

Local search for the probabilistic traveling salesman problem: correction to the 2-p-opt and 1-shift algorithms

Leonora Bianchi*, Joshua Knowles†, Neill Bowler‡

September 22, 2003

Abstract

The probabilistic traveling salesman problem concerns the best way to visit a set of customers located in some metric space, where each customer requires a visit only with some known probability. A solution to this problem is an a priori tour which visits all customers, and the objective is to minimize the expected length of the a priori tour over all customer subsets, assuming that customers in any given subset must be visited in the same order as they appear in the a priori tour. This problem belongs to the class of stochastic vehicle routing problems, a class which has received increasing attention in recent years, and which is of major importance in real world applications.

Several heuristics have been proposed and tested for the probabilistic traveling salesman problem, many of which are a straightforward adaptation of heuristics for the classical traveling salesman problem. In particular, two local search algorithms (2-p-opt and 1-shift) were introduced by Bertsimas.

In a previous report we have shown that the expressions for the cost evaluation of 2-p-opt and 1-shift moves, as proposed by Bertsimas, are not correct. In this paper we derive the correct versions of these expressions, and we show that the local search algorithms based on these expressions perform significantly better than those exploiting the incorrect expressions.

Keywords: Combinatorial optimization, probabilistic traveling salesman problem, heuristics, local search

1 Statement of the problem

The probabilistic traveling salesman problem (PTSP) is a traveling salesman problem (TSP) where each customer $i = 1, 2, \dots, n$ has a given probability p_i of requiring a visit. A feasible solution to the PTSP is an a priori tour τ visiting all customers. The a priori tour is interpreted as providing a template for constructing a tour of any particular subset of customers. More precisely, for any given set of customers requiring a visit, they should be visited in the same order that they appear in the a priori tour. This tour construction strategy enables, for example, a supermarket supplier, to rapidly adapt its daily delivery rounds to just those customers that require a supply that day. Thus, given this interpretation of the a priori tour, the objective of the PTSP is to minimize the expected length of the a priori tour over all customer subsets.

*IDSIA, Strada Cantonale, Galleria 2, CH-6928 Manno, Switzerland. E-mail address: leonora@idsia.ch

†IRIDIA, CP 194/6, Université Libre de Bruxelles, Avenue Franklin Roosevelt 50, 1050 Brussels, Belgium. E-mail address: jknowles@ulb.ac.be

‡Met Office, FitzRoy Road, Exeter, EX1 3PB, UK. E-mail address: Neill.Bowler@metoffice.com

The PTSP is an NP-hard problem [5, 2], and was introduced in 1985 in [12]. Heuristic approaches to this problem have been analyzed in [14, 2, 4] and an exact algorithm has been developed for small instances (up to 50 customers) in [13]. Theoretical properties of the PTSP (bounds, asymptotic analysis) have been derived in [5] and [4]. The PTSP has also recently been tackled, [6, 7, 10, 9], especially with new algorithmic approaches based on metaheuristics [11].

In this paper we restrict our attention to the homogeneous PTSP with symmetric distances, that is, a PTSP where each customer has the same probability p of requiring a visit, independently of the others, and where for every couple of customers i, j , the distance of traveling from i to j is equal to the distance of traveling from j to i .

Given an a priori tour $\tau = \tau(1), \tau(2), \dots, \tau(n)$, its expected length (that is, the objective function), is computed in $O(n^2)$ by the following expression [12]:

$$E[L(\tau)] = \sum_{i=1}^n \sum_{r=1}^{n-1} p^2 (1-p)^{r-1} d(\tau(i), \tau(i+r)). \quad (1)$$

Throughout the paper we use the convention that, for any customer $i \in \{1, 2, \dots, n\}$ and for any integer $r \in \{1, 2, \dots, n-1\}$,

$$i \pm r = \begin{cases} (i \pm r) \bmod n, & \text{if } i \pm r \neq 0 \text{ and if } i \pm r \neq n \\ n, & \text{otherwise.} \end{cases} \quad (2)$$

The meaning of equation (1) is as follows. In each term of the sum the distance between the i^{th} city $\tau(i)$ and the $(i+r)^{\text{th}}$ city $\tau(i+r)$ is weighted by the probability that the two nodes require a visit (p^2) while the $r-1$ nodes between them do not require a visit ($(1-p)^{r-1}$). In other words, each arc $(\tau(i), \tau(i+r))$ is weighted by a probability coefficient $p^2(1-p)^{r-1}$.

Using the full evaluation function (equation (1)) to evaluate the cost of moves in a local search is very expensive and so it would be desirable to have a ‘delta’ evaluation function (to calculate the cost difference between two neighboring tours) of a much lower complexity. To this end, Bertsimas [2, 4] has previously proposed delta evaluation expressions for two local search moves, namely 2-p-opt and 1-shift, described below, which compute the cost evaluations of the entire $O(n^2)$ -sized neighborhood of a tour in $O(n^2)$ time. Unfortunately, as we have shown in [8], they do not correctly calculate the change in expected tour length. It is the purpose of this paper to derive correct versions of them.

The remainder of the paper is organized as follows. In the next section we describe the 2-p-opt move and we derive the correct recursive delta evaluation expression. Section 3 follows a similar course but with respect to the 1-shift operator. In section 4 we perform computational experiments showing the effects of using the incorrect expressions in [2] and in [4].

2 The 2-p-opt

The 2-p-opt local search was introduced by Bertsimas in [2], later published in an article by Bertsimas and Howell [4], and applied again by Bertsimas, Chervi and Peterson in [3].

Given an a priori tour τ , its 2-p-opt neighborhood is the set of tours obtained by reversing a section of τ (that is, a set of consecutive nodes) and adjusting the arcs adjacent to the reversed section, as for example in Figure 1.

