Partial Neighborhoods of the Traveling Salesman Problem

Darrell Whitley
Department of Computer Science
Colorado State University
Fort Collins, CO 80524 USA
whitley@cs.colostate.edu

Gabriela Ochoa
School of Computer Science
University of Nottingham
Nottingham, NG81BB, UK
gxo@cs.nott.ac.uk

ABSTRACT

The Traveling Salesman Problem (TSP) is known to display an elementary landscape under all k-opt move operators. Previous work has also shown that partial neighborhoods may exist that retain some properties characteristic of elementary landscapes. For a tour of n cities, we show that the 2-opt neighborhood can be decomposed into $\lfloor n/2 - 1 \rfloor$ partial neighborhoods. While this paper focuses on the TSP, it also introduces a more formal treatment of partial neighborhoods which applies to all elementary landscapes. Tracking partial neighborhood averages in elementary landscapes requires partitioning the cost matrix. After every move in the search space, the relevant partitions must be updated. However, just as the evaluation function allows a partial update for the TSP, there also exists a partial update for the cost matrix partitions. By only looking at a subset of the partial neighborhoods we can further reduce the cost of updating the cost matrix partitions.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search

General Terms

Theory, Algorithms

Keywords

Fitness Landscapes, Elementary Landscapes

1. INTRODUCTION

The *fitness landscape* for a combinatorial problem instance is defined by a triple (X, N, f). The *objective function* f maps $f : X \mapsto \mathbb{R}$ and without loss of generality we can define f so as either to be minimized or maximized over X.

We define a *neighborhood operator* as a function N that maps candidate solutions in X to subsets of X. Given a candidate solution $x \in X$, $N(x)$ is the set of points reachable from x in one application of the neighborhood operator.

Elementary landscapes are a special class of fitness landscapes. For all elementary landscapes it is possible to compute $\bar{f}$, the average solution evaluation over the entire search space. It is also possible to compute $\text{avg}\{f(y)\}_{y \in N(x)}$, the average value of the fitness function f evaluated over all of the neighbors of x:

$$\underset{y \in N(x)}{\text{avg}}\{f(y)\} = \frac{1}{|N(x)|} \sum_{y \in N(x)} f(y)$$

Grover [?] originally showed that it is possible to compute $\text{avg}\{f(y)\}_{y \in N(x)}$ without actually evaluating the neighbors of x. He showed there exists neighborhoods for the Traveling Salesman Problem, Graph Coloring, Min-Cut Graph Partitioning, Weight Partition, as well as Not-all-equal-Sat (NAES) where this calculation is possible. Stadler [?] named this class of problems "elementary landscapes" and he showed that for these problems the objective function f is an eigenfunction of the Laplacian of the graph induced by the neighborhood operator. Other researchers have explored various properties of elementary landscapes [?] [?] [?] [?] [?] [?].

Whitley and Sutton [?] introduced another explanation of elementary landscapes. Elementary landscapes have objective functions that are a linear combination of components. Typically, the objective function uses a cost matrix to assign costs to edges in a graph. For any candidate solution, the "components" that make up the cost matrix can be decomposed into those that contribute to a solution x and those that do not contribute to the evaluation of $f(x)$. The components that do not contribute to the evaluation of solution x appear with uniform frequency in the neighborhood around x. The components that appear in solution x also appear with a uniform frequency in the neighborhood around x.

But we can take this idea one step further. Can the neighborhood be partitioned in such a way that we can explicitly calculate in an efficient manner which components of the cost matrix will be sampled in the various partitions of the neighborhood? In general the answer is yes. We show how the component model can be used to derive conditions under which there will exists partial neighborhoods of elementary landscapes that retain some of those properties that characterize elementary landscapes. In the next section we briefly review basic mathematical properties of elementary landscapes. Next we review the "component model" for elementary landscapes. The remainder of the paper illustrates partial neighborhoods for the Traveling Salesman Problem.

GECCO'11, July 12–16, 2011, Dublin, Ireland.

2. ELEMENTARY LANDSCAPES

Let X be a set of solutions, $f : X \to \mathbb{R}$ be a fitness function, and $N : X \to \mathcal{P}(x)$ be a neighborhood operator. We can represent the neighborhood operator by its *adjacency matrix*

$$\mathbf{A}_{xy} = \begin{cases} 1 & \text{if } y \in N(x) \\ 0 & \text{otherwise} \end{cases}$$

In this paper, we will restrict our attention to *regular* neighborhoods, where $|N(x)| = d$ for a constant d for all $x \in X$.

When a neighborhood is regular, the Laplacian operator can be defined as

$$\Delta = \mathbf{A} - d\mathbf{I}$$

where

$$\Delta f(x) = \sum_{y \in N(x)} (f(y) - f(x)) \qquad (1)$$

Stadler defines the class of *elementary landscapes* where the fitness function f is an eigenfunction of the Laplacian [**?**] with a constant offset of b; usually $b = \bar{f}$, the mean fitness value in X. In particular, Grover's wave equation can be written as

$$\Delta f + k(f - \bar{f}) = 0$$

where k is a positive constant. If we assume that function f is normalized such that $\bar{f} = 0$ then it follows that

$$\Delta f + k(f - \bar{f}) = \Delta f + kf = 0 \text{ and therefore } \Delta f = -kf$$

