A Note on Single Alternating Cycle Neighborhoods for the TSP

Birger Funke, Tore Grünert, and Stefan Irnich
Lehr- und Forschungsgebiet Operations Research und Logistik Management,
Rheinisch-Westfälische Technische Hochschule (RWTH) Aachen, Germany
e-mail: {birger,tore,sirnich}@or.rwth-aachen.de

Abstract

This paper investigates two different local search approaches for the TSP. Both approaches are based on the general concept of single-alternating cycle neighborhoods. The first approach, stems from the famous heuristic suggested by Lin and Kernighan and the second is based on the notion of stem-and-cycles developed by Glover in the early nineties. We show that the corresponding neighborhoods are not identical and that only a subset of moves can be found when Lin & Kernighan's gain criterion is applied.

Introduction

The Traveling Salesman Problem (TSP) is the most widely studied problem in combinatorial optimization, see, e.g., (Lawler et al., 1985; Gutin and Punnen, 2002). The objective is to find a cost-minimal Hamiltonian cycle, i.e. a cycle in a graph that contains each node exactly once. Although large instances of the problem can now be solved to optimality, see (Applegate et al., 2003a), heuristics and metaheuristics are needed if good results have to be computed within short time limits. In this note we will focus on the symmetric version of the problem, the *STSP* defined by a cost matrix $C = (c_{ij})$ with $c_{ij} = c_{ji}$ for all $i, j = 1, \ldots, n$.

The most successful and widely applied methods for solving large-scale TSPs to near optimality are variations of the Lin-Kernighan (LK) algorithm, (Lin and Kernighan, 1973; Applegate et al., 2003b; Johnson and McGeoch, 1997; Johnson and McGeoch, 2002). About 10 years ago, a similar, but different, approach was suggested by Glover in (Glover, 1991; Glover, 1992a). Recent computational experiments, see (Rego and Glover, 2002), indicate that methods based on so-called ejection chains (ECs) have the potential of outperforming the Lin-Kernighan-based approaches if implemented properly.

Both variants of LK and ECs are variations of the more general concept of local search performed along an alternating cycle in the following sense. The algorithms subsequently add and drop edges which share an endpoint so that after a number of iterations, a tour is transformed into another tour. Deleted and added edges form an alternating cycle. The Lin-Kernighan heuristic starts this process by deleting an edge and the Stem-and-Cycle methods, suggested by Glover, start by adding an edge. Since both methods generate alternating cycles, one might come to the conclusion that the same neighbor solutions can be reached by both methods and that they are, in fact, equivalent. We will show below, that this is not the case.

We start in Section 1 by defining the concepts of local search, k-Opt moves, alternating cycles and sequential search in the context of the STSP. Section 2 introduces the notion of cyclic-independence and show how it is related to the concept of sequential search introduced by Lin and Kernighan. A notation to classify different types of k-Opt moves is presented in Section 3. In Section 4, we discuss single-alternating cycle neighborhoods in general and show how the different neighborhoods are related to each other and the more general class of k-Opt neighborhoods. Finally, in Section 5 we give some conclusions and outline promising paths for future research.

1 Local Search, k-Opt, and Alternating Cycles

The STSP can be stated as a *combinatorial optimization problem* of the form $\min_{x \in X} c(x)$, where X is the set of feasible solutions (i.e. Hamiltonian cycles) and c the cost function. Clearly, the set X is in general too big to be searched entirely. The heuristics LK and EC are based on the *local search* (LS) concept. The heart of a local search procedure is the definition of a neighborhood $\mathcal{N}$ which is a mapping $\mathcal{N} : X \to 2^X : \mathcal{N}(x) \subset X$. Each element $x' \in \mathcal{N}(x)$ is called *neighbor of* x. Neighbors x' with cost $c(x') < c(x)$ are called *improving neighbors*. LS starts with a initial feasible solution $x^0 \in X$ and iteratively replaces the current solution x^t by an improving neighbor $x^{t+1} \in \mathcal{N}(x^t)$ as long as such an improving neighbor exists. The LS procedure terminates with a *local optimum*, i.e. a solution x^t for which the neigborhood $\mathcal{N}(x^t)$ contains no improving solution.

Neighborhoods for the TSP are usually defined implicitly by a set of *moves*. A move m transforms a solution into another solution. Many variations of this general approach exist, a good introduction can be found in the book (Aarts and Lenstra, 1997). For a precise definition of the term *move*, it is helpful to consider an enclosing superset of *solutions* $Z \supseteq X$. The idea of a solution $x \in Z$ is that some of the moves $m \in M$ might transform a *feasible solution* $x \in X$ into an object $m(x)$, which has a structure similar, but not identical to a tour. In general, we denote by M the *set of moves* where a move $m \in M$ is a map from Z to itself, i.e. $m : Z \to Z$. From the above discussion, it is clear that a move m does *not* necessarily map feasible solutions $x \in X$ into feasible solutions. For a given $x \in Z$, the *extended neighborhood* $\hat{\mathcal{N}}(x) = \{m(x) : m \in M\}$ contains all neighbors of x, either feasible or infeasible. Clearly, the neighborhood $\mathcal{N}(x) \subset X$ is given by $\mathcal{N}(x) = \hat{\mathcal{N}}(x) \cap X$. Every move, $m \in M$, with $m(x) \in X$ is called a *feasible move* w.r.t. x.

