver. The search is terminated at the end of this time limit, and the optimality gap is reported. On 40-node problems, this gap is found to be less than 5% on all the problem tested.

The capacity solution obtained at the end of Phase-I may or may not permit a feasible routing under all faults. In Phase-II we identify the violated faults by solving the multi-commodity flow LP for each fault. This LP contains only the flow variables and flow-conservation constraint for one specific fault, and therefore, can be solved very quickly. In our experience, most of the time the Phase-I solution turns out to be feasible with respect to all faults. In some instances, when it is infeasible, the number of violated faults is generally quite small. Let \( \mathcal{F} \) be the set of violated faults, and let \( \theta_k \) be the capacity installed at edge \( e \) in the Phase-I solution. Then, in Phase-II we solve the following MIP.

\[
\begin{align*}
\text{Minimize} & \quad \sum_{e \in \mathcal{E}} c_{xe} \\
\text{s.t.} & \quad \sum_{j \in \mathcal{N}} x_{e} - \sum_{j \in \mathcal{L}} x_{e} = d_{j}^{k} \quad \forall j \in \mathcal{N}, \forall k \in \mathcal{K}, \forall t \in \mathcal{F} \quad (1) \\
& \quad \sum_{e \in \mathcal{E}} x_{e} \leq \theta_k \quad \forall (i,j) \in \mathcal{E}, \forall t \in \mathcal{F} \quad (MIP-II) \\
& \quad x_{e} \leq \theta_k \text{ integer}
\end{align*}
\]

We note the two important respects in which this MIP differs from the original formulation SNDP-M. First, the number of variables and constraints in this MIP is much smaller, as it is defined over set \( \mathcal{F} \) of violated faults, rather than all faults. To illustrate this, if the number of faults is decreased from 100 to 10, the number of variables for a 40-node problem will decrease from 249,680 to 31,280. Similarly, the number of flow-conservation constraints will decrease from 256,000 to 32000.

The second, and perhaps more important, difference is that we impose lower bounds \( \theta_k \) on each variable \( x_e \) of the problem. Thus, the objective of this MIP is to find a minimum cost capacity addition to the Phase-I solution, which will permit a feasible routing for the violated faults. Imposing this lower bound considerably restricts the search space for the MIP, and the problem can be solved very efficiently despite a rather large number of variables and constraints. The computational results in the next section reveal that Phase-II, when it is required, usually adds only one or two additional facilities to the Phase-I solution to make it feasible.

It is possible that in some rare instances, the number of violated faults for the Phase-I solution may be quite large, which will make Phase-II problem difficult to solve. In that case, we can impose a reasonable upper limit \( k \) on the size of \( \mathcal{F} \), and include only the \( k \) most violated faults in it. If this is done, there is no guarantee that the solution of Phase-II MIP will be feasible with respect to the other violated faults that were not included in \( \mathcal{F} \). In that case, we solve the Phase-II problem once again with a newly defined set \( \mathcal{F} \), which contains only the faults which are still violated by the current solution. Strictly speaking, this step may need to be repeated several times before the solution is feasible with respect to all faults. However, our computational experience shows that a second application of Phase-II is never actually required even when we use a small value of \( k \), i.e. \( k=5 \).

If the Phase-I IP is solved optimally, it is obvious that the Phase-I solution provides a valid lower bound on the problem. When Phase-I IP is terminated prematurely, the least LP solution at any node of the branch-and-bound tree is taken as a valid lower bound. On the other hand, Phase-II solution is an upper bound. Thus, the gap between these two values is a measure of the quality of solution obtained. If the gap is sufficiently small, we can feel confident that solution obtained is close to optimal. Our computational experiments reveal that when the facility capacity is relatively large, the Phase-I solution is almost always feasible with respect to all faults, and is therefore the optimal. In all instances for problems of up to 40 nodes, the gap between the lower bound and the upper bound is less than 5% which demonstrates the effectiveness of this approach.

The above discussion is summarized the following algorithm.

**A Polyhedral Heuristic for SNDP**

(Phase – I)

Step 0: Form the Phase-I LP with no constraints. This LP has \( x_e = 0 \) \( \forall e \) as the optimal solution.

Step 1. Given the current capacity solution \( \{x_e\} \), generate a violated 2-, 3- or 4-partition inequality. If no such inequality can be generated, go to Step 3, otherwise go to Step 2.

Step 2. Add the violated inequality found in Step 1 to the LP, solve the LP, and go back to Step 1.

Step 3. Submit the Problem to CPLEX to solve the IP with a time limit \( T \). Let \( Z_L \) be the lower bound, \( Z_I \) the value of Phase-
I integer solution obtained, and \( \theta_e \) the value of variable \( x_e \) in this integer solution. Go to Step 4.

(Phase – II)

Step 4. Determine the set of violated faults \( \mathcal{F} \) by solving the multi-commodity flow LP for each fault for Phase - I capacity solution \{\( \theta_e \)\}. If the number of violated faults is more than \( k \), keep only the \( k \) most-violated faults in set \( \mathcal{F} \). If \( \mathcal{F} \) is empty, set \( Z_U = Z_L \) and go to Step 6. Otherwise go to Step 5.

Step 5. Solve the Phase-II MIP (MIP-II), and update the values of \( \theta_e \) in accordance with this solution. Go back to Step 4.

Step 6. Accept the current solution as final, with the optimality gap of \( Z_U - Z_L \), and STOP.

In our implementation, we have taken \( k = 5 \) and imposed a time limit of T=10 minutes on Phase-I IP.

In the next section we report the computational results obtained with this heuristic approach.

