Available online at www.sciencedirect.com
ScienceDirect
Operations Research Letters 35 (2007) 227–231
Operations Research Letters
www.elsevier.com/locate/orl

On the theoretical properties of swap multimoves

W. Bożejko[a,*], M. Wodecki[b]

[a] *Institute of Computer Engineering, Control and Robotics, Wrocław University of Technology, Janiszewskiego 11-17, 50-372 Wrocław, Poland*
[b] *Institute of Computer Science, University of Wrocław, Przesmyckiego 20, 51-151 Wrocław, Poland*

Received 22 September 2004; accepted 6 March 2006
Available online 5 June 2006

Abstract

We prove that any permutation can be transformed into any other permutation by the execution of at most two swap multimoves (i.e. the diameter of the neighborhood graph is 2). We also prove that it is NP-hard to search over such a neighborhood.

Keywords: Metaheuristics; Multimoves; Scheduling

1. Introduction

The natural representation of a solution for many combinatorial optimization problems is a permutation of integers $1, 2, \ldots, n$. Some important examples of these problems are: the traveling salesman problem, the assignment problem and the single and multimachine scheduling problems. These problems have both theoretical and practical significance. Each of them is strongly NP-hard, so finding the optimal solution is difficult. For a problem of large size the exact algorithm may not be useful since computationally it is too expensive. Approximate algorithms, which do not guarantee finding an optimal solution are a good alternative. An important class of heuristic algorithms are the so-called *local search* algorithms (see e.g. [1]). For a given starting solution, the elementary step of the algorithm is a search process through a subset of solutions called the *neighborhood*. This subset is covered by *moves* performed from the starting solution. Next, some element of this neighborhood is chosen which becomes the new starting solution in the next iteration of the algorithm. This process is repeated until some termination criterion is satisfied. The obtained solution is the local optimum.

Currently, *swap* and *insert* type moves are usually applied in heuristic algorithms presented in the literature. The first type consists of changing positions of any two elements in a permutation, the second type consists of the insertion of an element into a chosen position in the permutation. The neighborhoods generated by swap and insert moves have size $O(n^2)$.

* Corresponding author.
E-mail address: wojciech.bozejko@pwr.wroc.pl (W. Bożejko).

0167-6377/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.orl.2006.03.002

Moreover, neighborhoods of exponential size are also considered in literature. For the one machine batching problem, Hurink [10] considered a neighborhood generated by the compositions of independent swap moves, where only adjacent elements are allowed to change. Congram et al. [4] presented a neighborhood generated by compounded independent swaps for the one machine total tardiness problem. Grosso et al. [8] added to this approach independent insert moves. Ahuja et al. [2] presented a survey of techniques for searching such exponential neighborhoods.

A neighborhood metric studied in literature is the diameter of the neighborhood graph (see [2]). Gutin and Yeo [9] constructed a neighborhood of exponential size for the traveling salesman problem, where the corresponding neighborhood graph has diameter 4. In this paper, we introduce a neighborhood generated by the composition of all swap moves with disjoint supports (see Section 2)—that means that different swaps of a multiswap do not involve the same elements of the permutation (they are commutative). Such a neighborhood has an exponential number of elements. The diameter of the corresponding neighborhood graph is 2.

Three types of methods of searching an exponential neighborhood are discussed in the literature: (1) variable-depth methods in which exponential neighborhoods are searched by heuristic algorithms, (2) methods based on network flow techniques or dynamic programming, (3) neighborhoods induced by restrictions of the original problem. The neighborhood proposed in this work is used in papers [6,7] to diversify and intensify the calculations. Although the neighborhood is very large (we will prove that determining its optimal element is an NP-hard problem), a heuristic method is used to determine the suboptimal element in it.

Here, we will prove that any permutation can be transformed into any other permutation by the execution of at most two swap multimoves (Corollary 3). Therefore, executing a multimove simply moves the calculations to a distant area of the solution space. This explains in part why algorithms in which multimoves are applied (especially to intensify and diversify the calculations) obtain much better results than the classic algorithms based on single moves.

