MIT Sloan School of Management

Working Paper 4392-02
September 2002

THE EXTENDED NEIGHBORHOOD: DEFINITION AND CHARACTERIZATION

James B. Orlin and Dushyant Sharma

The Extended Neighborhood

James B. Orlin[1] and Dushyant Sharma[2]

Abstract

We consider neighborhood search defined on combinatorial optimization problems. Suppose that N is a neighborhood for combinatorial optimization problem $\mathbf{X}$. We say that N' is LO-equivalent (locally optimal) to N if for any instance of $\mathbf{X}$, the set of locally optimal solutions with respect to N and N' are the same. The union of two LO-equivalent neighborhoods is itself LO-equivalent to the neighborhoods. The largest neighborhood that is LO-equivalent to N is called the extended neighborhood of N, and denoted as N^*. We analyze some basic properties of the extended neighborhood. We provide a geometric characterization of the extended neighborhood N^* when the instances have linear costs defined over a cone. For the TSP, we consider 2-opt*, the extended neighborhood for the 2-opt (i.e., 2-exchange) neighborhood structure. We show that number of neighbors of each tour T in 2-opt* is at least $(n/2 -2)!$. We show that finding the best tour in the 2-opt* neighborhood is NP-hard. We also show that the extended neighborhood for the graph partition problem is the same as the original neighborhood, regardless of the neighborhood defined. This result extends to the quadratic assignment problem as well. This result on extended neighborhoods relies on a proof that the convex hull of solutions for the graph partition problem has a diameter of 1, that is, every two corner points of this polytope are adjacent.

(Last modified 09/29/02)

[1] MIT Sloan School of Management, E40-147, Cambridge, MA 02139, jorlin@mit.edu

[2] Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI 48109, dushyant@umich.edu

1. Introduction

Let **X** denote a combinatorial optimization problem, which is specified as a set of instances. An instance of **X** is a pair $\mathcal{I} = (S, f)$, where S denotes a finite set of feasible solutions and $f: S \to R$ is an objective function selected from some class $\mathcal{F}$ of objective functions. If $f(x) = cx$ for all $x \in S$, we alternatively express the instance as (S, c). A *neighborhood structure* for X is a mapping $N = (N_S)$, where for each feasible set S of solutions $N_S: S \to 2^S$ is a mapping from the set S to a subset of feasible solutions. For an instance (S, f), we usually refer to $N_S(x)$ as the *neighborhood of x with respect to N*, with the understanding that the neighborhood not only depends on x and on N but also on the instance. We note that the mapping N_S does not vary with the objective function f. We always assume that $x \in N_S(x)$.

We use the shorthand $S \in \mathbf{X}$ to mean that there is an instance (S, f) of **X** for some objective function f.

We say that a feasible solution x for (S, f) is *locally optimal* with respect to N if $f(x) \leq f(x')$ for all $x' \in N_S(x)$.

A *neighborhood search* algorithm is an iterative procedure that starts with an initial solution x. At each iteration the algorithm searches for a better solution in the neighborhood $N_S(x)$. If a better solution y is found, the current solution x is replaced by y and the algorithm continues. If there is no better solution in the neighborhood of the current solution x, then the algorithm returns the current solution, which is a locally optimal solution.

Neighborhood search is a popular method to solve difficult optimization problems (Aarts and Lenstra [1997], Voss et al. [1999]), largely because of its intuitive appeal and empirical success in solving difficult optimization problems. A number of neighborhood structures have been proposed in the literature for various combinatorial optimization problems. We refer the reader to the recent survey papers of Ahuja et al. [2001], Deineko and Woeginger [2000] for additional information.

We denote the set of locally optimal solutions with respect to a neighborhood structure N in the instance $\mathcal{I}$ by $LO_{\mathcal{I}}^N \subseteq S$. We say that two neighborhood structures N^1 and N^2 are *LO-equivalent* on the combinatorial optimization problem **X** if and only if $LO_{\mathcal{I}}^{N^1} = LO_{\mathcal{I}}^{N^2}$ for all instances $\mathcal{I}$ of **X**, i.e., N^1 and N^2 have the same set of local optima for all instances of **X**.

Suppose that N^1 and N^2 are two neighborhood structures defined on the same problem **X**. The union of N^1 and N^2 is the neighborhood function N such that $N_S(x) = N_S^1(x) \cup N_S^2(x)$ for all feasible sets $S \in \mathbf{X}$ and for all $x \in S$.

We state the following elementary property of LO-equivalence without proof.

Proposition 1. *If two neighborhood structures* N^1 *and* N^2 *are LO-equivalent then the neighborhood structure defined as* $N = N^1 \cup N^2$ *is LO-equivalent to both* N^1 *and* N^2.

We define the *extended neighborhood* of N for **X**, denoted by N^*, as the neighborhood structure that is LO-equivalent to N and has the following property: if N^1 is LO-equivalent to N over **X** and if S is any feasible set of solutions for an instance, then $N^1_S(x) \subseteq N^*_S(x)$ for all $x \in S$. That is, N^* is the largest neighborhood structure that is LO-equivalent to N over **X**. Proposition 1 shows that the extended neighborhood is well defined. A corollary of Proposition 1 is that if N^* denotes the extended neighborhood of N, then the extended neighborhood of N^* is N^*.

Searching the original neighborhood N using neighborhood search will also search the extended neighborhood, as we state more precisely in Section 2.

We say that a combinatorial optimization problem **X** *has linear costs defined over a cone* if the following is true:

(1) each instance (S, c) of **X** has a linear objective function;

(2) associated with each set $S \in$ **X**, with $S \subseteq \mathbb{R}^n$ is a polyhedral cone of objective functions $\mathcal{F}(S) \subseteq \mathbb{R}^n$. This means that for each set $S \in$ **X**, the pair (S, c) is an instance of **X** if and only if $c \in \mathcal{F}(S)$.

The major contributions of this paper are as follows:

(1) We define the extended neighborhood, and relate the extended neighborhood to other concepts in neighborhood search including very large scale neighborhood (VLSN) search, and domination number, and exact neighborhoods.

(2) We provide a geometric characterization of the extended neighborhood when the instances have linear costs defined over a cone.

(3) For the TSP, we consider 2-opt*, the extended neighborhood for the 2-opt (i.e., 2-exchange) neighborhood structure. We show that number of neighbors of each tour T in 2-opt* is at least $(n/2 - 2)!$.

(4) We show that finding the best tour in the 2-opt* neighborhood is NP-hard.

(5) We show that there are instances of the TSP, where a locally optimal tour T′ is reachable from a tour T with respect to the 2-opt* neighborhood, but not with respect to the 2-opt neighborhood.

(6) We show that the extended neighborhood for the graph partition problem is the same as the original neighborhood, **regardless of the neighborhood defined**. This result extends to the quadratic assignment problem as well. This result relies on a

proof that the convex hull of solutions for the graph partition problem has a diameter of 1, that is, every two corner points of this polytope are adjacent.

The rest of the paper is organized as follows. In Section 2, we give some additional definitions, notation, and some basic results regarding extended neighborhoods. In Section 3, we provide the characterization of the extended neighborhood in the case of combinatorial optimization problems with linear costs defined over a cone, and present some additional results for important special cases. In Section 4, we show that under commonly occurring conditions, one can recognize a solution in the extended neighborhood in polynomial time. In Section 5, we present results regarding properties of 2-opt[*], the extended neighborhood for the 2-opt neighborhood structure for the TSP, and establish the reachability result mentioned in point (5) above. In Section 6, we show that for any n-city tour T, the number of neighbors in 2-opt[*](T) is at least $(n/2 - 2)!$ In Section 7, we show that optimizing over 2-opt[*] is NP-hard. In Section 8, we show that if N is any neighborhood the extended neighborhood for the graph partition problem, then $N^* = N$. We summarize contributions and provide future research directions in Section 9.

2. Neighborhood Search, LO-equivalence and the Extended Neighborhood

In this section, we consider a variety of elementary results about neighborhood search, LO-equivalence, and the extended neighborhood. We will point out how the extended neighborhood relates to very large scale neighborhood (VLSN) search, and also draw a connection to the dominance number developed by Glover and Punnen (1997) and to exact neighborhoods (see for example Papadimitriou and Steiglitz [1982]).

Let $\mathbf{X}$ be a combinatorial optimization problem. For each $S \in \mathbf{X}$, let $\mathbf{X}_S$ be problem $\mathbf{X}$ as restricted to instances (S, f) for functions $f \in \mathcal{F}(S)$. Let $N = (N_S)$ denote a neighborhood function for problem $\mathbf{X}$, and let $N^* = (N_S^*)$ denote the extended neighborhood. The definition of LO-equivalence and the extended neighborhood apply equally to the problem $\mathbf{X}_S$.

The next observation formalizes the property that the extended neighborhood can be determined independently for each feasible region $S \in \mathbf{X}$.

Observation 1. *Let* N *be a neighborhood of* $\mathbf{X}$ *and let* N^* *be the extended neighborhood. Then for each feasible set* $S \in \mathbf{X}$, N_S^* *is the extended neighborhood of* N *for* $\mathbf{X}_S$.

We note that the extended neighborhood is defined even if there is a single instance (S, f) in $\mathbf{X}_S$. We describe the extended neighborhood in this situation as Proposition 2, which we state and prove next. We will give a polyhedral proof of Proposition 2 in Section 3 in the case that the objective function is linear. This second proof illustrates the main theorem in that section.

Proposition 2. *Suppose that* $\mathbf{X}$ *is a combinatorial optimization problem, and that* (S, f) *is the unique instance in* $\mathbf{X}_S$. *Let* N *be a neighborhood defined on* $\mathbf{X}$*, and let* L *be the set of locally*

optimal solutions for instance (S, f). *If* $x \notin L$, *then* $N_S^*(x) = S$. *If* $x \in L$, *then* $N_S^*(x) = \{y \in S: f(y) \geq f(x)\}$.