VII. COMPUTATIONAL EXPERIMENTS

We tested the performance of the heuristic on two sets of problems with 25 and 40 nodes respectively. We generate the coordinates of nodes randomly on a grid of 100 by 100. The traffic demands among various node-pair are generated from Uniform(1,10) distribution. The facility cost between two nodes has a fixed cost component of $100 and a distance cost of $10 per unit distance. In order to generate the edges of graph \( G \), we start with a complete graph, and gradually remove long edges if the shortest path distance between the end nodes of the edge would not increase significantly by removal of the edge. We also ensure that the degree of any node does not fall below 2 in this edge removal process. This edge removal process is continued until the number of edges in the graph decreases to the desired value (50 and 80, respectively, for 25- and 40-node problems). We report results on 5 instances of 25-node problems, and 5 instances of 40-node problems.

Tables 1 and 2 reports the results for 25- node problems with facility capacities of 100 and 200 respectively.

### Table 1: Results for 25-node problems with \( C=100 \)

<table>
<thead>
<tr>
<th>Instance</th>
<th>( Z_L )</th>
<th>( Z_I )</th>
<th>( Z_U )</th>
<th>% Gap</th>
<th>CPU (sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>15010</td>
<td>15010</td>
<td>15010</td>
<td>0.0</td>
<td>60</td>
</tr>
<tr>
<td>2</td>
<td>14590</td>
<td>14590</td>
<td>14590</td>
<td>0.0</td>
<td>58</td>
</tr>
<tr>
<td>3</td>
<td>15450</td>
<td>15450</td>
<td>15450</td>
<td>0.0</td>
<td>49</td>
</tr>
<tr>
<td>4</td>
<td>16110</td>
<td>16110</td>
<td>16110</td>
<td>0.0</td>
<td>47</td>
</tr>
<tr>
<td>5</td>
<td>13930</td>
<td>13930</td>
<td>13930</td>
<td>0.0</td>
<td>143</td>
</tr>
</tbody>
</table>

It is noteworthy that all instances of 35-node problems were solve optimally, and Phase-II was never required.

### Table 2 : Results for 25-node problems with \( C=200 \)

<table>
<thead>
<tr>
<th>Instance</th>
<th>( Z_L )</th>
<th>( Z_I )</th>
<th>( Z_U )</th>
<th>% Gap</th>
<th>CPU (sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8510</td>
<td>8510</td>
<td>8510</td>
<td>0.0</td>
<td>71</td>
</tr>
<tr>
<td>2</td>
<td>8200</td>
<td>8200</td>
<td>8200</td>
<td>0.0</td>
<td>65</td>
</tr>
<tr>
<td>3</td>
<td>8730</td>
<td>8730</td>
<td>8730</td>
<td>0.0</td>
<td>41</td>
</tr>
<tr>
<td>4</td>
<td>9080</td>
<td>9080</td>
<td>9080</td>
<td>0.0</td>
<td>40</td>
</tr>
<tr>
<td>5</td>
<td>7880</td>
<td>7880</td>
<td>7880</td>
<td>0.0</td>
<td>74</td>
</tr>
</tbody>
</table>

Tables 3 and 4 report the results for 40-node problems with facility capacities of 500 and 800, respectively.

### Table 3 : Results for 40-node problems with \( C=500 \)

<table>
<thead>
<tr>
<th>Instance</th>
<th>( Z_L )</th>
<th>( Z_I )</th>
<th>( Z_U )</th>
<th>% Gap</th>
<th>CPU (sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9471.41</td>
<td>9950</td>
<td>9950</td>
<td>4.81</td>
<td>799</td>
</tr>
<tr>
<td>2</td>
<td>10860.00</td>
<td>10860</td>
<td>10860</td>
<td>0.00</td>
<td>512</td>
</tr>
<tr>
<td>3</td>
<td>9483.67</td>
<td>9780</td>
<td>9780</td>
<td>3.03</td>
<td>930</td>
</tr>
<tr>
<td>4</td>
<td>9990.00</td>
<td>9990</td>
<td>9990</td>
<td>0.00</td>
<td>564</td>
</tr>
<tr>
<td>5</td>
<td>9760.00</td>
<td>9760</td>
<td>9920</td>
<td>1.61</td>
<td>540</td>
</tr>
</tbody>
</table>

Note that three out of these ten instances were solved optimally, and the maximum gap is less than 5%. The average gap for all ten instances is 1.89% which demonstrates the effectiveness of this approach. It is also noteworthy that in only one out of the ten instances (highlighted in Table 3) the Phase-I IP solution was not feasible, and a Phase-II LP had to be solved.

To gain further insight into the role of Phase-I and Phase-II of the heuristic, we depict the network solution for this instance of 40-node problem. This solution is shown in Figure 2.
Figure 2 : Solution of a 40-node Problem

Solid lines in the diagram show the Phase-I solution, which represents a lower bound on the solution value because the Phase-I IP was solve to optimality. Two dotted lines represent the additional facilities added to Phase-I solution to make it feasible with respect to all the faults. The cost of these two facilities represents the gap between the lower bound and the upper bound.

VIII. CONCLUDING REMARKS

We have presented an effective heuristic for solving the survivable network design problem using a polyhedral approach based on the integer programming formulation of the problem. Instances of up to 40 node problems have been solved, and the solutions obtained are either optimal or very close to optimal. The work is currently on to attempt solutions of even larger problems using this approach and we feel confident that by adopting suitable approximation schemes, this approach can be used to obtain close to optimal solutions for most of the real world problems.

REFERENCES