When f does not have zero mean, we can use this equation

$$\Delta f(x) = k\bar{f} - kf(x)$$

to calculate the *average fitness* across the neighborhood of any given candidate solution x. Using equation 1, we calculate this average fitness as follows:

$$\begin{aligned} \underset{y \in N(x)}{\text{avg}}\{f(y)\} &= \frac{1}{d} \sum_{y \in N(x)} f(y) \\ &= \frac{1}{d} \left(\sum_{y \in N(x)} f(y) - f(x) \right) + f(x) \\ &= \frac{1}{d} \Delta f(x) + f(x) \\ &= f(x) + \frac{k}{d}(\bar{f} - f(x)) \end{aligned}$$

2.1 Components and Partial Neighborhoods

Given a point x, and its evaluation $f(x)$ and the mean fitness $\bar{f}$, we will compute $\text{avg}\{f(y)\}_{y \in N(x)}$ based on the sampling rate of the components that make up the cost function. We need to define the set of components, denoted by C, that are used to construct the cost function. Typically the components are weights in a cost matrix, and every solution is a linear combination of components.

One of the things we want to do is to sum over all components. Furthermore in an elementary landscape there exists a ratio which we will denote by $0 < p_3 < 1$ such that

$$\bar{f} = p_3 \sum_{c \in C} c \quad \text{and therefore} \quad \sum_{c \in C} c = \bar{f}/p_3$$

Intuitively p_3 is the proportion of the total components in C that contribute to the cost function for any randomly chosen solution. It follows from this observation that all components are equally represented across the entire space of solutions under this model of elementary landscapes. Also, p_3 is independent of the neighborhood size.

In addition, there are ratios $0 < p_1 < 1$ and $0 < p_2 < 1$ that are used in the following equations.

$$\begin{aligned} \underset{y \in N(x)}{\text{avg}}\{f(y)\} &= f(x) - p_1 f(x) + p_2((\sum_{c \in C} c) - f(x)) \\ &= f(x) - p_1 f(x) + p_2(((1/p_3)\bar{f}) - f(x)) \end{aligned}$$

where p_1 is the proportion of components in x which change when a move occurs, relative to the total number of components that contribute to $f(x)$. Similarly, p_2 is the proportion of components in $C - x$ which change when a move occurs, relative to the total number of components in $C - x$. Both p_1 and p_2 can be expressed relative to d, the size of the neighborhood. The following theorem was proven by Whitley and Sutton [**?**].

THEOREM 1. *If p_1, p_2 and p_3 can be defined for any regular landscape such that the evaluation function can be decomposed into components where $p_1 = \alpha/d$ and $p_2 = \beta/d$ and*

$$\bar{f} = p_3 \sum_{c \in C} c = \frac{\beta}{\alpha + \beta} \sum_{c \in C} c$$

then the landscape is elementary.

Note that

$$p_1 + p_2 = p_2/p_3 = k/d \quad \text{and} \quad k = \alpha + \beta$$

where d is the size of the neighborhood and k is a constant. By simple algebra, one can show

$$\begin{aligned} \underset{y \in N(x)}{\text{avg}}\{f(y)\} &= f(x) - p_1 f(x) + p_2((1/p_3\bar{f}) - f(x)) \\ &= f(x) + \frac{\alpha + \beta}{d}(\bar{f} - f(x)) \end{aligned}$$

The key insight of this paper is that when $\alpha > \beta$ the components in x are changing at a higher frequency than the components in $(C - x)$. When this is the case, the neighborhood can be subdivided; how the neighborhood is subdivided is not important, as long as the ratio p_1 still holds and all of the components that make up solution x are uniformly sampled (included or removed) in each partial neighborhood. Furthermore, the ratio α/β measures how many partial neighbors exist if the partial neighborhoods are uniform in size. In the TSP for example, when the number of cities denoted by n is odd, $\alpha/\beta = \lfloor n/2 - 1 \rfloor$ yields the exact number of partial neighborhoods where p_1 holds. When n is even, there is one degenerate partial neighborhood.

While these partial neighborhoods uniformly sample the components in x, they sample only a subset of the components of $C - x$. The question then becomes whether the sampling of $C - x$ is regular enough to be concisely described. So how the neighborhood is subdivided becomes important.

While this paper focuses on the Traveling Salesman Problem as an example, the insights are general and can be applied to all elementary landscapes. This paper also provides a model and a framework for understanding how partial neighborhoods can be identified and described in other elementary landscapes.

3. THE TSP

Whitley and Sutton use the component model to show that the Traveling Salesman Problem (TSP) is elementary when the neighborhood is generated using 2-opt.

Let C denote the set of costs (weights) associated with edges between cities (vertices) in the graph. Let $w_{i,j}$ be the weight (or distance) associated with edge $e_{i,j}$. Note that the number of weights in the cost matrix is given by $|C| = n(n-1)/2$, which is the number of components in a lower triangle square matrix of dimension n.

There are n edges in a given solution and there are $n(n-1)/2$ edges in the cost matrix that are uniformly sampled over all possible solutions. Therefore:

$$p_3 = \frac{n}{|C|} = \frac{n}{n(n-1)/2} = \frac{2}{n-1}$$

$$\bar{f} = p_3 \sum_{c \in C} = \frac{2}{n-1} \sum_{e_{i,j} \in C} w_{i,j}$$

where $\sum_{e_{i,j} \in C} w_{i,j}$ counts each edge only once.

