Chapter 2

Reductions

A particular instance of the Steiner tree problem often can be reduced to a smaller one by examining local properties of the network G. Reductions fall into two main categories:

- identification of edges or non-terminals that do not belong to at least one Steiner minimal tree.
- identification of edges or non-terminals that belong to at least one Steiner minimal tree.

It is assumed that the reductions are applied one at a time. When an edge (v_i, v_j) belonging to a Steiner minimal tree is identified, G is reduced to a smaller network by the contraction along (v_i, v_j). The vertex to which (v_i, v_j) is contracted is then regarded as a terminal. Deletions of edges and non-terminals are straightforward.

Reductions decrease the cardinality of the vertex and edge sets. They are important since the performance of exact algorithms and heuristics for the Steiner tree problem is closely related to these parameters. Furthermore, heuristics may produce better solutions in the reduced networks. Reductions can also be applied at intermediate stages of any branch-and-bound algorithm.

Computational experiments indicate that a large number of problem instances can be solved by reductions alone. The remaining problem instances are reduced substantially (on average to one fourth of the original sizes).

Tests identifying edges or non-terminals not belonging to at least one Steiner minimal tree are discussed in Section 2.1. Inclusion tests for edges or non-terminals are covered in Section 2.2. The important issue of how to integrate exclusion and inclusion tests is addressed in Section 2.3. Effectiveness of reductions is discussed in Section 2.4.

Reductions of instances of the Steiner arborescence problem in directed networks are of similar nature as the reductions described below. The reader is referred to Voss [12] for a review.

2.1 Exclusion Tests

This section discusses various tests that can be used to identify edges and non-terminals not belonging to at least one Steiner minimal tree. Some very simple tests are given first. More complicated tests described next in fact generalize the simpler ones. The reason for including simpler tests is partly historical and partly because they often (but not always) are faster to check than their generalizations.

2.1.1 Non-Terminals of Degree 1 (NTD1)

Suppose that a network G contains a non-terminal v_k of degree 1. Let (v_k, v_i) be the edge incident with v_k. This edge cannot be in any Steiner minimal tree. If it did, its deletion would result in two components, one containing v_k alone, the other containing all terminals. Since $c_{v_k v_i} > 0$, the component spanning all terminals has a smaller total length than any Steiner minimal tree, a contradiction. Hence v_k and its incident edge (v_k, v_i) can be removed from G.

2.1.2 Non-Terminals of Degree 2 (NTD2)

Suppose that a network G contains a non-terminal v_k of degree 2. Let (v_i, v_k) and (v_k, v_j) denote the two edges incident with v_k. If $c_{v_i v_k} + c_{v_k v_j} \geq c_{v_i v_j}$, then there is at least one Steiner minimal tree not containing v_k. Hence v_k (and its two incident edges) can be deleted from G. If $c_{v_i v_k} + c_{v_k v_j} < c_{v_i v_j}$, then the edge (v_i, v_j) cannot belong to any Steiner minimal tree and can be deleted from G. Furthermore, the edges (v_i, v_k) and (v_k, v_j) can be replaced by a new edge (v_i, v_j) of length $c_{v_i v_k} + c_{v_k v_j}$, and the non-terminal v_k can be deleted from G (Fig. 2.1).

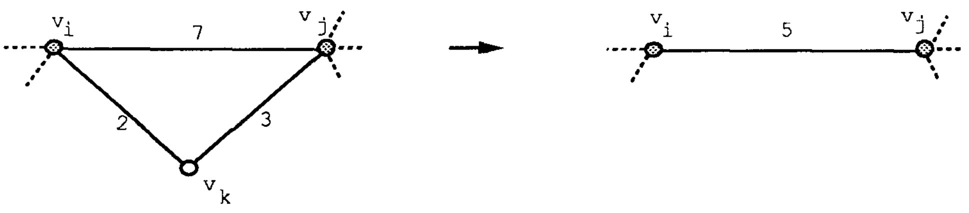

Figure 2.1: Removal of a non-terminal of degree 2

Suppose that the added edge (v_i, v_j) is eventually found to belong to a Steiner minimal tree for the reduced problem instance. In order to obtain a Steiner minimal tree for the original problem instance, the edge (v_i, v_j) has to be replaced by the original two edges (v_i, v_k) and (v_k, v_j) whereby the non-terminal v_k is reintroduced.

2.1.3 Non-Terminals of Higher Degree (NTDk)

Let v_i and v_j denote two vertices in a network G. Any path P from v_i to v_j breaks down into one or more *elementary paths* between v_i, successive terminals and v_j. A *Steiner distance* between v_i and v_j along P is the length of the longest elementary path in P. The *bottleneck Steiner distance* $b_{v_iv_j}$ between v_i and v_j is the minimum Steiner distance between v_i and v_j taken over all paths from v_i to v_j in G. Furthermore, the *restricted bottleneck Steiner distance* $\bar{b}_{v_iv_j}$ between v_i and v_j is the bottleneck Steiner distance between v_i and v_j in $G-(v_i,v_j)$.

Every path between v_i and v_j in G contains at least one elementary path of length at least $b_{v_iv_j}$. Any path between v_i and v_j in G containing an elementary subpath of length $b_{v_iv_j}$ is called a *bottleneck Steiner path* between v_i and v_j. It will be shown in Subsection 2.3.1 that bottleneck Steiner distances between all pairs of vertices in G can be determined in $O(v^2n)$ time.

Let Γ_{v_k} denote the vertices of G adjacent to a non-terminal v_k. Let B_{v_k} denote a complete network with Γ_{v_k} as its vertex set (Fig. 2.2). The length of an edge (v_i,v_j) in B_{v_k} is defined as the bottleneck Steiner distance $b_{v_iv_j}$ between v_i and v_j in G. Let B'_{v_k} denote a subnetwork of B_{v_k} induced by a subset Γ'_{v_k} of Γ_{v_k}. Finally, let T'_{v_k} denote a minimum spanning tree for Γ'_{v_k} in B'_{v_k}, and let C'_{v_k} be the set of edges in G connecting v_k with the vertices in Γ'_{v_k}.

Lemma 2.1 ([6]) *If*

$$|T'_{v_k}| \leq |C'_{v_k}|$$

for every subset Γ'_{v_k} of Γ_{v_k}, $|\Gamma'_{v_k}| \geq 3$, then the non-terminal v_k has degree at most 2 in at least one Steiner minimal tree.

Proof: If $deg_G(v_k) \leq 2$, then the lemma clearly holds. It will be therefore in the following assumed that $deg_G(v_k) > 2$.

Suppose that v_k has degree at least 3 in every Steiner minimal tree for N in G. Consider a Steiner minimal tree $T_G(N)$ for N in G where v_k has lowest possible degree l, $l \geq 3$. Let Γ'_{v_k} denote the set of vertices adjacent to v_k in $T_G(N)$. Remove v_k from $T_G(N)$. It then breaks into l subtrees $T_{v_1}, T_{v_2}, ..., T_{v_l}$, each containing one of the vertices previously adjacent to v_k.

Pick a shortest edge (v_i,v_j) in T'_{v_k}. Consider a bottleneck Steiner path $P_{v_iv_j}$ in G (in Fig. 2.2 $v_i = v_1$, $v_j = v_3$ and $P_{v_1v_3}$ is indicated by the shaded region). Traverse $P_{v_iv_j}$ from v_i toward v_j. Whenever a path $P_{v'_pv'_q}$ between vertices in two different trees T_{v_p} and T_{v_q} is encountered, T_{v_p} and T_{v_q} are interconnected by it. This interconnecting path $P_{v'_pv'_q}$ is a subpath of one of the elementary paths in $P_{v_iv_j}$. Consequently, $|P_{v'_pv'_q}| \leq b_{v_iv_j}$.

