Comparison of formulations and a heuristic for packing Steiner trees in a graph

Sunil Chopra

Department of Managerial Economics and Decision Sciences,
J.L. Kellogg Graduate School of Management, Northwestern University,
Evanston, IL 60208, USA

In this paper, we consider the problem of packing Steiner trees in a graph. This problem arises during the global routing phase of circuit layout design. We consider various integer programming formulations and rank them according to lower bounds they provide as LP-relaxations. We discuss a solution procedure to obtain both lower and upper bounds using one of the LP-relaxations. Computational results to test the effectiveness of our procedures are provided.

Keywords: Steiner tree, packing, LP-relaxation.

1. Introduction

In this paper, we consider the problem of packing Steiner trees in a graph. This is the fundamental problem in the global routing phase of circuit layout design. An excellent background on combinatorial problems in circuit layout design is contained in Lengauer [10]. The problem can be described as follows.

Consider an undirected graph $G = (V, E)$ with positive integer capacities c_e and nonnegative lengths $w_e \in \mathbb{R}_+$, for each edge in E. $\Omega = \{N_i, i = 1, 2, \ldots, k\}$ is a family of subsets of the node set V. Ω is referred to as a *netlist* and each subset N_i defines a *net* i. The nodes in N_i are the *terminals* of net i. Let $\Gamma = \{T_i, i = 1, \ldots, k\}$ be a family of trees such that T_i spans N_i. Such a tree T_i is called a *Steiner tree* for net i. Γ is referred to as a *routing*. The *traffic* $U(\Gamma, e)$ on an edge e for a routing Γ is the number of trees T_i in Γ that contain edge e.

$$U(\Gamma, e) = |\{i \mid T_i \in \Gamma,\ e \in T_i\}|. \tag{1.1}$$

The length of a routing Γ is given by

$$w(\Gamma) = \sum_{e \in E} w_e U(\Gamma, e). \tag{1.2}$$

The *load* on edge $e \in E$ in a routing Γ is defined to be

$$L(\Gamma, e) = \max\{0, U(\Gamma, e) - c_e\}. \tag{1.3}$$

An edge e is said to be *oversaturated* if $L(\Gamma, e) > 0$. Most of the above terminology is taken from Lengauer [10], and Lengauer and Lügering [11]. A routing is said to be *legal* if no edge is oversaturated.

Global routing problems may be constrained or unconstrained. In *constrained global routing* (CGR), we look for a legal routing of minimum length.

In *unconstrained global routing* (UGR), we look for a minimum length routing among those that minimize the maximum load on an edge. Essentially, we first look for the minimum excess capacity needed on each edge to produce a legal routing. Then we look for the minimum length legal routing given this excess.

Both versions of the problem are known to be NP-hard since they include as a special case the problem of finding a minimum length Steiner tree which is NP-hard (see Garey and Johnson [5]).

In this paper, we restrict attention to the unconstrained global routing problem. Most of the existing methods are sequential routing algorithms. The seminal work on sequential routing is that of Lee [9] and Moore [14]. These methods try to produce a routing by sequentially considering one net at a time. The method due to Raghavan and Thompson [16] is nonsequential. However, it requires that the Steiner trees under consideration be prespecified. This is not always feasible.

Integer programming methods have been considered by Lengauer [10], Lengauer and Lügering [11], and Grötschel et al. [7,8] (see also Martin [13]).

The objective of our study is to use integer programming methods to come up with good lower and upper bounds for UGR. We focus in particular on lower bounds, since this is one area that has traditionally been ignored. One simple lower bound for UGR can be obtained by ignoring the capacity constraints and constructing a routing using the minimum length Steiner trees for each net. The tighter the capacity constraints, the worse the lower bound provided by the above procedure. In this respect, good integer programming formulations can play a significant role. The hope is also that solutions giving good lower bounds can be used to obtain solutions giving tight upper bounds.

In section 2, we consider various integer programming formulations for UGR and compare them in terms of the LP-relaxation lower bound they provide. In section 3, we show how UGR can be solved by sequentially solving two related problems. Upper and lower bounding procedures are provided in each case. In section 4, we provide computational results testing our approach. In section 5, we discuss further work to be done.

We assume basic knowledge of graph theory. An edge e in an undirected graph with end nodes s and t will be referred to as $[s, t]$. An arc directed from s to t will be referred to as (s, t).

2. Integer programming formulations

In this section, we give four different formulations for UGR and compare the associated LP-relaxations.

2.1. THE UNDIRECTED FORMULATION

A formulation similar to the one we are about to give has also been considered by Lengauer [10], Lengauer and Lügering [11], and Grötschel et al. [7]. Define a variable $x_{i,e}$ for each net i and edge $e \in E$ where

$$x_{i,e} = \begin{cases} 1 & \text{if edge } e \text{ is in the Steiner tree spanning net } i, \\ 0 & \text{otherwise.} \end{cases}$$

Each partition $(X, V-X)$ of the nodes V of the graph G defines a cut. We call $(X, V-X)$ a *Steiner cut* for net i if $|X \cap N_i| \geq 1$ and $|(V-X) \cap N_i| \geq 1$. Let $\delta(X)$ be the set of edges in G with one end node in X and the other in $V-X$. For a net $i \in \{1, \ldots, k\}$ and each associated Steiner cut $(X, V-X)$, define the Steiner cut inequality

$$\sum_{e \in \delta(X)} x_{i,e} \geq 1. \tag{2.1}$$

Let u be the variable measuring the maximum load on edges. u is estimated by the following load inequality for $e \in E$:

$$\sum_{i=1}^{k} x_{i,e} - u \leq c_e. \tag{2.2}$$

Let M be a very large positive number. In particular, $M = k \sum_{e \in E} w_e$ suffices where $k = |\Omega|$. UGR can now be formulated as follows:

$$\begin{aligned} &\text{minimize} \quad Mu + \sum_{e \in E} w_e \left(\sum_{i=1}^{k} x_{i,e} \right) \\ &\text{subject to} \quad (x, u) \text{ satisfies (2.1), (2.2),} \\ &\qquad\qquad\quad x \geq 0, \quad u \geq 0, \quad x \text{ integer.} \end{aligned} \tag{2.3}$$

The large value of M ensures that we select only among those routings that minimize the maximum load on an edge. Define the polyhedra $\mathrm{LP1}(\Omega, G)$ and $\mathrm{IP1}(\Omega, G)$ where

$$\mathrm{LP1}(\Omega, G) = \{(x, u) \mid (x, u) \text{ satisfies (2.1), (2.2), } x \geq 0,\ u \geq 0\},$$

$$\mathrm{IP1}(\Omega, G) = \mathrm{conv}\{(x, u) \in \mathrm{LP1}(\Omega, G),\ x \text{ integer}\}.$$

2.2. THE DIRECTED FORMULATION

Given UGR on an undirected graph $G=(V, E)$, one can restate the problem on a corresponding directed graph as follows. Given $G=(V, E)$ with edge set E, construct the directed graph $D=(V, A)$ with arc set A, where arcs $a=(s, t)$ and $a'=(t, s)$ are in A if and only if edge $e=[s, t]$ is in E. The length w_a of an arc a is equal to the length of the corresponding undirected edge e. For each net i, we declare one of the nodes $r_i \in N_i$ as the root. An arborescence rooted at r_i is said to be a *Steiner arborescence* if it spans each node in N_i. A routing Γ on D consists of a set of Steiner aborescences – one for each net. Define a variable $y_{i,a}$ for each net i and arc $a \in A$ where

$$y_{i,a} = \begin{cases} 1 & \text{if arc } a \text{ is in the Steiner arborescence spanning net } i, \\ 0 & \text{otherwise.} \end{cases}$$

Define a cut $(X, V-X)$ to be a *directed Steiner cut* if $r_i \in X$ and $|(V-X) \cap N_i| \geq 1$. Define $\delta(X)$ to be the set of arcs directed from X to $V-X$ with one end in X and the other in $V-X$. For each net i and directed Steiner cut $(X, V-X)$, we obtain the directed Steiner cut inequality

$$\sum_{a \in \delta(X)} y_{i,a} \geq 1. \tag{2.4}$$

Consider an edge $e=[s, t]$ in the undirected graph G. Let $a=(s, t)$ and $a'=(t, s)$ be the corresponding arcs in the directed graph. Capacity is used up if either of the arcs a or a' is used by a Steiner arborescence. The load inequality is thus given as follows:

$$\sum_{i=1}^{k} y_{i,a} + \sum_{i=1}^{k} y_{i,a'} - u \leq c_e. \tag{2.5}$$

UGR can now be formulated as

$$\begin{aligned} &\text{minimize} && Mu + \sum_{a \in A} w_a \left(\sum_{i=1}^{k} y_{i,a} \right) \\ &\text{subject to} && (y, u) \text{ satisfies (2.4), (2.5),} \\ &&& y \geq 0, \quad u \geq 0, \quad y \text{ integer.} \end{aligned} \tag{2.6}$$

M is a large positive number as defined earlier. Define the polyhedra LP2(Ω, G) and IP2(Ω, G) where

$$\text{LP2}(\Omega, G) = \{(y, u) \mid (y, u) \text{ satisfies (2.4), (2.5), } y \geq 0,\ u \geq 0\},$$

$$\text{IP2}(\Omega, G) = \text{conv}\{(y, u) \in \text{LP2}(\Omega, G),\ y \text{ integer}\}.$$