Proof. Let us first define N_S' as follows. If $x \notin L$, then $N_S'(x) = S$. If $x \in L$, then $N_S'(x) = \{y \in S : f(y) \geq f(x)\}$. We want to establish that $N_S' = N_S^*$. Let L' denote the set of locally optimal solutions for S with respect to N_S'. Since $N_S(x) \subseteq N_S'(x)$ for all $x \in S$, it follows that $L' \subseteq L$. Conversely, any solution x in L is a locally optimal solution with respect to N_S' by its definition. Therefore $L \subseteq L'$. We conclude that $L' = L$, and N_S' is LO-equivalent to both N_S^* and N_S. Finally, we claim that N_S' is the largest neighborhood that is LO-equivalent to N_S. To see this, note that if $x \notin L$, then $N_S'(x) = S$, which is as large a neighborhood as possible. If $x \in L$, then $N_S'(x) = \{y \in S : f(y) \geq f(x)\}$. Adding any solution to $N_S'(x)$ will result in x not being locally optimal, in which case the resulting neighborhood would not be LO-equivalent. So any neighborhood that strictly contains N_S' would not be LO-equivalent to N_S. This completes the proof. ♦

Our definition of the extended neighborhood is in terms of LO-equivalence, which relates to our motivation for studying the extended neighborhood. Below, we provide an alternative characterization of the extended neighborhood that is sometimes simpler to use in proofs.

Lemma 1. *Suppose that* **X** *is a combinatorial optimization problem and that* N *is a neighborhood function defined on* **X**. *Suppose that* $S \in$ **X**, *and that* $x \in S$ *and* $x' \in S$. *Then* $x' \in N_S^*(x)$ *if and only if there is no instance* (S, f) *of* **X** *with the following properties:*

> (1) *x is locally optimal for* (S, f),
>
> (2) $f(x') < f(x)$.

Proof. Suppose first that there is an instance (S, f) of **X** with properties (1) – (2). Since x is locally optimal for (S, f), and $f(x') < f(x)$, it follows that $x' \notin N_S^*(x)$. Conversely, suppose that there is no instance S satisfying (1) and (2). Let N^* denote the extended neighborhood, and let N' be defined as follows: $N_S'(x) = N_S(x) \cup \{x'\}$. $N_S'(y) = N_S(y)$ for $y \neq x$, and $N_{S'}' = N_{S'}$ for $S' \neq S$. We claim that N' and N are LO-equivalent. To see this, first consider a feasible solution $y \neq x$. Since $N'(y) = N(y)$, it follows that y is locally optimal with respect to N, if and only if it is also locally optimal with respect to N'. We now consider x. Since x does not have the properties (1) and (2), it follows that x is locally optimal with respect to N if and only if it is also locally optimal with respect to N'. This establishes that N and N' are LO-equivalent. By Proposition 1, it follows that N and $N' \cup N^*$ are also LO-equivalent. Since N^* is the largest neighborhood that is LO-equivalent to N, it follows that $N' \subseteq N^*$, and thus $x' \in N^*(x)$, which completes the proof. ♦

We say that two problems **X** and **Y** *are defined over the same set of feasible regions* if the following is true: $S \in$ **X** if and only if $S \in$ **Y**. For example, suppose that **X** is the set of traveling

salesman instances with nonnegative costs, and **Y** is the set of traveling salesman instances with nonnegative costs satisfying the triangle inequality. Then **X** and **Y** are defined over the same set of feasible regions. Since every instance (S, f) of **Y** is also an instance of **X**, we would write that $Y \subseteq X$. If **X** and **Y** are defined over the same feasible regions, then any neighborhood function for **X** is also a neighborhood function for **Y**.

Proposition 3. *Suppose that* **X** *and* **Y** *are problems defined over the same set of feasible regions, and suppose that* $X \subseteq Y$. *Suppose further* N *is any neighborhood structure defined on* **X** *(and on* **Y***). Let* N^* *be the extended neighborhood of* N *for problem* **X** *and let* N′ *be the extended neighborhood of* N *for problem* **Y**. *Then for any feasible region* S *and for any* $x \in S$, $N'_S(x) \subseteq N^*_S(x)$.

Proof. Suppose that $x' \notin N^*_S(x)$. Then by Lemma 1, there is an instance (S, f) of **X** such that x is locally optimal for the instance, and such that $f(x') < f(x)$. Since (S, f) is also an instance of **Y**, it follows that $x' \notin N'_S(x)$, which completes the proof.   ◆

Proposition 3 establishes that the extended neighborhood decreases in size as one considers instances with a wider range of objective functions.

We now return our attention to properties of local search heuristics.

Let $\mathcal{I} = (S, f)$ be an instance of a combinatorial optimization problem **X** and let N be a neighborhood structure defined on X. We say that a sequence of solutions $x^1, x^2, ..., x^k$ is a *feasible search sequence* if

(1) $x^1 \in S$;

(2) $x^j \in N_S(x^{j-1})$ for each $j = 2$ to k;

(3) $f(x^j) < f(x^{j-1})$ for each $j = 2$ to k.

If $x^1, x^2, ..., x^k$ is a feasible search sequence, and if x^k is a locally optimal solution, we refer to the sequence as *terminating*.

Proposition 4. *Suppose that* $\mathcal{I}$ *is an instance of a combinatorial optimization problem* **X**. *Suppose that* N *is a neighborhood structure defined on* **X** *and let* N^* *denote its extended neighborhood. Then any feasible search sequence* $x^1, x^2, ..., x^k$ *for* N *on instance* $\mathcal{I}$ *is also a feasible search sequence for* N^* *on instance* $\mathcal{I}$. *Moreover, if* $x^1, x^2, ..., x^k$ *is terminating for* N, *then it is also terminating for* N^*.

Proof. The fact that $N_S(x) \subseteq N^*_S(x)$ for all x implies the first statement of the proposition. The fact that N^* has the same set of local optima as N implies the second.   ◆

Proposition 4 shows a close connection between searching a neighborhood and searching its extended neighborhood. There are times in which N is small, but N^* is exponentially large. In these cases, N is in some sense equivalent to an exponentially large neighborhood. Proposition 4 provides much of our motivation for studying properties of the extended neighborhood.

When N^* is exponentially large, then searching N^* is a special case of very large scale neighborhood (VLSN) search, as surveyed by Deineko and Woeginger [1997] in the case of the TSP and by Ahuja, Ergun, Orlin, and Punnen [2002] for the TSP and other problems. Ahuja et al [2002] provided the following rule of thumb in their survey: the larger the size of the neighborhood of each solution, the better is the quality of the locally optimal solutions. This rule of thumb is de facto violated in the case of the extended neighborhood since N and N^* have exactly the same set of locally optimal solutions.

Glover and Punnen [1997] introduced a concept related to extended neighborhoods. They defined the *domination ratio* of a heuristic algorithm α for a combinatorial problem as $dom(\alpha) = \inf_{(S, f)} \left| \{x \in S : f(x) \geq f(x_\alpha)\} \right| / |S|$ where x_α is the solution obtained by the heuristic α on instance (S, f). The domination ratio is the minimum fraction of solutions that are guaranteed to be worse than the solution obtained by the heuristic α. The concept is applicable to any heuristic including neighborhood search heuristics; however, it is not directly associated with the definition of a neighborhood structure. For example, if we generate a random solution from the set S, the domination number of our heuristic is approximately half of the total solutions, even though there is no neighborhood structure involved. We note that for neighborhood search heuristics, $\min (|N^*(x)| : x \in S)/|S|$ is a lower bound on the domination number of the heuristic on instances in which the feasible region is S. Other papers on domination analysis include Gutin and Yeo [2002], Punnen et al. [2002], Glover et al. [2001], and Punnen and Kabadi [2002].

A neighborhood structure is called *exact* for a combinatorial optimization problem if any locally optimal solution for an instance is also an optimal solution for the instance. One of the consequences of the definition of extended neighborhoods is an alternative characterization of exact neighborhood structures. The following proposition follows directly from the definition of the extended neighborhood of a neighborhood structure, and shows how exact neighborhoods are related to extended neighborhoods.

Proposition 5. *A neighborhood structure* N *is exact for a combinatorial optimization problem* $\mathbf{X}$ *if and only if for any instance* (S, f) *of* $\mathbf{X}$*, the extended neighborhood* N^* *satisfies the condition*: $N^*_S(x) = S$ *for every* $x \in S$.

3. Combinatorial Optimization Problems with linear cost objectives

In this section, we study some properties of the extended neighborhoods for combinatorial optimization problems with linear costs defined over a cone. In particular, we consider the set $S \in \mathbf{X}$ with $S \subseteq \mathbb{R}^n$. We say that the costs for S are defined over a cone if the

following is true: there is a polyhedral cone $\mathcal{F} = \{c: cA_S \geq 0\}$, where A_S is an n × m matrix for some m, and such that (S, c) is an instance of $\mathbf{X}_S$ if and only if $c \in \mathcal{F}$. As is suggested by our notation, A_S may depend on the feasible set of solutions S. Although A_S varies with the set S of feasible solutions, in most examples A_S will be the same for all instances of the combinatorial optimization problem that have the same dimension. We also assume here that m is finite, but the results will also carry through if m is infinite.

In this section, we provide a polyhedral description of the extended neighborhood in terms of the neighborhood structure N, the solution set S, and the constraint matrices A_S for cost vectors. We also consider the special cases in which all cost vectors are permitted, and when all non-negative cost vectors are permitted.