The traversal of $P_{v_iv_j}$ can interconnect several subtrees in $T_{v_1}, T_{v_2}, ..., T_{v_l}$ (e.g., $P_{v_1v_3}$ in Fig. 2.2 goes through T_{v_2}). If there is more than one subtree left when v_j is reached during the traversal of $P_{v_iv_j}$, the second shortest edge of T'_{v_k} and the corresponding bottleneck Steiner path in G are processed in the same manner. This continues until all subtrees $T_{v_1}, T_{v_2}, ..., T_{v_l}$ become interconnected.

Exactly $l-1$ subpaths must be added to $T_G(N)-v_k$ before it becomes connected. Their total length L, is at most $|T'_{v_k}|$. Since $T_G(N)$ is a Steiner minimal tree, L

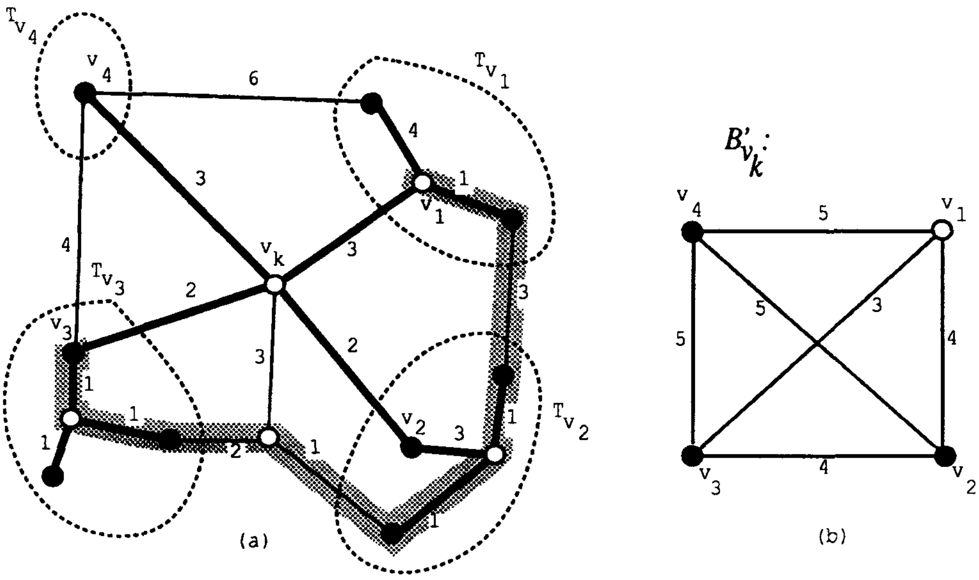

Figure 2.2: G, $T_G(N)$ and B'_{v_k}

must be at least $|C'_{v_k}|$. Hence, $L \leq |T'_{v_k}| \leq |C'_{v_k}| \leq L$. The tree obtained from $T_G(N) - v_k$ by adding the edges on the $l-1$ subpaths yields another Steiner minimal tree $T'_G(N)$.

Suppose that v_k has degree l in $T'_G(N)$. The length of l paths from v_k to $T_{v_1}, T_{v_2}, ..., T_{v_l}$ in $T'_G(N)$ is $|C'_{v_k}|$. Furthermore, $|C'_{v_k}| < L \leq |T'_{v_k}|$, a contradiction. Hence, $T'_G(N)$ is a Steiner minimal tree where v_k has degree less than l. This is in contradiction with the manner in which $T_G(N)$ was chosen. □

If the conditions of Lemma 2.1 are satisfied, then the non-terminal v_k appears in at least one Steiner minimal tree with degree at most 2. The original instance of the Steiner tree problem can be reduced in the following way. The non-terminal v_k is removed together with its all incident edges. The edges (v_i, v_j), $v_i, v_j \in \Gamma_{v_k}$, of length $c_{v_i v_k} + c_{v_k v_j}$ are added to G. If parallel edges arise as a result, only shorter ones are retained (Fig. 2.3). Once the smaller instance of the Steiner tree problem is solved, new edges appearing in the solution are replaced by the corresponding pairs of edges.

It can be easily verified that the NTDk-test is a generalization of the NTD1- and NTD2-tests.

The number of minimum spanning trees constructed in connection with the NTDk-test grows exponentially with $deg_G(v_k)$. Consequently, this test should be applied to non-terminals of low degree (typically not more than 4).

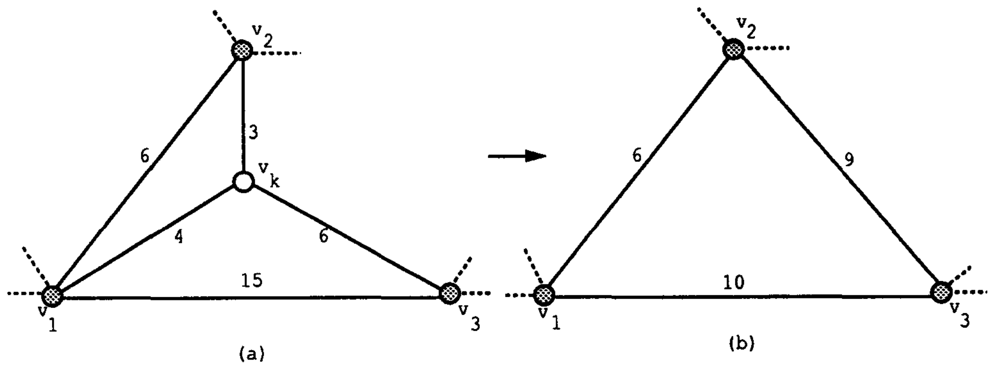

Figure 2.3: Removal of a non-terminal of degree 3

2.1.4 Long Edges (LE)

This test permits a deletion of long edges. The simple proof is omitted as the LE-test will be subsequently generalized.

Lemma 2.2 ([2]) *Any edge* (v_i, v_j) *with*

$$c_{v_i v_j} > d_{v_i v_j}$$

can be removed from G.

An application of the LE-test is shown in Fig. 2.4a where the edge indicated by the broken line segment can be removed. The LE-test can be generalized to cover the case when $c_{v_i v_j} = d_{v_i v_j}$ provided that there is a shortest path from v_i to v_j other than the shortest path consisting of the edge (v_i, v_j) alone.

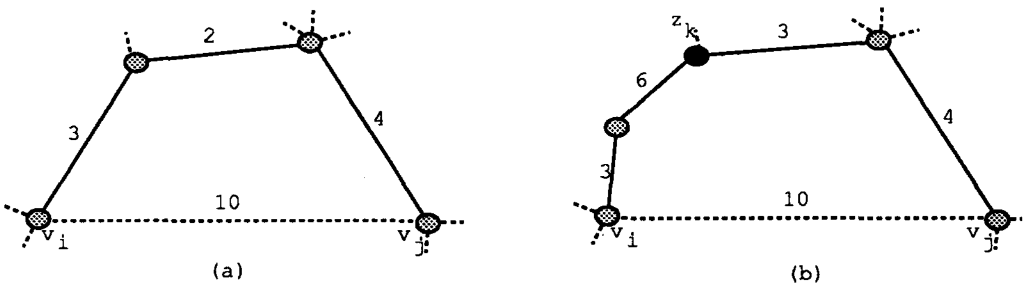

Figure 2.4: Removal of long edges

2.1.5 Paths with One Terminal (PT1)

This test is closely related to the LE-test. However, unlike the LE-test, it uses some information about the location of terminals in the network. Again, the simple proof is omitted as the PT1-test will be subsequently generalized.

Lemma 2.3 ([6]) *If G contains an edge (v_i, v_j) and there is a terminal z_k such that*

$$c_{v_i v_j} > \max\{d_{v_i z_k}, d_{v_j z_k}\}$$

then (v_i, v_j) can be removed from G.

An application of the PT1-test is shown in Fig. 2.4b where the edge indicated by the broken line segment can be removed. Contrary to the LE-test, the PT1-test can be successful in networks satisfying the triangle inequality and in Euclidean networks in particular.