A k-Opt move m deletes k edges $d_1, d_2, \ldots, d_k$ from a given tour $x \in X$ and adds k edges $a_1, a_2, \ldots, a_k$ in such a way that the result is another Hamiltonian cycle $x' \in X$. The symmetric difference of the edges taken from the current tour x and the new tour $x' = m(x)$ spans a subgraph $G(d_1, \ldots, d_k, a_1, \ldots, a_k)$, which decomposes into one or several *alternating cycles*. Note that the alternating cycles completely determine the move m. If $G(d_1, \ldots, d_k, a_1, \ldots, a_k)$ is connected, the moves (neighborhood) are called *single alternating cycle k-Opt moves (SAC-k-Opt) (neighborhoods, SACN).*

In order to analyze different moves, we will decompose them into smaller parts, the so-called *partial moves*. In the context of this note, only two types of partial move are considered, i.e. p^{add}_{ij} which adds the edge (i, j) to the current structure and p^{del}_{kl} which deletes the edge (k, l) from the current structure. A given decomposition $m = p_l \circ \ldots \circ p_2 \circ p_1$ of a move m into $l \geq 2$ partial moves $p_1, p_2, \ldots, p_l$ means that an $x \in Z$ is first transformed into $p_1(x)$, second $p_1(x)$ is transformed into $p_2(p_1(x))$, and so on. Of course, we have to consider the structures that occur after having applied one or several partial moves. In general, the ith partial move transforms elements of an intermediate structure Y_{i-1} to elements of another intermediate structure Y_i, while for the first and last structure $Y_0 = Y_l = Z$ holds. As a result, $m : Z \to Z$ decomposes into

$$m : \quad Z = Y_0 \xrightarrow{p_1} Y_1 \xrightarrow{p_2} Y_2 \xrightarrow{p_3} \quad \ldots \quad \xrightarrow{p_{l-1}} Y_{l-1} \xrightarrow{p_l} Y_l = Z.$$

There are some interesting cases that we would like to study. When some or all sets Y_i are identical, we refer to $Y = Y_i$ as *reference structures*. In this case it is possible to vary the number of intermediate partial moves, which results in chains of intermediate partial moves of variable length. Both the LK and the ECs are constructed in this way.

2 Cyclic Independence and Sequential Search

Given a move m and a decomposition of the move into partial moves, it is advantageous to study whether or not the *order* of application of the partial moves influences the resulting move. For the case $Y_i = Z$, the decomposition $m = p_l \circ \ldots \circ p_2 \circ p_1$ is *cyclic-independent* if

$$m(x) = p_{\pi(l)} \circ \ldots \circ p_{\pi(2)} \circ p_{\pi(1)}(x)$$

holds for all solutions $x \in Z$ and all *cyclic* permutations π of $\{1, 2, \ldots, l\}$.

Cyclic-independent neighborhoods are closely linked to the concept of *sequential search*, first developed by Lin and Kernighan. The basic idea of this approach is to consider all relevant partial moves of a cyclic independent neighborhood recursively. In sequential search, the elements are *not* selected according to any order specified by the current tour but rather using *neighbor lists*. These neighbor lists are sorted in the order of increasing cost of the edges. Sequential search is particularly attractive if the number of elements in every selection step (i.e. the length of the neighbor list) is reduced from $\mathcal{O}(n)$ to some small constant. In order to search a neighborhood exactly by sequential search, it has to be *cyclic-independent*, although it can be used heuristically with non-cyclic-independent neighborhood, as discussed below for the Lin-Kernighan heuristic. The attractiveness of sequential search in cyclic neighborhoods is due to the following theorem of Lin and Kernighan, (Lin and Kernighan, 1973):

Theorem 1 *If a sequence of numbers $(g_i)_{i=1}^k$ has a positive sum $\sum_{i=1}^k g_i > 0$, there is a cyclic permutation π of these numbers such that* every partial sum *is positive, i.e. $\sum_{i=1}^p g_{\pi(i)} > 0$ for all $1 \leq p \leq k$.*

Proof: Let q be the largest index for which $\sum_{i=1}^{q-1} g_i$ is minimum. Choose π such that $\pi(1) = q$. If $q \leq p \leq n$, $g_q + \ldots + g_p = (g_1 + \ldots + g_p) - (g_1 + \ldots + g_{q-1}) > 0$. If $1 \leq p < q$, then $g_q + \ldots + g_n + g_1 + \ldots + g_p \geq g_q + \ldots + g_n + g_1 + \ldots + g_{q-1} > 0$. $\diamond$

Denote by $g(p_i, x)$ the *partial gain* of the partial move p_i w.r.t. $x \in Z$. Obviously, we have $g(p_{ij}^{add}, x) = -c_{ij}$ and $g(p_{kl}^{del}, x) = c_{kl}$.