2-P-OPT

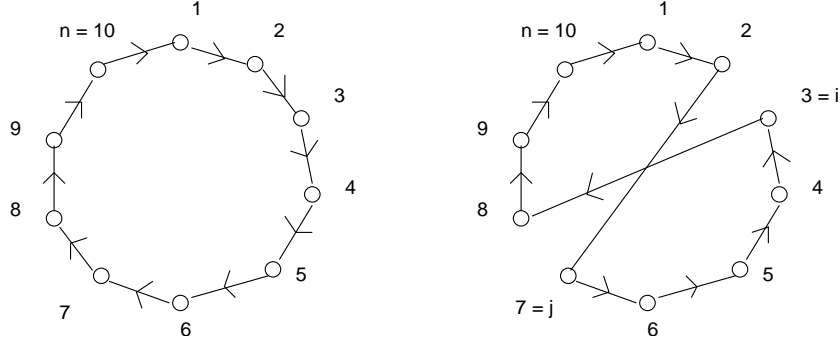


Figure 1: Tour $\tau = 1, 2, \dots, i, i + 1, \dots, j, j + 1, \dots, n$ (left) and tour $\tilde{\tau} = 1, 2, \dots, i - 1, j, j - 1, \dots, i, j + 1, \dots, n$ (right) obtained from τ by reversing the section $(i, i + 1, \dots, j)$, with $n = 10, i = 3, j = 7$.

2.1 Derivation of the correct expressions for the cost of a 2-p-opt move

Consider, without loss of generality, a tour $\tau = 1, 2, \dots, i, i + 1, \dots, j, j + 1, \dots, n$ and a tour $\tilde{\tau} = 1, 2, \dots, j, j - 1, \dots, i, j + 1, \dots, n$ obtained by reversing a section $(i, i + 1, \dots, j)$ of τ , where $i \in \{1, 2, \dots, n\}, j \in \{1, 2, \dots, n\}$, and $i \neq j$. Let $\Delta E_{i,j}$ denote the change in the expected length $E[L(\tilde{\tau})] - E[L(\tau)]$.

In the following, the correct recursive formula for $\Delta E_{i,j}$ is derived for the 2-p-opt move set. This derivation consists in first developing a formula for $\Delta E_{i,j}$ and then finding a recursive form of this.

Let $j = i + k$ (we are using the notation expressed by (2)). For $k = 1$ we believe the expression given in [2] and in [4] for $\Delta E_{i,i+k}$ (equation (18)) is correct. Therefore, we focus on the case $k \geq 2$ (and $k \leq n - 2$). Let us state a few definition.

Definition 1 *inside* = $\{i, \dots, i + k\}$, that is, the ‘inside’ of the reversed section

Definition 2 *outside* = all nodes – inside

Definition 3 $\Delta[S, T]_{i,i+k}$ = interaction between a set of nodes S and a set of nodes T , that is, the contribution to $\Delta E_{i,i+k}$ due to the arcs connecting nodes in S with nodes in T and vice versa.

The change in expected tour length $\Delta E_{i,i+k}$ is caused by the fact that arcs between some pairs of nodes are given different weights in the two tours τ and $\tilde{\tau}$. In other words, $\Delta E_{i,i+k}$ is due to the non-zero interactions between the nodes of the problem. It is not difficult to verify that the interactions between *outside* and *outside* and between *inside* and *inside* are zero. Therefore, the only contribution to $\Delta E_{i,i+k}$ is given by the interactions between *inside* and *outside*, that is,

$$\Delta E_{i,i+k} = \Delta[\textit{inside}, \textit{outside}]_{i,i+k}. \quad (3)$$

Let $q = 1 - p$. The contribution from arcs between *inside* and *outside* to the expected tour length of the undisturbed tour τ is

$$p^2 \sum_{t=1}^{k+1} q^{t-1} \sum_{r=1}^{n-k-1} q^{r-1} [d(i-r, i+t-1) + d(i+k-t+1, i+k+r)]. \quad (4)$$

In the disturbed tour $\tilde{\tau}$, the contribution of the same arcs is

$$p^2 \sum_{t=1}^{k+1} q^{t-1} \sum_{r=1}^{n-k-1} q^{r-1} [d(i+t-1, i+k+r) + d(i-r, i+k-t+1)]. \quad (5)$$

By subtracting (4) from (5), we obtain the interaction between *inside* and *outside*, that is, by equation (3):

$$\begin{aligned} \Delta E_{i,i+k} = p^2 \sum_{t=1}^{k+1} q^{t-1} \sum_{r=1}^{n-k-1} q^{r-1} [d(i+t-1, i+k+r) + d(i-r, i+k-t+1) \\ - d(i-r, i+t-1) - d(i+k-t+1, i+k+r)]. \end{aligned} \quad (6)$$

Consider now the case when the section from $i+1$ to $i+k-1$ of τ is reversed. Then the difference in expected length is

$$\begin{aligned} \Delta E_{i+1,i+k-1} = p^2 \sum_{t=1}^{k-1} q^{t-1} \sum_{r=1}^{n+1-k} q^{r-1} [d(i+t, i+k+r-1) + d(i+1-r, i+k-t) \\ - d(i-r+1, i+t) - d(i+k-t, i+k+r-1)]. \end{aligned} \quad (7)$$

From the two above expressions (6) and (7) it is possible to extract a recursive equation which relates $\Delta E_{i,i+k}$ to $\Delta E_{i+1,i+k-1}$

$$\Delta E_{i,i+k} = \Delta E_{i+1,i+k-1} + \epsilon. \quad (8)$$

Observe that the difference between $\Delta E_{i,i+k}$ and $\Delta E_{i+1,i+k-1}$ will only involve arcs which are connected to nodes i and $i+k$, that is,

$$\begin{aligned} \epsilon = & \text{ terms with } i \text{ and } i+k \text{ in } \Delta E_{i,i+k} \\ & - \text{ terms with } i \text{ and } i+k \text{ in } \Delta E_{i+1,i+k-1}. \end{aligned} \quad (9)$$

So, let us extract the terms which contain the appropriate arcs from (6) and (7). The terms with i in $\Delta E_{i,i+k}$ are (from equation (6))

$$p^2 \sum_{r=1}^{n-k-1} q^{r-1} [(1-q^k)d(i, i+k+r) + (q^k-1)d(i-r, i)], \quad (10)$$

and the terms with $i+k$ in $\Delta E_{i,i+k}$ are (from equation (6))

$$p^2 \sum_{r=1}^{n-k-1} q^{r-1} [(q^k-1)d(i+k, i+k+r) + (1-q^k)d(i-r, i+k)]. \quad (11)$$

The terms with i in $\Delta E_{i+1,i+k-1}$ are (from equation (7))

$$p^2 \sum_{t=1}^{k-1} q^{t-1} [(q^{n-k}-1)d(i+t, i) + (1-q^{n-k})d(i, i+k-t)], \quad (12)$$

and the terms with $i+k$ in $\Delta E_{i+1,i+k-1}$ are (from equation (7))

$$p^2 \sum_{t=1}^{k-1} q^{t-1} [(1-q^{n-k})d(i+t, i+k) + (q^{n-k}-1)d(i+k-t, i+k)]. \quad (13)$$