2.1.6 Minimum Spanning Trees (MST)

This test was originally given by Balakrishnan and Patel [1] in terms of three related tests. Only a unified version is presented here. Assume that G is a complete network. This is not a significant restriction as any instance of the Steiner tree problem can be transformed into an equivalent problem instance on a complete network by adding edges of "infinite" length (Subsection 1.4.2).

Lemma 2.4 ([7]) *Any edge (v_i, v_j) not in a minimum spanning tree for the subnetwork of G induced by $N \cup \{v_i, v_j\}$ can be removed from G.*

The proof is omitted as a more general test will be given below. An application of the MST-test to all edges in G is shown in Fig. 2.5. Not shown edges have "infinite" length. Edges that can be deleted by the MST-test are indicated by broken line segments.

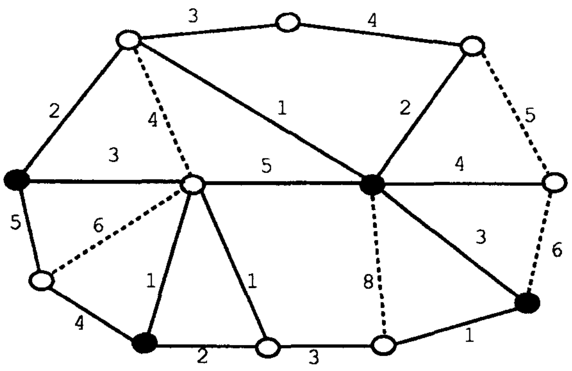

Figure 2.5: Removal of long edges based on the MST-test

2.1.7 Paths with Many Terminals (PTm)

This test is a generalization of both the LE- and PT1-tests. It can also be shown to be a generalization of the MST-test [7].

Lemma 2.5 ([6]) *Any edge (v_i, v_j) with*

$$c_{v_iv_j} > b_{v_iv_j}$$

can be removed from G.

Proof: Suppose that a Steiner minimal tree $T_G(N)$ contains an edge (v_i, v_j) with $c_{v_iv_j} > b_{v_iv_j}$. $T_G(N) - (v_i, v_j)$ consists of two components: C_{v_i} containing v_i and C_{v_j} containing v_j. Consider a bottleneck Steiner path $P_{v_iv_j}$ in G. Let P be the shortest subpath on $P_{v_iv_j}$ which has one end-vertex in C_{v_i}, and the other end-vertex in C_{v_j}. P always exists and is completely within one of the elementary paths of $P_{v_iv_j}$. Hence $|P| \leq b_{v_iv_j} < c_{v_iv_j}$. Reconnecting C_{v_i} and C_{v_j} by P yields a tree spanning all terminals and being shorter than $T_G(N)$, a contradiction. □

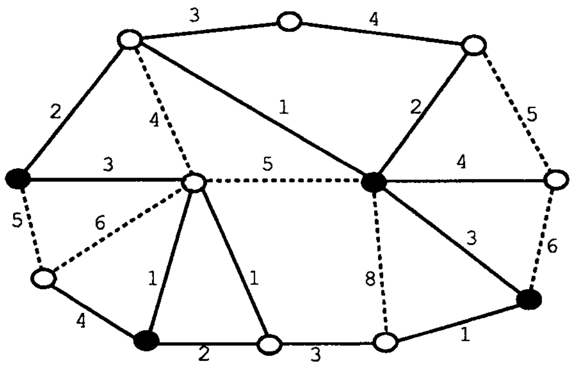

Figure 2.6: Removal of long edges based on the PTm-test

An application of the PTm-test to all edges of G is shown in Fig. 2.6. Edges that can be deleted by the PTm-test are indicated by broken line segments.

The PTm-test is also applicable when $c_{v_iv_j} = b_{v_iv_j} = \bar{b}_{v_iv_j}$. However, complications can arise if this extended PTm-test is applied more than once without the recalculation of the restricted bottleneck Steiner distances. The restricted bottleneck Steiner distances $\bar{b}_{z_1z_2}, \bar{b}_{z_1z_3}, \bar{b}_{z_2z_3}$ in Fig. 2.7 are all equal to 4. Any of the edges (z_1, z_2), (z_1, z_3), (z_2, z_3) can be deleted from G. Suppose that (z_1, z_3) is deleted. This causes both $\bar{b}_{z_1z_2}$ and $\bar{b}_{z_2z_3}$ to increase from 4 to 5 (assuming that all not shown edges of G have length at least 5). Consequently, the removal of for example (z_1, z_2) is now not allowed.

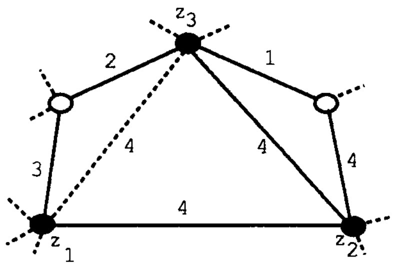

Figure 2.7: Improper removal of two edges

2.1.8 Reachability (R)

Suppose that a tree $T_G^*(N)$ spanning the terminals is available. $T_G^*(N)$ can for example be obtained by one of the heuristics for the Steiner tree problem discussed in Chapter 4. Let v_l denote any non-terminal of G not in $T_G^*(N)$. Let z_i and z_j denote terminals closest and second-closest to v_l in G. Let z_k denote a terminal farthest away from v_l in G.

Lemma 2.6 **([2,6])** *If*

$$d_{v_l z_i} + d_{v_l z_j} + d_{v_l z_k} \geq |T_G^*(N)|$$

then there exists at least one Steiner minimal tree where v_l has degree at most 2.

Proof: Assume first that the non-terminal v_l has degree at least 3 in some Steiner minimal tree $T_G(N)$. Let z'_k denote a terminal farthest away from v_l in $T_G(N)$. Since $T_G(N)$ is a subnetwork of G, z'_k is at least $d_{v_l z_k}$ away from v_l. Let z'_i denote a terminal closest to v_l in $T_G(N)$ and such that it can be reached from v_l by a path edge-disjoint from the path to z'_k. The length of this path from v_l to z'_i is at least $d_{v_l z_i}$. Let z'_j denote a terminal closest to v_l in $T_G(N)$ and such that it can be reached from v_l by a path edge-disjoint from the paths to z'_k and z'_i. The length of this path from v_l to z'_j is at least $d_{v_l z_j}$. The three edge-disjoint paths in $T_G(N)$ have therefore total length at least $d_{v_l z_i} + d_{v_l z_j} + d_{v_l z_k}$. Hence,

$$|T_G(N)| \geq d_{v_l z_i} + d_{v_l z_j} + d_{v_l z_k} \geq |T_G^*(N)|$$

implying that $T_G^*(N)$ is a Steiner minimal tree not containing v_l. □

If the condition of Lemma 2.6 is fulfilled, then v_l can be deleted from G in the way already described in Subsection 2.1.3. The addition of new edges between vertices adjacent to v_l can be completely avoided if

$$d_{v_l z_i} + d_{v_l z_k} \geq |T_G^*(N)|$$

To see this, suppose that there is a Steiner minimal tree $T_G(N)$ where v_l has degree 2. Let z'_k and z'_i be as defined above. The total length of paths from v_l to z'_k and z'_i

is at least $d_{v_l z_i} + d_{v_l z_k}$. Consequently, $|T_G(N)| \geq d_{v_l z_i} + d_{v_l z_k} \geq |T^*_G(N)|$, implying that $T^*_G(N)$ is a Steiner minimal tree not containing v_l.