The theorem implies that, for finding an improving move $m = p_k \circ \ldots \circ p_1$ within a given neighborhood which decomposes into cyclic-independent partial moves, we need only consider those moves where $G_i := \sum_{l=1}^{i} g(p_i, x) > 0$ for all $i = 1, \ldots, k$. The direct implication is that at stage i of the search we need only consider moves with a gain $g(p_i, x) > -G_{i-1}$. Thus, the total gain at stage $i-1$ limits the choice of a partial move at stage i. We refer to this rule as the *gain criterion*. The gain criterion is fundamental for the effectiveness of the Lin-Kernighan algorithm. We will see below that the reference structure used within the Lin-Kernighan neighborhood is *not* cyclic-independent. Therefore, Theorem 1 does not apply. Rather, it is used *heuristically* with excellent results. This should be kept in mind when considering the exploitation of the gain criterion within a search algorithm. The gain criterion is also exploited in effective algorithms for 2- and 3-Opt for the TSP, see, e.g., (Bentley, 1992; Johnson and McGeoch, 1997).

In order to describe a generic sequential search algorithm, consider the decomposition $m = p_k \circ \ldots \circ p_2 \circ p_1$ of a move m into $k \geq 2$ partial moves. The cyclic independence of the neighborhood implies that any sequence $p_{j-1} \circ \ldots \circ p_1 \circ p_k \ldots p_j$ with $1 \leq j \leq k$ represents the same move m. For the gain criterion to be applicable, the search algorithm has to guarantee that every cyclic permutation of the moves is generated.

Hence, the algorithm has to generate all the partial moves $p_1, \ldots, p_k$ on the first search level. Each of these partial moves has to satisfy the gain criterion, otherwise, it will be discarded immediately. On the second level, the composed partial moves $p_2 \circ p_1$, $p_3 \circ p_2, \ldots, p_k \circ p_{k-1}, p_1 \circ p_k$ that extend non-discarded partial moves from the first level have to be generated. Again, all move compositions on the second level have to satisfy the gain criterion and can otherwise be discarded. The same is true for the third level and so on until all k levels have been investigated.

3 abc-Notation for k-Opt Move Types

This section introduces a notation for different types of k-Opt moves. It is useful for distinguishing moves contained in the LK, EC, and SAC neighborhoods. All k-Opt moves can be described as a result of applying the following four operations:

1. **k-segmentation.** This operation removes k edges from the tour x resulting in k segments $s_1, \ldots, s_k$.
2. **k-inversion.** This operation inverts a subset of the k segments, i.e. $s_i^{\pm 1}$.
3. **k-permutation.** This operation changes the order of the segments $s_{\pi(1)}^{\pm 1}, \ldots, s_{\pi(k)}^{\pm 1}$ according to a cyclic permutation π.
4. **k-concatenation.** This operator concatenates the segments that result from applying the three first operators into a new tour x'.

The set of moves for which the operators inversion and permutation are identical define a *move type* or *edge exchange type*. When all inserted edges differ from the deleted edges, the move is called a *proper move*.

In order to give a clear description of different edge exchange types we introduce the *abc*-notation. In this notation, one uses the first k letters of the alphabet, where the ith letter corresponds to the ith segment of the tour. If the letter is uppercase, the corresponding segment is reversed. For example, the code $aBDc$ corresponds to moves where the second and the fourth segments are reversed and the third and fourth segments change position. Next, we want the abc-notation to be unambiguous, i.e. the notation should not depend on the numbering of the segments within the tour. Let us assume that the segment "a" is fixed at the first position and is not reversed by an appropriate definition, e.g., stating that segment "a" includes the node with index 1. As a result, any cyclic permutation π can be uniquely written in the abc-notation.

4 Single Alternating Cycle Neighborhoods

In this section we focus on the SACN neighborhood and its corresponding moves where the added and deleted edges form a *single* alternating cycle. The moves have the property of decomposing naturally into cyclic-independent partial moves. It should be noted that for a given k and the assumption that all added and deleted edges are pairwise disjoint, the SACN only contains moves that exchange exactly k edges with k other edges. Hence, the SACN is a subset of the neighborhood defined by the *proper* k-Opt moves. In this section we will analyze the relationship between SACN and the more restricted neighborhoods of LK and EC, which have been used in the TSP-literature. Before going into the details of SACN, we discuss k-Opt moves that have multiple alternating cycles.