Suppose that $S \in \mathbf{X}$ and that $N_S(x) = \{x^1, \ldots, x^K\}$ for some K. Then for each i = 1 to K, we let $v^i = x^i - x$. We refer to v^i as a *neighborhood vector*. We let $V_S(x)$ be a matrix whose j-th column is v^j. We will refer to $V_S(x)$ as the *matrix of neighborhood vectors* at the solution x. Thus, there is one column in $V_S(x)$ for every neighbor in $N_S(x)$.

We now characterize the extended neighborhood.

Theorem 1. *Suppose that* $\mathbf{X}_S = \{(S, c): cA_S \geq 0\}$. *Let* N_S *be any neighborhood for* $\mathbf{X}_S$, *and let* $V_S(x)$ *be the matrix of neighborhood vectors. Then the extended neighborhood of* N_S *is* N_S^*, *and*

$$N_S^*(x) = \{x' \in S: x' = x + V_S(x)\lambda + A_S\gamma\,;\, \lambda \geq 0\,,\, \gamma \geq 0\}.$$

Proof. Let $N_S'(x) = \{x' \in S: x' = x + V_S(x)\lambda + A_S\gamma\,;\, \lambda \geq 0\,,\, \gamma \geq 0\}$. We want to establish that $N_S'(x) = N_S^*(x)$. We first prove that N_S' is LO-equivalent to N_S.

Since $N_S(x) \subseteq N_S'(x)$ for all $S \in \mathbf{X}$ and for all $x \in S$, any locally optimal solution with respect to N_S' is also locally optimal with respect to N_S. So, we now consider the case that x is locally optimal with respect to N, and we will show that x is also locally optimal with respect to N'. Let c denote the cost function. By the definition of local optimality, we know that $c(x' - x) \geq 0$ for each neighbor x' of x, and so $cV_S(x) \geq 0$. Also, by assumption $cA_S \geq 0$. Hence for all $\lambda \geq 0$ and for $\gamma \geq 0$, it follows that

$$c[x + V_S(x)\lambda + A_S\gamma\,] \geq cx,$$

and so for all $x' \in N_S'(x)$, $cx' \geq cx$. Thus a locally optimal solution with respect to N is also a locally optimal solution with respect to N'.

To complete the proof that $N' = N^*$, we will prove that $N'(x)$ is the largest neighborhood structure that is LO-equivalent to N. To this end, we let x^* be any solution in S that is not in $N'(x)$. Since $x^* \in S\backslash N'(x)$, there is no feasible solution to the following linear inequality system:

$$x^* = x + V_S(x)\,\lambda + A_S\,\gamma,\ \lambda \geq 0,\ \gamma \geq 0.$$

It follows from Farkas Lemma that there exists a vector $w \in \mathbb{R}^n$ such that:

$$w(x^* - x') < 0,\ wV_S(x) \geq 0 \text{ and } wA_S \geq 0.$$

This implies that in the instance $(S,\ w)$, x is a locally optimal with respect to N but that x^* is a solution with lower cost. Since $(S,\ w)$ is a feasible instance, it follows that x^* is not in the extended neighborhood of x. Since x^* was an arbitrary element of $S\backslash N'_S(x)$ and since $N'_S(x) \subseteq N^*_S(x)$, it follows that $N' = N^*$, completing the proof. ♦

We state the following two special cases as corollaries, and subsequently give an alternative proof of Proposition 2 in the case that costs are linear. The first special case is one in which all linear objective functions are permitted, and thus $A_S = 0$. When all cost functions are permitted for a combinatorial optimization problem **X**, we say that **X** *has general linear costs*. The second special case is the one in which all non-negative linear objective functions are permitted, in which case A_S is the identity matrix. In this case, we say that **X** *has general nonnegative linear costs.*

Corollary 1. *Suppose that* **X** *is a combinatorial optimization problem with general linear costs, and that* N *is a neighborhood structure defined for* **X***. For each feasible region* $S \in$ **X***, let* $V_S(x)$ *be the matrix of neighborhood vectors. Then the extended neighborhood of* N *is* N*[*], where*

$$N^*_S(x) = \{x' \in S: x' = x + V_S(x)\,\lambda\ ;\ \lambda \geq 0\}.$$

Corollary 2. *Suppose that* **X** *is a combinatorial optimization problem with general nonnegatiave linear costs, and that* N *is a neighborhood structure defined for* **X***. For each feasible region* $S \in$ **X***, let* $V_S(x)$ *be the matrix of neighborhood vectors. Then the extended neighborhood of* N *is* N*[*], where*

$$N^*_S(x) = \{x' \in S: x' \geq x + V_S(x)\,\lambda;\ \lambda \geq 0\}.$$

The previous results follow directly from Theorem 1, and provide a geometric characterization of the extended neighborhood for the combinatorial optimization problem with linear costs defined over a cone. We next prove Proposition 2 in the case that the costs are linear.

Proposition 2′. *Suppose that* **X** *is a combinatorial optimization problem with linear costs, and that* $(S,\ c)$ *is the unique instance in* X_S *for which* $\sum_{i=1}^{n} c_i = 1$. *Let* N_S *be a neighborhood defined*

on **X***, and let* L *be the set of locally optimal solutions for instance* (S, f)*. If* $x \notin L$*, then* $N_S^*(x) =$ S*. If* $x \in L$*, then* $N_S^*(x) = \{y \in S : cy \geq cx\}$*.*

Proof. We prove the result directly from Theorem 1. Let A_S be a matrix such that $wA_S \geq 0$ implies that w is a non-negative multiple of c. Let us first define N_S' as follows. If $x \notin L$, then $N_S'(x) = S$. If $x \in L$, then $N_S'(x) = \{y \in S : f(y) \geq f(x)\}$. By Theorem 1,

$$N_S^*(x) = \{x' \in S: x' = x + V_S(x)\lambda + A_S\gamma \; ; \; \lambda \geq 0 , \gamma \geq 0\}.$$

Assume first that $y \in N_S^*(x)$ and $x \in L$. Then $c(y - x) = cV_S(x)\lambda + cA_S\gamma$ for some $\lambda \geq 0$ and $\gamma \geq 0$. By assumption, $x \in L$, and so $cv \geq 0$ for each column v of $V_S(x)$. Also by assumption, $cA_S \geq 0$. We conclude that $c(y - x) \geq 0$, and so $y \in N_S'(x)$.

Assume next $y \notin N_S^*(x)$ and $x \in L$. By Farkas Lemma there is a vector w such that (1) $w(y - x) < 0$, (2) $w V_S(x) \geq 0$, and (3) $w A_S \geq 0$. The inequality (3) implies that w is a nonnegative multiple of c, and it is a strictly positive multiple because $w(y - x) < 0$. Inequality (2) is a restatement that $x \in L$. Inequality (1) becomes "$c(y - x) < 0$", and so $y \notin N_S'(x)$. We have thus shown that $N_S'(x) = N_S^*(x)$ for $x \in L$.

We now consider the case that $x \notin L$. In this case, $N_S^*(x) \subseteq N_S'(x) = S$. We thus need to establish that $N_S^*(x) = S$. So, we suppose that $y \notin N_S^*(x)$, and we will derive a contradiction. By Farkas Lemma, there is a vector w satisfying (1) to (3) above. As before w is a positive multiple of c. But then condition (2) is equivalent to writing $c(x' - x) \geq 0$ for all $x' \in N_S(x)$, which contradicts that $x \notin L$. This contradiction shows that $y \in N_S^*(x)$, and thus $N_S'(x) = N_S^*(x) = S$ for $x \notin L$ completing the proof. ♦

We now illustrate Theorem 1 result using two examples of neighborhood structures in the literature.

Example 1: A linear program: min $\{cx: x \in P \subseteq \mathbb{R}^n\}$ over a polytope P can be viewed as a combinatorial optimization problem where the set of feasible solutions S is the set of extreme points of the polytope. We assume that all linear objectives are allowed in the linear program, and so we consider Corollary 1. The simplex method for the linear programming can be viewed as a neighborhood search algorithm, where the neighbors of an extreme point x are those extreme points that share an edge with x in the polytope P. It can be shown that for this neighborhood structure, $S \subseteq \{x': x' = x + V_S(x)\lambda, \lambda \geq 0\}$ for all $x \in S$. (This set is any solution that can be expressed as a non-negative linear combination of neighborhood vectors.) Corollary 1 implies that the $N_S^*(x) = S$ for all feasible regions S, and thus the neighborhood is exact, a well known result in linear programming.

Example 2: The traveling salesman problem is to find a minimum cost tour (or Hamiltonian cycle) in a graph $G = (V, E)$. It is often formulated as a combinatorial optimization problem with linear costs as follows. Each tour T is represented as a 0/1 vector $x \in \{0, 1\}^{|E|}$ where $x_e = 1$ means that the edge $e \in$ T. The set of feasible solutions of an instance is given by

$$\mathsf{S} = \{x \in \{0, 1\}^{|E|}: x \text{ represents a tour in G}\}.$$

A 2-opt *move* on a tour T removes two edges from T and adds two edges to T in order to get a new tour. The 2-opt neighborhood of a tour T consists of all tours that can be obtained from T by a 2-opt move. If a tour is represented as a 0-1 vector x, then a 2-opt move can be represented as a vector $v \in \{0, 1, -1\}^{|E|}$, where $v_e = 1$ means that the edge e is added to the cycle, $v_e = -1$ means that the edge is removed from the cycle, and $v_e = 0$ otherwise. From Theorem 1, the extended neighborhood of the 2-opt neighborhood structure contains all tours that can be obtained by non-negative combinations of 2-opt moves.

4. Recognizing Solutions that are in the Extended Neighborhood

In this brief section, we give sufficient conditions under which a solution in the extended neighborhood can be identified in polynomial time.

By the *local improvement problem for* N, we mean the following: given an instance (S, c) of $\mathbf{X}$ and a feasible solution x, determine that x is locally optimal, or else determine a neighbor x^* of x with $cx^* < cx$.