By subtracting (12) and (13) from (10) and (11) we obtain

$$\begin{aligned} \epsilon = p^2 \{ & (1 - q^k) \sum_{r=1}^{n-k-1} q^{r-1} [d(i, i+k+r) - d(i-r, i) + d(i+k, i+k+r) - d(i-r, i+k)] \\ & + (1 - q^{n-k}) \sum_{r=1}^{k-1} q^{r-1} [d(i+r, i) - d(i, i+k-r) - d(i+r, i+k) + d(i+k-r, i+k)] \}. \end{aligned} \quad (14)$$

Now, let us consider the two dimensional matrices of partial results A and B , as given in [4]

$$A_{i,k} = \sum_{r=k}^{n-1} q^{r-1} d(i, i+r) \quad \text{and} \quad B_{i,k} = \sum_{r=k}^{n-1} q^{r-1} d(i-r, i), \quad 1 \leq k \leq n-1, \quad 1 \leq i \leq n. \quad (15)$$

By expressing equation (14) in terms of A and B we obtain

$$\begin{aligned} \epsilon = p^2 [& (q^{-k} - 1)A_{i,k+1} + (q^k - 1)(B_{i,1} - B_{i,n-k}) \\ & + (q^k - 1)(A_{i+k,1} - A_{i+k,n-k}) + (q^{-k} - 1)B_{i+k,k+1} \\ & + (1 - q^{n-k})(A_{i,1} - A_{i,k}) + (1 - q^{k-n})B_{i,n-k+1} \\ & + (1 - q^{k-n})A_{i+k,n-k+1} + (1 - q^{n-k})(B_{i+k,1} - B_{i+k,k})]. \end{aligned} \quad (16)$$

Finally, the above expression, together with equation (8) leads to the following recursive equation for the change in expected tour length

$$\begin{aligned} \Delta E_{i,i+k} = \Delta E_{i+1,i+k-1} + p^2 [& (q^{-k} - 1)A_{i,k+1} + (q^k - 1)(B_{i,1} - B_{i,n-k}) \\ & + (q^k - 1)(A_{i+k,1} - A_{i+k,n-k}) + (q^{-k} - 1)B_{i+k,k+1} \\ & + (1 - q^{n-k})(A_{i,1} - A_{i,k}) + (1 - q^{k-n})B_{i,n-k+1} \\ & + (1 - q^{k-n})A_{i+k,n-k+1} + (1 - q^{n-k})(B_{i+k,1} - B_{i+k,k})]. \end{aligned} \quad (17)$$

This expression differs from the one proposed by Bertsimas in [2] in the sign of the last four terms inside the squared brackets, and from the expression proposed by Bertsimas-Howell in [4] also in the first term on the right side. Recall that equation (17) is valid for $k \geq 2$. In the case $k = 1$ the expression for $\Delta E_{i,i+k}$, as published in [2] and [4], is correct, and it is the following

$$\Delta E_{i,i+1} = p^3 [q^{-1}A_{i,2} - (B_{i,1} - B_{i,n-1}) - (A_{i+1,1} - A_{i+1,n-1}) + q^{-1}B_{i+1,2}]. \quad (18)$$

2.2 The 2-p-opt local search

The 2-p-opt local search proposed by Bertsimas and also implemented by us proceeds in two phases. The first phase consists of computing $\Delta E_{i,i+1}$ for every value of i (by means of equation (18)), and at the same time accumulating the two matrices of partial results A and B .

Note that since the computation of $\Delta E_{i,i+1}$ only involves two rows of A and B , one can proceed to the next pair of nodes without recomputing each entire matrix. Each time a negative $\Delta E_{i,i+1}$ is encountered, one immediately switches the two nodes involved. The n calculations of phase one require $O(n)$ time apiece, or $O(n^2)$ time in all. At the end of this phase, an a priori tour is reached for which every $\Delta E_{i,i+1}$ is positive,

```

procedure FirstPhase() of 2-p-opt and 1-shift local search
   $\tau :=$  starting solution
  for ( $i = 1, 2, \dots, n$ )
    compute rows  $i$  and  $i + 1$  of matrices  $A$  and  $B$ 
    compute  $\Delta E_{i,i+1}$  according to equation (18)
    if ( $\Delta E_{i,i+1} < 0$ )
       $\tau :=$  the tour obtained from  $\tau$  by switching node  $\tau(i)$  with node  $\tau(i + 1)$ 
    end if
  end for
  if (the starting solution has changed)
    re-compute the full matrices  $A$  and  $B$ 
  end if
end procedure

```

Figure 2: Pseudocode description of the first phase of the 2-p-opt and 1-shift algorithms, where only swapping of two consecutive nodes are considered.

and the matrices A and B are complete and correct for that tour. Figure 2 shows the pseudocode describing this first phase of the local search, which, as will be clear in section 3, is valid also for the 1-shift algorithm. The second phase of the local search consists of computing $\Delta E_{i,j}$ recursively by means of equation (17). Since each $\Delta E_{i,j}$ in phase two is computed in $O(1)$ time, this phase, and thus the entire 2-p-opt checking sequence, is performed in $O(n^2)$. See Figure 3 for the pseudocode of the 2-p-opt local search.

Note that this procedure is much faster than using the full evaluation function (equation (1)) for the cost of each 2-p-opt move. In fact, the full evaluation would require $O(n^2)$ time for the cost calculation of each move, and $O(n^4)$ time for the entire 2-p-opt checking sequence (since the neighborhood size of 2-p-opt is $O(n^2)$).

3 The 1-shift

The 1-shift local search was introduced by Bertsimas in [2], later published in an article by Bertsimas and Howell [4], and applied again by Bertsimas, Chervi and Peterson in [3]. Given an a priori tour τ , its 1-shift neighborhood is the set of tours obtained by moving a node which is at position i to position j of the tour, with the intervening nodes being shifted backwards one space accordingly, as for example in Figure 4.