2.3. THE EXPLICIT FORMULATION

This formulation has also been considered by Lengauer and Lügering [11]. For each net i, let $S_i = \{T_{ij}, j = 1, \ldots, n_i\}$ be the set of all Steiner trees in the graph G. n_i may be exponential in $|V|$. In this formulation, we explicitly define a variable z_{ij} for each Steiner tree T_{ij} for net i. The total number of variables is thus given by $\sum_{i=1}^{k} n_i$. Define the variable

$$z_{i,j} = \begin{cases} 1 & \text{if Steiner tree } T_{ij} \text{ is chosen to span net } i, \\ 0 & \text{otherwise.} \end{cases}$$

The following constraint ensures that exactly one Steiner tree is chosen for each net i:

$$\sum_{j=1}^{n_i} z_{i,j} = 1. \tag{2.7}$$

The load on each edge is measured by the following constraint:

$$\sum_{i=1}^{k} \sum_{j:e \in T_{ij}} z_{ij} - u \le c_e. \tag{2.8}$$

The length l_{ij} of a Steiner tree T_{ij} is given by

$$l_{ij} = \sum_{e \in T_{ij}} w_e.$$

UGR can now be formulated as

$$\begin{aligned} &\text{minimize} \quad Mu + \sum_{i=1}^{k} \sum_{j=1}^{n_i} l_{ij} z_{ij} \\ &\text{subject to} \quad (z, u) \text{ satisfies (2.7), (2.8),} \\ &\qquad\qquad z \ge 0, \quad u \ge 0, \quad z \text{ integer.} \end{aligned} \tag{2.9}$$

Define the polyhedra LP3(Ω, G) and IP3(Ω, G) where

$$\text{LP3}(\Omega, G) = \{(z, u) \mid (z, u) \text{ satisfies (2.7), (2.8), } z \ge 0, u \ge 0\},$$

$$\text{IP3}(\Omega, G) = \text{conv}\{(z, u) \in \text{LP3}(\Omega, G), z \text{ integer}\}.$$

A formulation similar to (2.9) has also been considered by Raghavan and Thompson [16].

2.4. A COMPARISON OF LP-RELAXATIONS

For any nonnegative edge lengths w_e, each of the three formulations (2.3), (2.6) or (2.9) will give the same optimal solution. The optimum to the LP-relaxation in each case gives a lower bound to the integer optimum. In this section, we compare the LP-relaxations in terms of the lower bound they provide. Define

$$V_1 = \min\left\{Mu + \sum_{e\in E} w_e\left(\sum_{i=1}^{k} x_{i,e}\right) \middle| (x,u) \in \mathrm{LP1}(\Omega, G)\right\},$$

$$V_2 = \min\left\{Mu + \sum_{a\in A} w_a\left(\sum_{i=1}^{k} y_{i,a}\right) \middle| (y,u) \in \mathrm{LP2}(\Omega, G)\right\},$$

$$V_3 = \min\left\{Mu + \sum_{i=1}^{k} \sum_{j=1}^{n_i} l_{ij} z_{ij} \middle| (z,u) \in \mathrm{LP3}(\Omega, G)\right\},$$

The following results rank the LP-relaxations in terms of the lower bound they provide.

THEOREM 2.4.1

V_1, V_2 and V_3 can be ordered as follows:

$$V_1 \le V_2 \le V_3. \qquad \square$$

The proof of theorem 2.4.1 is given as two separate propositions.

PROPOSITION 2.4.1

$$V_1 \le V_2.$$

Proof

Let $(\bar{y}, \bar{u}) \in \mathrm{LP2}(\Omega, G)$ be the extreme point defining V_2. Construct $(\bar{x}, \bar{u})$ where for each edge $e = [s, t]$ in the undirected graph and the corresponding arcs $a = (s, t)$, $a' = (t, s)$ in the directed graph, we have

$$\bar{x}_{i,e} = \bar{y}_{i,a} + \bar{y}_{i,a'}.$$

Clearly,

$$M\bar{u} + \sum_{e\in E} w_e\left(\sum_{i=1}^{k} \bar{x}_{i,e}\right) = M\bar{u} + \sum_{a\in A} w_a\left(\sum_{i=1}^{k} \bar{y}_{i,a}\right) = V_2.$$

Let $(X, V - X)$ be any Steiner cut for net i with $r_i \in X$.

$$\sum_{e \in \delta(X)} \bar{x}_{i,e} = \sum_{a \in \delta(X)} \bar{y}_{i,a} + \sum_{a \in \delta(V-X)} \bar{y}_{i,a} \geq \sum_{a \in \delta(X)} \bar{y}_{i,a} \geq 1.$$

Also,

$$\sum_{i=1}^{k} \bar{x}_{i,e} - \bar{u} = \sum_{i=1}^{k} (\bar{y}_{i,a} + \bar{y}_{i,a'}) - \bar{u} \leq c_e.$$

Thus, $(\bar{x}, \bar{u})$ satisfies both (2.1) and (2.2) and is a point in LP1(Ω, G). This shows that

$$V_1 \leq M\bar{u} + \sum_{e \in E} w_e \left(\sum_{i=1}^{k} \bar{x}_{i,e} \right) = V_2.$$ □

We now give an example where $V_1 < V_2$. Let $G = (V, E)$ be the complete graph on four nodes $V = \{1, 2, 3, 4\}$. Set $N_i = V - \{i\}$ for $i \in \{1, 2, 3, 4\}$. Let $w_e = 1$ and $c_e = 2$ for all edges in E. One can verify that $V_1 = 6$ and $V_2 = 8$ for arbitrary choice of r_i, $i \in \{1, 2, 3, 4\}$. In this case, V_2 is in fact the optimal solution.

PROPOSITION 2.4.2

$$V_2 \leq V_3.$$

Proof

Let $(\bar{z}, \bar{u}) \in$ LP3(Ω, G) be the extreme point defining V_3. Each variable z_{ij} corresponds to a Steiner tree T_{ij} for net i. The tree T_{ij} can be converted to an arborescence A_{ij} rooted at r_i by suitably directing the edges in T_{ij}. Form the vector $\bar{y}$ where for $i \in \{1, \ldots, k\}$, $a \in A$, we have

$$\bar{y}_{i,a} = \sum_{j: a \in A_{ij}} \bar{z}_{ij}. \tag{2.10}$$

Note that

$$V_3 = M\bar{u} + \sum_{i=1}^{k} \sum_{j=1}^{n_i} l_{ij} \bar{z}_{ij} = M\bar{u} + \sum_{a \in A} w_a \left(\sum_{i=1}^{k} \sum_{j: a \in A_{ij}} \bar{z}_{ij} \right)$$

$$= M\bar{u} + \sum_{a \in A} w_a \left(\sum_{i=1}^{k} \bar{y}_{i,a} \right).$$

Consider any directed Steiner cut defined by $(X, V - X)$ for the net i with $r_i \in X$. For $i \in \{1, \ldots, k\}$, we have

$$
\begin{aligned}
\sum_{a \in \delta(X)} \bar{y}_{i,a} &= \sum_{a \in \delta(X)} \sum_{j: a \in A_{ij}} \bar{z}_{ij} \\
&= \sum_{j=1}^{n_i} \sum_{a \in A_{ij} \cap \delta(X)} \bar{z}_{ij} \\
&\geq \sum_{j=1}^{n_i} \bar{z}_{ij} \qquad (\text{since } |A_{ij} \cap \delta(x)| \geq 1, \forall j) \\
&= 1 \qquad (\text{by (2.7)}).
\end{aligned}
$$

Further note that if $e = [s, t]$, $a = (s, t)$ and $a' = (t, s)$, then $|\{a, a'\} \cap A_{ij}| \leq 1$ for all i, j. Thus,

$$
\begin{aligned}
&\sum_{i=1}^{k} \bar{y}_{i,a} + \sum_{i=1}^{k} \bar{y}_{i,a'} - \bar{u} \\
&= \sum_{i=1}^{k} \left(\sum_{j: a \in A_{ij}} \bar{z}_{ij} + \sum_{j: a' \in A_{ij}} \bar{z}_{ij} \right) - \bar{u} \\
&= \sum_{i=1}^{k} \sum_{j: \{a, a'\} \cap A_{ij} = 1} \bar{z}_{ij} - \bar{u} \qquad (\text{since } |\{a, a'\} \cap A_{ij}| \leq 1) \\
&= \sum_{i=1}^{k} \sum_{j: e \in T_{ij}} \bar{z}_{ij} - \bar{u} \leq c_e \qquad (\text{by (2.8)}).
\end{aligned}
$$

Thus, $(\bar{y}, \bar{u}) \in \mathrm{LP2}(\Omega, G)$. This shows that

$$
V_2 \leq M\bar{u} + \sum_{a \in A} w_a \left(\sum_{i=1}^{k} \bar{y}_{i,a} \right) = V_3. \qquad \square
$$

Figure 1 contains an example for which $V_2 < V_3$. $G = (V, E)$ is shown in the figure with $N_1 = N_2 = \{1, 2, 4, 6\}$, $w_e = 1$, $\forall e \in E$, $c(e_i) = 2$ for $i \in \{1, 2, 3\}$, $c(e_i) = 1$ or $i \in \{4, \ldots, 9\}$. Set $r_1 = r_2 = 1$. One can verify that $V_2 = 9$, while $V_3 = 10$ (which is in fact the optimal solution).

