Discrete Particle Swarm Optimization for TSP: Theoretical Results and Experimental Evaluations

Matthias Hoffmann, Moritz Mühlenthaler, Sabine Helwig, Rolf Wanka

Department of Computer Science, University of Erlangen-Nuremberg, Germany
Matthias.Hoffmann@informatik.stud.uni-erlangen.de,
{muehlenthaler,sabine.helwig,rwanka}@cs.fau.de

Abstract. Particle swarm optimization (PSO) is a nature-inspired technique originally designed for solving *continuous* optimization problems. There already exist several approaches that use PSO also as basis for solving *discrete* optimization problems, in particular the Traveling Salesperson Problem (TSP). In this paper, (i) we present the first theoretical analysis of a discrete PSO algorithm for TSP which also provides insight into the convergence behavior of the swarm. In particular, we prove that the popular choice of using "sequences of transpositions" as the difference between tours tends to decrease the convergence rate. (ii) In the light of this observation, we present a new notion of difference between tours based on "edge exchanges" and a new method to combine differences by computing their "centroid." This leads to a more PSO-like behavior of the algorithm and avoids the observed slow down effect. (iii) Then, we investigate implementations of our methods and compare them with previous implementations showing the competitiveness of our new approaches.

1 Introduction

The problem. Particle Swarm Optimization (PSO) is a popular metaheuristic designed for solving optimization problems on *continuous* domains. It was introduced by Kennedy and Eberhard [11, 5] and has since then been applied successfully to a wide range of optimization problems. Since the structure of the PSO algorithm is relatively simple, PSO has to some extent been open for theoretical studies of the swarm behavior. Clerk and Kennedy [4], Trelea [17], and Jiang et al. [9] provide analyses of the convergence behavior of particle swarms, which offer some insights on how to select the swarm parameters, and the initial behavior of a swarm has been analyzed in [8]. Inspired by the performance of PSO on continuous optimization problems, several approaches have also been proposed for applying PSO to discrete problems, such as function optimization on binary domains [12], scheduling problems [1], and the Traveling Salesperson Problem (TSP) [2, 18, 6, 14, 15, 19].

The TSP is one of the classical problems in discrete optimization. A wealth of methods specifically tailored for solving TSP has been developed and mathematically and experimentally investigated. A comprehensive overview of this line of research can be found in [7].

But the TSP is also well suited to be approached by (meta-)heuristic methods like PSO. For discrete PSO, new interpretations of "movement" and "velocity" are necessary. The first approach to adapting the PSO scheme to TSP is due to Clerc [2, 3]. However, it turns out that this discrete PSO (DPSO) by itself is not as successful as the original PSO for continuous problems. Consequently, subsequent approaches to solving TSP by PSO typically rely on downstream optimization techniques such as k-OPT [15, 14] and Lin-Kernighan [6] applied after one PSO iteration to improve the quality of the solution obtained by PSO. Unfortunately, whereas these hybrid algorithms are evaluated experimentally by being run on benchmark instances, they are hard to analyze mathematically, and so far, no theoretical insights were gained about the particles' behavior in discrete PSO at all. In fact, the downstream optimization even conceals the performance of plain DPSO.

Our contribution. In this paper, we present the first theoretical analysis of the discrete PSO algorithms of Clerc [2] and Wang et al. [18], which, to some extent, also applies to the approach of Shi et al. [14]. In particular, we provide for the first time theoretical evidence for why the convergence behavior of these DPSO algorithms for the TSP is quite different from what we would expect from the classical PSO for continuous problems. The key insight is that in later stages of the optimization process, the particles are not likely to converge towards the best solution found so far. In fact, we prove that the distance to the best solution even remains more or less constant.

In the light of the theoretical findings, we then propose a novel interpretation of "particle motion" avoiding the convergence problem mentioned above. Our method is similar to computing the midpoint of a discrete line. Additionally, we introduce a new representation of the "velocity" of a particle, which is based on exchanging edges in a potential solution of a TSP instance. We evaluate our proposed DPSO with respect to seven instances from the TSPlib [13] with 52 to 105 cities. In these experimental evaluations, our focus is on the DPSO performance of different velocity representations because we are in this context mainly interested in the performance of the plain DPSO approaches without subsequent local improvement phases. Our results indicate that the combination of the midpoint-based particle motion with the edge-exchange operator outperforms the other operators as well as those methods which suffer from the identified convergence problem.

In order to also compare our DPSO to the previous approaches which use in the PSO iterations additional local improvement heuristics, we hybridize our DPSO iteration with a 2-OPT local optimization applied to the global attractor. Here, we make two observations: The first one is that better performance of the plain DPSO results in a better performance of the hybridized DPSO. And second, for the first time it is clearly documented that the huge performance gains (that is achieved when using local optimization in comparison to the plain DPSO) indicate that the quality of the solutions found by the DPSO algorithms with local optimization is almost completely determined by the quality of the local optimization. In the previous work [14, 18, 19], it is not differentiated between the contribution of the PSO and the additional local improvement methods.