3.1 Derivation of the correct expressions for the cost of a 1-shift move

Consider, without loss of generality, a tour $\tau = 1, 2, \dots, i, i + 1, \dots, j, j + 1, \dots, n$ and a tour $\tilde{\tau} = 1, 2, \dots, i - 1, i + 1, \dots, j, i, j + 1, \dots, n$ obtained from τ by moving node i to position j and shifting backwards one space the nodes $i + 1, \dots, j$, where $i \in \{1, 2, \dots, n\}$, $j \in \{1, 2, \dots, n\}$, and $i \neq j$. Let $\Delta E'_{i,j}$ denote the change in the expected length $E[L(\tilde{\tau})] - E[L(\tau)]$.

In the following, the correct recursive formula for $\Delta E_{i,j}$ is derived for the 1-shift move set. This derivation follows a line similar to the one for the 2-p-opt.

Let $j = i + k$ (we are using the notation expressed by (2)). Note that, for $k = 1$, the tour $\tilde{\tau}$ obtained by 1-shift is the same as the one obtained by 2-p-opt, for which we believe [2] and [4] gives the correct expression

```

procedure 2-p-opt()
   $\tau :=$  starting solution
  FirstPhase()
   $k := 2$ 
  (1) while (  $k \leq n - 2$  and time_is_not_over )
     $i := 1$ 
    (2) while (  $i \leq n$  )
      compute  $\Delta E_{i,i+k}$ 
      if (  $\Delta E_{i,i+k} < 0$  )
         $\tau :=$  tour obtained by inverting section  $\tau(i), \tau(i+1), \dots, \tau(i+k)$  of  $\tau$ 
        FirstPhase()
         $k := 2$ 
        go to (1)
      else  $i := i + 1$ 
      end if
    end while (2)
     $k := k + 1$ 
  end while (1)
end procedure

```

Figure 3: Pseudocode description of the 2-p-opt algorithm. The 2-p-opt is implemented as a first-improvement local search, that is, when a neighbor better than the current solution is found, the current solution is immediately updated with the neighbor solution, and the search is restarted. When there are no improving solutions in the neighborhood, or when the time is over, the search stops.

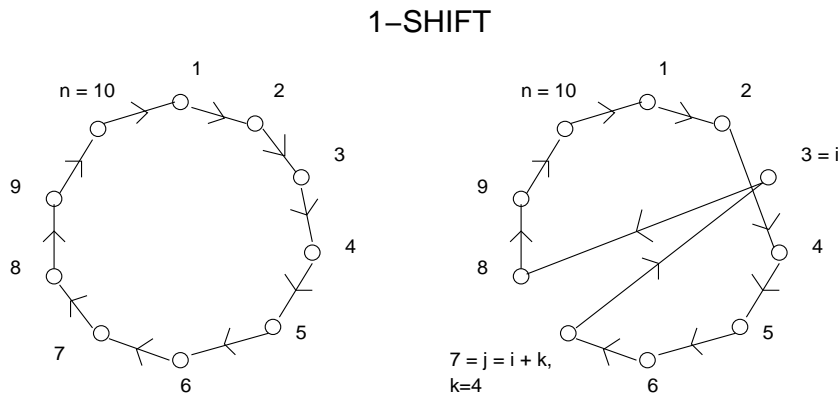


Figure 4: Tour $\tau = 1, 2, \dots, i, i+1, \dots, j, j+1, \dots, n$ (left) and tour $\tilde{\tau} = 1, 2, \dots, i-1, i+1, i+2, \dots, j, i, j+1, \dots, n$ (right) obtained from τ by moving node i to position j and shifting backwards one space the nodes $i+1, \dots, j$, with $n = 10, i = 3, j = 7$.

(equation (18)). Therefore, we focus on the case $k \geq 2$ (and $k \leq n - 2$).

Let us re-define the notions of *inside*, *outside* and *interaction* adapting them to the case of the 1-shift.

Definition 4 $inside = \{i + 1, \dots, i + k\}$, that is, the ‘inside’ of the shifted section

Definition 5 $outside = all\ nodes - inside - \{i\}$

Definition 6 $\Delta'[S, T]_{i, i+k} = interaction\ between\ a\ set\ of\ nodes\ S\ and\ a\ set\ of\ nodes\ T$, that is, the contribution to $\Delta'E'_{i, i+k}$ due to the arcs connecting nodes in S with nodes in T and vice versa.

The change in expected tour length $\Delta E'_{i, i+k}$ is caused by the fact that arcs between some pairs of nodes are given different weights in the two tours τ and $\tilde{\tau}$. In other words, $\Delta E'_{i, i+k}$ is due to the non-zero interaction between the nodes of the problem. It is not difficult to verify that the interaction between *outside* and *outside* and between *inside* and *inside* is zero. Therefore, the only contribution to $\Delta E'_{i, i+k}$ is given by the interaction between $inside \cup \{i\}$ and *outside*, and between $\{i\}$ and *inside*, that is

$$\Delta E'_{i, i+k} = \Delta'[inside \cup \{i\}, outside]_{i, i+k} + \Delta'[i, inside]_{i, i+k}. \quad (19)$$

In the following, we first derive a recursive expression for the interaction between $inside \cup \{i\}$ and *outside* (STEP A), and then for the interaction between $\{i\}$ and *inside* (STEP B).

STEP A: Interaction between $inside \cup \{i\}$ and *outside*.

Let $q = 1 - p$. The contribution from arcs between $inside \cup \{i\}$ and *outside* to the expected tour length of the undisturbed tour τ is

$$p^2 \sum_{t=1}^{k+1} q^{t-1} \sum_{r=1}^{n-k-1} q^{r-1} [d(i-r, i+t-1) + d(i+k-t+1, i+k+r)]. \quad (20)$$

In the disturbed tour $\tilde{\tau}$, the contribution of the same arcs is

$$p^2 \left\{ \sum_{r=1}^{n-k-1} q^{r-1} [q^k d(i-r, i) + d(i, i+k+r)] + \sum_{t=1}^k q^{t-1} \sum_{r=1}^{n-k-1} q^{r-1} [qd(i+k-t+1, i+k+r) + d(i-r, i+t)] \right\}. \quad (21)$$

