

Chapter 4

The Steiner Tree Problem

4.1 Introduction

In this chapter, we consider self-reduction operations for the Steiner Problem in Graphs. Duin and Volgenant published a wide variety of self-reductions in 1989 [21], and very recently Duin has published many improved forms of these tests [20]. Duin's terminology is used throughout the present work.

This chapter does not add any reduction rules to those that appear in [20]. Little, if any, theoretical analysis of these reductions appears to have been carried out, however. Some of the work contained this chapter also appears in [39].

Formally, let G be a connected graph with a positive weight (or "cost") $c(G, uv)$ associated with edge $uv \in E(G)$, and let $K(G)$ denote a subset of the vertex set of G known as the graph's *special vertices*. A *Steiner tree* for G is a subset T of the edges of G such that T is connected, and every vertex of $K(G)$ is adjacent to at least one edge of T . A *minimum Steiner tree* T for a graph G is a Steiner tree for G such that there is no Steiner tree U for G such that the sum of weights of U is less than that of T . We will denote the weight of a minimum Steiner tree on a graph G as $t(G)$. Figure 4.1 shows a graph with special vertices (solid), non-special vertices (hollow) and its minimum Steiner tree (heavy lines).

Analogously to our treatment of graph isomorphism in Chapter 3, for two graphs to be considered isomorphic we shall require that the edge weights of corresponding edges be equal.

The Steiner Problem arises in network layout problems. If each point is represented by a special vertex, and the possible network routes by non-special vertices and edges, a minimum Steiner tree represents the routing pattern of least cost.

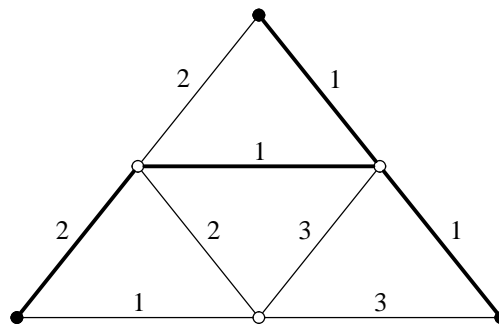


Figure 4.1: A minimum Steiner tree for a graph

4.1.1 Definitions

In the following, G will be a connected graph with special vertices, as described in the introduction. The *length* of a path P is the sum of the weights of the edges contained in P . The minimum spanning tree of a graph will be denoted by $\text{MST}(G)$.

Given two vertices u and v , the *distance* $d(G, u, v)$ between u and v is the minimum length of all u - v paths in G . The *distance graph* $D(G)$ is the complete undirected graph on $V(G)$ with edge uv having weight $d(G, u, v)$.

The *bottleneck length* of a path P in G is the maximum weight among weights on the edges of P . Given two vertices u and v , the *bottleneck distance* $b(G, u, v)$ between u and v is the minimum bottleneck length of all u - v paths in G .

A *special path* is a path in $D(G)$ such that all intermediate vertices (if any) are special. For two vertices u and v , the *special distance* $s(G, u, v)$ between u and v is the minimum bottleneck length over all special u - v paths in $D(G)$. If no special path exists, $s(G, u, v) = \infty$.

A *pseudo-elimination* of a vertex $x \in V(G)$ deletes x from G and for all distinct vertices $u, v \in N(G, x)$, adds an edge with weight $c(G, xu) + c(G, xv)$ if $c(G, xu) + c(G, xv) < c(G, uv)$. If a reduction pseudo-eliminates a vertex and one of the new edges appears in a Steiner tree on the reduced graph, this edge is replaced by the two corresponding edges adjacent to x to generate a Steiner tree on the original graph of the same weight.

4.2 Self-Reductions

Proofs for all of the theorems in this section are given in [21], [35] or [20]; where we have made some amendments to the reduction rules, the amended proofs are given here.

For the self-reductions in this chapter, it is convenient to denote the vertex subject to a reduction ξ by x^ξ , and the edge subject to a reduction ξ by $x^\xi y^\xi$.