2. Multimoves as involutions

Let $\pi = (\pi(1), \ldots, \pi(n))$ be a permutation of the n-element set $\{1, 2, \ldots, n\}$, and let $S(n)$ be the set of all such permutations. The composition (product) of permutations α and β, written as $\alpha\beta$, is a permutation such that $(\alpha\beta)(i) = \alpha(\beta(i))$, $i = 1, 2, \ldots, n$. Permutation π^{-1} is the inverse of the permutation $\pi \in S(n)$ if $\pi^{-1}(\pi(i)) = i$. By ε, we represent the neutral element, the identity permutation (i.e. $\varepsilon(i)=i$). Obviously $\pi\pi^{-1} = \varepsilon$ and $\pi^{-1}\pi = \varepsilon$.

Let the permutation $\pi \in S(n)$ and a move m (e.g. swap or insert) generate some permutation $\beta \in S(n)$ from π, i.e. $m(\pi) = \beta$. It is easy to see that in this case there exists a permutation $\gamma \in S(n)$ such that $\pi\gamma = \beta$ (it is enough to take $\gamma = \pi^{-1}\beta$). In this paper we assume that the move m can be identified with some permutation (permutation γ) and the move's execution consists of multiplying the permutation π by this permutation γ. (That is, the permutation γ did not depend on π.) Since a multimove is a composition (product) of some number of permutations, it is a permutation too. In metaheuristic algorithms, multimoves are applied to move (intensification or diversification) calculations to another area of the set of feasible solutions. In [5] a number of metrics defined on the permutation group are presented. Three of them are:

(1) *Kendall's tau*: $I(\pi, \sigma)$ = minimum number of pairwise adjacent transpositions taking π^{-1} to σ^{-1},
(2) *Cayley's distance*: $T(\pi, \sigma)$ = minimum number of transpositions taking π to σ,
(3) *Ulam's distance*: $L(\pi, \sigma) = n-$ (length of the longest increasing subsequence in $\sigma\pi^{-1}$).

Each of these metrics is directly connected with the typical moves and neighborhoods applied to local search methods. These metrics are used to estimate how distant is the permutation generated by a multimove. Kendall's tau metric is equivalent to the number of adjacent swap moves, the Cayley's metric is equivalent to the number of (not necessarily adjacent) swap moves, and the Ulam's metric is connected with the insertion moves, because the number of moves needed to transform permutation π into σ is equal to the value $L(\pi^{-1}, \sigma^{-1})$.

Definition 1 (*Knuth [11]*). Permutation σ is called an involution, if it is inverse to itself, i.e. $\sigma^2 = \varepsilon$.

Definition 2. The support $\mathrm{supp}(\sigma)$ of permutation $\sigma \in S(n)$ is the set

$$\mathrm{supp}(\sigma) = \{k : \sigma(k) \neq k, k = 1, 2, \ldots, n\}.$$

Permutations $\alpha, \beta \in S(n)$ are *independent* if their supports are disjoint, i.e. $\mathrm{supp}(\alpha) \cap \mathrm{supp}(\beta) = \emptyset$. The multimove consisting in a simultaneous exchange of different pairs of elements in a permutation (executing independent swap moves) is a composition of pair-independent transpositions. We call it a *multiswap* (as a move executed on some permutation) or *involution* (as a permutation, i.e. function).

We use cyclic notation in the following theorem. We represent a permutation σ as a product of cycles, and each cycle $(i_1, i_2, \ldots, i_k)$ is part of the cycle decomposition of σ, i.e. $\sigma(i_j) = i_{j+1}$ for $j = 1, 2, \ldots, k-1$ and $\sigma(i_k) = i_1$.

Now we will prove the theorem that every permutation can be obtained from any other permutation in at most two multiswaps.