$T^*_G(N)$ in Fig. 2.8 is indicated by thick line segments. In this particular example, all non-terminals not in $T^*_G(N)$ disappear due to the R-test. Consider for example the non-terminal v_5. Then $z_i = z_3$, $z_j = z_4$ and $z_k = z_1$. Furthermore $d_{v_5 z_3} = 2$, $d_{v_5 z_4} = 5$ and $d_{v_5 z_1} = 9$ while $|T^*_G(N)| = 10$. Hence v_5 can be deleted from G. For the non-terminal v_6, one has $z_i = z_2$, $z_j = z_4$ and $z_k = z_3$. Furthermore $d_{v_6 z_2} = 2$, $d_{v_6 z_4} = 4$ and $d_{v_6 z_3} = 6$. Hence v_6 can be deleted from G. However, in this case edges (z_2, v_7) and (v_7, v_8) of length 5 and 4 respectively must be added to G. There is no need of adding the edge (z_2, v_8) of length 3 since G already contains an edge between these two vertices of length 1.

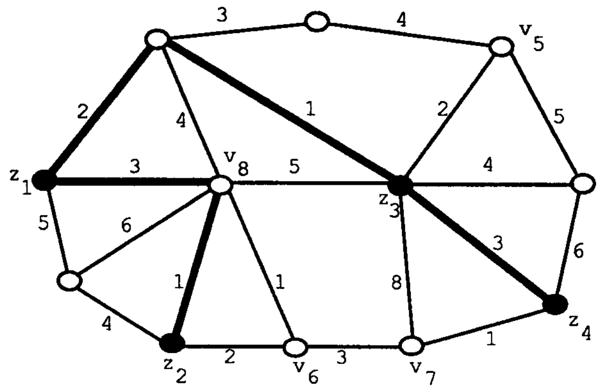

Figure 2.8: Removal of non-terminals based on the R-test

2.1.9 Cut Reachability (CR)

Let $T^*_G(N)$ and v_l be as in Subsection 2.1.8. Let $\bar{c}_{z_i}$ denote the length of the shortest edge in G incident with a terminal z_i. Select z_i and z_j such that $d_{v_l z_i} - \bar{c}_{z_i}$ is smallest possible, and $d_{v_l z_j} - \bar{c}_{z_j}$ is second smallest. Let $N_{ij} = N \setminus \{z_i, z_j\}$.

Lemma 2.7 ([9,6]) *If*

$$d_{v_l z_i} + d_{v_l z_j} + \sum_{z_k \in N_{ij}} \bar{c}_{z_k} \geq |T^*_G(N)| \tag{2.1}$$

then there exists a Steiner minimal tree not containing v_l.

Proof: Suppose that there is a Steiner minimal tree $T_G(N)$ containing v_l. At least one edge enters each terminal. There are at least two edge-disjoint paths leaving v_l toward two distinct terminals z'_i and z'_j. Furthermore, z'_i and z'_j can be chosen such that there is no terminal on the paths from v_l to z'_i and z'_j, respectively. Hence,

$$|T_G(N)| \geq d_{v_l z'_i} - \bar{c}_{z'_i} + d_{v_l z'_j} - \bar{c}_{z'_j} + \sum_{z_k \in N} \bar{c}_{z_k} \geq d_{v_l z_i} + d_{v_l z_j} + \sum_{z_k \in N_{ij}} \bar{c}_{z_k}$$

Consequently, $T^*_G(N)$ is a Steiner minimal tree not containing v_l. □

There is also an edge version of this test. Let (v_i, v_j) denote an edge of G not in $T^*_G(N)$. Select $z_i \neq v_j$ and $z_j \neq v_i$ such that $d_{v_i z_i} - \bar{c}_{z_i} + d_{v_j z_j} - \bar{c}_{z_j}$ is smallest possible.

Lemma 2.8 ([9,6]) *If*

$$d_{v_i z_i} + c_{v_i v_j} + d_{v_j z_j} + \sum_{z_k \in N_{ij}} \bar{c}_{z_k} \geq |T^*_G(N)|$$

then there is a Steiner minimal tree not containing the edge (v_i, v_j).

The proof, analogous to the proof of Lemma 2.7, is omitted. An application of the vertex version of the CR-tests is shown in Fig. 2.9 where $\bar{c}_{z_1} = 2$ while $\bar{c}_{z_2} = \bar{c}_{z_3} = \bar{c}_{z_4} = 1$. Non-terminals and their incident edges indicated by broken line segments can be removed from G. For instance, consider the non-terminal v_5: $d_{v_5 z_1} = 5$, $d_{v_5 z_2} = 8$, $d_{v_5 z_3} = 4$, $d_{v_5 z_4} = 7$. Furthermore, $d_{v_5 z_1} - \bar{c}_{z_1} = 3$, $d_{v_5 z_2} - \bar{c}_{z_2} = 7$, $d_{v_5 z_3} - \bar{c}_{z_3} = 3$, $d_{v_5 z_4} - \bar{c}_{z_4} = 6$. Let $z_i = z_1$ and $z_j = z_3$. Then the left side of (2.1) is equal to 11 while $|T^*_G(N)| = 10$. It can be easily verified that the vertex version of the CR-test may fail for a non-terminal v_l while the edge version is applicable to one of the edges incident with v_l. In particular, the edge (v_7, z_3) would be removed by the application of the edge version of the CR-test.

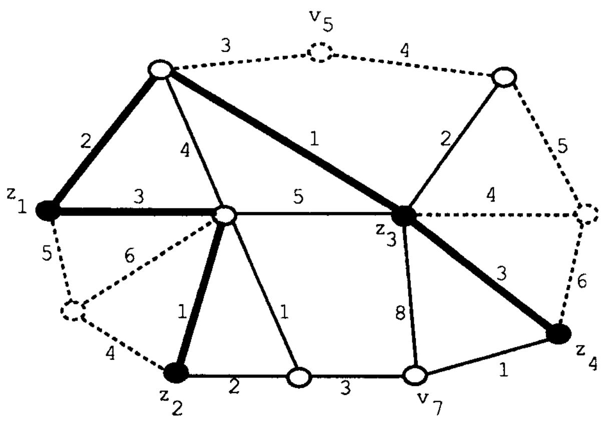

Figure 2.9: Removal of terminals based on the CR-test

2.2 Inclusion Tests

In this section tests that identify edges and non-terminals belonging to at least one Steiner minimal tree are discussed. Again, for similar reasons as when describing the exclusion tests, simple tests, which are subsequently generalized, are discussed first.

2.2.1 Terminals of Degree 1 (TD1)

Suppose that G contains a terminal z_i with $deg_G(z_i) = 1$. The unique edge (z_i, v_j) belongs to every Steiner minimal tree. G can be reduced by contracting it along (z_i, v_j).

2.2.2 Short Terminal-to-Terminal Edges (STTE)

If the nearest vertex adjacent to a terminal z_i also is a terminal, denoted by z_j, then the edge (z_i, z_j) must belong to at least one Steiner minimal tree. Note in particular that (z_i, z_j) belongs to a minimal spanning tree. In fact, even a more general result is valid. The straightforward proof is omitted as the STTE-test will be generalized.

Lemma 2.9 **([1])** *Any edge (z_i, z_j) of a minimum spanning tree belongs to at least one Steiner minimal tree.*

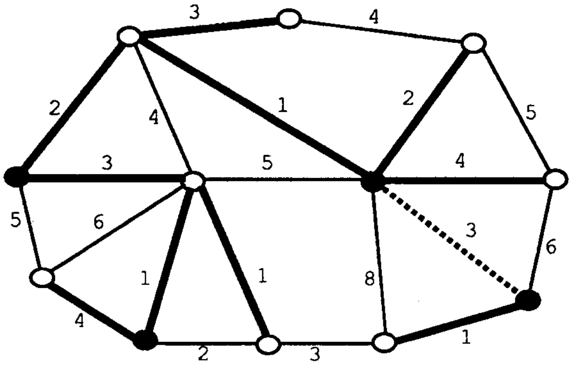

Figure 2.10: Inclusion of an edge based on the STTE-test

An application of the STTE-test is shown in Fig. 2.10. Edges indicated by heavy or broken line segments belong to the minimum spanning tree $T_G(V)$. The edge indicated by the broken line segment is the only edge identified by the STTE-test as belonging to at least one Steiner minimal tree.