4.2.1 Reachability Reduction (RT)

Let T be a Steiner tree on a graph G , and let x be a non-special vertex of G adjacent to no more than two edges of T . Let u_1, u_2 and u_∞ be the special vertices of G at, respectively, the least, second-least and most distance from x of all special vertices in G . If $d(G, x, u_1) + d(G, x, u_2) + d(G, x, u_\infty) \geq c(G, T)$, x is said to be *unreachable*.

Definition 4.2.1 A *reachability reduction* (RT, x) pseudo-eliminates an unreachable vertex x .

Theorem 4.1 If ξ is a reachability reduction on a graph G , then $t(G^\xi) = t(G)$.

Proof. This is shown in [35] and [20] for an unreachable x not adjacent to any edge of an upper bound tree T , but it is sufficient for x to be adjacent to no more than two edges of T . Suppose that x has degree at least three in a Steiner tree T' , and consider the subtree of T' containing u_∞ rooted at x and containing only one edge adjacent to x . Obviously this tree has cost of at least $d(G, x, u_\infty)$. Since x is adjacent to at least two more edges of T' , there must be at least two more similar trees of cost at least $d(G, x, u_1)$ and $d(G, x, u_2)$, respectively. Hence $c(G, T') \geq d(G, x, u_1) + d(G, x, u_2) + d(G, x, u_\infty) \geq c(G, T)$, and T' cannot be a better Steiner tree than T , showing the result. \square

4.2.2 Cut Reachability Reduction (CRT)

For a vertex u of a graph G , let $\hat{c}(G, u)$ denote $\min\{c(G, uv) \mid v \in N(G, u)\}$, and for a set $X \subseteq V(G)$ let $\hat{c}(G, X)$ denote $\sum_{x \in X} \hat{c}(G, x)$. Let T be a Steiner tree on a graph G , and let x be a non-special vertex of G that is not adjacent to any edge of T . Let u_1 and u_2 be distinct special vertices of G such that $d(G, x, u_1) - \hat{c}(G, u_1)$ and $d(G, x, u_2) - \hat{c}(G, u_2)$ are minimised. If $d(G, x, u_1) + d(G, x, u_2) + \hat{c}(G, K(G) - \{u_1, u_2\}) \geq c(G, T)$, then x is said to be *cut unreachable*.

Similarly, if there is an edge $xy \in E(G)$ such that $d(G, x, u_1) + c(G, xy) + d(G, x, u_2) + \hat{c}(G, K(G) - \{u_1, u_2\}) \geq c(G, T)$, then xy is also said to be cut unreachable.

Definition 4.2.2 A *vertex cut reachability reduction* $\langle v\text{CRT}, x \rangle$ on a graph G deletes a cut unreachable vertex x from G . An *edge cut reachability reduction* $\langle e\text{CRT}, xy \rangle$ on a graph G deletes a cut unreachable edge xy from G .

Theorem 4.2 If ξ is a cut reachability reduction on a graph G , then $t(G^\xi) = t(G)$.

Proof. This is proven in [36] and [35]. The proofs for the vertex and edge cases are similar; we will give only the vertex version here.

Suppose a Steiner tree T' contains a cut unreachable vertex x . There must be at least two edges of T' adjacent to x and hence two edge disjoint paths in

T' , not having any internal special vertices, to two special vertices. Considering these two paths, and that every special vertex must be adjacent to at least one edge of T' , we see that $c(G, T') \geq d(G, x, u_1) - \hat{c}(G, x, u_1) + d(G, x, u_2) - \hat{c}(G, x, u_2) + \hat{c}(G, K(G)) \geq c(G, T)$, and hence T' cannot be a better tree than T , showing the result. \square

4.2.3 Nearest Special or Chord Vertices Reduction (NSC)

Let xy be an edge of a minimum spanning tree T for G , and let T_x and T_y be the components of $T - xy$ containing x and y , respectively. Let $\bar{T}(xy)$ be the set of edges connecting a vertex of T_x to a vertex of T_y . Let u_x be the special vertex nearest x in T_x and u_y be the special vertex nearest y in T_y . If for all $vw \in \bar{T}(xy)$, $c(G, vw) \geq \min\{d(G, x, u_x), d(G, x, v)\} + c(G, xy) + \min\{d(G, y, u_y), d(G, y, w)\}$, then xy satisfies the conditions for an NSC reduction.