Theorem 2. *Let* $\mathbf{X}$ *be a combinatorial optimization problem with linear costs defined over a cone. Let* N *be a neighborhood structure for* $\mathbf{X}$. *Suppose that the local improvement problem for* N *can be solved in polynomial time and that the separation problem for the cone can be solved in polynomial time. Then there is a polynomial time algorithm to determine the membership in the extended neighborhood* N^*.

Proof. Let S denote a set of feasible solutions, and suppose that the costs are in the cone: $\{c : cA_\mathsf{S} \geq 0\}$. Suppose further that $x, x' \in \mathsf{S}$. By Theorem 1, $x' \in \mathsf{N}_\mathsf{S}^*(x)$ if and only if there is a feasible solution to the following system of inequalities and equalities:

$$x' = x + \mathsf{V}_\mathsf{S}(x)\lambda + A_\mathsf{S}\gamma \, ; \, \lambda \geq 0 \, , \, \gamma \geq 0. \tag{1}$$

By Farkas Lemma, we know that the linear system (1) has no feasible solution if and only if the following linear system has a feasible solution:

$$w(x' - x) < 0; \tag{2a}$$
$$w\mathsf{V}_\mathsf{S}(x) \geq 0; \tag{2b}$$
$$wA_\mathsf{S} \geq 0. \tag{2c}$$

We now claim that feasibility of (2) may be determined in polynomial time using the ellipsoid algorithm. It suffices to show that the separation problem for this linear program is solvable in

polynomial time. Let w' be any solution, not necessarily feasible for (2). If (2a) is not satisfied, then $w(x' - x) \geq 0$ is a separating constraint.

If (2b) is not satisfied, then there exists a column vector v of $V_S(x)$ such that $w'v < 0$. Equivalently, there is a vector $x' \in N_S(x)$ $x' = x + v$ such that $w'x' < w'x$. The separating constraint is $wv \geq 0$. Since we can solve the local improvement problem in polynomial time, we can determine this vector x' in polynomial time, and thus determine v in polynomial time.

If (2c) is not satisfied, then there is a column A_j of A_S so that $wA_j < 0$. By assumption, we can solve the separation problem for this cone in polynomial time. We conclude that we can separate w' in polynomial time, and so we can recognize vectors in $N^*(x)$ in polynomial time. ♦

While recognizing vectors in $N^*(x)$ is solvable in polynomial time, optimizing over $N^*(x)$ is often NP-hard. We prove an NP-hardness result for optimizing over $N^*(x)$ in the next section.

We next study the extended neighborhood for the well-known 2-opt neighborhood structure for the traveling salesman problem.

5. Some Properties of the Extended Neighborhood for TSP 2-opt

In this section and in the following two sections, we analyze some properties of the extended neighborhood 2-opt* of the 2-opt neighborhood structure for the TSP. We show that 2-opt* is exponentially large, and that optimizing over 2-opt* is NP-hard.

If T is a feasible tour with n cities, we let 2-opt(T) be the tours that can be obtained from T by a single 2-opt move, that is the neighbor T′ is obtained from T by adding two edges and deleting two edges. In previous sections, we used the index S when describing the neighborhood function. For the 2-opt neighborhood, the set S is obvious from context since it is the set of all tours on n cities. So, we will omit the index S when discussing the neighborhood 2-opt as well as its extended neighborhood 2-opt*.

Let G = (V, E) be a complete undirected graph with node set V = {1, …, n} and costs c_{ij} associated with each edge $(i, j) \in$ E. The traveling salesman problem (TSP) is to find the minimum weight tour in G. We represent a tour as a sequence $i_1, i_2, …, i_n, i_1$, where edges (i_k, i_{k+1}) for $k = 1, …, n$-1 and (i_n, i_1), belong to the tour. We assume without loss of generality that $i_1 = 1$. Let T be a tour. The *incidence vector* for T is the vector $x \in \{0, 1\}^{|E|}$ where $x_{ij} = 1$ for edges $(i, j) \in$ T and $x_{ij} = 0$ otherwise. The traveling salesman problem can be formulated as a combinatorial optimization problem with linear objective as follows. We define the set of feasible solutions as S = {$x \in \{0, 1\}^{|E|}$: x is incidence vector of some tour in G} and the linear objective function associated with each feasible solution x is $f(x) = cx$.

For some examples, we will assume that the starting tour is T =1, 2, 3, ..., n, 1. In this notationally simpler case, we will represent 2-opt moves using the following concise notation.

We represent a 2-opt move by the unordered pair $\{k, l\}$, where $k, l \in \{1,\ldots, n\}$ such that nodes k and l are not adjacent in the tour, i.e., $k \neq l\text{-}1$ and $k \neq l\text{+}1$. The move $\{k, l\}$ represents the following changes to tour T:

> 1. Remove the edges $(k, k\text{+}1)$ and $(l, l\text{+}1)$,
> 2. Add the edges (k, l) and $(k\text{+}1, l\text{+}1)$.

In this definition, we assume that $n\text{+}1 = 1$. We denote the set of 2-opt moves corresponding to tour T by 2-*optmove*(T). The 2-opt neighborhood for the tour T (solution x) is defined as all the tours (incidence vectors of tours) that can be obtained from T by performing a 2-opt move $\{k, l\}$ for some $\{k, l\} \in$ 2-*optmove*(T). We use $v^{kl} \in \{-1, 0, 1\}^{|E|}$ to denote the neighborhood vector in 2-opt$^V(x)$ corresponding to the 2-opt neighbor obtained by performing the move $\{k, l\}$, i.e.,

$$v^{kl}(e) = \begin{cases} -1 & \text{if } e = (i_k, i_{k+1}) \text{ or } e = (i_l, i_{l+1}) \\ 1 & \text{if } e = (i_k, i_l) \text{ or } e = (i_{k+1}, i_{l+1}) \\ 0 & \text{otherwise.} \end{cases}$$

For example, $x + v^{kl}$ to x gives the incidence vector of the tour obtained from the initial tour T by performing the move $\{k, l\}$.

We consider all possible cost vectors in TSP. We let 2-opt$^*(x)$ denote the extended neighborhood of x. Rather than use the matrix notation of Theorem 1, we represent solutions in the extended neighborhood as x plus linear combinations of neighborhood vectors. In particular,

$$2\text{-opt}^*(x) = \{x' \in S : x' = x + \sum\nolimits_{\{k,l\} \in 2\text{-}optmove(\text{T})} \lambda_{kl} v^{kl}, \; \lambda_{kl} \geq 0 \text{ for all } \{k, l\}\}. \qquad (3)$$

Often, the TSP is restricted to instances where the edge costs c satisfy the *triangle inequality*, that is: for any $i, j, k \in$ V, $c_{ij} + c_{jk} \geq c_{ik}$. We refer to this special case of the TSP as TSPTI. Non-negative cost functions satisfying the triangle-inequality form a cone, and so we may apply the results of Theorem 1, with an appropriately chosen matrix $A(\text{S})$. However, the extended neighborhood is the same as the one given in Theorem 1. We establish this result next.

Theorem 3. *Let* 2-optTI* *denote the extended neighborhood for* 2-opt *for the problem* TSPTI. *Then* 2-optTI* *is the same as* 2-opt*, *the extended neighborhood of 2-opt for TSP.*

Proof. By Proposition 3, every instance of TSPTI is also an instance of TSP, and so the extended neighborhood of TSPTI contains the extended neighborhood of TSP, that is for all feasible solutions x, 2-opt$^*(x) \subseteq$ 2-optTI$^*(x)$. Now let us suppose that x and x' are incidence vectors for tours on n cities, and that $x' \notin$ 2-opt$^*(x)$. Then there is an instance on n cities with cost function c' such that x is locally optimal with respect to the 2-opt neighborhood and such that $c'(x' - x) < 0$. We will next determine another cost vector c^* such that c^* is non-negative, satisfies the triangle inequality, and such that x is locally optimal with respect to c^* and such that $c^*(x' - x) < 0$. This will establish that $x' \notin$ 2-optTI$^*(x)$, which will complete the proof of the theorem.

Let c^M be obtained from c' by adding M to every component. If M is chosen large enough, then c^M will be strictly positive and will satisfy the triangle inequality. Moreover, for any solution y, $c^M(y - x) = c'(y - x)$. It follows that x is locally optimal for cost function c^M and that $c^M(x' - x) < 0$, completing the proof. ♦

Our next result concerns reachability. We say that a tour T′ is reachable from T with respect to neighborhood N if there is a sequence $T = T^1, T^2, \ldots, T^K = T'$ of tours so that

1. Tour $T^j \in N (T^{j-1})$ for j = 2 to K;

2. The cost of T^j is less than the cost of T^{j-1} for j = 2 to K.

We now address the following question: is reachability for 2-opt* the same or different from reachability for 2-opt. Clearly, any tour T′ that is reachable from T with respect to the 2-opt neighborhood is also reachable with respect to 2-opt*. In Theorem 4, we prove that the converse is not true.

Theorem 4. *Let* T *be any tour with at least* 11 *nodes. There is cost vector c and a tour* T′ *such that* T′ *is not reachable from* T *with respect to* 2-opt*, and so that* T′ *is reachable from* T *with respect to* 2-opt*.