To compute p_1 note there are n edges in any solution, and 2-opt changes exactly 2 edges. Across the entire neighborhood all edges in x are uniformly removed. Therefore

$$p_1 = 2/n = \frac{2(n-3)/2}{n(n-3)/2} = \frac{\alpha}{d}$$

To compute p_2 note there are $|C| - n$ edges in C with the edges in $f(x)$ removed, and 2 new edges are uniformly picked from these edges. Therefore

$$p_2 = \frac{2}{n(n-1)/2 - n} = \frac{2}{n(n-3)/2} = \frac{\beta}{d}$$

Adding the terms to the component model yields:

$$\operatorname*{avg}_{y \in N(x)}\{f(y)\} = f(x) + \frac{n-1}{n(n-3)/2}(\bar{f} - f(x))$$

where $k = n-1$ and the neighborhood size is $d = n(n-3)/2$.

3.1 Decomposing the TSP

Whitley and Sutton show that some neighborhoods of elementary landscapes can be decomposed into partial neighborhoods such that average can still be efficiently computed for those partial neighborhoods. But they did not explain why such decompositions exist, and (as we will see) they also failed to see that the neighborhood under 2-opt for the Traveling Salesman Problem is highly decomposable.

The key question is this: how can we partition the 2-opt neighborhood so that in a given partition every component of x is uniformly represented? It is easier to answer a complimentary form of this question: how can we select a subset of 2-opt moves that uniformly removes edges from x? If edges are uniformly removed, they will also be uniformly represented in the partial neighborhood.

The answer is to group all of the 2-opt moves according to the length of the segment that is reversed. For example, we can group together all of the 2-opt moves that reverse a segment of length 3. This works because we can reverse a segment starting at city 1, then at city 2, then at city 3 and so on until we reverse a segment at city n. Therefore every edge in solution x is cut exactly twice: when we cut at location i we also cut at location $i + 3$ and our index wraps around the TSP tour (Hamilton circuit).

For $n > 7$ the partial neighborhoods over 2-opt moves of length 3 must be of size n in our example if we wish for the sampling rate of the edges in solution x to be uniform in the partial neighborhood. This means we must apply a reversal starting at each of the n cities. This also means that all of the edges in solution x are found in the partial neighborhood $n-2$ times. To see that this is true, imagine we construct the neighbors by making n copies of x: each edge occurs n times in the duplicate partial neighborhood. Then we cut (remove) two edges from x in each copy, removing every edges exactly twice in the partial neighborhood. Thus, if the segment length is 3, then the ratio p_1 is still calculated using $p_1 = 2/n$ in the partial neighborhood.

The same logic applies to all segments of length $i < n/2$. ($i = n/2$ turns out to be a special case). When reversing 2-opt segments, the minimal segment length that can be meaningfully reverse is of length 2. Also, note that when a segment of length i is identified, there is also a corresponding segment of length $n - i$. Reversing either segment yields the same 2-opt move, so we will only consider segments of length $i \leq \lfloor n/2 \rfloor$.

3.2 An Illustration

Assume we have a 7 city TSP where the tour is represented by a permutation of integers.

The current solution x is: `1 2 3 4 5 6 7` . A reversed segment will be denoted by `<1 2>` . In general, since there are 7 cities, there are exactly 7 reversals of length $l < n/2$. We first consider all of the 2-opt moves that reverses a segment of length 2. We next look at all of the reverses of length 3 using the same notation. All of the moves are generated by first shifting the permutation, then reversing the first 2 or 3 cities.

length 2 moves	length 3 moves
<2 1> 3 4 5 6 7	<3 2 1> 4 5 6 7
<3 2> 4 5 6 7 1	<4 3 2> 5 6 7 1
<4 3> 5 6 7 1 2	<5 4 3> 6 7 1 2
<5 4> 6 7 1 2 3	<6 5 4> 7 1 2 3
<6 5> 7 1 2 3 4	<7 6 5> 1 2 3 4
<7 6> 1 2 3 4 5	<1 7 6> 2 3 4 5
<1 7> 2 3 4 5 6	<2 1 7> 3 4 5 6

It is critical to note that all of the length 2 moves form a partial neighborhood, and that all of the length 3 moves form a different partial neighborhood. While they introduce different sets of new edges, they break exactly the same set of edges: every edge in the solution x is broken twice. Thus, the calculation of $p_1 = 2/n$ remains constant for the partial neighborhoods.

We will construct an triangle matrix M where the entry at location m_{ij} can simultaneously describe both edges in the neighborhood and segments of the tour. We will assume row and column indices i and j correspond to the *position* of the cities in the current tour. When describing segments, an entry at location $m_{i,j}$ corresponds to a segment that is cut immediately after city i and is cut immediately after city j

and then is reversed. Note that this also creates a new edge in the resulting neighbor, since city i and j are now adjacent after the segment is reversed. Thus, an entry at location $m_{i,j}$ represents both the segment that is reversed and a new edge $e_{i,j}$ that is created by 2-opt. *We can therefore assign this new edge to a partial neighborhood.* What we will actually store at location $m_{i,j}$ is the name of the partition to which that edge belongs. Note that when $j = i + 1$ the segment is of length 1, and the move generates a duplicate of the current solution. Therefore, when $j = i + 1$ the entry $m_{i,j}$ corresponds to an edge $e_{i,j}$ in the current solution x. Also note that when edge $e_{i,j}$ is created by a 2-opt move, so is edge $e_{i+1,j+1}$. But since $(j + 1) - (i + 1) = (j - i)$, if the distance between cities i and j is p then the distance between cities $(i + 1), (j + 1)$ is also p. Therefore, all of the edges that belong to a particular partial neighborhood will lie one of two diagonals in the matrix.