Definition 4.2.3 A *nearest special or chord vertices reduction* $\langle \text{NSC}, xy, z \rangle$ on a graph G contracts G along a subject edge xy into a special vertex z with $c(G, zu) = \min\{c(G, xu), c(G, yu)\}$ for each edge zu adjacent to z .

Theorem 4.3 If ξ is a nearest special or chord vertices reduction on G , then $t(G^\xi) = t(G) - c(G, x^\xi y^\xi)$ and $x^\xi y^\xi$ is a part of a minimum Steiner tree for G .

Proof. This is discussed in Section 3.2 of [20]. \square

4.2.4 Smaller Special Distance Reduction (SD)

Definition 4.2.4 A *smaller special distance reduction* $\langle \text{SD}, xy \rangle$ on a graph G deletes an edge xy with $c(G, xy) > s(G, x, y)$ or $c(G, xy) > \max\{s(G, u, v) \mid u, v \in K(G)\}$ from G .

Theorem 4.4 If ξ is a smaller special distance reduction on a graph G , then $t(G^\xi) = t(G)$.

Proof. This is the combination of the “special distance” and “excessive cost” tests described in Section 2.2 of [20]. Suppose that an edge xy with $c(G, xy) > s(G, x, y)$ lies on a Steiner tree T of a graph G , and consider the components of $T - xy$. Consider the minimum special path from x to y in G . It crosses from one component of $T - xy$ to the other, and has less cost than xy . Hence adding the edges of this path to $T - xy$ forms a Steiner tree with cost less than T , showing that any such T is not minimum. Similarly, it is easy to see that no edge with greater cost than the maximum special distance between two special vertices can be in any minimum Steiner tree. \square

4.2.5 Local Steiner Cost Reduction (LSC)

Let $S(G, X)$ be the graph on vertex set $X \subseteq V(G)$ with $c(S(G, X), xy) = s(G, x, y)$ for every $x, y \in X$.

Definition 4.2.5 A *local Steiner cost reduction* $\langle \text{LSC}, x \rangle$ pseudo-eliminates a non-special vertex x with $t(S(G, N')) \leq \sum_{n \in N'} c(G, xn)$ for all sets $N' \subseteq N(G, x)$ and $|N'| \leq |K|$.

Theorem 4.5 If ξ is a local Steiner cost reduction on a graph G , then $t(G^\xi) = t(G)$.

Proof. This is discussed in Section 4.3 of [20]. \square

Duin also describes a similar test LSD^P that uses the minimum spanning tree costs instead of the minimum Steiner tree costs for performing the test. Analogous to pseudo-eliminating a vertex, an edge xy can be pseudo-eliminated by deleting it and replacing with edges yz for all $z \in N(G, x) - \{y\}$ of weight $c(G, xz) + c(G, xy)$ if $c(G, xz) + c(G, xy) < c(G, yz)$.

Definition 4.2.6 An *edge local special distance reduction* $\langle \text{eLSD}, xy \rangle$ pseudo-eliminates an edge xy with with

$$c(\text{MST}(S(G, N'))) \leq \sum_{n \in N'} c(G, xn)$$

for all sets $N' \subseteq N(G, x)$ containing y and $|N'| \leq |K|$. A *vertex local special distance reduction* $\langle v\text{LSD}, x \rangle$ pseudo-eliminates a non-special vertex x with

$$c(\text{MST}(S(G, N'))) \leq \sum_{n \in N'} c(G, xn)$$

for all sets $N' \subseteq N(G, x)$ and $|N'| \leq |K|$.