Propositions 2.4.1 and 2.4.2 together prove theorem 2.4.1.

Remark 2.4.1

Note that in general $V_1 < V_2$ and $V_2 < V_3$.

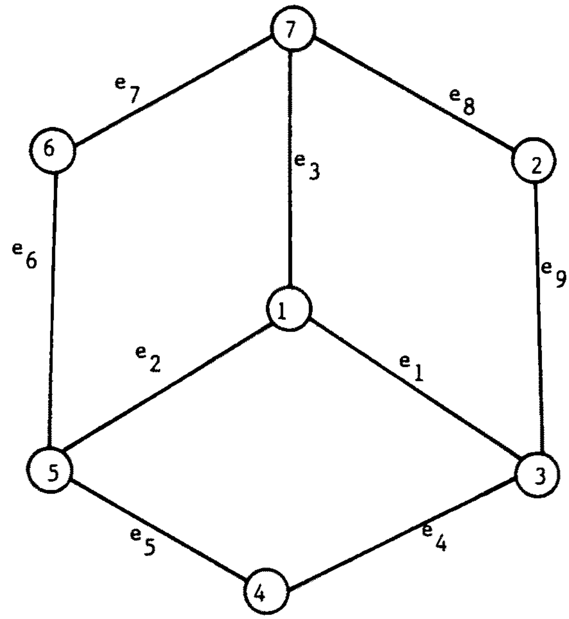

Fig. 1.

At this stage, we have shown that the LP-relaxation of formulation (2.9) is the strongest (in terms of lower bounds for the integer optimum), while the LP-relaxation of formulation (2.3) is the weakest. Notice that the total number of Steiner trees is exponential in the number of nodes and nets. Thus, the number of columns in the LP-relaxation of (2.9) is exponential in the size of the underlying Steiner packing instance. If column generation is used, we have to solve Steiner tree problems (which is NP-hard) to decide on incoming columns. Thus, solving the LP-relaxation of (2.9) is NP-hard. On the other hand, the LP-relaxations of (2.3) and (2.6) are polynomially solvable using results of Grötschel et al. [6], since Steiner cut inequalities can be identified in polynomial time. However, theorem 2.4.1 does not completely identify the gap between the LP-relaxation of formulations (2.9) and (2.6). We do so in the following development.

Given the directed graph $D=(V, A)$, a net i and a root $r_i \in N_i$, define the associated Steiner arborescence polyhedron

$$\mathrm{STP}(N_i, D) = \mathrm{conv}\{x(A_{ij}) \mid A_{ij} \text{ is a Steiner arborescence}\} + \mathbb{R}_+^A,$$

where $x(A_{ij})$ is the incidence vector of A_{ij}. Let $\alpha y \geq \alpha_0$ be any facet defining inequality for $\mathrm{STP}(N_i, D)$. Define the vector β where

$$\beta_{j,a} = \begin{cases} \alpha_a & \text{if } j = i, \\ 0 & \text{if } j \neq i. \end{cases}$$

The inequality $\beta y \geq \alpha_0$ is valid for IP2(Ω, G). Let $B^i y \geq b^i$ define the set of all valid inequalities that are obtained by lifting facet defining inequalities for STP(N_i, D) as described above. We should point out that a complete description of the set of inequalities $B^i y \geq b^i$ is unknown and is unlikely to be found since the Steiner tree problem itself is NP-hard. We strengthen formulation (2.6) by adding all inequalities of the form

$$B^i y \geq b^i \tag{2.11}$$

for $i = 1, 2, \ldots, k$. Define the polyhedron

$$\text{LP4}(\Omega, G) = \{(y, u) \mid (y, u) \text{ satisfies (2.4), (2.5), (2.11), } y \geq 0,\ u \geq 0\}$$

and

$$V_4 = \min\left\{Mu + \sum_{a \in A} w_a \left(\sum_{i=1}^{k} y_{i,a}\right) \Big| (y, u) \in \text{LP4}(\Omega, G)\right\}.$$

We now show that the optimum to the LP-relaxation of (2.9) is equal to the optimum obtained using LP4(Ω, G) as the LP-relaxation.

PROPOSITION 2.4.3

$$V_3 = V_4.$$

Proof

We first show that $V_4 \leq V_3$.

Let $(\bar{z}, \bar{u}) \in$ LP3(Ω, G) be the extreme point defining V_3. Define $\bar{y}$ as in (2.10). As in the proof of proposition 2.4.2, one can prove that $(\bar{y}, \bar{u})$ satisfies the inequalities (2.4) and (2.5). We now show that $(\bar{y}, \bar{u})$ also satisfies the inequalities (2.11). Each variable z_{ij} is associated with a Steiner tree T_{ij} for net i. One can construct the corresponding Steiner arborescence A_{ij} rooted at r_i by suitably directing the edges in T_{ij}. For each arborescence A_{ij}, we define its incidence vector t^{ij} for $i \in \{1, \ldots, k\}$, $j \in \{1, \ldots, n_i\}$, where

$$t_{s,a}^{ij} = \begin{cases} 1 & \text{if } s = i,\ a \in A_{ij}, \\ 0 & \text{otherwise,} \end{cases}$$

for $a \in A$, $s \in \{1, \ldots, k\}$. Each vector t^{ij} clearly satisfies the inequalities (2.11) for the corresponding net i. Note that

$$\bar{y} = \sum_{i=1}^{k} \sum_{j=1}^{n_i} \bar{z}_{ij} t^{ij}$$

and

$$\sum_{j=1}^{n_i} \bar{z}_{ij} = 1 \quad \text{for } i \in \{1, \ldots, k\}.$$

Thus, we have for $i \in \{1, \ldots, k\}$

$$\begin{aligned} B^i\bar{y} &= B^i \sum_{s=1}^{k} \sum_{j=1}^{n_s} \bar{z}_{sj} t^{sj} \\ &= \sum_{s=1}^{k} \sum_{j=1}^{n_s} \bar{z}_{sj} B^i t^{sj} \\ &= \sum_{j=1}^{n_i} \bar{z}_{ij} B^i t^{ij} \qquad \text{(since } B^i t^{sj} = 0 \text{ for } s \neq i) \\ &\geq \sum_{j=1}^{n_i} \bar{z}_{ij} b^i \qquad \text{(since } B^i t^{ij} \geq b^i) \\ &= b^i \qquad \text{(by (2.7)).} \end{aligned}$$

This shows that $(\bar{y}, \bar{u}) \in \text{LP4}(\Omega, G)$. Now

$$\begin{aligned} V_4 &\leq M\bar{u} + \sum_{a \in A} w_a \left(\sum_{i=1}^{k} \bar{y}_{i,a} \right) \\ &= M\bar{u} + \sum_{a \in A} w_a \left(\sum_{i=1}^{k} \sum_{j: a \in A_{ij}} \bar{z}_{ij} \right) \\ &= M\bar{u} + \sum_{i=1}^{k} \sum_{j=1}^{n_i} \bar{z}_{ij} \left(\sum_{a \in A_{ij}} w_a \right) = V_3. \end{aligned}$$

Now we consider the direction $V_3 \leq V_4$. Let $(\bar{y}, \bar{u})$ be the optimal vertex of LP4(Ω, G) defining V_4. Note that for net $i \in \{1, \ldots, k\}$, the inequalities (2.4) and (2.11) completely define the associated Steiner arborescence polyhedron STP(N_i, D). Given $i \in \{1, \ldots, k\}$, let $\bar{y}^i$ be the restriction of $\bar{y}$ to the variables $y_{i,a}$ for $a \in A$. $\bar{y}^i$ satisfies inequalities (2.4) and (2.11) for net i. This shows that $\bar{y}^i \in$ STP(N_i, D), i.e. $\bar{y}^i$ dominates the convex hull of a subset J_i of vertices of STP(N_i, D). Thus, we can write

$$\bar{y}_i = \sum_{j \in J_i} \alpha_j^i \bar{y}^{ij} + \delta_i, \tag{2.12}$$

where $\delta_i \geq 0$, $\alpha_j^i \geq 0$ and $\sum_{j \in J_i} \alpha_j^i = 1$. Further, $\bar{y}^{ij}$ is the incident vector of a Steiner arborescence A_{ij} for net i. For each vector $\bar{y}^{ij}$, define $\tilde{y}^{ij}$, where

$$\tilde{y}_{s,a}^{ij} = \begin{cases} \bar{y}_a^{ij} & \text{if } s = i, \\ 0 & \text{otherwise.} \end{cases}$$

Since $M>0$ and $w_a \geq 0$, $\forall a \in A$, we can in fact assume that $\delta=0$, $\mu=0$. Each vector $\tilde{y}^{ij}$ is the incidence of a Steiner arborescence A_{ij} (and a corresponding tree T_{ij}). Define $\bar{z}$, where

$$\bar{z}_{ij} = \alpha^i_j \quad \text{for } i \in \{1,\dots,k\}, \ j \in \{1,\dots,n_i\}.$$

Since $\sum_{j \in J_i} \alpha^i_j = 1$, we have

$$\sum_{j=1}^{n_i} \bar{z}_{ij} = 1 \quad \forall i \in \{1,\dots,k\}.$$

Also, for $e \in E$ (a and a' are the corresponding arcs in D)

$$\begin{aligned}\sum_{i=1}^{k} \sum_{j: e \in T_{ij}} \bar{z}_{ij} - \bar{u} &= \sum_{i=1}^{k} \sum_{j: e \in T_{ij}} \alpha^i_j - \bar{u} \\ &= \sum_{i=1}^{k} (\bar{y}_{i,a} + \bar{y}_{i,a'}) - \bar{u} \quad \text{(from (2.12))} \\ &\leq c_e .\end{aligned}$$

Thus, $(\bar{z}, \bar{u}) \in$ LP3(Ω, G). This shows that

$$V_3 \leq M\bar{u} + \sum_{i=1}^{k} \sum_{j=1}^{n_i} l_{ij}\bar{z}_{ij} = V_4.$$

The result thus follows. □

This shows that LP3(Ω, G) and LP4(Ω, G) are equivalent LP-relaxations in terms of the lower bound they provide.