4.1 Single and Multiple Alternating Cycles

All 2-Opt moves are SAC-2-Opt moves, all 3-Opt are SAC-3-Opt moves, and the smallest multiple alternating cycle examples are the *double-bridge move* (type adcb) and the *twisted double-bridge moves* (types adBC and aCDb), see Figure 1. These moves produce two

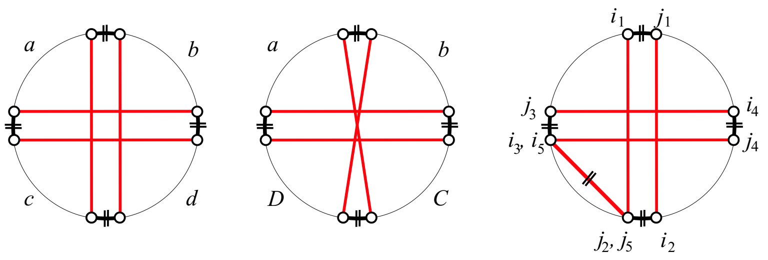

Figure 1: Double bridge move (type adcb), twisted double bridge move (type aCDb), and generation of the double bridge move by a SAC-5-Opt move

alternating cycles, each of which has two added and two deleted edges. Therefore, the (twisted) double-bridge move is not contained in the SAC-4-Opt neighborhood. However, it is important to mention that the double bridge move is included in SAC-5-Opt when one allows the same (or inverse) edge (i, j) to be added and deleted. Figure 1(c) shows the construction of the double bridge move as a SAC-5-Opt move.

Generally, it is possible to connect r alternating cycles with $r - 1$ edges which have to be added and deleted. Since each alternating cycle must consist of at least four (two added and two deleted) edges, the maximum number r^{max} of alternating cycles is $r^{max} = \lfloor k/2 \rfloor$. For instance, two alternating cycles are maximum for $k = 4$ and $k = 5$, and all these moves can be found in SAC-$(k + 1)$-Opt. Three alternating cycles are maximum for $k = 6$ and $k = 7$. All 6-Opt moves can be constructed in SAC-8-Opt.

Note that the above decomposition of multiple alternating cycle moves is *not* cyclic-independent, since deleting a previously added edge requires that one has to add a non-tour edge first and remove it later. These two operations cannot be interchanged. The same holds for adding a deleted edge, where the edge has to be part of the tour.

4.2 Decomposition of Single Alternating Cycle Moves

To extend the above notation of added and deleted edges to the case of *single alternating cycle moves*, we have to label the incident nodes. Tracing along the alternating cycle shows that there exist $2k$ (not necessarily different) nodes $i_1, i_2, \ldots, i_k$ and $j_1, j_2, \ldots, j_k$ such that the deleted edges can be written as $d_p = (i_p, j_p)$ and the added edges can be written as $a_p = (j_p, i_{p+1})$ for all $p = 1, \ldots, k$ (with the setting $i_{k+1} := i_1$). The move m is determined by the nodes $i_1, i_2, \ldots, i_k$ and $j_1, j_2, \ldots, j_k$ of the alternating cycle. From this representation one gets a *natural* decomposition of the move into an alternating sequence of "delete-edge" and "add-edge" partial moves, i.e. p_{ij}^{del} and p_{ji}^{add}. Using this notation and introducing some intermediate structures which will be explained below, the cyclic edge exchange move m can be written as

$$m^{SACN}_{\substack{i_1,\ldots,i_k,\\ j_1,\ldots,j_k}} : \quad Z_{smc} \xrightarrow{p^{add}_{j_1,i_2}} Y' \xrightarrow{p^{del}_{i_2,j_2}} Z_{smc} \xrightarrow{p^{add}_{j_2,i_3}} Y' \xrightarrow{p^{del}_{i_3,j_3}} Z_{smc} \xrightarrow{p^{add}_{j_3,i_4}} \ \ldots \ \xrightarrow{p^{add}_{j_k,i_1}} Y' \xrightarrow{p^{del}_{i_1,j_1}} Z_{smc}. \tag{1}$$

when the process is started with an edge addition, or alternatively

$$m^{SACN}_{\substack{i_1,\ldots,i_k,\\ j_1,\ldots,j_k}} : \quad Z_{smc} \xrightarrow{p^{del}_{i_1,j_1}} Y_{pmc} \xrightarrow{p^{add}_{j_1,i_2}} Z_{smc} \xrightarrow{p^{del}_{i_2,j_2}} Y_{pmc} \xrightarrow{p^{add}_{j_2,i_3}} Z_{smc} \xrightarrow{p^{del}_{i_3,j_3}} \ \ldots \ \xrightarrow{p^{del}_{i_k,j_k}} Y_{pmc} \xrightarrow{p^{add}_{j_k,i_1}} Z_{smc}. \tag{2}$$

when the process is started with an edge deletion. For the application of the gain criterion, we assume that two consecutive partial moves of the above decompositions are joined to a single partial move to which the gain criterion can be applied. We thus obtain k partial moves, where the ℓth partial move in decomposition (1) is given by $p_\ell := p^{del}_{i_{\ell+1},j_{\ell+1}} \circ p^{add}_{j_\ell,i_{\ell+1}}$ and the ℓth partial move in decomposition (2) is given by $p_\ell := p^{add}_{j_\ell,i_{\ell+1}} \circ p^{del}_{i_\ell,j_\ell}$.