Theorem 1. *Let $\sigma \in S(n)$ be a permutation, then there exist involutions $\alpha, \beta \in S(n)$ such that $\sigma = \alpha\beta$.*

Proof. Consider first the case that σ is a single cycle $C = (i_1, i_2, \ldots, i_r)$. In this case, let

$$\alpha = (i_1, i_r)(i_2, i_{r-1}) \ldots (i_k, i_{r-k+1}) \quad \text{and} \tag{1}$$

$$\beta = (i_1, i_{r-1})(i_2, i_{r-2}) \ldots (i_k, i_{r-k}) \quad \text{where } k = \lfloor r/2 \rfloor. \tag{2}$$

Permutations α, β are compositions of independent transpositions, hence they are involutions (multiswaps). The product of these involutions is

$$\alpha\beta = (i_1, i_2, \ldots, i_r) = \sigma.$$

In the case that σ is a product of disjoint cycles $C_1, C_2, \ldots, C_l$ (see [11]), then let $C_s = \alpha_s\beta_s$, $s = 1, 2, \ldots, l$ where α_s, β_s are two involutions associated with cycle C_s. In this case, permutation σ takes the form

$$\sigma = C_1C_2C_3 \cdots C_l = \alpha_1\beta_1\alpha_2\beta_2\alpha_3\beta_3 \cdots \alpha_l\beta_l.$$

All involutions $\alpha_1, \alpha_2, \ldots, \alpha_l$ are commutative and all involutions $\beta_1, \beta_2, \ldots, \beta_l$ are commutative (because of disjoint supports). Moreover, α_i and β_j are commutative, i.e. $\alpha_i\beta_j = \beta_j\alpha_i$ (because $\mathrm{supp}(\alpha_i) \cap \mathrm{supp}(\beta_j) = \emptyset$) for $i \neq j$, therefore

$$\begin{aligned}\sigma &= \alpha_1\beta_1\alpha_2\beta_2\alpha_3\beta_3 \cdots \alpha_l\beta_l \\ &= (\alpha_1\alpha_2 \cdots \alpha_l)(\beta_1\beta_2 \cdots \beta_l) = \alpha\beta,\end{aligned}$$

where $\alpha = \alpha_1\alpha_2 \cdots \alpha_l$ and $\beta = \beta_1\beta_2 \cdots \beta_l$. Now it is necessary to prove that the permutations α, β are indeed multiswaps, i.e. involutions. From the commutativity of $\alpha_1, \alpha_2, \ldots, \alpha_l$, it follows that

$$\begin{aligned}\alpha^2 &= (\alpha_1\alpha_2 \cdots \alpha_l)(\alpha_1\alpha_2 \cdots \alpha_l) \\ &= (\alpha_1\alpha_1)(\alpha_2\alpha_2) \cdots (\alpha_l\alpha_l) = \varepsilon,\end{aligned}$$

therefore α is an involution, so α is a multiswap. Similarly, we can prove that β is an involution, finishing the proof of the theorem. □

From Theorem 1, it follows that for any permutation $\delta \in S(n)$ there exist involutions $\alpha = \alpha_1\alpha_2 \cdots \alpha_l$ and $\beta = \beta_1\beta_2 \cdots \beta_l$, where $\alpha_s\beta_s$ $(s = 1, 2, \ldots, l)$ are independent cycles defined, respectively, by (1) and (2) and such that $\delta = \alpha\beta$. The method of construction of both involutions is precisely presented in the proof of the theorem.

Let $\delta = \alpha\beta$, where $\delta, \alpha, \beta \in S(n)$ and α, β are involutions. Because an involution is the inverse of itself (i.e. $\alpha^{-1} = \alpha$ and $\beta^{-1} = \beta$), multiplying the equation $\delta = \alpha\beta$ from the right first by β and then by α we obtain $\delta\beta\alpha = \varepsilon$. So we can transform any permutation δ into the identity permutation ε within two involutions (two multimoves).