2.5. A MULTI-COMMODITY FLOW FORMULATION

In this section, we give a multi-commodity flow based formulation that is shown to be equivalent to formulation (2.6) in terms of the lower bounds it provides. It does have the advantage of a polynomial number of variables and a polynomial number of constraints in the formulation.

Given the undirected graph $G=(V, E)$, form the directed graph $D=(V, A)$ as described in section 2.2. For each net i, one of the nodes $r_i \in N_i$ is declared as the root. For each net i and terminal $s \in N_i - \{r_i\}$, we define a commodity i, s. Given an arc $a=(f, h)$, let x^{is}_{fh} represent the flow of commodity i, s on arc a. In order to ensure a Steiner tree for each net $i \in \{1, \dots, k\}$, we must ensure a flow of one unit of commodity i, s from r_i to s for all $s \in N_i - \{r_i\}$. This is done using the following flow constraints:

$$\sum_{f \in V} x_{fh}^{is} - \sum_{f \in V} x_{hf}^{is} = \begin{cases} -1 & \text{if } h = r_i, \\ 1 & \text{if } h = s, \\ 0 & \text{if } h \neq r_i, s. \end{cases} \tag{2.13}$$

In constraints (2.13), if $(f, h) \notin A$, the variable x_{fh}^{is} is simply ignored. If we find a multi-commodity flow satisfying (2.13) for $s \in N_i - \{r_i\}$, the arcs with positive flow contain a Steiner arborescence spanning net i. To capture this, we define integer variables $y_{i,a}$, where

$$y_{i,a} = \begin{cases} 1 & \text{if there is a positive flow of any of the commodities } i, s \\ & \text{for } s \in N_i - \{r_i\} \text{ on arc } a, \\ 0 & \text{if the flow on arc } a \text{ is } 0. \end{cases}$$

The length of an arc $a = (f, h)$ is referred to as w_a or w_{fh}. UGR can then be formulated as follows:

$$\text{minimize} \quad Mu + \sum_{a \in A} w_a \left(\sum_{i=1}^{k} y_{i,a} \right)$$

$$\text{subject to} \quad x \text{ satisfies (2.13)},$$

$$x_{fh}^{is} \le y_{i,a} \quad \text{for } a = (f,h) \in A, \ i \in \{1, \dots, k\}, \ s \in N_i - \{r_i\}. \tag{2.14}$$

$$\sum_{i=1}^{k} y_{i,a} + \sum_{i=1}^{k} y_{i,a'} - u \le c_e$$

$$\text{for } a = (s,t), \ a' = (t,s), \ e = [s,t] \in E,$$

$$x, y, u \ge 0, \ y \in \{0,1\}. \tag{2.15}$$

This is a generalization of the multi-commodity flow formulation for the Steiner tree problem given by Wong [19]. Define the polyhedra LP5(Ω, G) and IP5(Ω, G), where

$$\text{LP5}(\Omega, G) = \{(x, y, u) \ge 0 \mid (x, y, u) \text{ satisfies (2.13), (2.14) and (2.15)}\},$$

$$\text{IP5}(\Omega, G) = \text{conv}\{(x, y, u) \in \text{LP5}(\Omega, G), y \text{ integer}\}.$$

Notice that LP5(Ω, G) is an LP-relaxation to this formulation of UGR and it is defined by a polynomial number of variables and constraints. This LP-relaxation can thus be solved in polynomial time. Define

$$V_5 = \min \left\{ Mu + \sum_{a \in A} w_a \left(\sum_{i=1}^{k} y_{i,a} \right) \middle| (x, y, u) \in \text{LP5}(\Omega, G) \right\}.$$

We now prove that the LP-relaxation lower bound provided by this formulation is equal to the lower bound obtained from the LP-relaxation to (2.6).

PROPOSITION 2.5.1

$$V_2 = V_5.$$

Proof

We first show that $V_2 \leq V_5$. Let $(\bar{x}, \bar{y}, \bar{u})$ be the extreme point of LP5(Ω, G) defining V_5. Notice that its restriction $(\bar{y}, \bar{u})$ satisfies inequality (2.5), which is equivalent to (2.15). Consider any directed Steiner cut $(X, V-X)$ with $r_i \in X$, $s \in (V-X) \cap N_i$. We have

$$1 \leq \sum_{(f,h)\in\delta(X)} \bar{x}^{is}_{fh} \quad \text{(since 1 unit of } i, s \text{ flows from } r_i \text{ to } s)$$

$$\leq \sum_{a\in\delta(X)} y_{i,a} \quad \text{(by (2.14), where } a = (f,h)).$$

This shows that $(\bar{y}, \bar{u})$ also satisfies inequalities (2.4), i.e. $(\bar{y}, \bar{u}) \in$ LP2(Ω, G). Thus, $V_2 \leq V_5$.

Conversely, let $(\bar{y}, \bar{u})$ be the extreme point of LP2(Ω, G) defining V_2. For each net i, consider the directed graph $D = (V, A)$ with arc capacities $\bar{y}_{i,a}$. For $s \in N_i - \{r_i\}$ we know that we can send one unit of flow from r_i to s in the above network, using the max-flow min-cut theorem (see Ford and Fulkerson [4]), since the minimum capacity cut separating r_i and s has capacity at least one. For each commodity i, s, send one unit of flow from r_i to s with arc capacities $\bar{y}_{i,a}$. Let $\tilde{x}^{is}_{fh}$ be the flow of commodity i, s on arc (f, h). For $a = (f, h)$, $i \in \{1, \ldots, k\}$, define

$$\tilde{y}_{i,a} = \max\{\tilde{x}^{is}_{fh} \mid s \in N_i - \{r_i\}\}.$$

For $a = (h,f)$, $a' = (f, h)$, $e = [h,f]$, define

$$\tilde{u} = \max_{a\in A}\left\{\sum_{i=1}^{k} \tilde{y}_{i,a} + \sum_{i=1}^{k} \tilde{y}_{i,a'} - c_e, 0\right\}.$$

It is easy to see that $(\tilde{x}, \tilde{y}, \tilde{u}) \in$ LP5(Ω, G). Further, by construction, $\tilde{y}_{i,a} \leq \bar{y}_{i,a}$, $\forall i \in \{1, \ldots, k\}$, $a \in A$, and $\tilde{u} \leq \bar{u}$. Since $M \geq 0$, $w_a \geq 0$, we have

$$V_2 = M\bar{u} + \sum_{a\in A} w_a \left(\sum_{i=1}^{k} \bar{y}_{i,a}\right) \geq M\bar{u} + \sum_{a\in A} w_a \left(\sum_{i=1}^{k} \tilde{y}_{i,a}\right) \geq V_5.$$

The result thus follows. □

This shows that LP2(Ω, G) and LP5(Ω, G) are equivalent in terms of the LP-relaxation lower bound they provide.

3. Obtaining the upper and lower bounds

In this section, we describe the procedure used to obtain upper and lower bounds for UGR. From theorem 2.1 and proposition 2.4.4, it follows that either LP3(Ω, G) or LP4(Ω, G) is likely to give the best lower bound. Unfortunately, both are NP-hard. Thus, as a relaxation to LP4(Ω, G), we have to use either LP1(Ω, G) or LP2(Ω, G). Computational results obtained in Chopra et al. [2] indicate that the directed Steiner cut inequalities (defining LP2(Ω, G)) alone often suffice to obtain an integer solution for the Steiner tree problem. On the other hand, the undirected Steiner cut inequalities were found to be very weak in general. In this study, we thus chose to use LP2(Ω, G), which can be solved in polynomial time and gives a tighter LP-relaxation than LP1(Ω, G).

Optimizing on LP2(Ω, G) is difficult in practice because of the constraints (2.5) (the size of the problems becomes very large for any cutting plane type approach). We relax these constraints in a Lagrangian fashion. The remaining problem breaks up into k independent problems, each of which is solved to optimality. The detailed procedure is described in the sequel.