It is interesting to consider the intermediate structures Z_{smc}, Y_{pmc}, and Y' which result from this decomposition. We will discuss them in detail below and show that the well-known Lin-Kernighan edge exchange neighborhood (Lin and Kernighan, 1973) as well as neighborhoods which rely on the stem-and-cycle reference structure introduced in (Glover, 1992a; Glover, 1992b) are subsets of the SACN. Furthermore, we will analyze the relationship between these neighborhoods.

First, the structure Z_{smc} has to contain all tours. In general, deleting an edge (i_1, j) from a cycle and adding an incident edge (j, i_2) creates a stem-and-cycle structure. A *stem-and-cycle* (SC) is a special graph structure that consists of a set of *cycle nodes* that all lie on a cycle or subtour of the routing graph, see Fig. 2. This cycle contains exactly one

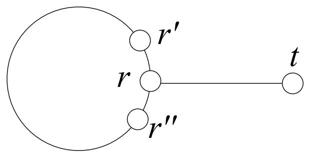

Figure 2: The stem-and-cycle structure with the root r, the subroots r' and r'', and the tip t

node, the *root* r, which has degree three and forms the beginning of a path, the so-called *stem*. The other endpoint of the stem is the *tip* t. The nodes adjacent to the root r in the cycle are the *subroots* r', r''. When the stem degenerates, i.e. tip and root coincide, one obtains a cycle. Therefore, the set Z_{SC} of all stem-and-cycle structures is a superset of the set X of Hamiltonian cycles.

We start with an analysis of the decomposition (1). Consider a given SC. Adding an edge (t, j) from the tip to the SC and subsequently deleting an incident edge (j, q) either creates another SC or a SC and a node disjoint cycle. The four different structures that can result from this operation are depicted in Fig. 3.

Therefore, the reference structure Z_{smc} consists of all unions of a single SC with $SC \in Z_{SC}$ and possibly $q \geq 0$ cycles $C_1, \ldots, C_q$, so that every node of the routing graph belongs

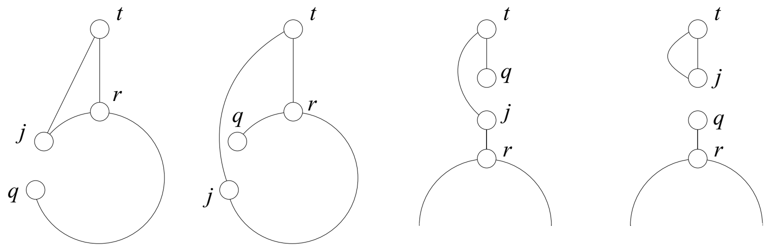

Figure 3: The three possible cases resulting from the rules R1 and R2 of Glover (Glover, 1992b) and the fourth possibility that transforms a stem-and-cycle into a stem-and-multicycle

to the SC or exactly one of the q cycles. The elements $z \in Z_{smc}$ are called *stem-and-multicycle*. An element of this reference structure is depicted in Fig. 4 and it exactly

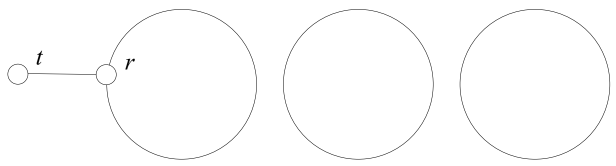

Figure 4: The stem-and-multicycle reference structure

describes the intermediate structure Z_{smc} of the decomposition (1), where incident edges are subsequently added and deleted.

In contrast, the decomposition (2) always deletes an edge (r, r') from the root r to one of the subroots of the SC. The deletion of an edge (r, r') transforms a SC into a single *path*. In general, the structures Y_{pmc} consist of a single path P possibly together with a set of $q \geq 0$ node disjoint cycles $C_1, \ldots, C_q$. We call Y_{pmc} the set of *path-and-multicycles*. Adding an edge from an endpoint (i.e. the former subroot r') of a path P to any other node of this path creates a SC. On the other hand, adding an edge from an endpoint of P to a node which belongs to one of the cycles, e.g., C_q, aggregates the path P and the cycle C_q into a SC. The result is the union of a SC with the $q-1$ cycles $C_1, \ldots, C_{q-1}$. Summing up both cases, the addition of an edge starting at an endpoint of the path of the path-and-multicycle $y \in Y_{pmc}$ gives a stem-and-multicycle $z \in Z_{smc}$. An important special case is the addition of an arc which connects both endpoints of the path, i.e. the transformation of the path into a single cycle. The last partial move in (2) must be of this type. This chain of arguments sufficiently explains the correctness of decomposition (2).

The same argument holds in the inverse case of decomposition (1), where subsequent add/delete partial moves transform one $z \in Z_{smc}$ into another $z' \in Z_{smc}$. This follows from the fact that an add/delete combination is the inverse of a delete/add combination. The advantage of this "symmetry argument" is that we can avoid analyzing the intermediate structures Y' which do not have a simple representation.