In the following example, we will assign the symbol `X` to a matrix location to indicate that an edge in the current solution (i.e., when $i = j + 1$). The symbol `A` marks the edges in the partial neighborhood corresponding to length 2 moves, and the symbol `B` marks the edges in the partial neighborhood corresponding to length 3 moves.

Assume the currently solution is

```
x = 1 2 3 4 5 6 7
```

The matrix M then has edges found in the following positions belonging to partial neighborhoods denoted by the labels `A` and `B`.

```
    1  2  3  4  5  6  7
--|-------------------
1 |    X  A  B  B  A  X
2 |       X  A  B  B  A
3 |          X  A  B  B
4 |             X  A  B
5 |                X  A
6 |                   X
```

The patterns which emerges can also be generalized. If $j - i = 1$ then the corresponding edges in the sorted cost matrix will belong to the current solution x. If $j - i = l$ and $1 < l < n/2$ then the corresponding edges in the sorted cost matrix correspond to a partial neighborhood defined over n edges where the reversed segment is of length l. If $j - i = l$ and $l > n/2$ then $n - l$ is the shorter of the two segments which can be reversed to yield the same move, so we classify this as a move of length $n - l$. This is why each partial neighborhood is distributed along diagonals of the matrix.

When n is odd, every partial neighborhood is defined by exactly n edges in matrix M. This is because every possible segment length from 2 to $(n - 1)/2$ produces a unique 2-opt move for every possible edge (i, j) where $i < j$.

When n is even, our observation as to what happens when $j - i = l$ and $l < n/2$ or $l > n/2$ still holds, but it does not cover the case where $l = n/2$. Consider the case when $n = 8$. In this example, the symbols `A` and `B` represent segments of length 2 and 3 respectively, and `C` denotes segments of length 4.

Given the tour `x = 1 2 3 4 5 6 7 8` we obtain the following matrix.

```
  | 2  3  4  5  6  7  8
--|-------------------
1 | x  A  B  C  B  A  x
2 |    x  A  B  C  B  A
3 |       x  A  B  C  B
4 |          x  A  B  C
5 |             x  A  B
6 |                x  A
7 |                   x
```

When $l = n/2$ there are only $n/2$ unique 2-opt moves of length l. When n is even there is one partial neighborhood of size $l = n/2$. This subset corresponds to a single diagonal of matrix M that lies at a centered diagonal in the matrix. We formalize these ideas in the following theorem.

THEOREM 2. *Let M correspond to an new upper triangle matrix where the indices of M also index a permutation x representing a tour in a Traveling Salesman Problem. When interpreted as edges, M corresponds to all of the edges in the cost matrix. Each edge in M where $j = i+1$ corresponds to an edge found in x. Each edge in M where $j \neq i+1$ corresponds to an edge found in the 2-opt neighborhood of x that is not found in x, since if they are not in x they are edges in the set (C - x). If $j-i = l$ or $j-i = n-l$ then the 2-opt move that generated edge $e_{i,j}$ was produced by reversing a segment of length l. This means the diagonals where $j - i = l$ or $j - i = n - l$ corresponds to a partial neighborhood that uniformly samples edges in solution x. Each edge on a diagonal appears twice in the corresponding partial neighborhood.*

PROOF:

Since there are exactly n pairs where $j = i + 1$, this must cover all of the edges in x and therefore all other edges in M must be in the set (C-x). When $j \neq i+1$, a segment of length $l = j - i$ exits that corresponds to a 2-opt move. When $l < n/2$ there must be exactly n such edges, since these edges must lie on one of two diagonals in matrix M. The neighborhood size for 2-opt is $d = n(n - 3)/2$, which is also the number of edges in matrix M that are not in solution x (the size of the matrix is $n(n-1)/2$ and there are n solutions in x). When n is odd there are $d/n = (n - 3)/2 = \alpha/\beta$ partial neighborhoods. When n is even there is an additional partial neighborhood of size $n/2$, and therefore there are $(d - n/2)/n = (n(n - 3)/2 - n/2)/n = (n - 4)/2$ partial neighborhoods of size n. This accounts for all of the elements in the matrix M.

Technically, the matrix M only counts the edge $e_{i,j}$ which is the "leading" edge that is introduced by 2-opt. When edge $e_{i,j}$ is created by a 2-opt move, so is edge $e_{i+1,j+1}$ which we will describe as a "following" edge. But since $(j+1)-(i+1) = (j - i)$, if the distance between cities i and j is d then the distance between cities $(i+1)$ and $(j+1)$ is also d. Therefore M only accounts for the "leading" edges. But note that M exactly covers each edge in the set (C - x) once. In the full 2-opt neighborhood, we know that $p_2 = \frac{2}{d}$ which means that 2-opt touches each edge in the set (C - x) twice.

Therefore, each edge in M appears once as a "leading" and once as a "following" edge under 2-opt. Therefore, the classification of edges in M applies equally to the sets of leading and following edges. □

3.3 Averages Over Partial Neighborhoods

Assume we have a solution x and we would like to calculate the averages over all partial neighborhoods. Our goal is to ask if some partial neighborhood has a lower average than another, and therefore can we guide search toward more promising partial neighborhoods.