The remainder of this paper is organized as follows. In Sec. 2, we provide relevant background information on TSP and PSO. Sec. 3 contains our theoretical analysis of the particle convergence behavior of discrete PSOs for permutation problems. In Sec. 4, we propose the new discrete PSO for the TSP which uses a centroid-based approach for the particle movement. Sec. 5 provides the experimental results which indicate that our proposed approach outperforms other discrete PSOs for the TSP.

2 Preliminaries

2.1 The Traveling Salesperson Problem

The Traveling Salesperson Problem (TSP) is a classical combinatorial optimization problem. An instance $I = (n, \mathrm{dist})$ of the TSP with n cities $\{1, \dots, n\}$ consists of the distances between each pair of cities, given as an $n \times n$-integer matrix 'dist'. The task is to find a tour with minimum length visiting each city exactly once, including the way back to the initial city. A tour is given as a permutation π of the cities $\{1, \dots, n\}$, where $\pi(i)$ denotes the ith visited city. Hence the set of optimal solutions of an instance $I = (n, \mathrm{dist})$ is

$$\underset{\pi \in S_n}{\operatorname{argmin}} \Big(\mathrm{dist}_{\pi(n),\pi(1)} + \sum_{1 \le i < n} \mathrm{dist}_{\pi(i),\pi(i+1)} \Big) \ ,$$

where S_n is the symmetric group on $\{1, \dots, n\}$ with the usual composition $\circ$ as group operation. The operation $\circ$ is used for exploring the search space. Note that in this formulation the search space consists of *all* elements of S_n, so one specific cycle of length n is represented by many permutations. The advantage of using all elements of S_n for the proposed discrete PSO is that the particles can move around freely in the search space without the danger of encountering permutations which do not correspond to a valid tour. The decision variant of TSP ("given an integer L, is there a tour in I of length at most L?") is NP-hard and hence, TSP can presumably not be solved exactly in polynomial time.

2.2 (Discrete) Particle Swarm Optimization

Introduced by Kennedy and Eberhard [11, 5], particle swarm optimization (PSO) is a population-based metaheuristic that uses a swarm of potential solutions called particles to cooperatively solve optimization problems. Typically, the search space of a problem instance is an n-dimensional rectangle $\mathcal{B} \subseteq \mathbb{R}^n$, and the objective function (often also called fitness function) is of the form $f : \mathbb{R}^n \to \mathbb{R}$. PSO works in *iterations*. In iteration t, each particle i has a *position* $\boldsymbol{x}_i^{(t)} \in \mathcal{B}$

and a *velocity* $\boldsymbol{v}_i^{(t)} \in \mathbb{R}^n$. While moving through the search space, the particles evaluate f at $\boldsymbol{x}_i^{(t)}$. Each particle remembers its best position $\boldsymbol{p}_i$ so far (called local attractor) and the best position $\boldsymbol{p}_{\text{glob}}$ of all particles in the swarm so far (called global attractor). In iteration t, the position and velocity of each particle is updated according to the following *movement equations*:

$$\boldsymbol{v}_i^{(t+1)} = a \cdot \boldsymbol{v}_i^{(t)} + r_{\text{loc}} \cdot b_{\text{loc}} \cdot (\boldsymbol{p}_i - \boldsymbol{x}_i^{(t)}) + r_{\text{glob}} \cdot b_{\text{glob}} \cdot (\boldsymbol{p}_{\text{glob}} - \boldsymbol{x}_i^{(t)}) \quad (1)$$

$$\boldsymbol{x}_i^{(t+1)} = \boldsymbol{x}_i^{(t)} + \boldsymbol{v}_i^{(t+1)}. \quad (2)$$

The parameters $a, b_{\text{loc}}, b_{\text{glob}} \in \mathbb{R}$ are constant weights which can be selected by the user. The *inertia* a adjusts the relative importance of the inertia of the particles, and the so-called *acceleration coefficients* b_{loc} and b_{glob} determine the influence of the local and the global attractor, resp. In every iteration, r_{loc} and r_{glob} are drawn uniformly at random from $[0, 1]$. Particles exchange information about the search space exclusively via the global attractor $\boldsymbol{p}_{\text{glob}}$. $\boldsymbol{p}_i - \boldsymbol{x}_i^{(t)}$ is the *local attraction* exerted by the local attractor on particle i, and $\boldsymbol{p}_{\text{glob}} - \boldsymbol{x}_i^{(t)}$ is the *global attraction* exerted by the global attractor.