The STTE-test leads to the following logical constraints [1]. Suppose that there is an edge (v_i, z_k) in a minimum spanning tree $T_G(V)$ between a terminal z_k and a non-terminal v_i. As soon as it is known that $v_i \in T_G(N)$, the edge (v_i, z_k) must belong to $T_G(N)$ as well. Unfortunately, this information about v_i is not available before $T_G(N)$ has been determined. But consider any integer programming formulation of the Steiner tree problem with x_{ij} denoting a binary variable with value 1 if $(v_i, v_j) \in T_G(N)$ and 0 otherwise. Then the above interdependence can be represented by the constraints

$$x_{ik} \geq x_{ij} \text{ for all } (v_i, z_k) \in T_G(V) \text{ and for all } (v_i, v_j) \in G$$

Although these inequalities are redundant in every integer programming formulation of the Steiner tree problem, they strengthen the linear programming and Lagrangean relaxations, and consequently lead to better lower bounds needed in connection with branch-and-bound algorithms for the Steiner tree problem (Chapter 3).

2.2.3 Nearest Vertex (NV)

This test identifies additional edges of a minimum spanning tree $T_G(V)$ as belonging to at least one Steiner minimal tree $T_G(N)$. The proof is omitted since the test will be generalized. Let z_k denote a terminal in G. Let (z_k, v_i) and (z_k, v_j) denote respectively the shortest and second shortest edge incident with z_k.

Lemma 2.10 ([2]) *If G contains a terminal z_l, $z_l \neq z_k$, satisfying*

$$c_{z_k v_j} \geq c_{z_k v_i} + d_{v_i z_l}$$

then the edge (z_k, v_i) belongs to at least one Steiner minimal tree.

An application of the NV-test is shown in Fig. 2.11 where $c_{z_k v_j} = 8$ while $c_{z_k v_i} + d_{v_i z_l} \leq 6$. Hence (z_k, v_i) belongs to every Steiner minimal tree.

If $deg_G(z_k) = 1$, then let $c_{z_k v_j} = \infty$ and conditions of Lemma 2.10 are fulfilled. The NV-test is therefore a generalization of the TD1-test (Subsection 2.2.1). If $v_i \in N$, then $z_l = v_i$ (i.e., $d_{v_i z_l} = 0$) and conditions of Lemma 2.10 are always satisfied. Hence, the NV-test is also a generalization of the STTE-test (Subsection 2.2.2).

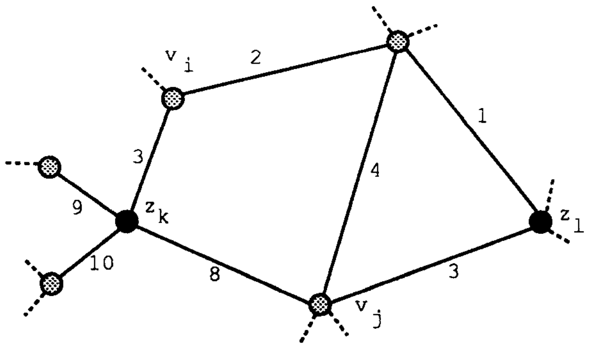

Figure 2.11: Inclusion of an edge based on the NV-test

2.2.4 Short Edges (SE)

The SE-test described in this subsection is a generalization of the TD1-, STTE- and NV-tests. Let (v_i, v_j) be an edge in a minimum spanning tree $T_G(V)$. Let $\bar{r}_{v_i v_j}$ denote the length of the shortest edge among longest edges taken over all paths from v_i to v_j in $G - (v_i, v_j)$. $\bar{r}_{v_iv_j}$ is referred to as the *restricted bottleneck edge length* between v_i and v_j. Note that if $N = V$, then $\bar{r}_{v_iv_j} = \bar{b}_{v_iv_j}$.

Lemma 2.11 ([6]) *If there is a pair of terminals z_k and z_l such that at least one shortest path from z_k to z_l goes through the edge (v_i, v_j) and*

$$\bar{r}_{v_iv_j} \geq d_{z_kz_l}$$

then the edge (v_i, v_j) belongs to at least one Steiner minimal tree.

Proof: Suppose that the edge (v_i, v_j) is not in any Steiner minimal tree for N in G. $T_G(V) - (v_i, v_j)$ consists of two components C_i and C_j such that $v_i \in C_i$ and $v_j \in C_j$. Since one of the shortest paths $P_{z_kz_l}$ contains the edge (v_i, v_j) and $\bar{r}_{v_iv_j} \geq d_{z_kz_l}$, the path $P_{z_kz_l}$ crosses the cut $\{C_i, C_j\}$ exactly once (at the edge (v_i, v_j)). Hence $z_k \in C_i$ and $z_l \in C_j$ or vice versa.

Consider the path from z_k to z_l in the Steiner minimal tree $T_G(N)$. It must contain an edge (v_p, v_q) of length at least $\bar{r}_{v_iv_j}$. $T_G(N) - (v_p, v_q)$ consist of two components, one containing z_k and the other containing z_l. $T_G(N) \cup P_{z_kz_l} - (v_p, v_q)$ is a connected network that spans N, contains (v_i, v_j), and has total length not greater than $|T_G(N)|$, a contradiction. $\square$

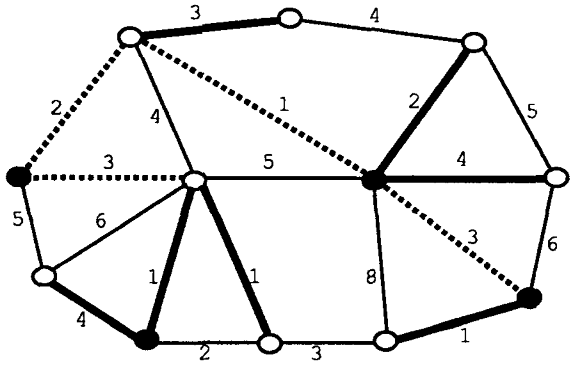

Figure 2.12: Inclusion of edges based on the SE-tests

An application of the SE-test is shown in Fig. 2.12. Edges indicated by heavy and broken line segments belong to the minimum spanning tree $T_G(V)$. Broken line segments indicate the edges that can be contracted.

The SE-test generalizes the NV-test: assume that v_i, v_j, z_k, z_l satisfy the conditions of Lemma 2.10. At least one shortest path from z_k to z_l is forced to go through the edge $(z_k, v_i) \in T_G(V)$. Furthermore, $\bar{r}_{z_kv_i} \geq c_{z_kv_j} \geq c_{z_kv_i} + d_{v_iz_l} \geq d_{z_kz_l}$.

To carry out the SE-test efficiently, one needs to restrict the number of candidates for z_k and z_l for a given edge $(v_i, v_j) \in T_G(V)$. In fact, it suffices to check the conditions of Lemma 2.11 for only one pair of terminals. If $d_{z_kv_i} < d_{z_kv_j}$ for all $z_k \in N$ (or $d_{z_k v_j} < d_{z_k v_i}$ for all $z_k \in N$), then no shortest path between any pair of terminals goes through (v_i, v_j). Hence conditions of Lemma 2.11 are fulfilled only if there is a terminal z_k closer to v_i than to v_j, and a terminal z_l closer to v_j than to v_i. Suppose that one of the shortest paths between z_k and z_l goes through (v_i, v_j) and $\bar{r}_{v_i v_j} \geq d_{z_k z_l}$. Suppose furthermore that there is a terminal z_p closer to v_i than to v_j and such that $d_{z_k v_i} > d_{z_p v_i}$. It follows immediately that $\bar{r}_{v_i v_j} \geq d_{z_k z_l} > d_{z_p z_l}$. Hence any shortest path between z_p and z_l goes through (v_i, v_j) unless $z_p \in C_j$ where C_j is the component of $T_G(V) - (v_i, v_j)$ containing v_j. Suppose that $z_p \in C_j$. The shortest path from z_p to v_i is not allowed to go through v_j; this would imply that z_p is closer to v_j than to v_i. But the shortest path from z_p to v_i has to cross the cut $\{C_j, C_i\}$, implying that $d_{z_p v_i} \geq \bar{r}_{v_i v_j} \geq d_{z_k z_l} > d_{z_k v_i}$, a contradiction. It follows that z_k can be chosen from among terminals closer to v_i than to v_j such that $d_{z_k v_i}$ is minimized. By a similar argument, z_l can be chosen from among terminals closer to v_j than to v_i such that $d_{v_j z_l}$ is minimized.