By subtracting (20) from (21), we obtain the interaction between $inside \cup \{i\}$ and *outside*

$$\begin{aligned} \Delta'[inside \cup \{i\}, outside]_{i, i+k} &= p^2 \left\{ \sum_{r=1}^{n-k-1} q^{r-1} [(q^k - 1)d(i-r, i) + (1 - q^k)d(i, i+k+r)] \right. \\ &\quad \left. + \sum_{t=1}^k q^{t-1} \sum_{r=1}^{n-k-1} q^{r-1} [(1 - q)d(i-r, i+t) + (q-1)d(i+k-t+1, i+k+r)] \right\}. \end{aligned} \quad (22)$$

Consider now the case where $j = i + k - 1$. Then the interaction between $inside \cup \{i\}$ and *outside* is

$$\begin{aligned} \Delta'[inside \cup \{i\}, outside]_{i, i+k-1} &= p^2 \left\{ \sum_{r=1}^{n-k} q^{r-1} [(q^{k-1} - 1)d(i-r, i) + (1 - q^{k-1})d(i, i+k-1+r)] \right. \\ &\quad \left. + \sum_{t=1}^{k-1} q^{t-1} \sum_{r=1}^{n-k} q^{r-1} [(1 - q)d(i-r, i+t) + (q-1)d(i+k-t, i+k-1+r)] \right\}. \end{aligned} \quad (23)$$

From the two above expressions (22) and (23) it is possible to extract a recursive equation which relates $\Delta'[\textit{inside} \cup \{i\}, \textit{outside}]_{i,i+k}$ to $\Delta'[\textit{inside} \cup \{i\}, \textit{outside}]_{i,i+k-1}$

$$\Delta'[\textit{inside} \cup \{i\}, \textit{outside}]_{i,i+k} = \Delta'[\textit{inside} \cup \{i\}, \textit{outside}]_{i,i+k-1} + \delta, \quad (24)$$

that we complete by expressing δ in terms of the matrices A and B . Observe that the difference between $\Delta'[\textit{inside} \cup \{i\}, \textit{outside}]_{i,i+k}$ and $\Delta'[\textit{inside} \cup \{i\}, \textit{outside}]_{i,i+k-1}$ will only involve arcs which are connected to nodes i and $i+k$, that is,

$$\delta = \begin{aligned} & \text{terms with } i \text{ and } i+k \text{ in } \Delta'[\textit{inside} \cup \{i\}, \textit{outside}]_{i,i+k} \\ & - \text{terms with } i \text{ and } i+k \text{ in } \Delta'[\textit{inside} \cup \{i\}, \textit{outside}]_{i,i+k-1}. \end{aligned} \quad (25)$$

So, let us extract the terms which contain the appropriate arcs from (22) and (23). The terms with i and $i+k$ in $\Delta'[\textit{inside} \cup \{i\}, \textit{outside}]_{i,i+k}$ are (from (22)):

$$\begin{aligned} p^2 \sum_{r=1}^{n-k-1} q^{r-1} [(q^k - 1)d(i-r, i) + (1 - q^k)d(i, i+k+r) \\ + (q-1)d(i+k, i+k+r) + (q^{k-1} - q^k)d(i-r, i+k)], \end{aligned} \quad (26)$$

and the terms with i and $i+k$ in $\Delta'[\textit{inside} \cup \{i\}, \textit{outside}]_{i,i+k-1}$ are (from (23)):

$$\begin{aligned} p^2 \left\{ \sum_{t=1}^{k-1} q^{t-1} [q^{n-k-1}(1-q)d(i+k, i+t) + (q-1)d(i+k-t, i+k)] \right. \\ \left. + \sum_{r=1}^{n-k} q^{r-1} [(q^{k-1} - 1)d(i-r, i) + (1 - q^{k-1})d(i, i+k+r-1)] \right\}. \end{aligned} \quad (27)$$

Now, by subtracting (27) from (26) and by applying the definition of A and B (15) we obtain the following expression

$$\begin{aligned} \delta = p^2 [(q^k - 1)(B_{i,1} - B_{i,n-k}) + (q^{-k} - 1)A_{i,k+1} \\ + (1 - q^{k-1})(B_{i,1} - B_{i,n-k+1}) + (1 - q^{-k+1})A_{i,k} \\ + (q-1)(A_{i+k,1} - A_{i+k,n-k}) + (q^{-1} - 1)B_{i+k,k+1} \\ + (1-q)(B_{i+k,1} - B_{i+k,k}) + (1 - q^{-1})A_{i+k,n-k+1}], \end{aligned} \quad (28)$$

which completes the recursive expression (24).

□ end of **STEP A**.

Now we need to evaluate the interaction between $\{i\}$ and *inside*, and this is done in **STEP B**.

STEP B: The contribution from arcs between $\{i\}$ and *inside* to the expected tour length of the undisturbed tour τ is

$$p^2 \sum_{t=1}^k [q^{t-1}d(i, i+t) + q^{n-k-1}q^{t-1}d(i+k-t+1, i)]. \quad (29)$$

In the disturbed tour $\bar{\tau}$, the contribution of the same arcs is

$$p^2 \sum_{t=1}^k [q^{t-1} d(i+k+1-t, i) + q^{n-k-1} q^{t-1} d(i, i+t)]. \quad (30)$$

By subtracting (29) from (30), we obtain the interaction between $\{i\}$ and *inside*

$$\Delta'[i, \textit{inside}]_{i, i+k} = p^2 (q^{n-k-1} - 1) \sum_{t=1}^k q^{t-1} [d(i, i+t) - d(i+k+1-t, i)]. \quad (31)$$

Consider now the case where $j = i+k-1$. Then the interaction between $\{i\}$ and *inside* is

$$\Delta'[i, \textit{inside}]_{i, i+k-1} = p^2 (q^{n-k} - 1) \sum_{t=1}^{k-1} q^{t-1} [d(i, i+t) - d(i+k-t, i)]. \quad (32)$$

From the two above expressions (31) and (32) it is possible to extract a recursive equation which relates $\Delta'[i, \textit{inside}]_{i, i+k}$ to $\Delta'[i, \textit{inside}]_{i, i+k-1}$

$$\Delta'[i, \textit{inside}]_{i, i+k} = \Delta'[i, \textit{inside}]_{i, i+k-1} + \gamma, \quad (33)$$

which implies

$$\gamma = \Delta'[i, \textit{inside}]_{i, i+k} - \Delta'[i, \textit{inside}]_{i, i+k-1}. \quad (34)$$

Now, by subtracting (32) from (31) and by applying the definition of A and B (15) we obtain the following expression

$$\begin{aligned} \gamma = p^2 [& (q^{n-k-1} - 1)(A_{i,1} - A_{i, k+1}) + (q^{k+1-n} - 1)B_{i, n-k} \\ & + (1 - q^{n-k})(A_{i,1} - A_{i, k}) + (1 - q^{k-n})B_{i, n-k+1}], \end{aligned} \quad (35)$$

which completes the recursive expression (33).