Proof. Without loss of generality, we may relabel the nodes so that $T = (1, 2, \ldots, n, 1)$. Suppose that all edges of T have a cost of 0, that is, $c_{i,i+1} = 0$ for i = 1 to n, and $c_{n1} = 0$. Suppose further that arcs (1,5), (2,6), (3,8), and (4,9) all have a cost of –2, arcs (6, 10) and (7, 11) have a cost of 1, and all other arcs have a cost of 10. The optimal tour is T′ = 1, 5, 4, 9, 10, 6, 2, 3, 8, 7, 11, 12, 13,… n, 1, which has a cost of –6. It can be obtained from T by performing the 2-opt moves {1,5}, {6, 10}, and {3,8} in that order. (Recall that move {i, j} adds arcs (i, j) and (i+1, j+1) and deletes arcs (i, i+1) and (j, j+1).) The sum of these three 2-opt moves is in 2-opt*(T), and thus T′ is (trivially) reachable from T with respect to 2-opt*. However, T′ is not reachable from T with respect to 2-opt. Indeed, if one performs either the move {1,5} or {3,8} on T, the resulting tour is locally optimal. ♦

The previous example permitted negative costs, and violated the triangle inequality. By adding 12 to each arc cost, one obtains an example that has positive costs and does satisfy the triangle inequality.

A consequence of Theorem 4 is that local search using the extended neighborhood fundamentally offers more possibilities than does local search using the original neighborhood. It is possible that there are locally optimal solutions that are reachable from the initial tour T using the extended neighborhood, but that are not reachable from T using the original neighborhood.

We next analyze the size of the extended neighborhood of a solution.

6. A Lower Bound on the Size of 2-opt[*]

Our original motivation for studying extended neighborhoods was in the context of very large scale neighborhood search. We hypothesized that neighborhoods with very large extended neighborhoods might reach better local optima. Whether this hypothesis is true awaits careful empirical analysis. However, it does suggest that the size of the extended neighborhood is an important parameter.   In this section, we give a lower bound on the size of 2-opt[*].

We note that the independent 2-opt neighborhood structure proposed by Potts and Velde [1995] is LO-equivalent to the 2-opt neighborhood structure, and is thus a subset of 2-opt[*]. They established the size of the independent 2-opt neighborhood structure for problems with n nodes to be $\Omega(1.75^n)$, it follows that the size of 2-opt[*] is $\Omega(1.75^n)$. In this section, we show for problems with at least n nodes that the size of the 2-opt[*] neighborhood of a solution is at least $h(n) = (n-3)(\lceil n/2 \rceil -3)!$, which for n > 6 is at least $\lceil n/2 - 2 \rceil!$ We shall establish this result by constructing a tree with at least $h(n)$ nodes such that each node of the tree is a tour in 2-opt[*](T). Our enumeration tree satisfies the following properties:

1. The root of the tree is T.

2. Each node of the tree is in the 2-opt[*] neighborhood of T.

3. If T′ is a node in the tree then the children of T′ are obtained from T′ by performing a 2-opt move that is also valid for T.

4. There are at least $h(n)$ nodes in the tree.

We describe the construction of the tree later in the section after introducing some preliminary results. Without loss of generality, we assume that T is given by the sequence 1, 2, 3, …, n, 1. We again represent 2-opt moves for tree T using an unordered pair $\{i, j\}$ of nodes such that $j \notin \{i$-1, i+1$\}$. Recall that the pair $\{i, j\}$ represents the following change to the tour T:

1. Remove edges $(i, i$+1$)$ and $(j, j$+1$)$.

2. Add edges (i, j) and $(i$+1, j+1$)$.

Let T′ be any tour represented as a sequence. If node k+1 immediately follows node k in T′, we say that edge $(k, k$+1$)$ is a *forward edge* of T′. If node k immediately follows node k+1 in T′, we say that $(k, k$+1$)$ is a *backward edge* of T′. (For convenience, we let both 1 and n+1 refer to node 1.)

In our construction below, T′ will be tour obtained from T by a sequence of 2-opt moves, each of the form $\{j, k\}$ for some j and k. We will say that the move $\{j, k\}$ is *feasible* for tour T′ if T′ $- \{(j, j$+1$), (k, $k+1$)\} \cup \{(j, $k+1$), (j$+1, $k)\}$ is a tour. The following lemma gives necessary and sufficient conditions for a move $\{j, k\}$ to be feasible with respect to a tour T′.

Lemma 2. *Let* T′ *be a tour. The move* $\{j, k\}$ *is feasible for* T′ *if and only if* $(j, j+1)$ *and* $(k, k+1)$ *are both forward edges of* T′ *or* $(j, j+1)$ *and* $(k, k+1)$ *are both backward edges of* T′.

Proof. Suppose first that $(j, j+1)$ and $(k, k+1)$ are both forward edges of T′. Let us write T′ as i_1, i_2, …, i_r, i_{r+1}, …, i_n, where $i_1 = j$, $i_2 = j+1$, and $i_r = k$. In this case the move $\{j, k\}$ is feasible, and results in the tour i_1, i_r, i_{r-1}, i_{r-2}, …, i_2, i_{r+1}, i_{r+2}, …, i_n. Similarly, the move $\{j, k\}$ is feasible if $(j, j+1)$ and $(k, k+1)$ are both backward edges of T′. This establishes the "if part" of the lemma.

We now consider the only if part. If $(j, j+1)$ or $(k, k+1)$ is not an edge of T′, then clearly $\{j, k\}$ is not feasible. So, the remaining cases are when $(j, j+1)$ is a forward edge and $(k, k+1)$ is a backward edge or when $(j, j+1)$ is a backward edge and $(k, k+1)$ is a forward edge. Let us consider the case when $(j, j+1)$ is a forward edge and $(k, k+1)$ is a backward edge. Again, let T′ be written as i_1, i_2, …, i_r, i_{r+1}, …, i_n, where $i_1 = j$, $i_2 = j+1$, and $i_r = k$. In this case, deleting edges $(j, j+1)$ and $(k, k+1)$ from T′ and adding edges (j, k) and $(j+1, k+1)$ creates two subtours: i_2, i_3, …, i_{r-1}, i_2 and i_r, i_{r+1}, …, i_n, i_1, i_r, and is thus not a feasible move. Similarly, when $(j, j+1)$ is a backward edge of T′ and $(k, k+1)$ is a forward edge, then $\{j, k\}$ is not a feasible move, establishing the only if part of the lemma. ♦

We will now constructively prove that 2-opt*(T) has at least h(n) elements when T is a tour on n cities. We do so by creating a tree with at least h(n) vertices, where the root vertex of the tree corresponds to tour T, and where each non-root vertex of the tree corresponds to a tour in 2-opt*(T). (We are referring vertices of the tree so as to distinguish the terminology from the nodes of the tour.) We assume without loss of generality that T = 1, 2, …, n, 1. Moreover, in our construction, each child vertex in the tree is obtained from its parent vertex by a move of the form $\{j, k\}$ for suitable choices of $\{j, k\}$.

For any tour T′, we let F(T′) denote the set of edges $(i, i+1)$ (where $n+1$ represents 1) in T′ that are forward edges of the tour. We let B(T′) denote the set of edges $(i, i+1)$ in T′ that are backward edges of T′. The initial tour is T = 1, 2, 3, …, n, 1, and F(T) is the set of edges of T, and B(T) = ∅. At intermediate stages of the algorithm, we say that a tour T′ of the tree is *eligible* if T′ is a tour such that $|F(T')| \geq 3$ or $|B(T')| \geq 3$, and if T′ has no children. We are now ready to state our algorithm to generate a tree of tours. Figure 1 gives the statement of our algorithm.

The tree contains the tour T as the root vertex. The root vertex has $(n-3)$ children that are obtained using the $(n-3)$ moves $\{1, j+2\}$ for $j = 1, 2, …, n-3$, on T. For each node T′ ≠ T, the children are generated as follows. If $|F(T')| \geq |B(T')|$ then we find an forward edge $(i, i+1)$ in the sequence of T′ such that $(i-1, i)$ is not an edge, and generate the children of T′ as tours obtained by performing those 2-opt moves $\{i, j\}$ of T on T′ that are feasible for T′. (We note that if $(i, i+1)$ is a forward edge of T′, then $(i-1, i)$ cannot be a backward edge. So, Step 7 is equivalent to determining an index i such that $(i-1, i)$ is not an edge in T′ but $(i, i+1)$ is. Since $|F(T')| < n$, such an index i always exists.) If $|F(T')| < |B(T')|$ then we find a backward edge $(i, i+1)$ such that $(i-1,$

i) is not a (backward) edge in T′. In this case, we again obtain children of T′ by performing those 2-opt moves $\{i, j\}$ of T on T′ that are feasible for T′.

Procedure *Tree-of-Tours*;
begin
1.　　　Let T be the root vertex of the tree;
2.　　　Let the j-th child of T be the tour obtained from T by performing the move $\{1, j+2\}$ for j
　　　　　 $= 1,…, n$-3;
3.　　　**while** there is an eligible tour T′ in the tree **do**
4.　　　**begin**
5.　　　　　**if** $|F(T′)| \geq |B(T′)|$ **then do**
6.　　　　　**begin**
7.　　　　　　 select a node i such that $(i, i+1) \in F(T′)$ and such that $(i$-$1, i) \notin T′$;
8.　　　　　　 for each node j such that $\{i, j\}$ is a feasible move for T′, create a child of T′ by
　　　　　　　　 performing move $\{i, j\}$;
9.　　　　　　 – Note: the number of children of T′ created in this way is at least $|F(T′)|$ - 2
10.　　　　**end**
11.　　　　**else begin**
12.　　　　　 select a node i such that $(i, i+1) \in B(T′)$ and such that $(i$-$1, i) \notin T′$;
13.　　　　　 for each node j such that $\{i, j\}$ is a feasible move for T', create a child of T′ by
　　　　　　　　 performing move $\{i, j\}$;
14.　　　　　 – Note: the number of children of T′ created in this way is at least $|B(T′)|$ - 2
15.　　　　**end**
16.　　**end;**
end.

Figure 1. Procedure to generate tree of some tours in 2-opt[*].