By using inverted indexing schemes we would calculate the averages over all partial neighborhoods in $O(n^2)$ by assigning each weight in the cost matrix to some partial neighborhood. However, just as we can do a partial update to the function $f(x)$ when computing $f(y)$ where $y \in N(x)$, we can also use partial updates to calculate how partial neighborhoods change.

We again consider some key examples. Assume we consider a 3-opt move of length 3 in a 7 city problem.

We can shift the tour x so that the segment to be reversed is at the beginning of the tour. Now, if the length of the 2-opt is l we only need the first l rows of matrix M to determine how the assignment of edges to partial neighborhoods change. We identify the reversed segment using the notation `< >` in the columns and `>` in the rows.

```
     4  5  6  7  1  2  3          <6  5 4>  7  1  2  3
   ---------------------        ---------------------
4 |   X  A  B  B  A  X       6>|   X  A  B  B  A  X
5 |      X  A  B  B  A       5>|      X  A  B  B  A
6 |         X  A  B  B       4>|         X  A  B  B
```

We can calculate the change in the averages of the partial neighborhoods by direct observation. Actually, we will compute the sum of weights in lieu of the average. Let S_A denote the sum of the weights in the partial neighborhood A before the move and let S_A* denote the sum of the weights in the partial neighborhood A after the move. Similarly, we will use S_B and S_B* to denote the partial neighborhood B before and after the move.

$$
\begin{aligned}
S_A* &= S_A + w_{2,6} - w_{2,4} + w_{1,4} - w_{1,6} \\
S_B* &= S_B - w_{2,6} + w_{2,4} - w_{1,4} + w_{1,6} \\
&\quad - w_{4,7} + w_{4,3} - w_{3,6} + w_{6,7}
\end{aligned}
$$

Clearly there must be a conservation of edges when a move occurs. Edges are exchanged between sets A and B, as well as between B and X. These exchanges form cycles, so we can look at which subsets of edges are exchanged in a cycle and group them accordingly.

$$
\begin{aligned}
E1 =& w_{4,7} - w_{4,3} + w_{3,6} - w_{6,7} \\
E2 =& w_{2,6} - w_{2,4} + w_{1,4} - w_{1,6} \\
S_A* =& S_A + E1 \quad \text{and} \quad S_B* = S_B - E1 - E2 \\
f(y) =& f(x) + E2
\end{aligned}
$$

Let $\mathcal{S}$ be the sum over all the weights in the cost matrix. Since the solution x and the partial neighborhoods A and B fully partition the entire cost matrix, it follows that when we move from solution x to solution y that:

$$
\mathcal{S} = f(x) + S_A + S_B = f(y) + S_{A*} + S_{B*}
$$

Therefore if we calculate the change in weights in S_{A*} we can subsequently calculate $S_{B*} = \mathcal{S} - f(y) - S_{A*}$ in one step.

Note that when the length of the segment being reversed is 3, only 2 columns and rows of the matrix change. This is because the middle row acts as a pivot row that does not move. Therefore updating the sums over the partial neighborhoods for a segment of length 2 and a segment length 3 has similar costs.

We can generalize this for all segments of length l. If l is even, then the cost of updating the sums of the cost matrix for the partial neighborhoods associate with segments of length l and $l+1$ is similar, because $l+1$ is odd and there will be pivot row in the cost matrix that does not change when all segments of length $l+1$ are reversed.

While this 7 city example has produced useful insight, there are only moves of length 2 and length 3. We next consider a 12 city example. Now we have segments of length 2, 3, 4, 5 and 6.

```
     1  2  3  4  5  6  7  8  9  10  11  12
   ---------------------------------------
1 |     X  A  B  C  D  E  D  C   B   A   X
2 |        X  A  B  C  D  E  D   C   B   A
3 |           X  A  B  C  D  E   D   C   B
4 |              X  A  B  C  D   E   D   C
5 |                 X  A  B  C   D   E   D
6 |                    X  A  B   C   D   E
7 |                       X  A   B   C   D
8 |                          X   A   B   C
9 |                              X   A   B
10|                                  X   A
11|                                      X
```

For now, we will only look at 2-opts of length 3 and 5. In the following matrices, only those entries in the matrix that correspond to a change in the partial neighborhood are shown. In this case, we will use the first 5 rows to illustrate how multiple 2-opts can be related. Both the length 3 and length 5 2-opts will use row 3 as the pivot row. Edges that are not affected by the 2-opt are replaced by the * symbol, including the pivot row.

```
2-opt of length 3

     1 <4  3 2>  5  6  7  8  9  10  11  12
   ---------------------------------------
1 |     x  *  B  *  *  *  *  *   *   *   *
4>|        *  *  B  C  D  E  *   C   B   A
3>|           *  *  *  *  *  *   *   *   *
2>|              x  A  B  C  *   E   D   C
5 |                 *  *  *  *   *   *   *
```

```
2-opt of length 5

    <5  4  3  2 1>  6  7  8  9  10  11  12
   ---------------------------------------
5>|     *  *  *  *  D  E  D  *   B   A   x
4>|        *  *  *  C  D  E  *   C   B   A
3>|           *  *  *  *  *  *   *   *   *
2>|              *  A  B  C  *   E   D   C
1>|                 x  A  B  *   D   E   D
```

We will refer to the average before the updates as

$$
S_A, S_B, S_C, S_D, S_E
$$

and the average after the updates as follows:

For length 3 the labels are: $S_{A3}, S_{B3}, S_{C3}, S_{D3}, S_{E3}$

For length 5 the labels are: $S_{A5}, S_{B5}, S_{C5}, S_{D5}, S_{E5}$

We will take advantage of the fact that the changes in edges are symmetric. We will again assume that the tour has been shifted so that the first l cities to be reversed are in the first l positions. First, all of the edges in the first l columns involve symmetric changes; thus we only need to consider edges starting at row $l+1$. When the edge at $(r, l+g)$ moves between partitions, so do edges at $(r, n+1-g)$ and at $(l+1-r, l+g)$ and at $(l+1-r, n+1-g)$. For convenience, we assume that $(r, l+g)$ are the two smallest indices. If the edge at $(r, l+g)$ belows to partition K and the edge at $(r, n+1-g)$ belows to partition Z, then the edges at $(l+1-r, l+g)$ will belong to Z and the edge at $(l+1-r, n+1-g)$ will belong to K. This creates a cycle. This means that all movement of edges between partitions can be grouped into sets of four edges. Two edges move into a partition, and two edges move out. We next formalize and prove these ideas.

Theorem 3. *Assume a 2-opt move of length l reverses the first l cities. When the edge at $(r, l+g)$ moves between partitions, so do edges at locations $(r, n+1-g)$ and at $(l+1-r, l+g)$ and at $(l+1-r, n+1-g)$. Assume partitions are numbered so that the partition corresponding to length s moves is numbered $s-1$. Let $Partition(i,j)$ be a function that return the partition number associate with matrix location $m_{i,j}$. Then for rows $r=1$ to $r=\lfloor l/2 \rfloor$, and for columns $c=l+1$ to $c=\lfloor (n-l)/2 \rfloor$ the following assignments track all possible exchanges of edges between partitions:*

$$Partition(r, l+g) = Partition(l+1-r, n+1-g) = z+l-r$$

$$Partition(l+1-r, l+g) = Partition(r, n+1-g) = z+r-1$$

In special cases this must be adjusted: if the value for the partition is $\lceil (n-3)/2+2 \rceil + j$ where $j > 0$ then the adjusted value is $\lceil (n-3)/2+2 \rceil - j$.

PROOF

Assume we have a 2-opt that reverses a segment of length l. This impacts only the first l rows of matrix M. We can divide matrix M into 5 regions.

Region 0. In the first l rows and columns of M, the segment is reversed. Within this region index (i, j) is remapped to index$(l+1-i, l+1-j)$ to reverse the relevant rows and columns. Note that all $j-i=c$ are on a diagonal and therefore are assigned to the same partition. Also, note that $j-i=(l+1-i)-(l+1-j)=c$. Therefore, none of the assignments of edges to partitions change in this region.

Region 1. This region includes the first $\lfloor l/2 \rfloor$ rows and those columns numbered from $l+1$ to $\lfloor l+(n-l)/2 \rfloor$. When n and l is even, this region is 1/4 of the remaining elements in the first l rows of matrix M excluding region 0. This creates 4 regions. When $n-l$ is odd, we remove a pivot column which lies between the regions. When l is odd, we remove a pivot row between the regions.

After removing any pivot column or row, Region 1 is 1/4 of the remaining elements in the first l rows of matrix M excluding region 0. We index Region 1 using $(r, l+g)$. The other 3 regions will be defined relative to $(r, l+g)$ and Region 1.

Region 2 is indexed by $(l+1-r, l+g)$.

Region 3 is indexed by $(r, n+1-g)$.

Region 4 is indexed by $(l+1-r, n+1-g)$.

We first consider the case where $l \leq \lceil (n-3)/2 \rceil$. In this case, all of the partition numbers found in any column or row of Region 2 and 3 are monotonically increasing or decreasing. We will assume all partition numbers are monotonically increasing or decreasing, and will compute a correction when this is not the case.

Note that row 1 and row l can be used as reference vectors to compute the other partitions for any given location. In row 1 we will count backwards from column n; the partition numbers are monotonically increasing starting with partition 0 for $\lfloor (n-l)/2 \rfloor$ steps (those positions in Region 3). In row l in region 2, starting at column $l+1$ the partitions are monotonically increasing starting with partition 0, again for $\lfloor (n-l)/2 \rfloor$ steps (those positions in Region 2).

When $Partition(l, l+g) = Partition(1, n+1-g) = z$ we obtain the following result.

$$Partition(l+1-r, l+g) = Partition(r, n+1-g) = z+r-1$$

$$Partition(r, l+g) = Partition(l+1-r, n+1-g) = z+l-r$$

Since the columns in Region 1 are numbered from $l+1$ to $\lfloor l+(n-l)/2 \rfloor$ the maximal value for z is $\lfloor (n-l)/2 \rfloor - 1$ (the first partition `X` has number zero). The minimal value for r is 1. Therefore, the maximal partition number is $\lfloor (n-l)/2 \rfloor + l - 2$.

In some cases (e.g., when $l/2 > \lceil (n-3)/2 \rceil$) there can be a few partitions that are out of bounds in the sense that the partition numbers are no longer monotonically increasing or decreasing. However, the partition number can be at most $\lceil (n-3)/2 \rceil$. When the partition number is out of bounds, the partition numbers repeat in reverse order. Therefore, if the calculated value for the partition is $\lceil (n-3)/2 \rceil + j$ where $j > 0$ then the correct partition is $\lceil (n-3)/2 \rceil - j$. □

Note that the out-of-bound partition numbers occur in predictable locations (e.g. the upper right corner of Region 1).