The PSO algorithm was originally designed for solving optimization problems on continuous domains. Inspired by the success and conceptual simplicity of PSO, several approaches have been proposed to adapt the PSO dynamics to discrete problems including the TSP [2, 6, 15, 14]. In order to adapt PSO's movement equations (1) and (2) to the discrete domain of the TSP, Clerk suggests in [2] the following modifications (or new interpretations) of the terms involved:

- The particle's *position* $x_i^{(t)}$ is a permutation π of the cities, i. e., $\pi = (c_{i_1} c_{i_2} \dots c_{i_n})$ which corresponds to the tour $c_{i_1} \to c_{i_2} \to \dots \to c_{i_n} \to c_{i_1}$.
- The *difference* $x - y$ between two positions x and y (also called the attraction of x to y) is represented by a shortest sequence of transpositions $T = t_1, \dots, t_k$ such that $y \circ T = x$. Transposition $t = (c_m c_r)$ exchanges the two cities c_m and c_r in a given round-trip.
- The *length* of a difference is the length of the sequence T of transpositions.
- The *multiplication* $s \cdot T$ of a difference $T = t_1, \dots, t_k$ with a scalar s, $0 < s \leq 1$, is defined as $t_1, \dots, t_{\lceil s \cdot k \rceil}$. For $s = 0$, $s \cdot T = \emptyset$. Here, we omit the cases $s > 1$ and $s < 0$ since they do not occur in our proposed PSO.
- The *addition* $T_1 + T_2$ *of differences* $T_1 = t_1^1, \dots, t_k^1$ and $T_1 = t_1^2, \dots, t_l^2$ is defined as $T_1 + T_2 = t_1^1, \dots, t_k^1, t_1^2, \dots, t_l^2$.
- The *addition of a difference and a position* is defined as applying the transpositions of the difference to the position.

A small example is presented after the proof of Theorem 1.

In [2], [18] and [14], the difference between two positions x and y in the search space is represented as a list of transpositions that transform round-trip x into round-trip y. In [6], this representation is restricted to adjacent transpositions. In our new approach in Sec. 4, we replace the transposition by a representation which successively exchanges two *edges* in a round-trip. Hence, the difference of two positions x and y is a sequence of edge exchanges of minimal length which transforms x into y.

3 Theoretical Analysis

In this section, we prove that under certain conditions the previously developed variants of DPSO mentioned in Sec. 2 behave counterintuitively when compared to the classical continuous PSO since the convergence rate in DPSO is slowed down. More specifically, we show that transpositions which occur both in the local attraction and in the global attraction cancel each other and prevent the particle from moving closer to the global and local attractor. This is quite different from the behavior observed in continuous PSO where common components in these attractions even result in an amplified attractive force.

The phenomenon that the local and the global attraction in previous approaches have a lot of transpositions in common in the later stages of the optimization process can be observed experimentally. Evaluating the two attractions $p_i - x_i^{(t)}$ and $p_{\text{glob}} - x_i^{(t)}$ for sample runs (see Fig. 1), we see that (in this example) on average about 30% of the transpositions occur in both attractions. Summing the particles in the right half of the bins in Fig. 1, we can conclude that for roughly 20% of the particles, more than a half of the transpositions are shared by the two attractions. We analyzed the DPSO methods from [2, 18] that

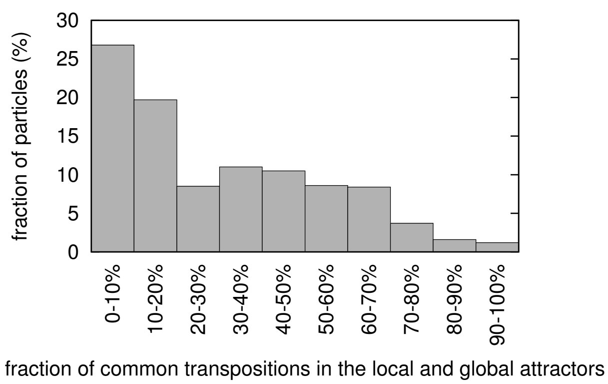


Fig. 1. Similarity of local and global attraction on Clerc's DPSO [2], averaged over 100 runs on the TSP instance berlin52, considering all particles in iterations 990 to 1000.

use transpositions for representing distances between particles in what we call the *Long Term Discrete PSO* model (LTD). In this model, we assume that the following four conditions hold:

- Differences between positions are represented by sequences of transpositions.
- $p_i = p_{\text{glob}} =: p$, for all particles i.
- $a = 0$
- r_{loc} and r_{glob} are uniformly distributed.