We divide UGR into two distinct problems and reformulate. In the first stage, we consider the problem of finding the minimum excess capacity needed to obtain a legal routing. This can be formulated as follows:

$$u^* = \min\{u \mid (y, u) \in \text{LP2}(\Omega, G), y \text{ integer}\}. \tag{3.1}$$

If u^* is the optimal solution to (3.1), the problem of finding a minimum length routing as follows:

$$\begin{aligned} W^* = \text{mininimize} \quad & \sum_{a \in A} w_a \left(\sum_{i=1}^{k} y_{i,a} \right) \\ \text{subject to} \quad & y \text{ satisfies (2.4)}, \\ & \sum_{i=1}^{k} y_{i,a'} + \sum_{i=1}^{k} y_{i,a} \le c_e + u^*, \\ & y \ge 0, \quad y \text{ integer}. \end{aligned} \tag{3.2}$$

Problem (3.1) is known to be NP-hard even in the special case that G is a series–parallel graph (see Richey and Parker [17]). Problem (3.2) contains the Steiner tree problem as a special case and is thus hard (see Garey and Johnson [5]). However, we solved the LP-relaxation in each case. We found that (3.1) is more simple to solve in practice that either (2.6) or (3.2) and this allows us to obtain much better bounds for (3.2) than would arise from (2.6). This is because in (3.2) the objective function minimizes the length of the routing, while in (2.6) the objective function has two parts – one to minimize excess capacity and one to minimize length. Also solving (3.1) would be useful even if we are to use formulation (2.9).

3.1. SOLVING (3.1) TO OBTAIN MINIMUM EXCESS CAPACITY

Notice that the constraints (2.5) defining LP2(Ω, G) are linking constraints and their removal breaks the remaining problem into k independent problems which can be solved efficiently using branch and cut (see Chopra et al. [2]) or Lagrangian relaxation (see Beasley [1]). In a parallel implementation, each independent problem can be solved simultaneously. Solving (3.1) directly is difficult because of the size of the problems involved. A problem with with 1000 edges and 100 nets would result in 200,000 variables (on directing). Removal of the load constraints (2.5) results in k independent problems, which is much more manageable. When solving (3.1), we remove the constraints (2.5) using a Lagrangian relaxation. For details regarding Lagrangian relaxation, see Fisher [3]. Let $\lambda_e \geq 0$ be the Lagrangian multipliers for constraints (2.5). Corresponding to the variable u, the dual to the LP-relaxation of (3.1) contains the inequality $\sum_{e\in E}\lambda_e \leq 1$, with $\lambda_e \geq 0$. All other inequalities have a right side of zero. In case the optimal solution of u^* is strictly positive, by complementary slackness we have $\sum_{e\in E}\lambda_e = 1$. If $u^* = 0$, we can scale λ to impose this equality. While updating the multipliers λ, we thus impose this equality. A lower bound to u^* can be obtained by solving the following problem:

$$\text{minimize} \quad \sum_{a\in A}\mu_a\left(\sum_{i=1}^{k} y_{i,a}\right) - \sum_{e\in E}\lambda_e c_e$$

$$\text{subject to} \quad y \text{ satisfies (2.4)},\ y \geq 0,\ y \text{ integer}, \tag{3.3}$$

where $\mu_a = \mu_{a'} = \lambda_e$ for $e = [s, t]$, $a = (s, t)$, $a' = (t, s)$. The objective function for (3.3) eliminates u since $\sum_{e\in E}\lambda_e = 1$. Notice that (3.3) can be solved as k independent problems. Solving (3.3) is NP-hard. However, in practice, branch and cut using the constraints (2.4) has proved very effective in solving this problem (see Chopra et al. [2]). Each routing $\Gamma = \{T_i, i = 1, \ldots, k\}$ obtained as a solution to (3.3) gives an upper bound to u^* as $\max\{L(G, e) \mid e \in E\}$ and the objective function value gives a lower bound.

We use a subgradient procedure to maximize the lower bound obtained from solving (3.3). The multipliers are updated using subgradient steps, and for each set of multipliers we solve (3.3). We first give two improvement heuristics that significantly reduce the gap between the upper and lower bound.

REDUCING THE UPPER BOUND (THE PACKING HEURISTIC)

Given any set of multipliers $\bar{\lambda}$ (and corresponding $\bar{\mu}$), one can construct a "packed routing" $\Gamma_p(\bar{\lambda})$ as follows. We refer to $\Gamma_p(\bar{\lambda})$ simply as Γ_p.

Step 1. Set $i = 1$, $c_e^1 = c_e$ for all $e \in E$, $\Gamma_p = \emptyset$, $u^+ = 0$.

Step 2. Check if net i can be spanned using a Steiner tree T, where $c_e^i > 0$, $\forall e \in T$. This can be checked using breadth-first search from each node in N_i using edges e for which $c_e^i > 0$. If no such tree exists, set $c_e^i = c_e^i + 1$, $\forall e \in E$, $u^+ = u^+ + 1$. Now every edge has $c_e^i > 0$. Find the minimum length (with arc a getting a weight $\bar{\mu}_a$) Steiner tree T_i spanning net i. In our implementation, we use the method of Takahashi and Matsuyama [18] to obtain a heuristic solution T_i for net i. Set $\Gamma_p = \Gamma_p \cup \{T_i\}$.

Step 3. If $i = k$, stop. Else, set

$$c_e^{i+1} = \begin{cases} c_e^i & \text{for } e \notin T_i, \\ c_e^i - 1 & \text{for } e \in T_i. \end{cases}$$

$i = i + 1$. Go to step 2.

Γ_p gives a packed routing and u^+ the excess capacity required by this routing. Clearly, the order in which the nets are packed may affect the value of u^+. Korte et al. [12] suggest that ordering the nets in ascending order of $|N_i|$ is effective. One can also further try to improve the packing by trying to construct a Steiner tree for each net i using fewer edges with load $L(\Gamma_p, e) \geq u^+$ than the current tree. These procedures together constitute the "packing heuristic" which outputs the routing Γ_p with maximum load u^+.

IMPROVING THE LOWER BOUND (THE PARTITION HEURISTIC)

The lower bounding heuristic is based on the following observation. Let $\pi = (V_i, i = 1, 2, \ldots, r)$ be any partition of the node set V, where $|V_i| \geq 1$, $V_i \cap V_j = \emptyset$ for $i \neq j$ and $\cup_i V_i = V$. Let $E(\pi)$ be the set of edges with end nodes in two different subsets of π, i.e.

$$E(\pi) = \{e \in E \mid e = [s,t],\ \{s,t\} \not\subseteq V_i \ \text{ for } i = 1,2,\ldots,r\}.$$

The capacity of the partition is given by

$$c(\pi) = \sum_{e \in E(\pi)} c_e.$$

For each net i, let γ_i be the number of subsets of π containing at least one node from N_i, i.e.

$$\gamma_i = |\{j : |N_i \cap V_j| \geq 1, \ j \in \{1, 2, \ldots, r\}\}|.$$

Each Steiner tree spanning net i must use at least $\gamma_i - 1$ edges from $E(\pi)$. Define $g(\pi) = \sum_{i=1}^{k} \gamma_i - k$. The total capacity of all edges in $E(\pi)$ that is used up by any routing is at least $g(\pi)$. Define $u^- = \lceil (g(\pi) - c(\pi))/|E(\pi)| \rceil$. $\max\{0, u^-\}$ is clearly a lower bound for u^*.

To try to identify a suitable partition π, we use the best packed routing Γ_p, with maximum load u^+, provided by the packing heuristic. Given an integer $r \leq u^+$, define

$$E_r(\Gamma_p) = \{e \in E \mid L(\Gamma_p, e) \geq r\}.$$

The graph $G_r = (V, E - E_r)$ consists only of edges with load less than r. For $r \leq u^+ - 1$, G_r is not connected since Γ_p is a result of the packing heuristic. Thus, we have a partition $\pi_r = (V_i, i = 1, 2, \ldots, q)$ of the node set defined by G_r, where each subset V_i induces a connected component of G_r and there is no edge in $E - E_r$ with one end node in V_i and another in V_j for $i \neq j$. Each of the edges $E(\pi_r)$ is over capacity for the routing Γ_p. Thus, it is quite likely that $g(\pi_r) - c(\pi_r)$ has a large value. We further try to increase $g(\pi_r) - c(\pi_r)$ using the following procedure.

Let V_j and V_n be any two subsets in π_r. Consider the partition $\bar{\pi}_r$ formed by combining the subsets V_n and V_j into one, i.e. replace V_j and V_n by $V_j \cup V_n$. Define

$$E_{nj} = \{e \in E \mid e = [s, t], \ s \in V_n, \ t \in V_j\}$$

and

$$c_{nj} = \sum_{e \in E_{nj}} c_e.$$

Let g_{nj} be the number of nets with a terminal in each of V_j and V_n, i.e.

$$g_{nj} = |\{i : |V_n \cap N_i| \geq 1 \quad \text{and} \quad |V_j \cap N_i| \geq 1\}|.$$

It is easy to verify that $g(\bar{\pi}_r) = g(\pi_r) - g_{nj}$ and $c(\bar{\pi}_r) = c(\pi_r) - c_{nj}$. If $g_{nj} < c_{nj}$, then $g(\bar{\pi}_r) - c(\bar{\pi}_r) > g(\pi_r) - c(\pi_r)$. For each pair of subsets V_j and V_n in π_r, we find g_{nj} and c_{nj} and identify the two subsets into one if $g_{nj} < c_{nj}$. We continue this procedure iteratively until either $g_{nj} \geq c_{nj}$ for each pair of subsets in the current parition or there is only a single subset remaining. Let π^* be the resulting partition. Clearly, $g(\pi^*) - c(\pi^*) \geq g(\pi_r) - c(\pi_r)$. If π^* has only one subset, $u^- = 0$, else

$$u^- = \lceil (g(\pi^*) - c(\pi^*))/|E(\pi^*)| \rceil.$$

THE SUBGRADIENT PROCEDURE

Now we describe the subgradient procedure used to obtain lower and upper bounds for (3.3).