Theorem 5. *The tree generated by procedure Tree-of-Tours has at least $h(n) = (n-3)(\lceil n/2 \rceil - 3)!$ distinct vertices when applied to a* TSP *with n nodes. Moreover, each non-root vertex of the tree corresponds to a tour in* 2-opt[*](T)*.*

Proof. First of all, all of the non-root vertices of the tree correspond to feasible tours because they are obtained from their parent by a feasible 2-opt move of the form $\{j, k\}$, which deletes edges $(j, j+1)$ and $(k, k+1)$ from a tour and adds edges (j, k) and $(j+1, k+1)$. Moreover, each tour is in 2-opt[*](T) since it can be expressed as T plus the sum of 2-opt moves of the form $\{j, k\}$ (This follows from Theorem 1). We also note that for any tour T′ in the tree, the descendents of T′ in the tree are obtained from T′ by a sequence of feasible 2-opt moves.

Observation 1. If a tour T′ of the tree does not have edge $(j, j+1)$ then neither does any of its descendents in the tree. The observation is true because no move can add back $(j, j+1)$.

Observation 2. If a tour T′ of the tree has edge (j, k) for $k \neq j$-1 or $j+1$, then each descendent of the tour T′ also has the edge (j, k). The observation is true because no move can delete edge (j, k).

　　　　To complete the proof, we need to establish that the tours in the tree are all distinct, and that there are at least $h(n)$ tours in the tree. We next show that any two distinct vertices in the tree

correspond to distinct tours. Let T^1 and T^2 be tours corresponding to two distinct vertices in the tree generated by *Tree-of-Tours*. Let T^3 be the tour corresponding to the vertex that is the least common ancestor of the vertices for T^1 and T^2 in the tree. Let T^4 be the child of T^3 whose descendent is T^1. (Possibly $T^4 = T^1$). Let T^5 be the child of T^3 whose descendent is T^2. (Possibly $T^5 = T^2$). Figure 2 illustrates the situation.
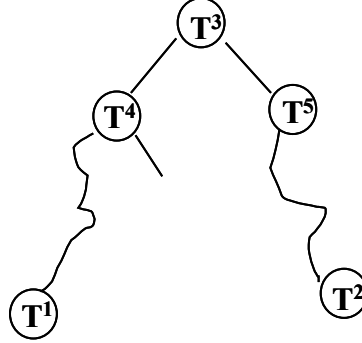

Figure 2. T^3 is the common ancestor of T^1 and T^2.

In creating children for T^3 in the procedure "Tree of Tours" a node i is selected either in Step 7 or Step 11 and all children of T^3 are obtained by performing a move $\{i, j\}$ for some j. In either case $(i\text{-}1, i)$ is not an edge of T^3. Let (i, r) and $(i, i+1)$ denote the two edges of T^3 that are incident to node i.

Suppose that the tour T^4 is obtained from the tour T^3 by performing the move $\{i, j\}$, and that the tour T^5 is obtained from the tour T^3 by performing the move $\{i, k\}$. Then T^4 and all of its descendents (including T^1) contain the edges (i, j) and (i, r), whereas T^5 and its descendents (including T^2) contain the edges (i, k) and (i, r). Since $k \neq j$, it follows that tours T^1 and T^2 are distinct.

We next establish that the tree has at least h(n) vertices. For a vertex T' of the tree, each child of T' is obtained by performing a 2-opt move $\{j, k\}$ for some j and k, obtaining a child T''. This deletes edges $(j, j+1)$ and $(k, k+1)$ from T', and so $|T'' \cap T| = |T' \cap T| - 2$. It is easy to establish inductively that for a vertex T' at level l in the tree (where T is at level 0), $|F(T')| + |B(T')| = n - 2l$. The number of children generated for the node T' is at least max $\{|F(T')|, |B(T')|\} - 2 \geq \lceil n/2 \rceil - l - 2$. The number of children of the root node is $(n - 3)$. Thus the number of leaf nodes in the tree is at least $h(n) = (n-3)(\lceil n/2 \rceil - 3)!$. Hence the total number of nodes in the tree is at least h(n) as well. This completes the proof. ◆

7. The Complexity of Optimizing Over 2-opt[*]

By definition of LO-equivalence and the extended neighborhood, the 2-opt neighborhood of a tour contains a better solution if and only if the 2-opt[*] neighborhood of the tour contains a better solution. Therefore, the complexity of finding a better solution in the 2-opt[*] neighborhood is the same as that for the 2-opt neighborhood, and this requires $O(n^2)$ time. In this section, we

consider the problem of determining an optimal solution in 2-opt[*] and show that the problem of optimizing over the extended neighborhood is strongly NP-hard. More precisely, we formulate a decision version of the optimization problem, and show that the decision version is NP-complete.

2-Opt[*] Search Problem

INPUT: An undirected complete graph $G = (V, E)$, where $V = \{1,\ldots, n\}$, an integer K, and integer edge costs c_{ij} for $(i, j) \in E$, and a tour T.

QUESTION: Is there a tour $T' \in$ 2-opt[*](T) such that $\sum_{(i,j)\in T'} c_{ij} \leq K$?

Our proof that the 2-opt[*] Problem is NP-complete will rely on a transformation from the Hamiltonian path problem, as well as on some properties of the neighborhood 2-opt[*](T). We next prove a lemma that is needed for the NP-completeness proof.

Lemma 3. *Let $n = 4p$ for some $p > 1$. Suppose that* T *is the tour* T $= 1, 2, 3, \ldots, n, 1$. *For any ordering P of the nodes $\{3, 7,\ldots, 4r\text{-}1,\ldots,4p\text{-}1\}$ there exists a tour* T' $\in$ 2-opt[*](T) *such that 1-2-P is the subpath of the first $2 + n/4$ nodes of* T'.

Proof. We establish this lemma constructively using an approach similar to that used in constructing the tree of tours. We first divide the set of edges in T into p groups of four consecutive edges each, where the rth group is $E^r =\{(4r\text{-}3, 4r\text{-}2), (4r\text{-}2, 4r\text{-}1), (4r\text{-}1, 4r), (4r, 4r+1)\}$. Note that E^r has 5 nodes, and the middle node is $4r$-1. Let $P = (4r_1\text{-}1), (4r_2\text{-}1) , \ldots , (4r_p\text{-}1)$ be an arbitrary ordering of the middle nodes of these p groups. We will construct a tour T' in 2-opt[*] such that $1 - 2 - P$ are the first $p+2$ nodes of T'.

More specifically, the procedure *Construct-Tour* given below in Figure 3 will create a sequence $T^0, T^1, T^2, T^3, \ldots, T^p$ of tours with the following properties:

1. $T^0 = T$,

2. For $k = 1$ to p, the first two nodes in T^k are 1 and 2; the j-th node is $(4r_{j-2} - 1)$ for $j = 3$ to $k+2$, and the node in position $k+3$ is either $4r_k$ or $(4r_k - 2)$; in the former case $(4r_k - 1, 4r_k) \in F(T^k)$; in the latter case $(4r_k - 1, 4r_k - 2) \in B(T^k)$;

3. $E^r \subseteq T^k$ for all $1 \leq k < r \leq p$;

4. T^k is in 2-opt[*](T) for $k = 1$ to p.

Property 3 is needed within the proof itself. If we establish that properties 1, 2 and 4 are satisfied, then $T' = T^p$ satisfies the conditions of the lemma, and the lemma is proved.

It remains to show that properties 1-4 are satisfied. They are satisfied for T^1. We now assume that they are satisfied for T^k and iteration k, and establish that they are also satisfied at iteration k+1 and for T^{k+1}. We consider the four cases in the "for loop" of the procedure separately.

Procedure *Construct-Tour*;
begin
    if $r_1 = 1$, **then** $T^1 = T$;
    else T^1 is obtained from T by performing the 2-opt move $\{2, 4r_1-1\}$;
    for k = 1 to p-1 **do**
        if $(4r_k - 1, 4r_k) \in F(T^k)$ **and if** $E^{k+1} \subseteq F(T^k)$, **then** T^{k+1} is obtained from T^k by performing
            the 2-opt move $\{4r_k - 1, 4r_{k+1} - 1\}$;
        else if $(4r_k - 1, 4r_k) \in F(T^k)$ **and if** $E^{k+1} \subseteq B(T^k)$, **then** T^{k+1} is obtained from T^k by first
            performing the 2-opt move $\{4r_{k+1} - 3, 4r_{k+1}\}$, and subsequently performing the 2-opt
            move $\{4r_k - 1, 4r_{k+1} - 1\}$;
        else if $(4r_k - 1, 4r_k - 2) \in B(T^k)$ **and if** $E^{k+1} \subseteq B(T^k)$, **then** T^{k+1} is obtained from T^k by
            performing the 2-opt move $\{4r_k - 2, 4r_{k+1} - 2\}$;
        else if $(4r_k - 1, 4r_k - 2) \in B(T^k)$ **and if** $E^{k+1} \subseteq F(T^k)$, **then** T^{k+1} is obtained from T^k by first
            performing the 2-opt move $\{4r_{k+1} - 3, 4r_{k+1}\}$ and subsequently performing the 2-opt
            move $\{4r_k - 2, 4r_{k+1} - 2\}$;
end.

Figure 3. Procedure to generate trees T^k, $k = 1, …, p$.