When the full swarm converges to a common best solution p, all local and global attractors are identical. If $p_i = p_{\text{glob}}$ for a certain particle, then it has visited the global attractor at least once. We assume the inertia a of the particles being 0 since in our experiments, the performance of the PSO algorithm even becomes worse if the inertia weight is set to a higher value. r_{loc} and r_{glob} are quite often uniformly distributed in practice. This assumption is also made in the mathematical analysis in [17].

For Theorem 1, we assume $b_{\text{loc}} = b_{\text{glob}}$, which allows for a closed and simple representation. After its proof, we deal with the more general case which can be analyzed analogously and present a small example.

Theorem 1. *Let $s \in [0, 1]$, and let $b_{\text{loc}} = b_{\text{glob}} = b$. The probability that in the* LTD *model a certain particle reduces its distance to p in an iteration by a factor of at least $b \cdot s$, is $(1 - s)^2$.*

Proof. As $a = 0$, the two movement equations (1) and (2) can be reduced to one:

$$x_i^{(t+1)} = x_i^{(t)} + r_{\text{loc}} \cdot b \cdot (p - x_i^{(t)}) + r_{\text{glob}} \cdot b \cdot (p - x_i^{(t)})$$

Let d be the number of transpositions in the attraction $(p - x_i^{(t)})$. Since we multiply the difference with $r_{\text{loc}} \cdot b$ and $r_{\text{glob}} \cdot b$, resp., we apply the first $r_{\text{loc}} \cdot b \cdot d$ and then the first $r_{\text{glob}} \cdot b \cdot d$ transpositions to $x_i^{(t)}$. Both differences have a common part consisting of the first $\min(r_{\text{loc}}, r_{\text{glob}}) \cdot b \cdot d$ transpositions.

By applying the first $r_{\text{loc}} \cdot b \cdot d$ transpositions, for each transposition an element of $x_i^{(t)}$ reaches the place that it also has in p. However, when applying the transpositions of the second difference, the common part of both differences is applied twice and the elements of the permutation that were already at the right place move now to another place. To bring the elements back to the original place we have to apply the inverse of the common part. Since the inverse of the common part has exactly the same number of transpositions as the common part, the distance to p is only reduced by the transpositions that are not common in both differences and so are only applied once.

The number of the transpositions that are applied only once is $|r_{\text{loc}} - r_{\text{glob}}| \cdot b \cdot d$. Only these transpositions contribute to the convergence towards p because the other transpositions move the particle further away from p when they are applied a second time. Therefore, we call transpositions that are applied only once "effective transpositions."

The probability that the fraction of effective transposition is at least $b \cdot s$ is, is given by

$$\mathrm{P}\Big(\frac{|r_{\text{loc}} - r_{\text{glob}}| \cdot b \cdot d}{d} \geq b \cdot s\Big) = \mathrm{P}(|r_{\text{loc}} - r_{\text{glob}}| \geq s) \ .$$

Since r_{loc} and r_{glob} are uniformly distributed, we may conclude (see also Fig. 2 choosing $b_{\text{loc}} = b_{\text{glob}} = b$): $\mathrm{P}(|r_{\text{loc}} - r_{\text{glob}}| \geq s) = (1 - s)^2$ □

If $b_{\text{loc}} \neq b_{\text{glob}}$, we analogously get the following expression for the probability q_s of the fraction of effective transpositions being larger than s:

$$\begin{aligned} q_s &= \mathrm{P}(|r_{\text{loc}} \cdot b_{\text{loc}} - r_{\text{glob}} \cdot b_{\text{glob}}| \geq s) \\ &= \mathrm{P}\Big(r_{\text{glob}} \leq \frac{b_{\text{loc}} \cdot r_{\text{loc}} - s}{b_{\text{glob}}}\Big) + \mathrm{P}\Big(r_{\text{glob}} \geq \frac{b_{\text{loc}} \cdot r_{\text{loc}} + s}{b_{\text{glob}}}\Big) \\ &= \int_0^1 \Big(\min\Big\{1, \max\Big\{0, \tfrac{b_{\text{loc}} \cdot r_{\text{loc}} - s}{b_{\text{glob}}}\Big\}\Big\} + \min\Big\{1, \max\Big\{0, \tfrac{b_{\text{loc}} \cdot r_{\text{loc}} + s}{b_{\text{glob}}}\Big\}\Big\}\Big) \,\mathrm{d}r_{\text{loc}} \end{aligned}$$

The probability q_s can be visualized like shown in Fig. 2, where q_s amounts to the shaded area.