4.3 The Lin-Kernighan Neighborhood

The Lin-Kernighan neighborhood (LK) is a restricted SAC-k-Opt neighborhood with moves m^{LK} of variable length, where the move always starts with the deletion of an edge. LK restricts the steps of the decomposition (2) to the case where after the deletion of an edge, the generated structure is a *(Hamiltonian) path* $y \in Y_P$, and not a path-and-multicycle. The intermediate objects z that result from the addition of an edge have to be simple stem-and-cycles $z \in Z_{SC}$. Consequently, a LK move decomposes into

$$m^{LK}_{\substack{i_1,\ldots,i_k,\\ j_1,\ldots,j_k}} : \quad Z_{SC} \xrightarrow{p^{del}_{i_1,j_1}} Y_P \xrightarrow{p^{add}_{j_1,i_2}} Z_{SC} \xrightarrow{p^{del}_{i_2,j_2}} Y^*_P \xrightarrow{p^{add}_{j_2,i_3}} Z_{SC} \xrightarrow{p^{del}_{i_3,j_3}} \quad \ldots \quad \xrightarrow{p^{del}_{i_k,j_k}} Y_P \xrightarrow{p^{add}_{j_k,i_1}} Z_{SC}. \qquad (3)$$

Furthermore, all deleted edges $d_1 = (i_1, j_1), d_2 = (i_2, j_2), \ldots$ and all added edges $a_1 = (j_1, i_2), a_2 = (j_2, i_3), \ldots$ have to be disjoint. From our point of view, the above decomposition is the kernel of the famous Lin and Kernighan neighborhood, suggested in (Lin and Kernighan, 1973). Its variability relies on the fact that at each add-step one can either add the edge (j_p, i_1), which transforms the current path into a tour (i.e. a closing partial move), or one may add a "short" edge (j_p, i_{p+1}) with $i_{p+1} \neq i_1$ which continues the alternating delete-add-process. The second alternative allows finding/generating edge exchanges with variable length.

In (Lin and Kernighan, 1973), the authors allow the second deletion of an edge to either create a Hamiltonian path or a path with a single cycle. In (3) the symbol Y_P^* denotes the set of path-and-cycle structures which contains Hamiltonian paths as well as the union of a path with a cycle, which together cover all nodes of the routing graph. Clearly, $Y_P^* \supset Y_P$ holds. This slightly changes the LK-neighborhood into a larger neighborhood LK*. The motivation for this modification according to (Lin and Kernighan, 1973, p. 506) is that LK* contains all 3-Opt moves, while LK does not. More specifically, the Or-Opt move (type acb), suggested in (Or, 1976), creates a path-and-cycle after the second deletion step, whereas the first and the third steps create Hamiltonian paths, respectively. This

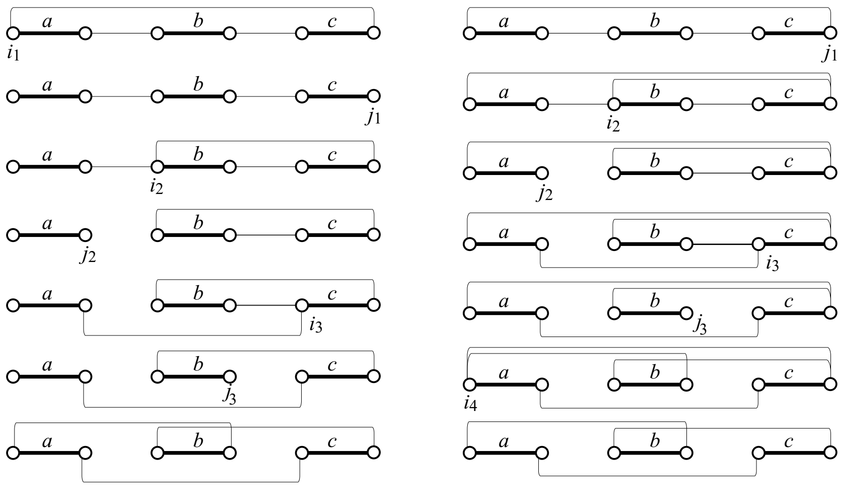

Figure 5: The original tour and the resulting configuration for the Or-Opt move after the six delete and add steps. The left hand side shows the decomposition with the path and multicycle reference structure, which starts with a *delete step*. The right hand side shows the SC decomposition with the stem and cycle reference structure, which starts with an *add step*.

is shown in Fig. 5, which depicts the original tour and the six add/delete steps of the decomposition. The Or-Opt move 'acb' is completely symmetric. Hence, the occurrence of the path-and-cycle is independent from the choice of the first node i_1 from which the search process is started.

When looking at other moves, e.g., the 'acB' 3-Opt move, the occurrence of a path-and-cycle depends on the initial choice of the node i_p, $p = 1, 2, 3$. It is easy to verify that in only one out of these three cases an intermediate path-and-cycle structure is generated. The consequence is that the LK neighborhood is *not* cyclic-independent. This is interesting, since the argument of a cyclic neighborhood is used as a motivation for the gain criterion by Lin and Kernighan.