Obviously, any vertex subject to LSD reduction is also subject to LSC reduction. However, Duin shows how to improve the basic LSD test by increasing the size of the tree rooted at x that is examined by the test. If the LSD test failed on a vertex x , then the stars induced by $N' \cup x$ are extended from the non-special vertices of N' to form all the trees such that no leaf is more than two edges from x , and all the interior vertices are non-special. If this LSD² test fails, the trees are extended again, and so on, until the trees cannot be extended any further or a limit imposed on the number of extensions is reached.

4.2.6 Local Bottleneck Distance Reduction (LBD)

Let $B'(G, Z, x)$ be the graph on a set of vertices $Z \subseteq V(G)$ with $c(B'(G, Z, x), yz) = b(G - x, y, z)$ for all $y, z \in Z$. Let x be a non-special vertex and ϕ be a one-to-one mapping from $X \subset N(G, x)$ to $Y \subseteq K(G)$ such that $\sum_{n \in X} (c(G, xn) + d(G, n, \phi(n))) \leq c(\text{MST}(B'(G, Y, x)))$. If such a mapping ϕ exists, then x satisfies the conditions for an LBD reduction.

Definition 4.2.7 A *local bottleneck distance reduction* $\langle \text{LBD}, X, x \rangle$ adds a subject vertex x to the set of special vertices.

Theorem 4.6 If ξ is a local bottleneck distance reduction on G , then $t(G^\xi) = t(G)$.

Proof. This is discussed in Section 4.2 of [20]. □

Clearly, finding the mapping required by the definition of the LBD test may require exponential time. Hence, Duin suggests that only a few sets be chosen by heuristics for trial. He suggests the set of vertices adjacent to x in the minimum spanning tree of G , and the assignment that minimises $\sum_{n \in X} d(G, n, \phi(n))$.

4.2.7 Change Edge-Costs Reduction (CEC)

Theorem 4.7 Let x be a non-special vertex of a graph G , and let xu and xv be distinct edges of G . Let G' be a graph the same as G except that $c(G', xu) = c(G, xu) + a$ and $c(G', xv) = c(G, xv) - a$ for some positive number a . If xu and xv are both in a minimum Steiner tree for G and G' when x is considered to be special, then a minimum Steiner tree on G' is also a minimum Steiner tree on G .

Theorem 4.7 does not reduce the problem in itself. However, suppose that the NSC test fails on xu and xv in G , but succeeds in the modified graphs. Then the edge costs can be changed as in the theorem, and reduction can proceed in G' .

Duin and Volgenant's rule proceeds as follows (replacing their "nearest special vertices" test with the stronger NSC test):

Let x be a non-special vertex of G and xu and xv be distinct edges on a minimum spanning tree of G such that u is a special vertex. If the NSC test succeeds on xv on G with x considered special, then xu can have its weight increased by a and xv have its weight decreased by a for $a \leq c(G, xv)$ and $a < b(G - xu, x, u) - c(G, xu)$.

4.3 Maintaining the Upper Bound

The tree used by Theorems 4.1 and 4.2 can be obtained by an approximation algorithm for the Steiner Problem in Graphs. We will assume that any approximate T is minimal, that is, that there is no Steiner tree T' such that $T' \subset T$. Any realistic approximation algorithm will have this property, and, in any case, it is trivial to compute a minimal tree from a non-minimal one.

When we consider the properties of sets containing the RT and CRT reductions, it is necessary to consider what happens to the approximate tree after a reduction. If a new tree were to be computed from scratch after every reduction, it seems unlikely that any results could be obtained at all, or, at least, none that

were independent of the particular algorithm used for computing the tree. This method of maintaining an upper bound seems highly inefficient and unlikely to be desirable in practice, in any case.