An application of the SE-test to edges of a minimum spanning tree $T_G(V)$ only is not restrictive. The conditions of Lemma 2.11 cannot be fulfilled if $(v_i, v_j) \notin T_G(V)$. To see this, suppose that (v_i, v_j) is an edge not in any minimum spanning tree $T_G(V)$, and there is a pair of terminals z_k and z_l such that the shortest path from z_k to z_l goes through (v_i, v_j). Then, $\bar{r}_{v_i v_j} < c_{v_i v_j} \leq d_{z_k z_l}$.

Finally, logical constraints discussed in connection with the STTE-test can be generalized to all edges in a minimum spanning tree $T_G(V)$.

2.2.5 Length Transformations (LT)

Reductions can sometimes become applicable after a suitable modification of edge lengths. One such method was suggested by Duin and Volgenant [6]. Let v_k be a non-terminal. Let (v_k, v_i) and (v_k, v_j) denote two edges incident with v_k. Let G' be a network obtained from G by changing the lengths of (v_k, v_i) and (v_k, v_j) to

$$c'_{v_k v_i} = c_{v_k v_i} + \alpha \text{ and } c'_{v_k v_j} = c_{v_k v_j} - \alpha$$

where $\alpha > 0$. A simple proof of the following lemma is omitted.

Lemma 2.12 ([6]) *If the edges (v_k, v_i) and (v_k, v_j) belong to Steiner minimal trees for $N \cup v_k$ in both G and G', then Steiner minimal trees for N in G and in G' have the same length.*

Suppose that a minimum spanning tree $T_G(V)$ contains two edges (v_k, z_i) and (v_k, z_j), where z_i and z_j are terminals and v_k is a non-terminal. Suppose furthermore that the SE-test does not apply while it does apply to both (v_k, z_i) and (v_k, z_j) when the set of terminals is extended by v_k. Note that (v_k, z_i) and (v_k, z_j) are shortest paths from v_k to z_i and z_j respectively. The length of (v_k, z_i) can be reduced by some amount α, $0 < \alpha \leq c_{v_k z_i}$. This will not influence the applicability of the SE-test for (v_k, z_i) with respect to $N \cup v_k$: $\bar{r}_{v_k z_i}$ remains unchanged while (v_k, z_i) remains the shortest path (of smaller length). Also $\bar{r}_{v_k z_j}$ remains unchanged: every path from v_k to z_j over the edge (v_k, z_i) contains another edge of length at least $c_{v_k z_i}$ (otherwise (v_k, z_i) would not belong to $T_G(V)$). In addition, (v_k, z_j) remains the shortest path from v_k to z_j.

Let α in addition be chosen such that $\alpha < \bar{r}_{v_k z_j} - c_{v_k z_j}$. The increase of the length of (v_k, z_j) by α will not influence the applicability of the SE-test for (v_k, z_j) with respect to $N \cup v_k$. This is due to the fact that $\bar{r}_{v_k z_j}$ remains unchanged. The length of (v_k, z_j), even if it increases, will be strictly smaller than $\bar{r}_{v_k z_j}$. Hence, (v_k, z_j) remains the shortest path from v_k to z_j. Finally, $\bar{r}_{v_k z_i}$ can only increase as a result of increasing the length of (v_k, z_j) and (v_k, z_i) remains the shortest path from v_k to z_i.

Thus, if α is chosen as large as possible subject to $\alpha \leq c_{v_k z_i}$ and $\alpha < \bar{r}_{v_k z_j} - c_{v_k z_j}$, the decrease of the lenght of (v_k, z_i) can make the SE-test applicable to both (v_k, z_i) and (v_k, z_j) in G'. Any tree containing both these edges will have the same length in both G' and G. Hence, in view of Lemma 2.12, a Steiner minimal tree for N in G' will be a Steiner minimal tree for N in G. This is for example the case in Fig. 2.13 [6]. Before the transformation, the shortest path from z_1 to z_3 goes through (z_1, v_5). But $\bar{r}_{z_1 v_5} = 70$ while $d_{z_1 z_3} = 80$. So the SE-test for (z_1, v_5) based on terminals z_1 and z_3 will fail. If v_5 is regarded as a terminal, the shortest path from z_1 to v_5 goes through (z_1, v_5). Furthermore, $\bar{r}_{z_1 v_5} = 70$ and $d_{z_1 v_5} = 30$. Consequently, the SE-test is successful for (v_5, z_1). Similarly, the shortest path from v_5 to z_2 goes through (v_5, z_2). Furthermore, $\bar{r}_{v_5 z_2} = 70$ and $d_{v_5 z_2} = 50$. Consequently, the SE-test is successful for (v_5, z_2). Next, $\min\{c_{v_5 z_1}, \bar{r}_{v_5 z_2} - c_{v_5 z_2}\} = \{30, 70 - 50\} = 20$. If the length of (z_1, v_5) is reduced by less than 20 (e.g., by 19) and the length of (v_5, z_2) is increased by the same amount, then the SE-test remains successful for both (z_1, v_5) and (v_5, z_2). The search for a Steiner minimal tree can now continue in G'. The shortest path from z_1 to z_3 goes again through (v_5, z_1). Furthermore, $\bar{r}_{z_1 v_5} = 70$ while $d_{z_1 z_3} = 61$. Hence, the edge (z_1, v_5) belongs to at least one Steiner minimal tree for N in G'. The vertex v_5 can now be regarded as a terminal. This implies that (v_5, z_2) also belongs to a Steiner minimal tree for N in G'. The length of this Steiner minimal tree is the same in both G and G'. From Lemma 2.12 follows that there is a Steiner minimal tree for N in G' containing both (v_5, z_1) and (v_5, z_2).

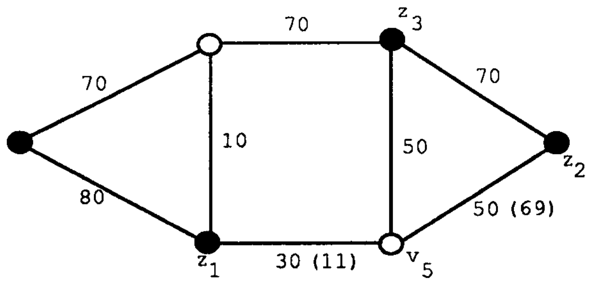

Figure 2.13: Length transformation enabling additional reductions

2.3 Integration of Tests

The tests described in this chapter vary from very simple tests of limited applicability to very general tests. Fig. 2.14 summarizes the interdependencies between various tests. Tests higher up in the hierarchies generalize all tests below.