Consider the following small example. Let $x_i^{(t)} = (1\,5\,2\,7\,3\,9\,4\,6\,8)$ and $p = (1\,2\,3\,4\,5\,6\,7\,8\,9)$. Then $p - x_i^{(t)} = ((2\,3)\,(3\,5)\,(4\,7)\,(6\,8)\,(8\,9))$. With $b = 0.8$, we have $b \cdot (p - x_i^{(t)}) = ((2\,3)\,(3\,5)\,(4\,7)\,(6\,8))$. With $r_{\text{loc}} = 0.75$ and $r_{\text{glob}} = 0.5$, we get $r_{\text{loc}} \cdot b \cdot (p - x_i^{(t)}) = ((2\,3)(3\,5)(4\,7))$ and $r_{\text{glob}} \cdot b \cdot (p - x_i^{(t)}) = ((2\,3)\,(3\,5))$, and finally $x_i^{(t+1)} = x_i^{(t)} + r_{\text{loc}} \cdot b \cdot (p - x_i^{(t)}) + r_{\text{glob}} \cdot b \cdot (p - x_i^{(t)}) = (1\,5\,2\,4\,3\,9\,7\,6\,8)$.

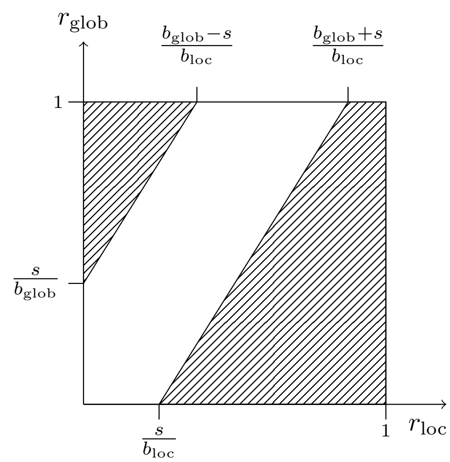


Fig. 2. The shaded area denotes q_s

The transposition $(4\,7)$ is the only effective transposition. By Theorem 1, the probability that the particle reduces its distance to p by at least 25% is $(1 - 0.3125)^2 \approx 0.47$.

Our analysis directly applies to Clerc's DPSO [2]. The algorithm proposed by Wang et al. [18] works a bit different with respect to the scaling of the attractions. In [18], Wang et al. proposed to scale the attractions by $b_{\text{loc}}, b_{\text{glob}} \in [0, 1]$ keeping each transposition with probability b_{loc} and b_{glob}, resp., in the attraction. So in the LTD model, the movement equations (1) and (2) reduce to

$$x_i^{(t+1)} = x_i^{(t)} + b_{\text{glob}} \cdot (p - x_i^{(t)}) + b_{\text{loc}} \cdot (p - x_i^{(t)}) \enspace .$$

A transposition becomes an effective transposition if it is kept in exactly one of the two attractions. Therefore, effective transpositions occur with probability $b_{\text{loc}} \cdot (1 - b_{\text{glob}}) + b_{\text{glob}} \cdot (1 - b_{\text{loc}}) = b_{\text{loc}} + b_{\text{glob}} - 2b_{\text{glob}} \cdot b_{\text{loc}}$. This is also the expected value of the fraction of effective transpositions.

The coefficients b_{loc} and b_{glob} are intended to adjust the weight of the local and the global attractor. Intuitively, if the attractors should exert a large influence on the particles, b_{loc} and b_{glob} are set to 1. This works fine in the classical PSO for continuous problems. In the discrete case however, whenever the LTD model applies, the local and global attractions do not pull the particles closer to the attractors at all.

4 A new DPSO for the TSP

4.1 Centroid-based particle movement: a new interpretation of "addition"

As described in Sec. 2.2, concatenation is used in [2] and [18] as "addition" of differences, i. e., attractions. In Sec. 3, we showed that this approach has the disadvantage that after some time the expected progress becomes considerably slow. Now we propose a new method of combining the attractions that avoids this disadvantage.

Instead of composing two attractions to a long list of operators, we look at the destinations, which are the points the different weighted attractions lead to, and compute the centroid of those destinations. In our approach, we use no inertia (i. e., we set $a = 0$), but only the attraction to the local and to the global attractor, each weighted in accordance with equation (1). Since we have only two attractors, the centroid can be calculated easily by computing the difference between the destinations, scaling them by one half and adding the result to the first destination. The PSO movement equations can now be expressed with the destination points of the attraction to the local attractor, to the global attractor and a random velocity:

$$\begin{aligned} d_{\text{loc}} &= x_i^{(t)} + r_{\text{loc}} \cdot b_{\text{loc}} \cdot (p_i - x_i^{(t)}) \\ d_{\text{glob}} &= x_i^{(t)} + r_{\text{glob}} \cdot b_{\text{glob}} \cdot (p_{\text{glob}} - x_i^{(t)}) \\ v_{\text{rand}} &= r_{\text{rand}} \cdot b_{\text{rand}} \cdot (p_{\text{rand}} - x_i^{(t)}) \\ x_i^{(t+1)} &= d_{\text{glob}} + \tfrac{1}{2} \cdot (d_{\text{loc}} - d_{\text{glob}}) + v_{\text{rand}} \end{aligned}$$

The random movement in the end ensures that the swarm does not converge too fast. A graphical, "continuous" representation is depicted in Fig. 3.