Duin and Volgenant re-compute the upper bound only when the graph can no longer be reduced by the SD, LSD, NSC or CEC reductions. However, it seems profitable both in theory and practice to maintain the upper bound by modifying the tree in a straightforward manner for every reduction performed on the graph, since re-computing the tree from scratch is expensive.

It is trivial to maintain the tree for the RT and CRT reductions, and obviously the reduced tree will be no heavier than the original tree.

If an SD reduction deletes an edge xy from the tree, then the tree will be re-connected by inserting the portion of the minimum special x - y path that crosses between the two components of the tree. It is easy to see that the new upper bound will be strictly lesser than the old upper bound.

If an LSC reduction deletes a vertex x from the tree, the tree will be re-connected using the special distance Steiner tree for all of the vertices adjacent to x . If, for tree-vertices $u, v \in N(G, x)$, uv is an edge of the reduced graph, this edge will be inserted into the tree. Otherwise, the tree will be re-connected as if for an SD reduction of uv . Since this Steiner tree must be of less or equal weight to the edges adjacent to x , the upper bound cannot be increased by this procedure. Where x is not part of the tree but modifies an edge that is, the procedure for an RT reduction will be followed.

If an NSC reduction contracts an edge that is part of the tree, the tree will be contracted with it. If an NSC reduction causes a cycle to appear in the tree, the cycle will be broken by removing the heaviest edge on this cycle. Again, the new tree will be no heavier than the original tree.

4.4 Confluence

Theorem 4.8 The set $\{\text{RT}\}$ is confluent.

Proof. Consider RT reductions ξ and ζ on a graph G . As RT reduction cannot decrease distances in the graph, and cannot increase the weight of the upper bound tree, x^ζ can be deleted from G^ξ by RT reduction to form a graph $G^{\xi\zeta}$. Similarly, a graph $G^{\zeta\xi}$ can be formed from G^ζ , and this graph is isomorphic to $G^{\xi\zeta}$, showing that $G^\xi \downarrow G^\zeta$. Hence $\{\text{RT}\}$ is locally confluent, and confluence follows from Theorem 2.2 since the set is obviously terminating. \square

Theorem 4.9 The set $\{\text{LBD}\}$ is confluent.

Proof. The proof is similar to that of Theorem 4.8. \square

Lemma 4.1 Let G' be the graph obtained from G by application of a reduction from the set $\{\text{NSC}, \text{SD}, \text{LSD}, \text{LBD}, \text{RT}\}$. For any distinct $u, v \in V(G')$, $s(G', u, v) \leq s(G, u, v)$.

Proof. It is obvious that the NSC and LBD reductions can only decrease special path length since the former contracts edges (making all paths through the contracted edge shorter), and the latter can only increase the number of possible special paths without changing any lengths. Clearly an SD reduction cannot increase minimum special path lengths because the deleted edge had weight greater than the minimum special path between its endpoints. For the LSD and RT reductions, consider a special path passing through the pseudo-eliminated vertex. Since the edges adjacent to the pseudo-eliminated vertex are replaced by an edge of at most equal length, and the pseudo-eliminated vertex was not special, a special path (using the new edge) of no greater length than the deleted one must exist in the reduced graph. \square

Corollary 4.9.1 The set $\{\text{SD}\}$ is confluent.

Proof. The proof is similar to that of Theorem 4.8. \square

Theorem 4.10 The set $\{\text{RT}, \text{SD}\}$ is confluent.

Proof. Let ξ and ζ be reductions on a graph G . If ξ and ζ are both RT reductions or both SD reductions, then $G^\xi \downarrow G^\zeta$ from Theorem 4.8 or Corollary 4.9.1. Otherwise, without loss of generality, let ξ be the RT reduction.

From Lemma 4.1, ζ is a legal reduction in G^ξ , yielding a graph $G^{\xi\zeta}$.