Case 1. $(4r_k - 1, 4r_k) \in F(T^k)$ and $E^{k+1} \subseteq F(T^k)$. In this case, the sequence for T^k starts with $1 - 2 - (4r_1-1) - (4r_2-1) -... - (4r_k-1) - 4r_k$ by Property 2. By assumption, $(4r_{k+1}-1, 4r_{k+1}) \in F(T^k)$, and so the 2-opt move $\{4r_k-1, 4r_{k+1}-1\}$ is feasible for T^k, and applying this move to T^k yields a new tour $T^{k+1} \in$ 2-opt*(T). The move $\{4r_k-1, 4r_{k+1}-1\}$ on T^k removes edges $(4r_k-1, 4r_k)$, $(4r_{k+1}-1, 4r_{k+1})$ and adds the edges $(4r_k-1, 4r_{k+1}-1)$, $(4r_k, 4r_{k+1})$ to T^k. Hence $(4r_{k+1}-2, 4r_{k+1}-1) \in B(T^{k+1})$ Property 2 remains satisfied. Property 3 remains satisfied after the move because no edge in E^r, $r > k+1$ is deleted by the move.

Case 2. $(4r_k - 1, 4r_k) \in F(T^k)$ and $E^{k+1} \subseteq B(T^k)$. In this case, the sequence for T^k starts with $1 - 2 - (4r_1-1) - (4r_2-1) -... - (4r_k-1) - 4r_k$ by Property 2. The algorithm then performs the 2-opt move $\{4r_{k+1}-3, 4r_{k+1}\}$ on T^k, creating an intermediate tour T". It follows that T" is in 2-opt*(T) and also satisfied Properties 2 and 3. The edge $(4r_{k+1}-1, 4r_{k+1}) \in F(T")$, and so the 2-opt move $\{4r_k-1, 4r_{k+1}-1\}$ is feasible for T". Applying this move to T" yields a new tour $T^{k+1} \in$ 2-opt*(T). Just as shown in Case 2, Properties 2 and 3 remain satisfied after the move.

Case 3. $(4r_k-1, 4r_k-2) \in B(T^k)$ and $E^{k+1} \subseteq B(T^k)$. In this case, the sequence for T^k starts with $1 - 2 - (4r_1-1) - (4r_2-1) -... - (4r_k-1) - 4r_k-2$ by Property 2. Since $(4r_k-2, 4r_k-1) \in B(T^k)$ and $(4r_{k+1}-2, 4r_{k+1}-1) \in B(T^k)$, we can apply the 2-opt move $\{4r_k-2, 4r_{k+1}-2\}$ to T^k, obtaining a tour $T^{k+1} \in$ 2-opt*(T). Using argument similar to Case 1, it is easy to see that T^{k+1} satisfied Property 2 and Property 3.

Case 4. $(4r_k-1, 4r_k-2) \in B(T^k)$ and $E^{k+1} \subseteq F(T^k)$. In this case, the sequence for T^k starts with $1 - 2 - (4r_1-1) - (4r_2-1) -... - (4r_k-1) - 4r_k-2$ by Property 2. The algorithm then performs the 2-opt move $\{4r_{k+1}-3, 4r_{k+1}\}$ on T^k, creating an intermediate tour T". It follows that T" is in 2-opt*(T) and also satisfied Properties 2 and 3. Since $(4r_k-2, 4r_k-1) \in B(T")$ and $(4r_{k+1}-2, 4r_{k+1}-1) \in B(T")$, we can apply the 2-opt move $\{4r_k-2, 4r_{k+1}-2\}$ to T", obtaining a tour $T^{k+1} \in$ 2-opt*(T). Using argument similar to Case 1, it is easy to see that T^{k+1} satisfied Property 2 and Property 3.

Based on our construction, the tour T^p satisfies the property claimed in the lemma. ♦

To establish the NP-completeness of the 2-opt* search problem, we will rely on the construction in Lemma 3, and carry out a transformation from the Hamiltonian Path Problem.

Hamiltonian Path Problem

INPUT: An undirected graph $G = (V, E)$, where $V = \{1,\ldots, n\}$.

QUESTION: Is there a Hamiltonian path from node 1 to node n in G?

This problem is known to be NP-complete (Garey and Johnson [1979]).

Theorem 6. *The* 2-opt* *Search Problem is NP-Complete.*

Proof. The 2-opt* Search Problem is in the Class NP by Theorem 2. The rest of our proof will rely on a transformation from the Hamiltonian Problem as well as Lemma 3. Let $G = (V, E)$ be an instance of the Hamiltonian Path problem. For notational convenience, we will assume that the n nodes of V are labeled 3, 7, …, $4n$-1, and the question is whether there is a Hamiltonian path in G from node 3 to node $4n$-1. We now create a complete graph $G' = (V', E')$ with node set $V' = \{1, 2, \ldots, 4n\}$ and let T be the tour $1 - 2 - 3 - \ldots - 4n - 1$. Using our notation, we may view V as a subset of V'. We construct the edge weights c for G' as follows:

1.   If $i \in V \setminus \{3, 4n\text{-}1\}$ and $j \in V' \backslash V$, then $c_{ij} = 1$

2.   If $i \in V$ and $j \in V$ and $(i, j) \notin E$, then $c_{ij} = 1$.

3.   For all other arcs $c_{ij} = 0$. (E' is a complete graph).

We now claim that G has a Hamiltonian path from node 3 to node 4n-1 if and only if there is a tour T' in 2-opt*(T) such that $\sum_{(i,j) \in T'} c_{ij} = 0$.

Suppose first that there is a Hamiltonian Path P from node 3 to node $4n$-1 in G. By Lemma 3, there is a tour T' in 2-opt*(T) such that the first n+2 nodes of T' are 1-2-P. All nodes following P are in V'\V. Regardless of how the remaining nodes in T' are ordered, the cost of T' is 0 by our construction of the edge weights.

We now consider the case that there is a tour T' in 2-opt*(T) with cost 0. Since the first and last nodes of T' are in V'\V, there are at least two nodes of V are incident to nodes of V'\V in T'. However, $c_{ij} = 1$ for $i \in V\backslash\{3, 4n\text{-}1\}$ and $j \in V'\backslash V$. Since none of these arcs can be in T' (which has a cost of 0), we conclude that exactly two nodes of V are incident to nodes of V'\V and these nodes are 3 and 4n-1. This means that the nodes in V are consecutive in T'. Let P denote the subpath formed by nodes in set V in tour T'. We now claim that P is Hamiltonian Path in G. To see this, note that any arc of P that is not in E must have a cost of 1 in G'. This completes the proof. ♦

Although it is NP-hard to optimally search the 2-opt* neighborhood, there are some known large subsets of the 2-opt* neighborhood that are searchable in polynomial time. These include the independent 2-opt neighborhood of Potts and Velde [1995] and the subsets of the twisted neighborhood structure of Aurenhammer [1988].

8. The Extended Neighborhood for Graph Partition Problem

In the two examples provided in Section 3, the extended neighborhoods are much larger than the neighborhood structures themselves. However, for some combinatorial optimization problems, it can be shown that the extended neighborhood is always equal to the neighborhood structure itself. In this section, we consider the graph partition problem given below.

Graph Partition Problem

INPUT:　　　A complete undirected graph $G = (V, E)$ containing $2n$ nodes and weights w_{ij} associated with each edge $(i, j) \in E$.

OBJECTIVE: Partition the set of nodes V into two subsets V^1 and V^2 such that $|V^1| = |V^2|$ and the weight of edges crossing the partition, $\sum_{\{(i,j):i\in V^1 \text{and } j\in V^2\}} w_{ij}$, is minimum.

The Graph Partition problem is a widely studied problem, including several papers devoted to neighborhood search. It is known to be NP-hard (Garey and Johnson [1979]). The most popular algorithms for the Graph Partition problem are based on the search heuristics of Kernighan and Lin [1970] and Fiduccia and Mathias [1982]. These search heuristics employ a variable depth search procedure to identify a new solution starting from a solution to the Graph Partitioning. We note that there is no neighborhood structure associated with each solution in these search heuristics. Johnson et al. [1989] performed detailed computational experiments with the search heuristic of Kernighan and Lin and a simulated annealing based neighborhood search algorithm. They concluded that although the neighborhood search based algorithm worked better on some instances, the heuristic of Kernighan and Lin performed well overall.

The main result of this section is that for any neighborhood function N of the graph partition problem, the extended neighborhood $\mathsf{N}^* = \mathsf{N}$. Our result relies on an important property of the convex hull of solutions for the graph partition problem (when represented as an integer program): all feasible partitions are adjacent corner points in the convex hull.

Because the graph partition problem is a special case of the quadratic assignment problem, this result applies to the quadratic assignment problem as well.

Before proving the result for the Graph Partition Problem, we establish a more general theorem concerning the extended neighborhood.

Let $\mathsf{P} = Conv(\mathsf{S})$ denote the convex hull of the feasible solution set of a combinatorial optimization problem. In order to show the next result we assume that every feasible solution in S is an extreme point of P. This property is automatically satisfied if $\mathsf{S} \subseteq \{0, 1\}^n$.

In the following theorem, we refer to adjacency in a polytope in the usual linear programming sense of adjacency. That is, two corner points x and y are adjacent if one can obtain y from x by a single linear programming pivot.

Theorem 7. *Let* **X** *be a combinatorial optimization problem with general linear costs. Suppose for every feasible region* $S \in \mathbf{X}$*, the following is true*:

(1) *the set of corner points of Conv*(S) *is the set* S.

(2) *every two corner points of the polytope* $P = Conv(S)$ *are adjacent;*

Then for any neighborhood structure N *defined on* **X***,* $N^* = N$.