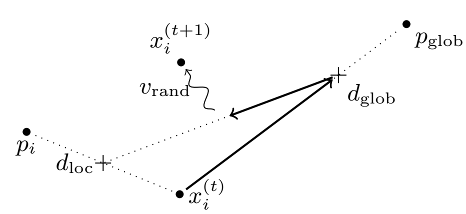


Fig. 3. Centroid-based particle movement

The advantage of this model is that it takes the spatial structure of the search space into account. Since the centroid is the mean of the destinations, our factors b_{loc} and b_{glob} can be transferred easily to the classical PSO by dividing them by 2.

4.2 Edge recombination: a new interpretation of "velocity"

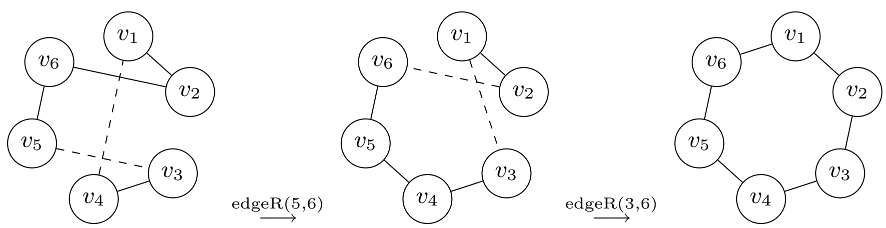
In [2], [18] and [14], the difference between two positions b and a in the search space, i. e., the velocity or the attraction of b to a is expressed as a list of transpositions that transforms one sequence of cities into the other. Here, we propose a new method that is based on edge exchanges. Edge exchanges are a common technique used in local search methods for the TSP [7]. The idea is to improve a solution by exchanging crossing edges, which results in a shorter tour. For an example, see the transformations from Figures 4(a) to (c).

A generalization of this operation is the edge recombination operator. Given a list $\ell = (c_1 c_2 \dots c_n)$ of cities representing a round-trip, the edge recombination operator edgeR(i, j) inverts the sequence of cities between indices i and j:

$$\ell \circ \text{edgeR}(i,j) = (c_1 \dots c_{i-1}\ \underline{c_j\ c_{j-1}\ c_{j-2} \dots c_{i+2}\ c_{i+1}\ c_i}\ c_{j+1} \dots c_n)$$

In our approach, we use this operator to express the difference between two particles b and a. Instead of a list of transpositions, the difference (or velocity, or attraction) is now a list of edge recombination operators that yields b if the operators are applied to a.

For example, the difference between $a = (1\ 2\ 6\ 5\ 3\ 4)$ and $b = (1\ 2\ 3\ 4\ 5\ 6)$ is $b - a = (\text{edgeR}(5, 6)\ \text{edgeR}(3, 6))$ (see Fig. 4).



(a) (1 2 6 5 3 4) (b) (1 2 6 5 4 3) (c) (1 2 3 4 5 6)

Fig. 4. Visualization of two edge recombinations

The problem of finding the minimum number of edge exchanges that is needed to transform one permutation into the other is NP-complete [16]. Therefore, we use in our experiments the simple and fast approximation algorithm GetListOfEdgeExchanges from [10] that is similar to the algorithm that finds the minimum number of transpositions. The solution found by the approximation algorithm GetListOfEdgeExchanges has in the worst case $\frac{1}{2}(n-1)$ times more edge exchanges than the optimal solution [10].

5 Experimental Results

In our experiments, we have compared our new approaches to the previous existing ones. We have on purpose initially not done any local optimization between

Table 1. Comparison of different distance representations and movement types without local optimization (left) and with an additional 2-OPT-based local optimization of the global attractor (right).