As ζ does not reduce distances in the graph and does not increase the weight of the tree, ξ is a legal reduction on G^ζ unless x^ξ is adjacent to more than two edges of the reduced upper bound tree. As x^ξ was adjacent to no more than two in the original tree, this would imply that ζ caused the tree to be re-connected by joining an edge to x^ξ . But any tree in which x^ξ is adjacent to more than two vertices must have weight at least as great as that of the original tree, contradicting the properties of an SD test performed on an edge of the tree. Hence ξ is a legal reduction in G^ζ , forming a graph $G^{\xi\zeta}$.

$G^{\xi\zeta}$ and $G^{\zeta\xi}$ are obviously isomorphic unless $x^\zeta y^\zeta$ is adjacent to x^ξ in G , that is, without loss of generality, $x^\xi = x^\zeta$. In this case, $G^{\xi\zeta}$ may have extra edges linking y^ζ and other neighbours of x^ξ . Consider such a neighbour z . The edge $y^\zeta z$ has weight $c(G, x^\xi z) + c(G, x^\xi y^\zeta)$. By traversing the minimum special x^ζ - y^ζ path starting at z , we see that $s(G^{\xi\zeta}, z, y^\zeta) \leq s(G, x^\zeta, y^\zeta) + c(G, x^\xi z)$. Since $x^\zeta y^\zeta$ was subject to SD in G , $s(G^{\xi\zeta}, z, y^\zeta) < c(G, x^\xi y^\zeta) + c(G, x^\xi, z)$ and hence $y^\zeta z$ is subject to SD deletion in $G^{\xi\zeta}$. By applying the SD reduction to all such edges, we can obtain a graph isomorphic to $G^{\zeta\xi}$.

Hence $G^\xi \downarrow G^\zeta$, showing local confluence. The set is obviously terminating, and so confluence follows from Theorem 2.2. \square

Note, however, that Theorem 4.10 fails if we use Duin and Volgenant's original RT reduction, which is slightly weaker than our version as it does not eliminate unreachable vertices adjacent to any edges of the upper tree (in all other respects, it is the same as the version used here). In this case, an RT vertex might lie on the minimum special path used to re-connect the tree after an SD reduction and consequently be unavailable for further reduction, as shown in Fig. 4.2. Heavy lines show the upper bound tree.

Unlike the reductions mentioned so far, the set $\{\text{NSC}\}$ is not confluent, as demonstrated by Fig. 4.3. It is easy to check that both of the reduced graphs are irreducible under RT and SD, and so we immediately obtain the result that the set $\{\text{RT}, \text{SD}, \text{NSC}\}$ and all of its subsets containing NSC are not confluent.

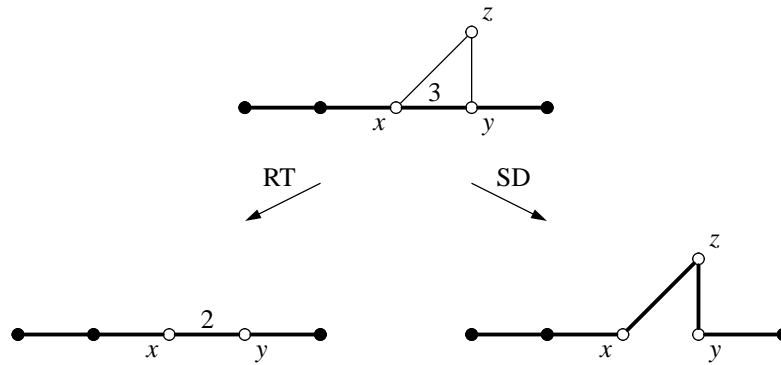


Figure 4.2: Two non-isomorphic irreducible graphs produced by SD reduction and Duin and Volgenant's original RT reduction.