□ end of **STEP B**.

Finally, by considering equations (19), (24) and (33), we can write

$$\Delta' E_{i, i+k} = \Delta' E_{i, i+k-1} + \delta + \gamma,$$

which, by considering the expressions for δ (28) and γ (35), becomes,

$$\begin{aligned} \Delta' E_{i, i+k} = & \Delta' E_{i, i+k-1} + p^2 [(q^{-k} - q^{n-k-1})A_{i, k+1} \\ & + (q^k - q^{k-1})B_{i, 1} + (q^{k+1-n} - q^k)B_{i, n-k} \\ & + (q^{n-k-1} - q^{n-k})A_{i, 1} + (q^{n-k} - q^{1-k})A_{i, k} \\ & + (q^{k-1} - q^{k-n})B_{i, n-k+1} \\ & + (q-1)(A_{i+k, 1} - A_{i+k, n-k}) + (q^{-1} - 1)B_{i+k, k+1} \\ & + (1 - q^{-1})A_{i+k, n-k+1} + (1 - q)(B_{i+k, 1} - B_{i+k, k})]. \end{aligned} \quad (36)$$

```

procedure 1-shift()
   $\tau :=$  starting solution
  FirstPhase()
  while (not_in_a_local_optimum and time_is_not_over)
    initialize_best_neighbor()
    for ( $i = 1, 2, \dots, n$ )
      for ( $k = 2, \dots, n - 2$ )
        compute  $\Delta' E_{i,i+k}$ 
        update_best_neighbor()
      end for
    end for
    if (best_neighbor_is_better_than_ $\tau$ )
       $\tau :=$  tour obtained from  $\tau$  by inserting  $\tau(i)$  after  $\tau(i+k)$ 
      FirstPhase()
    else STOP
    end if
  end while
end procedure

```

Figure 5: Pseudocode description of the 1-shift algorithm. The 1-shift is implemented as a best-improvement local search, that is, the whole neighborhood is explored and the current solution is updated with the best (improving) neighbor solution. When there are not improving solutions in the neighborhood, or when the time is over, the search stops.

This expression differs from the one proposed by Bertsimas in [2] and [4] by the first six terms on the right side inside the square brackets, and it can be further simplified to

$$\begin{aligned}
\Delta' E_{i,i+k} = & \Delta' E_{i,i+k-1} + p^2[(q^{n-k} - q^{-k})(qA_{i,k} - A_{i,k+1}) \\
& + (q^{k-n} - q^{k-1})(qB_{i,n-k} - B_{i,n-k+1}) \\
& + (1 - q^{-1})(q^k B_{i,1} - B_{i+k,k+1}) \\
& + (q^{-1} - 1)(q^{n-k} A_{i,1} - A_{i+k,n-k+1}) \\
& + (q - 1)(A_{i+k,1} - A_{i+k,n-k}) \\
& + (1 - q)(B_{i+k,1} - B_{i+k,k})].
\end{aligned} \tag{37}$$

Recall that equation (37) is valid for $k \geq 2$, and that in the case $k = 1$ the expression for $\Delta' E_{i,i+k}$ as published in [2] and [4] (equation (18)) is correct.

3.2 The 1-shift local search

The 1-shift algorithm follows the same lines as the 2-p-opt algorithm: all phase one computations, including the accumulation of matrices A and B , proceed in the same way (see Figure 2). The second phase of the local search consists of computing $\Delta E'_{i,j}$ recursively by means of equation (36). Like 2-p-opt, since each $\Delta E'_{i,j}$ in phase two is computed in $O(1)$ time, this phase, and thus the entire 1-shift checking sequence, is performed

in $O(n^2)$. Using the full evaluation, the entire 1-shift checking requires $O(n^4)$ time. See Figure 5 for the pseudocode of the 1-shift local search.

4 Computational results

4.1 Experimental setup

We considered instances with $n = 100$ and with customers coordinates uniformly and independently distributed on the square $[0, 1]^2$. For generating random instances we used the Instance Generator Code of the 8th DIMACS Implementation Challenge at <http://www.research.att.com/~dsj/chtsp/download.html>. Distances were computed with the Euclidean metric. For each of the common demand probabilities $p = 0.1, 0.2, \dots, 0.9$ we generated 10 problems. Experiments were run on a PowerPC G4 400MHz, the code has been written in C++ and it is available at <http://www.idsia.ch/~leo>.

4.2 Comparison between correct and incorrect local search

We tested the 2-p-opt local search algorithm with three delta objective evaluation expressions: the correct one (equation (17)), the incorrect one published in [2], and the incorrect one published in [4]. We also tested two versions of the 1-shift local search algorithm: one with the correct delta objective evaluation (equation (36)) and one with the incorrect delta objective evaluation published in [2] (which is the same as that published in [4]). In the following, we denote by the prefix ‘C’, the algorithms involving the the correct delta objective evaluations (e.g., C-1shift), and by the prefix ‘I’ the ones implementing the incorrect delta objective evaluations as published in [4] (e.g., I-1shift). Since we obtained similar results with the incorrect expressions of [2] and [4], in the following we only show results obtained with the delta objective evaluation expression of [4].

For each C- and I-local search, we considered two types of starting solutions, generated with two different solution construction heuristics, namely the the Radial Sort and the Space Filling Curve heuristic. These solution construction heuristics have been extensively applied in previously published papers on the PTSP. Radial Sort builds a tour by sorting customers by angle with respect to the ‘center of mass’ of the customer spatial distribution. The ‘center of mass’ coordinates are computed by averaging over the customers coordinates. The Space Filling curve heuristic uses a function of the class of Sierpinski curves to map customers coordinates to numbers, representing the position of customers inside the a priori solution (for details, see [1]).

The absolute solution quality of the tested algorithms was evaluated with respect to near-optimal solutions which were heuristically generated in the following way. Consider three solution construction heuristics, namely Radial Sort, Space Filling Curve and Random. The Random heuristic builds a solution by generating a random permutation of the customers, and the other two heuristics are described in the previous paragraph. Consider as local search heuristics the application of both C-1shift and C-2-p-opt one after the other. This results in two local search heuristics (first C-1shift and after C-2-p-opt and vice versa). By combining the three solution construction with the two local search heuristics one obtains six different heuristics. Return the best solution found by these six heuristic combinations.