Proof. Let $S \in \mathbf{X}$, and let us suppose that $N_S(x) \neq N_S^*(x)$ for some $x \in S$. Let x' be a solution in $N^*(x) \backslash N_S(x)$. By hypothesis x and x' share a common edge in the polytope. It is well known (see for example Papadimitriou and Steiglitz [1982]) that in this case there must exist a cost vector w satisfying: $wx' < wx$ and $wx \leq wx''$ for all $x'' \in S \backslash \{x, x'\}$. However, in the problem instance (S, w), the solution x is locally optimal with respect to N but not with respect to N^*, which is a contradiction. Hence it must be the case that $N^*(x) = N(x)$ for all $x \in S$. ♦

The Graph Partition Problem can be formulated as an integer programming problem as follows: We represent each node partition (V^1, V^2) as a vector $x \in \{0, 1\}^{|E|}$ such that $x_{ij} = 1$ if $i \in V^1$ and $j \in V^2$ and $x_{ij} = 0$ otherwise. Thus $S = \{x \in \{0, 1\}^{|E|}: x$ represents a node partition$\}$. The Graph Partition problem is Minimize $(wx : x \in S)$. We will next show that every two corner points in *Conv*(S) are adjacent. Therefore, the extended neighborhood of any neighborhood structure for the Graph Partition problem is the same as the neighborhood structure for all instances of the problem and for any neighborhood structure.

Theorem 8. *Let* $S = \{x \in \{0, 1\}^{|E|}: x$ *represents a node partition of* $G = (V, E)\}$. *Then every two corner points in Conv*(S) *are adjacent in Conv*(S).

Proof. Let S denote the set of feasible 0-1 solutions of the Graph Partition problem. Let x be a solution representing the node partition (V^1, V^2). Let $x' \in S$ be any other solution, and let $(\hat{V}^1, \hat{V}^2)$ denote its partition. We shall construct a weight function w satisfying: $wx' = wx$ and $wx < wx''$ for all $x'' \in S \backslash \{x, x'\}$. The existence of such a weight function implies that x and x' share an edge in the polytope *Conv*(S), proving the theorem.

Let $A = V^1 \cap \hat{V}^1$, $B = V^2 \cap \hat{V}^2$, $C = V^1 \cap \hat{V}^2$, and $D = V^2 \cap \hat{V}^1$. Therefore, $V^1 = A \cup C$, $V^2 = B \cup D$, $\hat{V}^1 = A \cup D$, and $\hat{V}^2 = B \cup C$. The partition $(\hat{V}^1, \hat{V}^2)$ can be obtained from (V^1, V^2) by moving the nodes in C from V^1 to V^2 and nodes in D from V^2 to V^1. We note that the following must hold: $|A| = |B|$ and $|C| = |D|$. We assign the weights in the graph as follows:

1. $w_{ij} = M$ if $i, j \in A$ or $i, j \in B$ or $i, j \in C$ or $i, j \in D$, where $M > |V|^2$.

2. $w_{ij} = 1$ if $i \in A \cup B$ and $j \in C \cup D$.

3. $w_{ij} = 0$ for all other edges.

By our construction, $wx = |A|\,|D| + |B|\,|C|$, and $wx' = |A|\,|C| + |B|\,|D|$. Since $|A| = |B|$ and $|C| = |D|$, it follows that $wx = wx' = 2\,|A|\,|C| < |V|^2$. We now claim that any other partition $x'' = (W^1, W^2)$ has a higher cost. If there is an edge (i, j) with $i \in A \cap W^1$ and $j \in A \cap W^2$, then $w_{ij} = M > wx$. We conclude that if $wx'' < M$, then $A \subseteq W^1$ or $A \subseteq W^2$. Similarly for $S' = B$ or C or D, $S' \subseteq W^1$ or $S' \subseteq W^2$. Without loss of generality, we assume that $A \subseteq W^1$. This leads to only one other possibilities other than x and x'. It follows that $W^1 = A \cup B$. (If $W^1 = A$, it would not be a partition, nor would it be a partition if W^1 were larger than $A \cup C$.) Since $W^1 = A \cup B$, it follows that $wx'' = |A|\,|C| + |A|\,|D| + |B|\,|C| + |B|\,|D| = 4\,|A|\,|C| > wx$. This proves the theorem. ♦

Putting together the results of Theorems 7 and 8 yields the following theorem.

Theorem 9. *Suppose that all cost vectors are feasible for the Graph Partition Problem. Then for any neighborhood structure* N *for the Graph Partition Problem,* $\mathsf{N}^*(x) = \mathsf{N}(x)$ *for all* $x \in \mathsf{S}$.

We observe that we could restrict attention to costs that are non-negative since adding a constant M to every edge adds the constant value $|V/2|^2$ to every partition. We have also assumed that every instance of the graph partition problem is complete. We can relax this assumption to permit problems in which the graph $G = (V, E)$ is not complete. If E is not complete, then we can create an equivalent problem in which G is complete by assigning a weight of 0 to arcs not in E.

9. Conclusions

In this paper, we have introduced the concepts of LO-equivalence of neighborhoods and of the extended neighborhood of a neighborhood structure for the combinatorial optimization problems. These concepts are motivated in part by the study of the independent 2-opt neighborhood for the TSP, which is exponentially large even though it is LO-equivalent to the 2-opt neighborhood. It is also motivated in part by the concept of dominance as per Glover and Punnen. It is also motivated in part as a generalization of "exactness" of neighborhoods.

The size of the extended neighborhood provides an alternative metric for a neighborhood structure. A small neighborhood structures can appear to be very large scale if its extended neighborhood is of exponential size.

We showed in the case of combinatorial optimization problems with linear costs defined over a cone that there is a geometric characterization of the extended neighborhood for any neighborhood structure. We used this geometric characterization to study the size and other properties for the extended neighborhood for the 2-opt neighborhood structure for TSP. We showed that the number of neighbors of each solution in 2-opt$^*(x)$ is at least $((n\text{-}4)/2)!$. We also showed that the 2-opt* neighborhood differs from the 2-opt neighborhood in important ways other

than its size. We showed that optimizing over the 2-opt* neighborhood is NP-hard. We also showed that there are possibly local optima reachable from a tour T with respect to the 2-opt* neighborhood that are not reachable with respect to the 2-opt neighborhood.

We showed that for certain problems such as Graph Partition problem, the extended neighborhood is always equal to the neighborhood structure. This relied on a theorem stating that all corner points in the convex hull of partitions are adjacent.

There is an interesting related property of VLSN search for the TSP and quadratic assignment problem. Deineko and Woeginger [1997] point out that there are many known exponentially large neighborhoods of the traveling salesman problem that can be searched in polynomial time, but that there is no known exponentially large neighborhood of the quadratic assignment problem that can be solved in polynomial time. We mention this related property because the quadratic assignment problem contains the graph partition problem as a special case, and we wonder whether our results in Section 8 and their observations are connected.

We note that our definition of the extended neighborhood of a neighborhood structure is dependent implicitly on the class of objective functions that are allowed for a problem. In the future, it may be worthwhile to analyze the extended neighborhoods for specific neighborhood structures as well as restricted classes of objective functions.

References

Aarts, E. M. L., and J. K. Lenstra. 1997. *Local search in Combinatorial Optimization*. John Wiley.

Ahuja, R. K., O. Ergun, J. B. Orlin, and A. P. Punnen. 2001. A Survey of Very Large-Scale Neighborhood Search Techniques. *Discrete Applied Mathematics* **23**, p75-102.

Deineko, V. G., and G. J. Woeginger. 2000. A study of exponential neighborhoods for the traveling salesman problem and the quadratic assignment problem. *Mathematical Programming* **87**, p519-542.

Ergun, O. 2001. New Neighborhood Search Algorithms Based on Exponentially Large Neighborhoods. PhD Thesis, Operations Research Center, MIT, Cambridge, Massachusetts, USA.

Fiduccia, C. M., and R. M. Mattheyes. 1982. A linear time heuristic for improving network partitions. *ACM IEEE Nineteenth Design Automation Conference Proceedings*, IEEE Computer Society, p175-181.

Garey, M., and D. S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman.

Glover, F., G. Gutin, A. Yeo, and A. Zverovich. 2001. Construction heuristics for the asymmetric TSP. *European Journal of Operational Research* **129**, p555-568.

Glover, F., and A. P. Punnen. 1997. The Traveling Salesman Problem: New Solvable Cases and Linkages with the Development of Approximation Algorithms. *Journal of the Operations Research Society* **48**, p502-510.

Gutin, G., and A. Yeo. 2002. Polynomial approximation algorithms for the TSP and the QAP with a factorial domination number. *Discrete Applied Mathematics* **119**, p107-116.

Johnson, D. S., C. R. Aragon, L. A. McGeoch, C. Schevon. 1989. Optimization by Simulated Annealing: An Experimental Evaluation; Part 1, Graph Partitioning. *Operations Research* **37**(6), p865-892.

Johnson, D. S., and L. A. McGeoch. 2002. Experimental Analysis of Heuristics for the STSP. Gutin and Punnen (eds.), *The Traveling Salesman Problem and its Variations*, Kluwer Academic Publishers.

Kernighan, B. W., and S. Lin. 1970. An efficient heuristic procedure for partitioning graphs. *Bell System Technical Journal* **49**, p291-307.

Papadimitriou, C. H., and K. Steiglitz. 1982. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall.

Potts, C. N., and S. L. van de Velde. 1995. Dynasearch – Iterative local improvement by dynamic programming: Part 1, The traveling salesman problem. Technical Report, University of Twente, The Netherlands.

Punnen, A. P., and S. N. Kabadi. 2002. Domination Analysis of some heuristics for the asymmetric traveling salesman problem. *Discrete Applied Mathematics* **119**, p117-128.

Punnen, A. P., F. Margot, and S. N. Kabadi. 2002. TSP heuristics: domination analysis and complexity. *Algorithmica* (accepted).

Savage, S. L. 1973. The solution of discrete linear optimization problems by neighborhood search techniques. PhD Thesis, Department of Computer Science, Yale University.

Sharma, D. 2002. Cyclic Exchange and Related Neighborhood Structures for Combinatorial Optimization Problems. Ph.D. Thesis. Operations Research Center, MIT, Cambridge, MA.

Voss, S., S. Martello, C. Roucairol, I. H. Osman. 1999. *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*. Kluwer.