4.4 Edge Ejection Chains – Stem-and-Cycle

The inverse strategy of first adding an edge and subsequently deleting an incident edge is closely related to the work on ejections chains and the stem-and-cycle reference structure presented by F. Glover in (Glover, 1992b).

We assume that the decomposition (1) is restricted to the case

$$m^{SC}_{\substack{i_1,\ldots,i_k,\\ j_1,\ldots,j_k}}: \quad Z_{SC} \xrightarrow{p^{add}_{j_1,i_2}} Y' \xrightarrow{p^{del}_{i_2,j_2}} Z_{SC} \xrightarrow{p^{add}_{j_2,i_3}} Y' \xrightarrow{p^{del}_{i_3,j_3}} Z_{SC} \xrightarrow{p^{add}_{j_3,i_4}} \ \ldots \ \xrightarrow{p^{add}_{j_k,i_1}} Y' \xrightarrow{p^{del}_{i_1,j_1}} Z_{SC}. \qquad (4)$$

where the reference structure is a stem-and-cycle (SC) (but not a stem-and-multicycle). This exactly corresponds to the transformation of a given SC with tip t and root r according to the following two rules: (R1) Add an edge $\{t, j\}$ where j belongs to the cycle. Identify the deleted edge $\{j, q\}$ to be either of the two edges of the cycle incident at j. (R2) Add an edge $\{t, j\}$ where j belongs to the stem. Identify the deleted edge $\{j, q\}$ by requiring q to lie on the portion of the stem between j and t. In both rules, R1 and R2, node q becomes the new tip of the new SC while the root node r remains fixed. The three possible cases resulting from rules R1 and R2 are depicted as the three leftmost cases in Figure 3.

Glover shows that ejection chains of the form (1) can be used to transform a given tour x into any other tour x' when deleting an added edge is allowed. Moreover, the chain can be started from an arbitrarily chosen node, an added edge might be deleted later (but no deleted edge is added later), $2k$ additions and deletions are sufficient (when x' contains k edges not contained in x), and the chain can be chosen in such a way that no two consecutive partial moves of deleting previously added edges occur (Glover, 1992b, Thm. 1).

The original decomposition (1) with the more general stem-and-multicycle reference structure corresponds with a rule set R1*–R4*, where the first two rules coincide with the rule R1 and R2 given above and the other two rules allow the creation of an additional cycle or the merger of a SC and a cycle into a single SC. These four rules are sufficient to transform any given tour x into any other tour x' within k add/delete steps where $2k$ is the cardinality of the symmetric difference of x and x' (Glover, 1992b, Thm. 4).

For the remaining part of this subsection we require the decomposition (4), i.e. the move m^{SC}, to contain only disjoint added and deleted edges. Then SC is a proper superset of LK, i.e. each Lin-Kernighan move m^{LK} can be replaced by an SC-move m^{SC} with the same number of partial moves and there exist SC-moves with no equivalent LK-move. In order to see this, consider an arbitrarily chosen LK-move given by (3) (with Y_P^* replaced by Y_P). This move can be constructed by the move (4) starting at the second partial move $p^{add}_{j_1,i_2}$ and ending with the partial move $p^{del}_{i_1,j_1}$. The requirement of LK to generate a single path after each deletion partial move corresponds to the requirement of SC not to generate a stem-and-multicycle. It is shown in Fig. 5 that the Or-Opt move (type acb) is contained in SC, making SC a proper superset of LK.

There are moves in SACN which are not contained in SC. The SAC-5-Opt move of type aedcb which reverses the ordering of all five segments without inverting any segment is not in SC. For other types of moves, e.g. the SAC-4-Opt move acbD, the question whether the decomposition (1) generates a stem-and-multicycle depends on the node/edge chosen for starting the ejection chain. A proof of these simple results is straightforward.

5 Conclusions

In this note we have investigated an important class of neighborhoods which are relevant for local search in the context of the TSP. It could be shown that $LK \subset SC \subset SACN \subset$

$k - Opt$. Fig. 6 contains examples (with a minimum number k of added/deleted edges) of moves contained in one of the neighborhoods but not in the corresponding smaller ones. We have also shown that the concept of cyclic independence is at the core of a