We have shown that if edge $(r, l+g)$ belows to partition Q then edge $(l+1-r, n+1-g)$ also belows to partition Q. And if edge $(l+1-r, l+g)$ belows to partition R then edge $(r, n+1-g)$ also belows to partition R. If $R \neq Q$ then an exchange of four edges happens between partition Q and R. We denote the exchange of edges by E.

$$E = w_{r,l+g} + w_{l+1-r,n+1-g} - w_{l+1-r,l+g} - w_{r,n+1-g}$$

This means that all exchanges of edges can be organized into groups of 4 edges. One of these groups will correspond to the update to the evaluation function, because partition 0 corresponds to the set `X` that corresponds to the current solution.

Returning to our example, when 2-opt reverses a segment of length 5 in a 12 city problem, we obtain the following updates. Let E_i denote a set of edges that change together as a group.

$$
\begin{aligned}
E_1 =& w_{2,6} + w_{4,6} - w_{2,12} - w_{4,12} \\
E_2 =& w_{2,7} + w_{4,11} - w_{2,11} - w_{4,7} \\
E_3 =& w_{2,8} + w_{4,10} - w_{2,10} - w_{4,8} \\
E_4 =& w_{1,6} + w_{1,12} - w_{5,6} - w_{5,12} \\
E_5 =& w_{1,7} + w_{1,11} - w_{5,7} - w_{5,11} \\
E_6 =& w_{1,8} + w_{1,10} - w_{5,8} - w_{5,10}
\end{aligned}
$$

E_3 is already computed by the fitness update for $f(y)$ for the length 5 2-opt move.

$$
\begin{aligned}
f(y) =& f(x) + E_4 \\
S_{A5} =& S_A + E_1 - E_5 \\
S_{B5} =& S_B + E_2 - E_6 \\
S_{C5} =& S_C - E_1 + E_3 \\
S_{D5} =& S_D - E_2 - E_4 + E_6 \\
S_{E5} =& S_E - E_3 + E_5
\end{aligned}
$$

We can next update the edges for the nested 3-opt move. We will need one addition set of four edges. Because we have *not* shifted the matrix relative to the length 3 move but instead relative to the length 5 update, the computation is calculated relatively to the position $r, l + g = 2, 5$ and thus $(l + 1 - r), (n + 1 - g) = 13, 4 = 1, 4$ (due to the mod effect: $13 = n + 1 = 1$).

$$E_7 = w_{2,5} + w_{1,4} - w_{1,2} - w_{4,5}$$

$$
\begin{aligned}
f(y) =& f(x) + E_7 \\
S_{A3} =& S_A + E_1 \\
S_{B3} =& S_B - E_7 + E_2 \\
S_{C3} =& S_C - E_1 + E_3 \\
S_{D3} =& S_D - E_2 \\
S_{E3} =& S_E - E_3
\end{aligned}
$$

We will use this example to make 2 observations.

First, using this method, reversing shorter segments results in a lower cost for updating the averages of the partial neighborhoods. We can compute the maximal number of edges that can move.

Second, even when long segments are reversed, there can be sequences of moves that reduce the cost of updating the partial neighborhood. Suppose we decide to reverse the segment `<1 2 3 4 5>` as an improving move, and immediately after this we check the segment `<2 3 4>` and find that it is also an improving move. We then update the partial neighborhood after making 2 2-opt moves. This means that even fewer edges must be updated, as the following illustrates (those position marked by * do not move).

```
 <5 <2  3  4> 1> 6  7  8  9  10  11  12
----------------------------------------
5>|  x  *  B  *  D  E  D  *   B   A   x
2>|     *  *  B  *  *  *  *   *   *   *
3>|        *  *  *  *  *  *   *   *   *
4>|           x  *  *  *  *   *   *   *
1>|              x  A  B  *   D   E   D
```

The updates after the nested set of 2-opt moves for the 12 city problem is as follows:

$$
\begin{aligned}
S_{A*} =& S_A - E_5 \\
S_{B*} =& S_B - E_6 + E_7 \\
S_{C*} =& S_C \qquad \text{no change} \\
S_{D*} =& S_D + E_6 - E_4 \\
S_{E*} =& S_E + E_5
\end{aligned}
$$

Recall that E_7 and E_4 are needed to update the evaluation function, so only E_5 and E_6 (8 edges) must be updated in this case. In general, Region 1 includes only $\lfloor n - 5/2 \rfloor$ edges. Nevertheless, this kind of situation is surely unusual.

4. A COST BENEFIT TRADE OFF

Another approach to reducing computation costs is to aggregate the partitions; with fewer partitions, there will be less movement between partitions. Because of the bias in the update costs, we can keep more partitions for shorter lengths, and aggregate partitions associated with long length. This may also make sense, if most improving moves are associated with reversing shorter segments. Also, note that an update to the partition information is only required *after* an improving move is accepted. In our example, we might keep information about partition `A` and `B` and aggregate the rest (`C` and `D` and `E`) into a partition `Z`. Then, we can compute the sum of weights in the partitions as follows:

$$
\begin{aligned}
f(y) =& f(x) + E_4 \\
S_{A5} =& S_A + E_1 - E_5 \\
S_{B5} =& S_B + E_2 - E_6 \\
S_Z =& \sum_{w \in C} w - f(x) - S_{A5} - S_{B5}
\end{aligned}
$$

Of course, the sum over all the weights in the cost matrix need only be done once. This kind of update can be done in constant time if we bound the number of partitions that we wish to track.