For any permutations $\pi, \delta \in S(n)$ let $\gamma = \delta^{-1}\pi$, $\gamma \in S(n)$. Based on previous considerations there exist involutions $\alpha, \beta \in S(n)$ such that $\gamma\alpha\beta = \varepsilon$. Multiplying this equation from the left side by δ we obtain $\delta\gamma\alpha\beta = \delta\varepsilon$. Because $\gamma = \delta^{-1}\pi$, therefore $\pi\alpha\beta = \delta$. It follows that by executing two multimoves we can generate every permutation of the set $S(n)$ from any other permutation of $S(n)$.

Definition 3 (*Ahuja et al. [2]*). Neighborhood graph, defined with respect to a specific problem instance, is a directed graph with one node for each feasible solution (and/or instance of a nonfeasible reference structure) created, and with an arc (s, t) whenever a solution t belongs to a set of neighbors of a solution s.

Corollary 1. *The diameter of the neighborhood graph for multiswaps is 2.*

Remark 1. In one of the advanced heuristic methods—path relinking (see e.g. [12])—for two permutations π and δ a permutation γ lying on a path between π and δ is constructed. Because $\pi\alpha\beta = \delta$, where α, β are involutions, so the permutation $\gamma = \pi\alpha$, which is generated from π by the involution (multimove) α, lies on the path between π and δ. It is possible to construct such involutions α, β that the permutation $\gamma = \pi\alpha$ is in the required range of distance to π (with respect to Cayley, Kendall or Ulam measure). In particular, construction (1)–(2) can be successfully applied to diversify the calculations, as a fully deterministic method. From construction (1)–(2), it follows that the permutation $\pi\alpha$ has many transpositions, so it is at a long distance (counted as the number of swap moves) from π.

Remark 2. For any permutation $\pi \in S(n)$ the set of all permutations $S(n)$ can be divided into three subsets (orbits): $O_1 = \{\pi\}$, $O_2 = \{\pi\alpha : \alpha \in S(n)$ and α is an involution$\}$, $O_3 = \{\pi\alpha\beta : \alpha, \beta \in S(n)$ and α, β are involutions$\}$. The dynasearch neighborhoods from the algorithms proposed in the works [4] and [5] are generated by single multimoves and these neighborhoods are subsets of the set O_2.

Remark 3. In works [3,6,7] multimoves are applied to intensify and diversify the calculations. Because these moves are compositions of independent moves, one can estimate the goal function of the permutation generated by these moves with good precision.

Remark 4. In this paper, we consider only swap multimoves. But it is easy to prove that any swap move can be obtained as a composition of exactly two insert moves. So if we consider insert multimoves, Theorem 1 implies that the diameter of the corresponding neighborhood graph is at most 4, which means that any permutation can be transformed into any other permutation by executing at most four insert multimoves.

Remark 5. The neighborhood based on swap multimoves is of very large-scale (the number of its elements is asymptotically $(1/\sqrt{2})(n/e)^{n/2}e^{(\sqrt{n}-1/4)}$, see [11]).

Lemma 1. *The problem of determining whether there is a multiswap with cost less than or equal to k in the multimove neighborhood is NP-complete.*

The proof of this lemma is given in Appendix A.

From Theorem 1, it follows that any permutation from the group $S(n)$ can be transformed into any other permutation by executing at most two multimoves. Therefore, executing a single multimove is a way of going considerably away from the current solution. This explains in part why application of multimoves to metaheuristic algorithms (in addition to single moves) can significantly improve their computational results.

Acknowledgment

The authors wish to thank the referee for his detailed comments and constructive criticisms of the initial draft.

Appendix A. Proof of Lemma 1

We carry out a transformation from the Hamiltonian Path Problem on undirected graphs.

Hamiltonian Path Problem.

INPUT. An undirected graph $G = (V, E)$, where $V = \{1, 2, \ldots, n\}$.

QUESTION. Is there a path in G starting at node 1 and including every other node of V?

We transform the Hamiltonian Path Problem into a special case of the multiswap problem with on integers from 1 to $2n$ as follows. Let us consider a graph $G' = (U, E')$ where the set of nodes $U = \{1, 2, \ldots, n, n+1, n+2, \ldots, 2n\}$ and a set of edges $E' = \{\{i, j\} : i \neq j,\ i, j \in U\}$. For each edge $\{i, j\} \in E'$ we associate a cost $c_{i,j}$.