Table 1 shows the results obtained with the correct algorithms, and the percent increase resulting from the use of Bertsimas’ expressions. CPU running times in seconds are shown in parenthesis. In each line of the table the data show the average over 10 different instances. For each instance, each algorithm was run only once. Note, to evaluate the I-algorithms we checked the final solutions obtained using the full evaluation

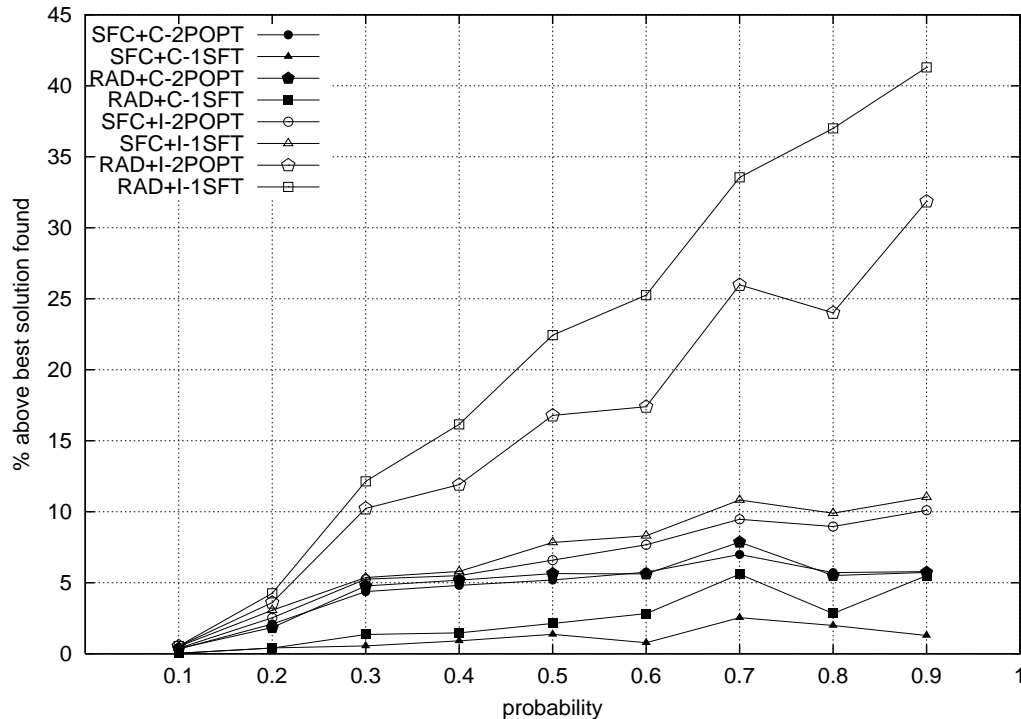


Figure 6: percent distance from the best solution found for C-(Correct) and I-(Incorrect) local search heuristics, combined with the Radial Sort and the Space Filling Curve solution construction heuristics. Black symbols refer to C-algorithms, and white symbols refer to I-algorithms. Each point is an average over 10 different Euclidean instances with 100 customers and customer coordinates in the unit square. The customer probability is on the horizontal axis.

equation (1). For the C-algorithms we confirm that this check was unnecessary because the algorithms exactly calculated the improvements obtained by the local search. In Figure 6 the relative difference between each algorithmic combination and the best near-optimal solution found is shown in a graphical form. It is clear from both Table 1 and Figure 6 that I-algorithms always give worse solution quality than to C-algorithms, as expected. From Table 1 we see that for small customer probability ($p=0.1$) the error in Bertsimas' expressions results in a very small worsening effect, smaller than 1%, for all heuristic combinations. The table shows that the higher the probability, the worse are I-algorithms with respect to C-algorithms. This effect is bigger for the algorithms involving the Radial Sort, where I-algorithms obtain results more than 30% worse than C-algorithms.

The running times of all algorithmic combinations are under the second, and also obtaining the best near-optimal result only took a time of the order of the second. Running times of I-algorithms (not shown), are very similar to those of C-algorithms, except for some cases, where the solution quality of a I-algorithm is very bad. In those cases the I-local search would stop improving very early, because it starts cycling between two solutions, never stopping until the maximum allowed run time is exceeded. In order to understand why cycling may occur in the I-local search, consider for instance the I-2-p-opt local search. Given a solution τ ,

p	SFC+2POPT		SFC+1SFT		RAD+2POPT		RAD+1SFT		best
	C	+I%	C	+I%	C	+I%	C	+I%	C
0.1	2.988 (0.20)	0.1	2.980 (0.29)	0.5	2.990 (0.14)	0.2	2.980 (0.21)	0.5	2.979 (1.28)
0.2	4.127 (0.15)	0.4	4.059 (0.33)	2.6	4.117 (0.22)	1.7	4.058 (0.44)	3.8	4.042 (1.06)
0.3	5.089 (0.11)	0.8	4.903 (0.33)	4.8	5.109 (0.33)	5.2	4.942 (0.49)	10.6	4.876 (1.11)
0.4	5.757 (0.10)	0.7	5.542 (0.28)	4.8	5.777 (0.32)	6.4	5.573 (0.44)	14.5	5.492 (1.25)
0.5	6.441 (0.12)	1.3	6.206 (0.31)	6.4	6.468 (0.39)	10.5	6.253 (0.52)	19.9	6.122 (1.34)
0.6	7.065 (0.11)	1.8	6.733 (0.31)	7.5	7.057 (0.39)	11.2	6.871 (0.50)	21.8	6.682 (1.28)
0.7	7.436 (0.12)	2.3	7.126 (0.32)	8.1	7.496 (0.46)	16.8	7.339 (0.57)	26.5	6.950 (1.51)
0.8	7.819 (0.11)	3.1	7.543 (0.27)	7.8	7.804 (0.48)	17.5	7.606 (0.55)	33.2	7.396 (1.62)
0.9	8.340 (0.12)	4.1	7.985 (0.30)	9.6	8.335 (0.49)	24.7	8.316 (0.58)	33.9	7.883 (0.47)