Move type	Composition	Centroid	Centroid	Centroid	Composition	Centroid	Centroid	Centroid
Distance Repr.	Transposition	Adj. Transposition	Transposition	edgeR	Transposition	Adj. Transposition	Transposition	edgeR
Problem								
berlin52	**104.6%**	**194.6%**	**70.5%**	**22.5%**	**24.2%**	**186,2%**	**8,2%**	**7.0%**
(7542)	18780	23193	15103	10233	10333	23181	8964	8614
	15431.9	22216.8	12862.3	9240.4	9368.1	14692.9	8157.6	8071.5
	±1046.8	±531.6	±863.0	±410.8	±362.6	±1539.6	±241.5	±187.0
	12588	20755	10977	8259	8877	14369	7542	7708
pr76	**220.9%**	**317.7%**	**156.5%**	**88.9%**	**56.9%**	**229.1%**	**5.8%**	**4.7%**
(108159)	416630	468479	347091	249993	19479	460894	128969	122794
	347107.5	451771.6	277404.3	204322.5	169716.3	356021.9	114454.6	113311.6
	±18966.6	±6919.1	±21902.2	±19535.6	±8340.2	±76522.9	±3516.6	±2518.3
	304195	431144	228652	158676	149895	195103	109669	109543
gr96	**310.3%**	**430.4%**	**220.8%**	**128.5%**	**82.9%**	**368.1%**	**9.7%**	**6.3%**
(55209)	251167	302477	204837	149542	121949	299384	63962	63846
	226531.6	292815.8	177107.8	126126.1	100986.2	258421.8	60576.6	58716.1
	±9631.1	±4398.8	±14565.0	±8618.0	±5929.8	±38302.9	±1785.1	±1511.0
	203617	278382	138255	107835	83344	122110	56085	56393
kroA100	**377.2%**	**529.2%**	**238.0%**	**111.2%**	**85.4%**	**401.9%**	**7.4%**	**5.5%**
(21282)	115730	137761	83682	55383	45010	136703	25009	24195
	101552.2	133913.6	71933.4	44945.0	39464.4	106817.7	22864.25	22472.3
	±4468.6	±2291.3	±4702.6	±3498.7	±1908.0	±23555.6	±620.1	±485.8
	91137	125127	60524	37289	35600	40279	21794	21431
kroC100	**386.7%**	**537.4%**	**256.2%**	**133.9%**	**90.0%**	**435.9%**	**8.2%**	**7.1%**
(20749)	112320	135960	84903	58291	45690	136150	24600	24087
	100988.8	132261.1	73903.8	48538.7	39438.7	111195.0	22440.8	22229.4
	±4189.6	±2107.8	±5234.0	±3093.7	±1834.0	±23114.4	±728.4	±655.7
	90375	125228	63107	41237	33682	42998	20996	21034
kroD100	**364.2%**	**503.1%**	**239.0%**	**127.7%**	**86.1%**	**368.9%**	**7.9%**	**7.1%**
(21294)	106556	131549	83922	58077	47876	130923	25091	23941
	98847.9	128438.0	72194.1	48487.5	39625.9	99852.8	22983.5	22808.7
	±4650.5	±1902.7	±4865.6	±3227.4	±2180.0	±21125.3	±589.7	±512.5
	90347	121195	62665	41828	32814	46429	21860	21665
lin105	**421.8%**	**575.8%**	**305.3%**	**188.5%**	**104.4%**	**475.5%**	**18.0%**	**7.1%**
(14379)	87392	100671	69741	49740	33460	98936	18546	16496
	75032.0	97170.6	58284.6	41484.5	29383.8	82754.7	16968.0	15404.8
	±3967.1	±1506.7	±4905.4	±3041.1	±1734.8	±12847.2	±626.2	±414.9
	67391	92116	48110	34121	25383	48322	15306	14600

two iterations to see the clear impact of exchanging the existing approaches with ours. The swarm we use consists of 100 particles and we use 1000 iterations to optimize the function. Each configuration is run 100 times to compute the mean error and the standard deviation. The entries in Table 1 provide data in the following format:

Problem name	**relative error**
(optimal value)	maximal value found by the algorithm
	mean value ± standard deviation
	best solution found by the algorithm

In Table 1, the left four result columns present our results obtained with the proposed DPSO variants without local optimization. In order to make our results also comparable to other approaches, we have added the local optimization method from Shi et al. in [14]. These results are shown in the right four columns of Table 1. Similarly to our proposed approach, the method of Shi also avoids the convergence problems analyzed in Sec. 3, but seems to result in a smaller relative error.

In every four columns block, the first column shows the results of the method representing differences as transpositions and using a simple composition to combine the differences. The other three columns show the results obtained by the centroid-based method from Sec. 4.1. The centroid-based approach is combined with various representations of differences, namely adjacent transpositions, transpositions and the edge recombinations introduced in Sec. 4.2.

Our centroid-based approach avoids the counter-intuitive convergence behavior explained by the theoretical analysis in Sec. 3. The experiments show that this method is a better choice than the simple composition of differences. Another crucial factor is the choice of the representation of particle velocities. Our experimental results show that transpositions are better than adjacent transpositions and that the proposed edge recombination method performs best. Finally, a comparison between different approaches of discrete PSO can only be significant, if the actual contribution of the PSO algorithm is not obfuscated by an additional local search procedure. This is why we show the results of the pure PSO without local optimization, which can serve as a reference for future DPSO variants for the TSP.