THEOREM 4. *Assume that we wish to track the sum of weights in partitions associated with the g shortest lengths of 2-opt moves, where g is fixed. At most $2(g(g + 1))$ edges must be tracked.*

PROOF:

To compute an upper bound, we will assume $g < l/2$.

Between the g row and the $l - g$ rows are all the partitions that are numbered greater than g. Assume all partitions greater than g are assigned to a single partition Z. Between the $l + 1 + g$ column and the $n + 1 - g$ are all columns assigned to partition Z. This cuts the counting problem into 4 symmetric cases that fall in Regions 1, 2, 3 and 4. We can then count the number of updates in Region 1; by symmetry, the same count applies to Regions 2, 3 and 4. There are g distinct partitions in row 1 (including X), there are $g - 1$ in row 2, etc. Therefore there are

$$g(g + 1)/2 = \sum_{i=1}^{g} i$$

distinct partitions when $g < l/2$.

This means the total number of updates that must be made is $4(g(g+1)2) = 2(g(g+1))$. This result is independent of problem size, and if $g \leq l/2$ then the result is exact and independent of the length of the segment reversed by 2-opt. When $g > l/2$ the updates number of updates is less than $2g(g+1)$ due to overlap in the counted rows.

When $g > l/2$ the exactly number of updates that must be make in one region is given by

$$\sum_{i=1}^{g} i - \sum_{i=1}^{g-l/2} i = (g(g+1)/2 - ((g-l/2)(1+g-l/2)/2)$$

which is clearly less than $g(g+1)/2$ for each of the four Regions. □.

This means that if we wish to track partitions corresponding to 2-opt reversal of length 2, 3, 4 and 5 we need to tracking changes to at most $5(6)2 = 60$ edges. If we wished to track all reversals up to length 10, this would require tracking at most $10(11)2 = 220$ edges. Thus, these computations represent an $O(1)$ update that is independent of problem size.

5. DISCUSSION AND CONCLUSIONS

The main contribution of this paper is to show that when $\alpha > \beta$ in an elementary landscape, the neighborhood can be decomposed such that the components that make up x are not only uniformly represented across all of the neighbors of x, the components that make up x are also uniformly distributed across a set of partial neighborhoods. This means that some of the properties that hold over the full neighborhood still hold for the partial neighborhood.

We can compute neighborhood averages over partial neighborhoods if we can still efficiently compute how the partial neighborhood samples the set $C - x$. Tracking the averages over all of the partial neighborhoods for the Traveling Salesman Problems requires $O(n^2)$ time in the worst case. Note that we only need to track the averages after a new candidate solution is accepted. But if we only wish to track the averages of partial neighborhoods corresponding to moves up to length g, the complexity is only $O(g^2)$. As we have just seen, when $g = 10$, the computational cost involves tracking only 220 edges, which is reasonable for large problem sizes.

It would seem likely that this information could be used to guide a local search strategy. However, we have not yet tested this idea. And any particular implementation would be only one way of utilizing partial neighborhoods. The more fundamental point, however, is that this work reveals a new structure that is common to all elementary landscapes where $\alpha > \beta$. There could be many different ways that this information might be exploited by local search heuristics on different problems displaying elementary landscapes.

6. ACKNOWLEDGMENTS

Dr. Whitley would like to thank the School of Computer Science at the University of Nottingham for support during his sabbatical visit. This research was also sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant number FA9550-08-1-0422. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

7. REFERENCES

[1] J. W. Barnes, B. Dimova, and S. P. Dokov. The theory of elementary landscapes. *Applied Mathematics Letters*, 16(3):337–343, April 2003.

[2] B. Codenotti and L. Margara. Local properties of some NP-complete problems. Technical Report TR 92-021, International Computer Science Institute, Berkeley, CA, 1992.

[3] Bruce Colletti and J. Wesley Barnes. Linearity in the traveling salesman problem. *Applied Mathematics Letters*, 13(3):27–32, April 2000.

[4] B. Dimova, J. Wesley Barnes, and E. Popova. Arbitrary elementary landscapes & ar(1) processes. *Appl. Math. Lett.*, 18(3):287–292, 2005.

[5] Lov K. Grover. Local search and the local structure of NP-complete problems. *Operations Research Letters*, 12:235–243, 1992.

[6] Andrew Solomon, J. Wesley Barnes, Steftcho P. Dokov, and Raul Acevedo. Weakly symmetric graphs, elementary landscapes, and the tsp. *Appl. Math. Lett.*, 16(3):401–407, 2003.

[7] Peter F. Stadler. Toward a theory of landscapes. In R. Lopéz-Peña, R. Capovilla, R. García-Pelayo, H. Waelbroeck, and F. Zertruche, editors, *Complex Systems and Binary Networks*, pages 77–163. Springer Verlag, 1995.

[8] Peter F. Stadler. Landscapes and their correlation functions. *Journal of Mathematical Chemistry*, 20:1–45, 1996.

[9] Darrell Whitley and Andrew M. Sutton. Partial neighborhoods of elementary landscapes. In *Proceedings of the Genetic and Evolutionary Computation Conference*, Montreal, CA, July 2009.