Step 1. Set initial values of $ub1 = k$, $ub2 = k$, $ub = k$, $lb1 = 0$, $lb = 0$ and Lagrange multipliers $\lambda_e = 1/|E|$ for all $e \in E$.

Step 2. Solve (3.3) with the current set of multipliers. Let $\overline{\Gamma}$ be the resulting routing, $\bar{z}$ the optimal value of the objective function, and $\bar{u} = \max\{L(\overline{\Gamma}, e) \mid e \in E\}$. Set $lb = \max\{lb, \lceil \bar{z} \rceil\}$, $lb1 = \max\{lb1, \bar{z}\}$.

Step 3. If $ub1 < \bar{u}$, go to step 4, else run the packing heuristic with multipliers λ to obtain routing $\overline{\Gamma}_{\mathrm{p}}$ and $\bar{u}^+ = \max\{L(\overline{\Gamma}_{\mathrm{p}}, e) \mid e \in E\}$. Run the partition heuristic on $\overline{\Gamma}_{\mathrm{p}}$ to obtain $\bar{u}^-$. Set $ub1 = \bar{u}$, $ub2 = \min\{ub2, \bar{u}^+, \lfloor \bar{u} \rfloor\}$, $lb = \max\{lb, \bar{u}^-\}$ and $ub = \min\{ub, ub2\}$. If $ub = lb$, stop.

Step 4. Given $e = [s, t]$, $a = (s, t)$, $a' = (t, s)$, calculate the subgradients

$$\delta_e = -c_e + \sum_{i=1}^{k} (\bar{x}_{i,a} + \bar{x}_{i,a'}),$$

where $\bar{x}$ is the incidence vector of routing $\overline{\Gamma}$. Adjust the subgradients as follows

$$\delta_e = 0 \quad \text{if} \quad \lambda_e = 0 \quad \text{and} \quad \delta_e < 0.$$

Stop if $\sum_{e \in E} \delta_e^2 = 0$.

Step 5. Define a step size

$$\eta = f(ub - \bar{z}) \Big/ \sum \delta_e^2.$$

f is initially set to 1 and reduced by a factor of 0.6 each time there are fifteen consecutive iterations with no improvement in $lb1$. Update the Lagrange multipliers by

$$\lambda_e = \max\{0, \lambda_e + \eta\delta_e\}, \quad \forall e \in E.$$

Rescale λ such that $\sum_{e \in E} \lambda = 1$.

Step 6. If $f < 0.005$ and $\eta < 0.005$, stop. Else, go to step 2.

The output of this procedure is the routing Γ_1 defining ub and a value for both the upper bound ub and lower bound lb for u^*.

3.2. OBTAINING THE MINIMUM LENGTH ROUTING

Here also, we relax constraints (2.5) in a Lagrangian fashion. Lower bounds to the optimal solution of (3.2) can be obtained by solving the following Lagrangian relaxation:

$$\begin{aligned} &\text{minimize} \quad \sum_{a \in A} \overline{w}_a \left(\sum_{i=1}^{k} y_{i,a} \right) - \sum_{e \in E} \lambda_e (c_e + u^*) \\ &\text{subject to} \quad y \text{ satisfies (2.4)}, \ y \geq 0, \ y \text{ integer}, \end{aligned} \tag{3.4}$$

where for $e = [s, t]$, $a = (s, t)$ or (t, s)

$$\overline{w}_a = w_a + \lambda_e. \tag{3.5}$$

Equation (3.4) once again breaks up into k independent Steiner tree problems. Solving (3.4) is NP-hard in general. However, in Chopra et al. [2], we observed that solving the corresponding LP-relaxation (using (2.4)) gives very good lower bounds. We run a packing heuristic similar to that in section 3.1. The only difference is that we construct the packed routing $\Gamma_p(\overline{w})$ instead of $\Gamma_p(\lambda)$ as described earlier (set $\mu_a = \mu_{a'} = \overline{w}_a$). Let $u^+ = \max\{L(\Gamma_p(\overline{w}), e) \mid e \in E\}$ and $W(\Gamma_p(\overline{w}))$ be the length of the routing $\Gamma_p(\overline{w})$, i.e.

$$W(\Gamma_p(\overline{w})) = \sum_{e \in E} w_e U(\Gamma_p(\overline{w}), e).$$

If $u^+ \leq u^*$, then $W(\Gamma_p(\overline{w}))$ provides an upper bound to W^*, the optimum for (3.2). The optimal solution to (3.4) gives a lower bound for W^*.

The subgradient procedure can now be described as follows:

Step 1. Set $ub = W(\Gamma_1)$, where Γ_1 is the routing defining u^* obtained as a solution to (3.3) and $lb = 0$. Initially, set $\lambda_e = 0$, $\forall e \in E$ and $u^+ = k$.

Step 2. Solve (3.4) with the current set of multipliers and $\overline{w}$ as defined in (3.5). Let $\overline{\Gamma}$ be the resulting routing, $\overline{z}$ the optimal value of the objective function and $\overline{u} = \max\{L(\overline{\Gamma}, e) \mid e \in E\}$. Set $lb = \max\{lb, \overline{z}\}$. If $\overline{u} \leq u^*$, set $ub = \min\{ub, W(\overline{\Gamma})\}$.

Step 3. If $\overline{u} \leq u^*$ or $u^+ < \overline{u}$, go to step 4, else set $u^+ = \min\{u^+, \overline{u}\}$ and run the packing heuristic (using arc weights μ_a) to obtain $\overline{\Gamma}_p$ with $\overline{u}_p = \max\{L(\overline{\Gamma}_p, e) \mid e \in E\}$. If $\overline{u}_p \leq u^*$, set $ub = \min\{ub, W(\overline{\Gamma}_p)\}$.

Step 4. Given $e = [s, t]$, $a = (s, t)$, $a' = (t, s)$, calculate the subgradients

$$\delta_e = -c_e + \sum_{i=1}^{k} (\overline{x}_{i,a} + \overline{x}_{i,a'}),$$

where $\overline{x}$ is the incidence vector of the routing $\overline{\Gamma}$. Adjust the subgradients as follows:

$$\delta_e = 0 \quad \text{if} \quad \lambda_e = 0 \quad \text{and} \quad \delta_e < 0.$$

Stop if $\sum_{e \in E} \delta_e^2 = 0$.

Step 5. Define a step size

$$\eta = f(ub - \bar{z}) / \sum \delta_e^2.$$

f is initially set to 1 and reduced by a factor of 0.6 each time there are fifteen consecutive iterations with no improvement in lb. Update the Lagrange multipliers by

$$\lambda_e = \max\{0, \lambda_e + \eta\delta_e\}, \quad \forall e \in E.$$

Step 6. If $f < 0.005$ and $\eta < 0.005$, stop; else, go to step 2.

The output of this procedure is a routing Γ^* defining ub and a value for the upper bound ub and lower bound lb for W^*.

4. Computational results

In this section, we present computational results for the solution procedure described in section 3. As described, we separate UGR into two distinct problems – one to find the minimum excess capacity u^* (formulated as (3.1)) and the other to find a minimum weight legal routing for excess capacity u^* on each edge (formulated as (3.2)).

We used problems on rectilinear grid graphs and randomly generated graphs to test our solution procedure. Rectilinear grid graphs most closely approximate the problem arising in practice.

Rectilinear grids of size 10×10 and 15×15 were used. For each net, first the number of nodes was randomly generated between 2 and 10 (using a uniform distribution). The nodes of the net were then randomly placed on grid points (using a uniform distribution for each of the x and y coordinates).

For the case where the underlying graph $G = (V, E)$ was randomly generated, we studied two graph densities $|E| = 3|V|, 5|V|$. The edge capacities were integers uniformly generated between 1 and 4. The edge lengths were integers uniformly generated between 1 and 10. The netlists were generated randomly as follows. For each net i, we first generated the size $|N_i|$ uniformly between 3 and 10. We then generated $|N_i|$ nodes from V to obtain the net i.

In these runs, we made the following change from the procedure described in section 3. Each subgradient step requires the solution of (3.3) (or (3.4)). Obtaining the exact solution at each step would have been very time consuming since we solve k Steiner tree problems for each subgradient step. So we use a heuristic at each step to solve the Steiner tree problems. The heuristic used is the one proposed by Takahashi and Matsuyama [18]. Since the heuristic does not give us a lower bound ((3.3) or (3.4) has to be solved exactly to obtain a Lagrangian lower bound), we multiply the heuristic solution by 0.90 and use this value as a temporary lower bound. Finally, we use the multipliers giving the highest temporary lower bound

Table 1(a)

Rectilinear grid graphs.