Table 1: Average solution values obtained with C-(Correct) local search algorithms, and percent increase resulting from the use of I-(Incorrect) local search algorithms. The column named ‘best’ shows the values of the best near-optimal solutions found heuristically, as described in section 4.2. For each probability p the average result over 10 different instances is reported (problems and experiments are the same as the ones used for obtaining Figure 6). CPU run times in seconds on a PowerPC G4 400MHz are shown in parentheses.

suppose that the section $\tau(i), \tau(i+1), \dots, \tau(i+k)$, with $k \geq 2$ is reversed, leading to a new solution $\tilde{\tau}$, with $\tilde{\tau}(i) = \tau(i+k), \tilde{\tau}(i+1) = \tau(i+k-1), \dots, \tilde{\tau}(i+k) = \tau(i)$. This means that $\Delta E_{i,k}(\tau) < 0$. Now, due to the incorrectness of the expression for evaluating $\Delta E_{i,k}$ of [2, 4], it may happen that also $\Delta E_{i,k}(\tilde{\tau}) < 0$, and that the section $\tilde{\tau}(i), \tilde{\tau}(i+1), \dots, \tilde{\tau}(i+k)$ is reversed, going back to solution τ . The same situation may occur in the I-1shift local search.

From Figure 6 we can see that C-1shift consistently gives the best results (among single heuristic combinations), with both the Space Filling Curve and the Radial Sort. The C-2-p-opt heuristic always gives worse results, but it is not clear if it works better with Space Filling Curve or with Radial Sort. When using I-algorithms, it seems that the starting solution is more important than the local search, since, as we see from Figure 6, the Space Filling Curve gives better results than the Radial Sort, no matter if it is combined with I-2-p-opt or with I-1shift. This is a side effect of the inaccuracy resulting from the use of incorrect local searches.

5 Conclusion

In this paper we derived the correct expressions for the efficient computation of the cost of 2-p-opt and 1-shift moves, in the case of homogeneous PTSP. With this derivations we confirm that it is possible to compute the cost evaluations of the entire neighborhood of a solution in $O(n^2)$ time, with both 2-p-opt and 1-shift local search algorithms. Moreover, we showed experimentally the effects of using the incorrect expressions in [2] and in [4], and we conclude that, in addition to giving incorrect estimations of solution quality, the local search algorithms based on these expressions lead to worse solutions than when the correct expressions are used.

6 Acknowledgments

The authors wish to thank Marco Dorigo and Luca Maria Gambardella for their useful comments. This research has been partially supported by the Swiss National Science Foundation project titled “On-line fleet management”, grant 16R10FM, and by the “Metaheuristics Network”, a Research Training Network funded by the Improving Human Potential programme of the CEC, grant HPRN-CT-1999-00106. Joshua Knowles gratefully acknowledges the support of a Marie Curie postdoctoral research fellowship of the CEC, contract number HPMF-CT-2000-00992. The information provided in this paper is the sole responsibility of the authors and does not reflect the Community’s opinion. The Community is not responsible for any use that might be made of data appearing in this publication.

References

- [1] J. J. Bartholdi III and L. K. Platzman. An $O(n \log n)$ planar travelling salesman heuristic based on spacefilling curves. *Operations Research Letters*, 1(4):121–125, 1982.
- [2] D. J. Bertsimas. *Probabilistic Combinatorial Optimization Problems*. PhD thesis, MIT, Cambridge, MA, 1988.
- [3] D. J. Bertsimas, P. Chervi, and M. Peterson. Computational approaches to stochastic vehicle routing problems. *Transportation Science*, 29(4):342–352, 1995.
- [4] D. J. Bertsimas and L. Howell. Further results on the probabilistic traveling salesman problem. *European Journal of Operational Research*, 65(1):68–95, 1993.
- [5] D. J. Bertsimas, P. Jaillet, and A. Odoni. A priori optimization. *Operations Research*, 38(6):1019–1033, 1990.
- [6] L. Bianchi, L. M. Gambardella, and M. Dorigo. An ant colony optimization approach to the probabilistic traveling salesman problem. In J. J. Merelo Guervós et al., editor, *Proceedings of PPSN-VII, Seventh International Conference on Parallel Problem Solving from Nature*, volume 2439 of *Lecture Notes in Computer Science*, pages 883–892. Springer Verlag, Berlin, Germany, 2002.
- [7] L. Bianchi, L. M. Gambardella, and M. Dorigo. Solving the homogeneous probabilistic traveling salesman problem by the ACO metaheuristic. In M. Dorigo, G. Di Caro, and M. Sampels, editors, *Proceedings of ANTS 2002 – From Ant Colonies to Artificial Ants: Third International Workshop on Ant Algorithms*, volume 2463 of *Lecture Notes in Computer Science*, pages 176–187. Springer Verlag, Berlin, Germany, 2002.
- [8] L. Bianchi and J. Knowles. Local search for the probabilistic traveling salesman problem: a proof of the incorrectness of Bertsimas’ proposed 2-p-opt and 1-shift algorithms. Technical Report IDSIA-21-02, IDSIA, Lugano, Switzerland, 2002.
- [9] N. E. Bowler, T. M. A. Fink, and R. C. Ball. Characterization of the probabilistic traveling salesman problem. *Physical Review E*, 68(036703), 2003.
- [10] J. Branke and M. Guntsch. New ideas for applying Ant Colony Optimization to the Probabilistic TSP. In *Proceedings of EvoCOP 2003 – Third European Workshop on Evolutionary Computation in*

Combinatorial Optimization, volume 2611 of *Lecture Notes in Computer Science*, pages 165–175. Springer Verlag, Heidelberg, Germany, 2003.

- [11] D. Corne, M. Dorigo, and F. Glover, editors. *New Ideas in Optimization*. McGraw-Hill, 1999.
- [12] P. Jaillet. *Probabilistic Traveling Salesman Problems*. PhD thesis, MIT, Cambridge, MA, 1985.
- [13] G. Laporte, F. Louveaux, and H. Mercure. An exact solution for the a priori optimization of the probabilistic traveling salesman problem. *Operations Research*, 42(3):543–549, 1994.
- [14] F. A. Rossi and I. Gavioli. Aspects of heuristic methods in the probabilistic traveling salesman problem. In *Advanced School on Statistics in Combinatorial Optimization*, pages 214–227. World Scientific, Singapore, 1987.