For $1 \leqslant j, k \leqslant n$, we let

$$c_{jk} = \begin{cases} 1 & \text{if } (j,k) \notin E, \\ 0 & \text{if } (j,k) \in E. \end{cases}$$

For $n+1 \leqslant j, k \leqslant 2n$, we let $c_{jk} = 0$.

For $1 \leqslant j \leqslant n$ and $n+1 \leqslant k \leqslant 2n$, we let

$$c_{jk} = c_{kj} = \begin{cases} 1 & \text{if } j = 1, \\ 0 & \text{if } j \neq 1. \end{cases}$$

We start with the natural permutation $\varepsilon \in S(2n)$, that means we are searching a multiswap of permutation ε which leads to a permutation (Hamiltonian path in G') with cost 0.

We now claim that there is a Hamiltonian path in G if and only if there is a multiswap that leads to a solution with cost 0.

First assume that there is a Hamiltonian path in G, say $i_1, i_2, \ldots, i_n$. Then swap the elements so that the integers $i_1, i_2, \ldots, i_n$ appear in positions $n+1, n+2, \ldots, 2n$. This leads to a permutation of cost 0.

Now assume that there is a multiswap that leads to a permutation (Hamiltonian path in G') of cost 0. By the way that c_{jk} is defined for $1 \leqslant j \leqslant n < k \leqslant 2n$, it follows that the first n integers of the permutation after the multiswap are from 1 to n, or the last n integers are from 1 to n. In either case, the integers 1–n induce a Hamiltonian path in G starting at node 1.

References

[1] E. Aarts, J.K. Lenstra, Local Search in Combinatorial Optimization, Wiley, New York, 1997.

[2] R.K. Ahuja, O. Ergun, J.B. Orlin, A.P. Punnen, A survey of very large-scale neighborhood search techniques, Discrete Appl. Math. 123 (2002) 75–102.

[3] W. Bożejko, M. Wodecki, Parallel genetic algorithm for the flow shop scheduling problem, Lecture Notes in Computer Science, vol. 3019, Springer, Berlin, 2004, pp. 566–571.

[4] R.K. Congram, C.N. Potts, S.L. Van de Velde, An iterated dynasearch algorithm for the single-machine total weighted tardiness scheduling problem, INFORMS J. Comput. 14 (1) (2002) 52–67.

[5] P. Diaconis, Group representations in probability and statistics, Lecture Notes—Monograph Series, vol. 11, Institute of Mathematical Statistics, Harvard University, 1988.

[6] J. Grabowski, M. Wodecki, A very fast tabu search algorithm for the permutation flow shop problem with makespan criterion, Comput. Oper. Res. 31 (2004) 1891–1909.

[7] J. Grabowski, M. Wodecki, A very fast tabu search algorithm for the job shop problem, in: C. Rego, B. Alidaee (Eds.), Adaptive Memory and Evolution; Tabu Search and Scatter Search, Kluwer Academic Publishers, Dordrecht, 2005.

[8] A. Grosso, F. Della Croce, R. Tadei, An enhanced dynasearch neighborhood for single-machine total weighted tardiness scheduling problem, Oper. Res. Lett. 32 (2004) 68–72.

[9] G.M. Gutin, A. Yeo, Small diameter neighbourhood graphs for the traveling salesman problem: at most four moves from tour to tour, Comput. Oper. Res. 26 (1999) 321–327.

[10] J. Hurink, An exponential neighborhood for a one machine batching problem, OR Spektrum 21 (1999) 461–476.

[11] D.E. Knuth, The Art of Computer Programming, vol. 3, second ed., Addison Wesley Longman, 1998.

[12] C. Reeves, T. Yamada, Genetic algorithms, path relinking and the flowshop sequencing problem, Evol. Comput. 6 (1988) 45–60.