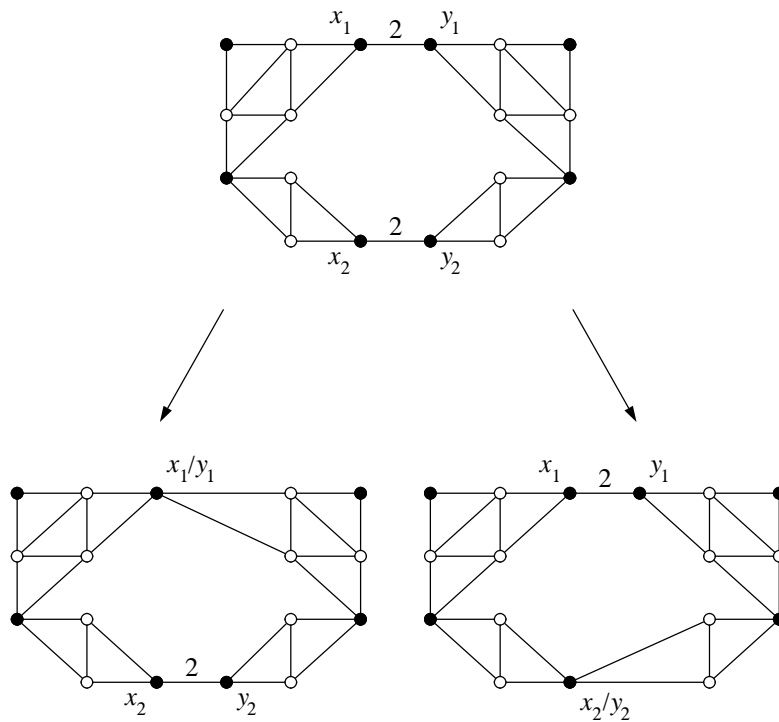


Figure 4.3: Two non-isomorphic irreducible graphs produced by NSC reduction.

Note that the graph of Fig. 4.3 has several distinct minimum spanning trees, and that the two edges subject to NSC reduction lie on different spanning trees. In [39], we showed that Duin and Volgenant’s “nearest special vertices” test, a special case of Duin’s NSC test, is confluent over the set of all graphs with a unique minimum spanning tree. This result is easily extended for the stronger NSC test.

Lemma 4.2 If ξ and ζ are distinct NSC reductions on a graph G , and G has a unique minimum spanning tree, then $x^\zeta y^\zeta$ is subject to NSC reduction on G^ξ .

Proof. It is obvious that $x^\zeta y^\zeta$ is on a minimum spanning tree of G^ξ , since it was on the minimum spanning tree of G . As ξ does not create edges or alter any paths that are not along the minimum spanning tree of G , any chord of $\bar{T}_{x^\zeta y^\zeta}$ of G^ξ must also be a chord of $\bar{T}_{x^\zeta y^\zeta}$ of G .

Consider such a chord $vw \in \bar{T}_{x^\zeta y^\zeta}$ in G^ξ . Obviously

$$\min\{d(G^\xi, x^\zeta, u_{x^\zeta}), d(G^\xi, x^\zeta, v)\} \leq \min\{d(G, x^\zeta, u_{x^\zeta}), d(G, x^\zeta, v)\}$$

and similarly

$$\min\{d(G^\xi, y^\zeta, u_{y^\zeta}), d(G, y^\zeta, w)\} \leq \min\{d(G, y^\zeta, u_{y^\zeta}), d(G, y^\zeta, w)\}.$$

As $c(G^\xi, vw) = c(G, vw)$ and $c(G^\xi, x^\zeta y^\zeta) = c(G, x^\zeta y^\zeta)$, and vw satisfied the NSC conditions in G , it is clear that vw still satisfies the NSC conditions in G^ξ and the result follows. \square

Corollary 4.10.1 The set {NSC} is confluent over the set of all graphs with unique minimum spanning trees.