6 Conclusions

In our theoretical analysis of discrete PSO for the TSP we showed that the convergence behavior the convergence behavior differs significantly from what we expect from a PSO for continuous problems. Our analysis can be applied mainly to the DPSO variants of Clerc in [2] and Wang in [18] but can also be extended to the approaches of Shi et al. [14]. The convergence behavior can be observed whenever the local and the global attractor of a particle nearly coincide. If this is the case the transpositions occurring in respective velocities cancel each other out. This slows down the particles and prevents convergence.

We proposed a new model for particle motion which from a theoretical point of view does not suffer from the aforementioned convergence problem. This is backed by our experiments, which show a clear improvement of the DPSO performance with this model. Additionally we introduced a representation for particle velocities, which is based on edge exchanges in a tour. Our evaluation shows that the edge exchange-based representation produces better results than traditional approaches from the literature.

References

1. D. Anghinolfi and M. Paolucci. A new discrete particle swarm optimization approach for the single-machine total weighted tardiness scheduling problem with sequence-dependent setup times. *European Journal of Operational Research*, 193:73–85, 2009.

2. M. Clerc. Discrete Particle Swarm Optimization, illustrated by the Traveling Salesman Problem. Website, 2000.
http://clerc.maurice.free.fr/pso/pso_tsp/Discrete_PSO_TSP.zip.
3. M. Clerc. Discrete particle swarm optimization, illustrated by the traveling salesman problem. In G. C. Onwubolu and B. V. Babu, editors, *New Optimization Techniques in Engineering*, Studies in Fuzziness and Soft Computing, chapter 8, pages 219–239. Springer, 2004.
4. M. Clerc and J. Kennedy. The particle swarm – Explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, 6:58–73, 2002.
5. R. C. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. In *Proc. 6th International Symposium on Micro Machine and Human Science*, pages 39–43, 1995.
6. E. F. G. Goldbarg, G. R. de Souza, and M. C. Goldbarg. Particle swarm for the traveling salesman problem. In *Proc. 6th European Conference on Evolutionary Computation in Combinatorial Optimization (EvoCOP)*, pages 99–110, 2006.
7. G. Gutin and A. P. Punnen, editors. *The Traveling Salesman Problem and Its Variations*, volume 12 of *Combinatorial Optimization*. Springer, 2002.
8. S. Helwig and R. Wanka. Theoretical analysis of initial particle swarm behavior. In *Proc. 10th Int. Conf. on Parallel Problem Solving from Nature (PPSN)*, pages 889–898, 2008.
9. M. Jiang, Y. P. Luo, and S. Y. Yang. Stochastic convergence analysis and parameter selection of the standard particle swarm optimization algorithm. *Inf. Process. Lett.*, 102:8–16, 2007.
10. J. Kececioglu and D. Sankoff. Exact and approximation algorithms for sorting by reversals, with application to genome rearrangement. *Algorithmica*, 13:180–210, 1995.
11. J. Kennedy and R. C. Eberhart. Particle swarm optimization. In *Proc. IEEE International Conference on Neural Networks*, volume 4, pages 1942–1948, 1995.
12. J. Kennedy and R. C. Eberhart. A discrete binary version of the particle swarm algorithm. In *Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics*, volume 5, pages 4104–4108, 1997.
13. G. Reinelt. TSPLIB – A traveling salesman problem library. *ORSA Journal on Computing*, 3(4):376–384, 1991.
14. X. H. Shi, Y. C. Liang, H. P. Lee, C. Lu, and Q. X. Wang. Particle swarm optimization-based algorithms for TSP and generalized TSP. *Inf. Process. Lett.*, 103:169–176, 2007.
15. X. H. Shi, Y. Zhou, L. M. Wang, Q. Wang, and Y. C. Liang. A discrete particle swarm optimization algorithm for travelling salesman problem. In *Proc. 1st Int. Conf. on Computation Methods (ICCM)*, volume 2, pages 1063–1068, 2004.
16. A. Solomon, P. Sutcliffe, and R. Lister. Sorting circular permutations by reversal. In *Proc. 8th Int. W'shop on Algorithms and Data Structures (WADS)*, pages 319–328, 2003.
17. I. C. Trelea. The particle swarm optimization algorithm: Convergence analysis and parameter selection. *Inf. Process. Lett.*, 85:317–325, 2003.
18. K. P. Wang, L. Huang, C. G. Zhou, and W. Pang. Particle swarm optimization for traveling salesman problem. In *Proc. 2nd Int. Conf. on Machine Learning and Cybernetics*, volume 3, pages 1583–1585, 2003.
19. W. Zhong, J. Zhang, and W. Chen. A novel discrete particle swarm optimization to solve traveling salesman problem. In *Proc. IEEE Congress on Evolutionary Computation (CEC)*, pages 3283–3287, 2007.