Problem	*size*	*k*	*nsize*	u_1^+	u_1	u_2^+	u_2	*iter*	*time*
grid101	10 × 10	25	6.56	5	2	4	3	195	13.0
grid102	10 × 10	25	6.52	5	2	4	3	205	13.7
grid103	10 × 10	25	6.48	5	2	4	3	190	13.8
grid104	10 × 10	50	6.28	9	3	7	6	185	30.5
grid105	10 × 10	50	6.24	8	3	7	6	193	32.5
grid106	10 × 10	50	5.86	9	3	7	5	211	44.8
grid107	10 × 10	75	5.52	12	4	10	8	183	31.0
grid108	10 × 10	75	6.43	13	4	11	9	215	41.4
grid109	10 × 10	75	6.11	12	4	10	9	195	37.9
grid1010	10 × 10	100	6.03	16	5	13	12	205	39.2
grid1011	10 × 10	100	6.19	16	5	13	12	172	30.7
grid1012	10 × 10	100	6.10	16	5	13	11	183	53.5
grid151	15 × 15	25	5.08	4	1	3	2	230	83.5
grid152	15 × 15	25	5.92	4	1	3	2	195	84.9
grid153	15 × 15	25	6.12	4	1	3	2	191	113.6
grid154	15 × 15	50	6.32	7	2	5	4	200	311.9
grid155	15 × 15	50	5.90	6	2	5	3	188	306.8
grid156	15 × 15	50	6.10	7	2	5	4	185	305.2
grid157	15 × 15	75	6.12	9	3	7	5	200	454.7
grid158	15 × 15	75	6.15	9	3	7	5	200	475.6
grid159	15 × 15	75	5.89	9	3	7	5	199	415.7
grid1510	15 × 15	100	5.96	12	3	9	7	201	761.2
grid1511	15 × 15	100	5.69	11	3	8	6	214	876.2
grid1512	15 × 15	100	5.60	11	3	8	6	201	826.2

and solve (3.3) and (3.4) exactly using branch and cut. The branch and cut procedure is described in Chopra et al. [2]. This gives us accurate lower bounds for (3.1) and (3.2), respectively.

The computational runs are on an IBM RS6000 (powerstation 520). All algorithms were coded in FORTRAN.

Tables 1(a) (grid graphs) and 1(b) (randomly generated graphs) contain the results of solving (3.1) for each problem. Here, we try to identify the minimum excess capacity needed to obtain a legal solution. The various headings for tables 1(a) and 1(b) are described below.

Table 1(b)

Randomly generated graphs.

Problem	$\|V\|$	$\|E\|$	k	*nsize*	u_1^+	u_1	u_2^+	u_1	*iter*	*time*
g100101	100	300	100	6.25	8	1	3	3	10	1.5
g100102	100	300	100	6.64	5	1	2	2	18	2.9
g100103	100	300	100	6.72	10	1	3	3	1	0.6
g100104	100	500	100	6.41	5	0	0	0	1	0.6
g100105	100	500	100	6.20	5	0	1	0	166	19.9
g100106	100	500	100	6.39	5	0	1	0	111	13.9
g100201	100	300	200	6.39	14	2	7	7	1	1.1
g100202	100	300	200	6.40	13	2	8	8	4	1.9
g100203	100	300	200	6.47	11	2	6	6	6	3.9
g100204	100	500	200	6.68	6	1	2	2	40	36.3
g100205	100	500	200	6.44	12	1	2	2	1	4.2
g100206	100	500	200	6.41	3	1	2	1	232	131.1
g100301	100	300	300	6.44	14	4	9	9	7	9.9
g100302	100	300	300	6.30	10	4	9	8	165	195.5
g100303	100	300	300	6.46	10	4	9	8	212	215.7
g100304	100	500	300	6.44	8	2	4	3	245	212.5
g100305	100	500	300	6.41	16	2	5	5	1	2.1
g100306	100	500	300	6.59	10	2	4	3	184	164.0
g200201	200	1000	200	6.34	6	0	1	1	1	3.8
g200202	200	1000	200	6.88	7	0	0	0	1	4.6
g200203	200	1000	200	6.42	5	0	0	0	1	3.7
g200204	200	600	200	6.68	4	1	3	2	382	570.0
g200205	200	600	200	6.45	12	1	3	3	1	5.9
g200206	200	600	200	6.30	12	1	5	5	1	3.6
g200401	200	1000	400	6.36	11	0	2	2	1	7.5
g200402	200	1000	400	6.57	14	0	1	1	1	13.1
g200403	200	1000	400	6.63	3	0	1	0	170	586.0
g200404	200	600	400	6.62	17	3	9	9	1	7.5
g200405	200	600	400	6.53	22	3	13	13	1	7.6
g200406	200	600	400	6.60	24	3	8	8	1	8.0

k = number of nets in netlist.
size = size of grids for grid graphs (table 1(a)).
nsize = average net size.
u_1^+ = best upper bound for excess capacity without running packing heuristic.
u_2^+ = best upper bound for excess capacity after running packing heuristic.
u_1 = best lower bound for excess capacity without running partition heuristic (this is obtained as $\lceil \bar{z} \rceil$, where $\bar{z}$ is the Lagrangian lower bound).
u_2 = best lower bound for excess capacity after running partition heuristic.
iter = number of subgradient iterations.
time = total time taken (including input and output) in minutes.

As can be seen from tables 1(a) and 1(b), the packing and partition heuristics prove to be quite effective. In each problem, the packing heuristic lowers the upper bound. The partition heuristic raises the lower bound in 47 of the 54 problems attempted. In the rectilinear grid graphs, the gap between the upper and lower bound was reduced to 1 for 13 of the 24 problems attempted, and 2 for the rest. For the randomly generated problems, 21 of the 30 problems resulted in the optimal solution for the excess capacity. In the remaining instance, the gap was reduced to 1.

The results in tables 1(a) and 1(b) do not fully indicate the benefit of using subgradient optimization to solve the LP-relaxation since most of the gap is closed using the packing and partitioning heuristic. To judge the effect, we also ran the same set of problems using only the packing and partitioning heuristics. The results

Table 2

Problem	GAP1	GAP2
grid101	2	1
grid102	1	1
grid103	1	1
grid104	2	1
grid105	3	2
grid106	3	1
grid107	2	2
grid108	3	2
grid109	3	1
grid1010	3	1
grid1011	3	1
grid1012	3	2

are given in table 2. GAP1 indicates the difference between the upper and lower bounds when only the heuristics were run. GAP2 indicates the difference when the

Table 3(a)

Grid graphs.

Problem	*lb*1	*lb*	*ub*	% imp	% gap	*iter*	*time*
grid101	350	683	820	70.8	20.0	184	10.6
grid102	644	792	1030	38.3	30.0	204	7.9
grid103	295	512	600	71.1	17.2	196	6.3
grid104	1143	1420	1810	41.5	27.5	192	14.3
grid105	839	1290	1530	65.3	18.6	165	10.8
grid106	1023	1360	1460	77.1	7.4	154	10.0
grid107	1558	1770	2610	20.1	47.4	152	11.3
grid108	1868	2010	3050	12.0	51.7	154	15.9
grid109	1402	2312	2620	74.7	13.3	205	21.3
grid1010	1817	2818	3710	52.9	31.7	193	27.2
grid1011	1810	2512	3260	48.4	29.7	255	34.9
grid1012	1812	2740	3240	64.8	18.2	197	28.1
grid151	567	743	770	86.7	3.6	164	16.4
grid152	837	968	1030	67.9	6.4	185	23.3
grid153	644	748	1050	25.6	40.4	152	21.6
grid154	1301	1587	2170	32.9	36.7	184	50.6
grid155	1246	1432	1650	46.0	15.2	152	35.8
grid156	1643	1972	2800	28.4	41.9	152	41.8
grid157	2677	2983	4060	22.1	36.1	152	65.2
grid158	2340	2743	3880	26.2	41.4	206	84.4
grid159	2490	2818	3560	30.6	26.3	152	51.2
grid1510	3220	3612	4760	25.5	31.8	152	78.2
grid1511	3021	3432	4860	22.3	41.6	165	82.9
grid1512	3100	3592	4860	27.9	35.3	180	86.9

heuristics were run together with subgradient optimization. In 9 out of 12 instances, we found that using the subgradient optimization and heuristics resulted in a smaller gap than using the heuristics alone.