Proof. First, it is easy to see the the set of all graphs with unique minimum spanning trees is closed under NSC reduction, as the minimum spanning tree contracts with the reduction. The proof is then similar to that of Theorem 4.8. \square

If we assume that the minimum spanning tree used by a reduction algorithm is contracted with an NSC reduction (rather than being re-computed from

scratch), Corollary 4.10.1 can be used to show that, given a minimum spanning tree on an arbitrary initial graph, the irreducible graph that results from a sequence of NSC reductions is unique. That is to say, selection of the minimum spanning tree on the original graph is the only choice of any consequence for NSC reduction.

Figure 4.4 shows two non-equivalent, maximal sequences of LSC reductions on a graph, showing that LSC is not confluent. In [41], the present author conjectured that the set $\{\text{LSD}\}$ is confluent; however, all the reductions in Fig. 4.4 are valid LSD (and LSD^P) reductions and the final graphs are irreducible under LSD^P , which refutes this conjecture. As the reduced graphs are irreducible under RT, CRT and SD reduction, we immediately obtain the result that the set $\{\text{RT}, \text{CRT}, \text{SD}, \text{LSC}, \text{LSD}^P\}$, and its subsets containing LSC or LSD^P , are not confluent.

Observing that distances can only be increased, and the upper bound left unchanged, by a CRT reduction, it seems unlikely that a vertex subject to a CRT reduction could ever cease to be subject to CRT reduction by the action of other CRT reductions.

Conjecture 4.1 The set $\{\text{CRT}\}$ is confluent.

Regardless of whether or not Conjecture 4.1 holds, it is easy to see from the proof of Theorem 4.2 and that if a vertex or edge is subject to a CRT reduction, then there is always a minimum Steiner tree not containing this vertex for so long as only CRT reductions are performed on the graph. Hence, it is possible to “force confluence” of $\{\text{CRT}\}$ by testing all edges and vertices in the graph and marking them all for deletion without checking the CRT conditions again.

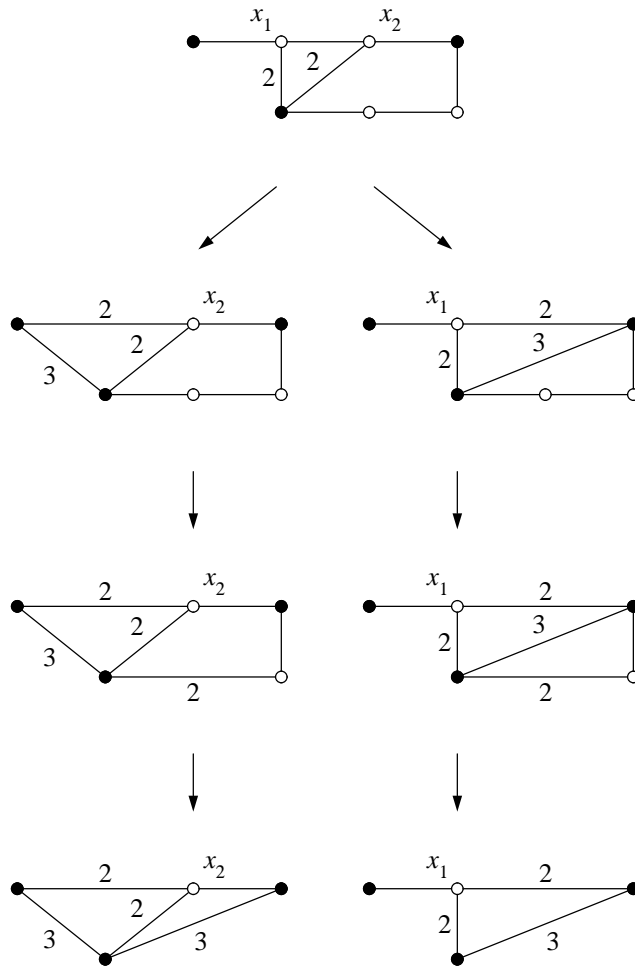


Figure 4.4: Two non-isomorphic irreducible graphs produced by LSC reduction.