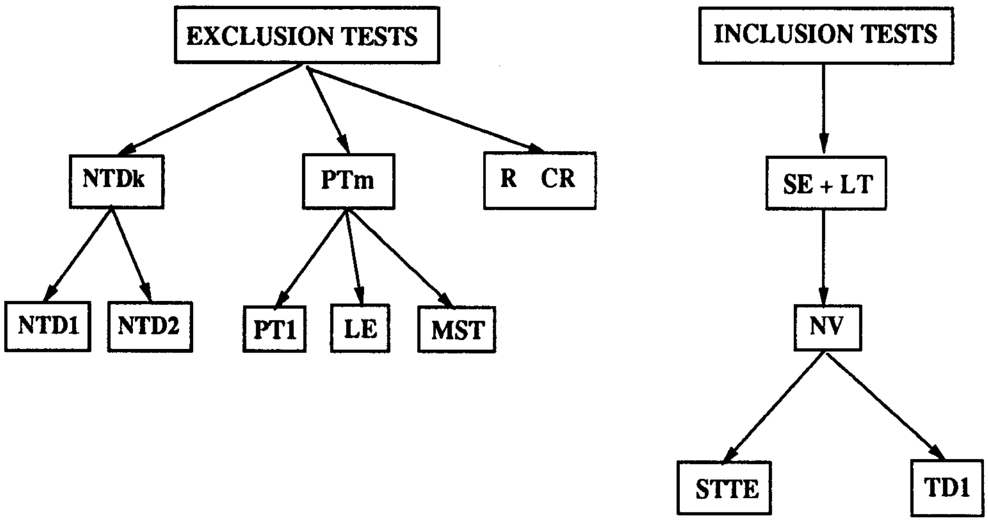

Figure 2.14: Test hierarchies

One reduction may trigger other reductions based on the same or different tests. It is therefore very important to establish the order in which the tests are to be applied. Furthermore, the reduction process should be terminated if the computational effort per test becomes too large. Another very important issue is what modifications of distance and bottleneck matrices are needed after each reduction type.

2.3.1 Data and Algorithms

When the applicability of some particular test is to be established, various types of information are required. Table 2.1 indicates what is required for each non-dominated test type. Rows correspond to the following information: shortest distances between vertices (represented by an $v \times v$ matrix D), bottleneck Steiner distances (represented by an $v \times v$ matrix B), a minimum spanning tree $T_G(V)$, and a suboptimal tree $T_G^*(N)$ spanning all terminals.

Some of the entries in D and B rows are marked with downward or upward arrows. This means that a particular test remains valid even if entries are respectively lower and upper bounds of the exact values. The issue of inexact information will be addressed in Subsection 2.3.3.

	NTDk	PTm	R	CR	SE+LT
D			$\bullet\downarrow$	$\bullet\downarrow$	$\bullet\uparrow$
B	$\bullet\uparrow$	$\bullet\uparrow$			
$T_G(V)$					$\bullet$
$T_G^*(N)$			$\bullet$	$\bullet$	

Table 2.1: Data requirements

In order to simplify the discussion below, assume that D and B are represented by $v \times v$ matrices. $T_G(V)$ and $T_G^*(N)$ are represented by edge lists. It should however be emphasized that more elaborate data structures such as forward stars, heaps and disjoint sets representation [10] could be employed to speed-up the reduction process.

The distance matrix D containing lengths of shortest paths between all pairs of vertices in V can be determined in $O(v^3)$ by for example Dijkstra's algorithm [4] (applied to each vertex as a source). More sophisticated algorithms are also available [10].

The bottleneck Steiner distance matrix B for G can be determined in $O(v^2 n)$ by a simple modification of Dijkstra's algorithm (applied to the distance network $D_G(V)$ rather than to G). More specifically, the following algorithm finds bottleneck Steiner distances from a vertex v_i to all other vertices in G.

- **STEP 1:** $b_{v_i v_j} := d_{v_i v_j}$ for all $v_j \in V$. $L := \{v_i\}$.
- **STEP 2:** If $N \subseteq L$ then Stop.
- **STEP 3:** Let $z \in N \setminus L$ with $b_{v_i z}$ smallest possible. $L := L \cup z$.
- **STEP 4:** $b_{v_i v_j} := \min\{b_{v_i v_j}, \max\{b_{v_i z}, d_{z v_j}\}\}$ for every $v_j \in V \setminus L$.
  If v_j is a non-terminal and $b_{v_i v_j} \leq b_{v_i z}$, then $L := L \cup v_j$.
  Go to **STEP 2**.

A minimum spanning tree $T_G(V)$ can be determined in $O(v^2)$ time using for example Prim's algorithm which is closely related to the Dijkstra's algorithm for shortest paths [10]. Alternatively, more sophisticated algorithms using Fibonacci heaps [8] or leftist heaps [10] could be used.

Several good polynomial heuristics for the Steiner tree problem in networks are described in Chapter 4. In particular, the shortest paths heuristic (Subsection 4.1.1) that requires $O(n(e + v \log v))$ time is well-suited for the reduction purposes as it basically requires the distance matrix D. Another more straightforward possibility is to use the simple heuristic which deletes (one at a time) non-terminals of degree 1 from the minimum spanning tree $T_G(V)$ (Subsection 4.2.1). However, this heuristic usually finds suboptimal solutions that are worse than those found by the shortest paths heuristic.

Some tests need additional information which can be easily derived from C (edge length matrix), D, and $T_G(V)$. More specifically, the NTDk-test needs to identify non-terminals of low degree. This information is readily available from C. In addition, the NTDk-test requires the determination of minimum spanning trees for small complete networks with the bottleneck Steiner distances as edge lengths. The R- and CR-tests need to know the closest, second-closest, and farthest terminals from each non-terminal. In addition, the CR-test needs to know the closest vertex to each terminal. All this information can be obtained in $O(v)$ time from D. The SE-test requires the restricted bottleneck edge length $\bar{r}_{v_iv_j}$ between vertices v_i and v_j adjacent in the minimum spanning tree $T_G(V)$. It is equal to the length of the shortest edge in $G-(v_i,v_j)$ crossing the cut $\{C_i,C_j\}$ obtained by deleting the edge (v_i,v_j) from $T_G(V)$. It can be found in $O(e)$ time.

2.3.2 Updating

When edges and non-terminals are deleted or edges are contracted, it is not always necessary to determine D, B, $T_G(V)$ and $T_G^*(N)$ from the scratch. For instance, when an edge (v_i,v_j) is deleted from G and it does not belong to $T_G(V)$, the minimum spanning tree for the smaller network remains unchanged. If (v_i,v_j) is deleted but belongs to $T_G(V)$, new minimum spanning tree can be determined in $O(e)$ time by reconnecting the two components of $T_G(V)-(v_i,v_j)$. Similar simplifications can be applied when edges are contracted or non-terminals are deleted. Such simplifications are also possible in connection with suboptimal trees spanning the terminals. The determination of shortest paths when edges are deleted or contracted can also be speed-up by the use of distance information in the original network [5]. These methods can also be adapted in connection with the determination of new bottleneck Steiner distances.

2.3.3 Use of Inaccurate Information

The computational effort needed to obtain and update the information necessary to verify the applicability of reductions can be substantial. It can be advantageous to use inaccurate information even if it can fail to recognize some reductions. This is in particular the case during the initial phases of the reduction process when the network is relatively large. Furthermore, rather than updating the information after each reduction, original information can sometimes be used even if it becomes inaccurate.

A simple upper bound for $b_{v_iv_j}$ is $c_{v_iv_j}$. A slightly better upper bound is obtained as follows.

$$b^+_{v_iv_j} = \min\{c_{v_iv_j}, \min_{v_k\notin N}\{c_{v_iv_k}+c_{v_kv_j}\}, \min_{z_l\in N}\{\max\{c_{v_iz_l}, c_{z_lv_j}\}\}\}$$

In other words, $b^+_{v_iv_j}$ is the bottleneck Steiner distance with respect to paths from v_i to v_j with at most two edges.

Table 2.2 shows what happens to D, B, $T_G(V)$, and $T_G^*(N)$ when NTDk-, PTm-, R-, CR- and SE-tests are satisfied and the corresponding reductions are applied.