Tables 3(a) (grid graphs) and 3(b) (randomly generated graphs) contain the results of solving (3.2) for each problem. For the instances that (3.1) has been solved to optimality, the excess capacity u^* is known with precision. In other instances, we use the best known upper bound as u^*. This guarantees us a feasible solution to (3.2). One trivial lower bound for (3.2) can be obtained by ignoring the capacity constraints and simply finding a minimum length Steiner tree for each net to be assigned routing. The length of such a routing is clearly a lower bound for

Table 3(b)

Randomly generated graphs

Problem	*lb*1	*lb*	*ub*	% imp	% gap	*iter*	*time*
g100101	3446	4110	4468	64.9	8.7	309	45.5
g100102	3533	4547	5088	65.2	11.9	307	45.5
g100103	3546	4386	4731	70.9	7.9	301	45.5
g100104	2365	3485	4070	65.7	16.8	267	48.5
g100105	2289	2991	3289	70.2	10.0	292	48.0
g100106	2374	3012	3364	64.6	11.6	277	45.6
g100201	6889	8520	9168	71.5	7.6	265	72.2
g100202	6747	8116	8708	69.8	7.3	273	69.9
g100203	6909	8911	9959	65.6	11.8	272	77.4
g100204	5080	7430	8650	65.8	16.4	304	112.1
g100205	4787	7066	8073	69.3	14.2	347	115.8
g100206	4854	7153	8364	65.6	16.9	313	111.8
g100301	10222	13747	16350	57.5	18.9	285	116.7
g100302	9990	13382	14933	68.6	11.6	333	131.3
g100303	10311	13870	16664	56.0	20.1	321	130.8
g100304	7232	10627	12398	65.7	16.7	335	168.5
g100305	7280	9944	11088	70.0	11.5	361	181.6
g100306	7462	11118	15187	47.3	36.6	354	191.2
g200201	3680	4219	4621	57.3	9.5	247	251.9
g200202	6114	8133	8978	70.5	10.4	342	352.2
g200203	5817	7453	8287	66.2	11.2	308	294.7
g200204	8551	10639	12206	57.1	14.7	279	265.9
g200205	8213	10245	11479	62.2	12.0	358	321.9
g200206	8124	9008	9740	54.7	8.1	259	222.8
g200401	11530	15865	17733	69.9	11.8	308	721.5
g200402	11755	17357	20295	65.6	16.9	304	715.7
g200403	12013	17574	20007	69.6	13.8	329	730.3
g200404	17047	20788	22547	68.0	8.5	369	662.4
g200405	16708	18414	19560	59.8	6.2	323	627.8
g200406	16686	20663	23182	61.2	12.2	348	641.8

the length of the optimal routing. The length of this routing is listed as *lb*1 in table 3. A comparison of the gap between the upper bound and the best lower bound and the upper bound and *lb*1 gives a measure of the effectiveness of our lower bounding procedure. The various headings for tables 3(a) and 3(b) are as follows:

*lb*1	= length of routing with minimum length Steiner trees for each net, ignoring capacity constraints.
lb	= best lower bound obtained from Lagrangian relaxation.
ub	= best upper bound obtained.
% imp = $100(lb - lb1)/(ub - lb1)$	= percent improvement in gap by subgradient procedure.
% gap = $100(ub - lb)/lb$	= percent gap remaining.
iter	= number of subgradient iterations.
time	= time in minutes.

In the problems considered so far, all nets were of size between three and ten. For comparison, we solved six problems where all nets were of size two. The edge capacities were uniformly generated to be either 1 or 2. The Steiner tree problem in this case reduces to a shortest path problem which can be solved in polynomial time. The results are contained in tables 4 and 5. The headings for tables 4 and 5

Table 4

Problem	$\|V\|$	$\|E\|$	k	*nsize*	u_1^+	u_1	u_2^+	u_2	*iter*	*time*
g100107	100	300	100	2	5	0	1	1	1	0.1
g100108	100	300	100	2	4	0	3	3	1	0.1
g100109	100	300	100	2	3	0	1	0	145	3.2
g100110	100	500	100	2	2	0	0	0	1	0.1
g100111	100	500	100	2	3	0	0	0	1	0.1
g100112	100	500	100	2	3	0	0	0	1	0.1

Table 5

Problem	*lb*1	*lb*	*ub*	% imp	% gap	*iter*	*time*
g100107	1024	1189	1213	87.3	2.0	441	12.4
g100108	1026	1067	1068	97.6	0.1	342	8.3
g100109	997	1142	1156	91.2	1.2	448	12.2
g100110	710	874	919	78.4	5.1	449	16.1
g100111	712	881	892	93.9	1.2	484	17.3
g100112	718	889	918	85.5	3.3	361	13.0

are as for tables 1 and 3, respectively. The gap between the upper and lower bounds was fairly small in this case (the largest gap is about 5 percent).

5. Conclusions and further research

The object of this study was to use integer programming foundations to obtain good lower and upper bounds for UGR. The results from section 4 seem encouraging in this regard, particularly when finding the minimum excess capacity required to obtain a legal solution. When searching for a minimum length solution, we would like to point out that the simple lower bound obtained by ignoring capacity constraints (listed under $lb1$ in tables 3 and 5) would have been a poor lower bound. The subgradient procedure used by us significanly raises this lower bound. If the LP-relaxation itself could be solved exactly, we would obtain an even better lower bound. However, using cutting planes to solve even the smallest problems considered by us (9000 variables) would have been very time comsuming. Thus, the subgradient procedure seems reasonable. Good lower bound solutions are also more likely to provide good upper bound solutions upon application of heuristics such as the packing heuristic. This was the case in our study where the packing heuristic, when applied to the solution giving $lb1$, rarely resulted in a feasible solution and gave poor (or no) upper bounds. The packing heuristic when applied to the solutions giving the lower bound lb, on the other hand, results in fairly good upper bounds. The packing heuristic we use is very simple. It may be worthwhile to try more sophisticated heuristics.

We have not considered the multi-commodity flow formulation of section 2.5 in our computational experiments. However, this may be worth pursuing if a good dual ascent procedure can be devised. The comparison of course would be between the dual ascent procedure and the Lagrangian relaxation we have considered.

Given the encouraging results, we feel there are two avenues that need to be explored further. The major advantage of relaxing the capacity constraints in a Lagrangian fashion is that the remaining problem is the solution of k independent Steiner tree problems. To exploit this structure, we plan to test an implementation of our procedure on a parallel machine. This would allow us to solve exactly the Steiner tree problems at each iteration in a reasonable amount of time. We feel this should further allow us to lower the gap.

Another avenue to be pursued further is formulation (2.9). Given its strength in providing lower bounds, we plan to use a column generation scheme to bring in Steiner tree variables as needed in solving the LP-relaxation of (2.9).

Finally, we feel that improvement can be made in the partition and packing heuristic. In the problems that we were unable to find the optimal excess capacity, we were able to close the gap to within two. In each of these cases, the number of edges with load higher than u_2 was very small (varied from 0.5% to 2% of the edges). We feel that the upper bound was tight in these instances, but the lower bound could not be raised any further by the partition heuristic. An improved heuristic would be useful in obtaining optimal solutions.

Acknowledgement

The author thanks Edgar Gorres for help with the computational results.

References

[1] J.E. Beasley, An SST-based algorithm for the Steiner Tree problem in graphs, Networks 19(1989) 1–16.
[2] S. Chopra, E. Gorres and M.R. Rao, Solving the Steiner Tree problem on a graph using branch and cut, ORSA J. Comp. 4(1992)320–335.
[3] M.L. Fisher, The Lagrangian relaxation method for solving integer programming problems, Manag. Sci. 27(1981)1–18.
[4] L.R. Ford, Jr. and D.R. Fulkerson, *Flows in Networks* (Princeton University Press, Princeton, NJ, 1962).
[5] M. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (Freeman, San Francisco, 1979).
[6] M. Grötschel, L. Lovasz and A. Schrijver, The ellipsoid method and its consequences in combinatorial optimization, Combinatorica 1(1981)169–191.
[7] M. Grötschel, A. Martin and R. Weismantel, Packing Steiner trees: Polyhedral investigations, Konrad-Zuse-Zentrum für Informationstechnik Berlin, Preprint SC 92-8 (1992).
[8] M. Grötschel, A. Martin and R. Weismantel, Packing Steiner tress: A cutting plane algorithm and computational results, Konrad-Zuse-Zentrum für Informationstechnik Berlin, Preprint SC 92-9 (1992).
[9] C.Y. Lee, An algorithm for path connection and its applications, IRE Trans. Electron. Comp. EC-10(1961)346–365.
[10] T. Lengauer, *Combinatorial Algorithms for Integrated Circuit Layout* (Wiley, 1990).
[11] T. Lengauer and M. Lügering, Integer program formulations of global routing and placement problems, Research Report, University of Paderborn, Germany.
[12] B. Korte, H.J. Prömel and A. Steger, Steiner trees in VLSI layout, in: *Paths, Flows, and VLSI-Layout*, ed. B. Korte, L. Lovasz, H.J. Prömel and A. Schrijver (Springer, 1990).
[13] A. Martin, Packen von Steinerbäumen: Polyredrische Studien und Anwendung, Ph.D. Thesis, Technische Universität Berlin (1992).
[14] E.F. Moore, Shortest path through a maze, in: *Proc. Int. Symp. on Switching Circuits* (Harvard University Press, 1959) pp. 285–292.
[15] A.P.C. Ng, P. Raghavan and C.D. Thompson, Experimental results for a linear program global router, Comp. Art. Int. 6(1987)229–242.
[16] P. Raghavan and C.D. Thompson, Randomized rounding: A technique for provably good algorithms and algorithmic proofs, Combinatorica 7(1987)365–374.
[17] M.B. Richey and R.G. Parker, On multiple Steiner subgraph problems, Networks 16(1986)423–438.
[18] H. Takahashi and A. Matsuyama, An approximate solution for the Steiner problem in graphs, Math. Japonica 24(1980)573–577.
[19] R.T. Wong, A dual ascent approach for Steiner Tree problems on a directed graph, Math. Progr. 28(1984)271–287.