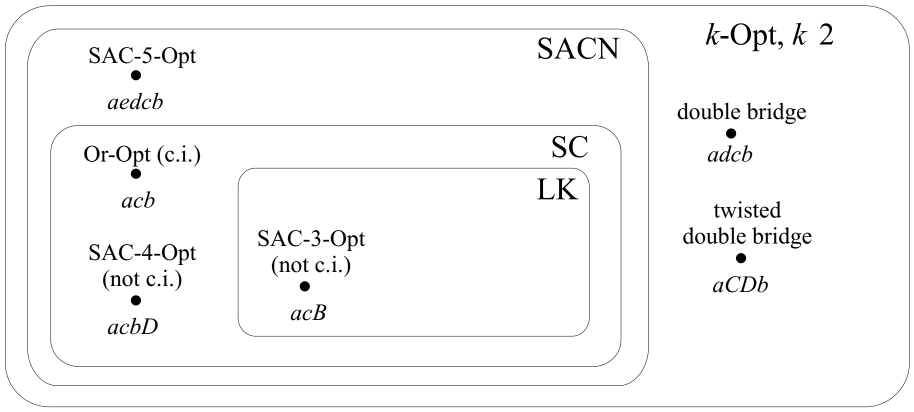

Figure 6: The relationship between different subsets of the k-Opt neighborhood. c.i. = cyclic independent

sequential search algorithm, such as the method by Lin and Kernighan. Interestingly, the move decomposition given by Lin and Kernighan is *not* cyclic independent and, thus, the application of the gain criterion is a heuristic and *not* an exact search algorithm for scanning the entire neighborhood. The same holds in the case of SC. Hence, in both cases the application of sequential search is a heuristic and not an exact technique.

Finally, it should be noted that Glover (Glover, 1992b, Thm. 4) extends ideas of the stem-and-multicycle reference structure also to the case of k-Opt moves with *multiple* alternating cycles and to the asymmetric case. In addition, he defines *subpath ejection chains* and the double-rooted reference structures *bicycle* and *tricycle.* It remains an open question whether algorithms based on Stem-and-Cycle can outperform those based on Lin-Kernighan in practice. Theoretically, this seems possible, since the Lin-Kernighan neighborhood is strictly contained in the Stem-and-Cycle neighborhood. However, the success of such an approach will depend on the potentials for finding efficient data structures which allow the partial moves in a Stem-and-Cycle to be performed efficiently, see (Fredman et al., 1995) for the Lin-Kernighan algorithm. Finding such data structures is an interesting open research problem.

References

Aarts, E. and Lenstra, J. (1997). *Local search in combinatorial optimization.* Wiley, Chichester.

Applegate, D., Bixby, R., Chvatal, V., and Cook, W. (2003a). Implementing the dantzig-fulkerson-johnson algorithm for large traveling salesman problems. *Mathematical Programming, Series B*, 97(1–2):91–153.

Applegate, D., Cook, W., and Rohe, A. (2003b). Chained Lin-Kernighan for large traveling salesman problems. *INFORMS Journal on Computing*, 15(1):82–92.

Bentley, J. (1992). Fast algorithms for geometric traveling salesman problems. *Operations Research Society of America*, 4(4):387–411.

Fredman, M., Johnson, D., McGeoch, L., and Ostheimer, G. (1995). Data structures for traveling salesman. *Journal of Algorithms*, 18:432–479.

Glover, F. (1991). Multilevel tabu search and embedded search neighborhoods for the travling salesman problem. Technical report, US West Chair in Systems Science, University of Colorado, Boulder, School of Business, Campus Box 419, Boulder, CO, 80309.

Glover, F. (1992a). Ejection chains, reference structures and alternating path methods for traveling salesman problems. Technical report, US West Chair in Systems Science, University of Colorado, Boulder, School of Business, Campus Box 419, Boulder, CO, 80309.

Glover, F. (1992b). New ejection chain and alternating path methods for traveling salesman problems. In Balci, O., Sharda, R., and Zenios, S., editors, *Computer Science and Operations Research - New developments in their interfaces*, pages 491–508. Pergamon Press.

Gutin, G. and Punnen, A., editors (2002). *The Traveling Salesman Problem and Its Variations*, volume 12 of *Combinatorial Optimization*. Kluwer, Dordrecht.

Johnson, D. and McGeoch, L. (1997). The traveling salesman problem: A case study in local optimization. In Aarts, E. and Lenstra, J., editors, *Local Search in Combinatorial Optimization*, chapter 8, pages 215–310. Wiley, Chichester.

Johnson, D. and McGeoch, L. (2002). Experimental analysis of heuristics for the stsp. In Gutin, G. and Punnen, A., editors, *The Traveling Salesman Problem and Its Variations*, volume 12 of *Combinatorial Optimization*. Kluwer, Dordrecht.

Lawler, E., Lenstra, J., Rinnooy Kan, A., and Shmoys, D., editors (1985). *The Traveling Salesman Problem. A Guided Tour of Combinatorial Optimization*. Wiley-Interscience Series in Discrete Mathematics. Wiley, Chichester.

Lin, S. and Kernighan, B. (1973). An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, 21:498–516.

Or, I. (1976). *Traveling Salesman-Type Problems and their Relation to the Logistics of Regional Blood Banking*. PhD thesis, Department of Industrial Engineering and Management Sciences. Northwestern University, Evanston, IL.

Rego, C. and Glover, F. (2002). Local search and metaheuristics. In Gutin, G. and Punnen, A., editors, *The Traveling Salesman Problem and Its Variations*, volume 12 of *Combinatorial Optimization*, chapter 8, pages 309–368. Kluwer, Dordrecht.