When applying the NTDk-, PTm-, and SE-tests, the bottleneck Steiner distances between the remaining vertices are either unchanged or become smaller [6]. Hence, the original bottleneck Steiner distances are upper bounds for the actual

	NTDk	PTm	R+CR	SE+LT
D	unchanged	increased	increased	decreased
B	unchanged	unchanged	increased	decreased
T	can change	unchanged	can change	unchanged
T*	can change	unchanged	unchanged	unchanged

Table 2.2: Changes caused by reductions

bottleneck Steiner distances. Updating B can be postponed for as long as desired when applying the NTDk-, PTm-, and SE-tests. However, updating should be done from time to time in order to catch the NTDk- and PTm-tests missed by the use of upper bounds.

PTm-tests do not use shortest distances (although their application can cause shortest distances between the remaining vertices to increase). Hence, there is no need to update D until no more PTm-tests can be applied.

NTDk-tests do not use shortest distances. When they apply, shortest distances between the remaining vertices remain unchanged.

SE-tests use shortest distances but when they apply, an edge is contracted. Consequently, the original shortest distances are upper bounds in the reduced network. Updating D can be postponed for as long as desired when applying the SE-tests. However, this can cause some SE-tests to be missed.

When a non-terminal or an edge is deleted by the R- or CR-tests, shortest distances between the remaining vertices may increase. Consequently, the original distances are lower bounds. Updating D can be postponed for as long as desired when applying the R- and CR-tests. Again, this can cause some R- and CR-tests to be missed.

The edges deleted by the PTm-tests cannot belong to any minimum spanning tree. Hence, $T_G(V)$ remains unchanged when such edges disappear. The edges contracted by the SE-test belong to $T_G(V)$. Hence, a minimum spanning tree for the contracted network can be obtained by contracting $T_G(V)$ along the same edge.

Non-terminals or edges deleted by the R- or CR-tests do not belong to the suboptimal tree $T_G^*(N)$. Hence, there is no need to recompute $T_G^*(N)$ as long as these tests are applicable. If $T_G^*(N)$ is determined by deleting degree 1 non-terminals from $T_G(V)$, then $T_G^*(N)$ is unchanged when PDm- and SE-tests are applied.

2.3.4 Ordering of Tests

The general ordering of tests is shown in Fig. 2.15. More detailed orderings (involving in particular NTD3- and PTm-tests with B^+ rather than B were suggested in [6,12]). For the sake of clarity, it is assumed that the distance matrix D, the bottleneck Steiner distance matrix B, the minimum spanning tree $T_G(V)$ and the suboptimal Steiner tree $T_G^*(N)$ are determined before the reduction process begins, and they are updated after each reduction. But as discussed in Subsection 2.3.3, these updates could be delayed. All tests are applied as many times as possible before continuing with next test type. Upward-going arrows are followed only if at least one reduction was carried out since previous upward step.

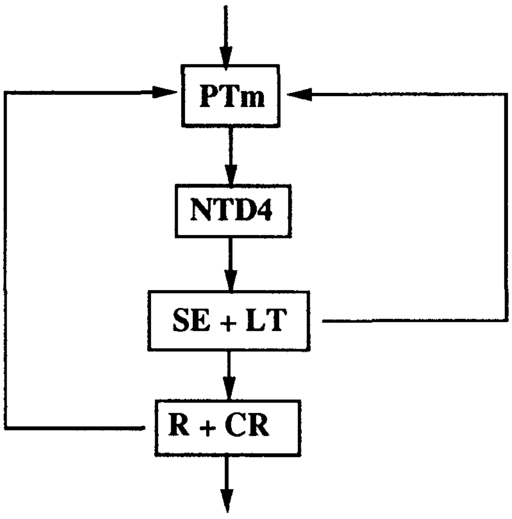

Figure 2.15: Ordering of reductions

The ordering of the tests is not accidental. The PTm-tests should in general be performed before the NTDk-tests since the application of the PTm-tests reduces the degrees of vertices. Since both the PTm- and NTDk-tests delete edges (and possibly replace pairs of edges by a single edge of length equal to the sum of lengths of the pair), these tests can only improve the performance of the SE-tests.

The R- and CR-tests are of rather limited applicability and should only be applied when all other tests fail. The same applies to the LT-transformations which can sometimes trigger additional reductions.

2.4 Effectiveness of Reductions

Although it is not difficult to find networks for which none (or few) of the tests apply (networks satisfying triangle inequality are usually of this type), there are on the other hand many networks for which considerable reductions can be attained (this is in particular the case for sparse networks and for networks with many terminals).

2.4.1 Theoretical Results

Theoretical results concerning the effectiveness of reductions are very limited. For complete graphs, it can be shown [1] that after the STTE-tests, the expected number of contracted edges is $n(n-1)/v$. In other words, the STTE-tests reduce the number of terminals by a factor of $(n-1)/v$.

For complete graphs, it can be shown [1] that after the MST-tests, the expected number of edges in the reduced network is at most v^2/n. Similar results concerning the SE-tests are given in [7].

2.4.2 Computational Results

For the computational results concerning the effectiveness of reductions, the reader is referred to [1,6,12]. Below the conclusions derived from the test runs reported there are mentioned.

- The speed-up when solving the Steiner tree problem with the reductions is considerable. It can be up to a factor ten for sparse problem instances.
- Reductions perform better for problem instances with larger ratios of n/v. This is mainly due to the fact that the SE-tests are then applicable more often. Also, a good upper bound is usually obtained for this type of problem instances. This makes R- and CR-tests more powerful.
- Problem instances are often completely solved by the reductions alone. This is in particular the case for sparse problem instances ($e \leq 2v$) with many terminals ($n \geq v/2$). This observation also applies for networks with randomly generated edge lengths (with uniform distribution).
- The R- and CR-tests proved to be of limited importance. When applicable, they can often be replaced by other tests.

An important issue which so far has not been addressed adequately in the literature is that of suitable data structures to represent the information needed by various reduction tests. In particular, such data structures must be of dynamic nature as networks undergo non-trivial changes during the reduction process. Another important issue that remains to be addressed in more detail is the applicability of reductions before and in particular during any branch-and-bound algorithm. Developing suitable data structures in this context is even more complicated since networks undergo changes not only when reduced but also during the branching and in particular during the backtracking stages. The use of some reductions in branch-and-bound algorithms have been discussed by Beasley [3] and Voss [11].

References

[1] A. Balakrishnan and N. R. Patel, Problem reduction methods and a tree generation algorithm for the Steiner network problem, *Networks* **17** (1987) 65-85.

[2] J. E. Beasley, An algorithm for the Steiner problem in graphs, *Networks* **14** (1984) 147-159.

[3] J. E. Beasley, An SST-based algorithm for the Steiner problem in graphs, *Networks* **19** (1989) 1-16.

[4] E. W. Dijkstra, A note on two problems in connection with graphs, *Numer. Math.* **1** (1959) 269-271.

[5] R. Dionne and M. Florian, Exact and approximate algorithms for optimal network design, *Networks* **9** (1979) 37-59.

[6] C. W. Duin and A. Volgenant, Reduction tests for the Steiner problem in graphs, *Networks* **19** (1989) 549-567.

[7] C. W. Duin and A. Volgenant, An edge elimination test for the Steiner problem in graphs, *Oper. Res. Lett.* **8** (1989) 79-83.

[8] M. L. Fredman and R. E. Tarjan, Fibonacci heaps and their uses in improved network optimization algorithms, *J. Assoc. Comput. Mach.* **34** (1987) 596-615.

[9] A. Iwainsky, E. Canuto, O. Taraszow and A. Villa, Network decomposition for the optimization of connection structures, *Networks* **16** (1986) 205-235.

[10] R. E. Tarjan, *Data Structures and Network Algorithms*, SIAM (1983).

[11] S. Voss, A reduction based algorithm for the Steiner problem in graphs, *Methods Oper. Res.* **58** (1987) 239-252.

[12] S. Voss, *Steiner-Probleme in Graphen*, Mathematical Systems in Economics **120**, Anton Hein, Frankfurt-am-Main (1990).