Effect of Using Partial Solutions in Creating New Candidate Solutions with EDA and ACO Schemes

Shigeyoshi Tsutsui
Department of Management Information
Hannan University
Matsubara, Osaka 580-8502 Japan

Abstract

In previous studies, we have proposed an algorithm named the edge histogram based sampling algorithm (EHBSA) with EDA scheme for permutation domains. In EHBSA, new solutions are obtained by combining partial solutions which exist in the current population, and partial solutions newly generated according to an edge histogram model of the current population. In this paper, we show using partial solutions can maintain the diversity of the population, resulting in a successful search. We also show that using partial solutions in ACO can greately enhance its performance.

1 Introduction

Genetic Algorithms (GAs) are widely used as robust black-box optimization techniques applicable across a broad range of real-world problems. GAs should work well for problems that can be decomposed into sub-problems of bounded difficulty [1]. However, fixed, problem-independent variation operators are often incapable of effective exploitation of the selected population of high-quality solutions [1, 2]. One of the most promising research directions is to look at the generation of new candidate solutions as a learning problem, and use a probabilistic model of selected solutions to generate the new ones [3, 4]. The algorithms based on learning and sampling a probabilistic model of promising solutions to generate new candidate solutions are called estimation of distribution algorithms (EDAs) [5] or probabilistic model-building genetic algorithms (PMBGAs) [3].

Most work on EDAs focuses on optimization problems where candidate solutions are represented by fixed-length vectors of discrete or continuous variables. However, for many combinatorial problems permutations provide a much more natural representation for candidate solutions. Despite the great success of EDAs in the domain of fixed-length discrete and continuous vectors, only few studies can be found on EDAs for permutation problems [6, 7].

In previous studies [8, 9], we have introduced a promising approach to learning and sampling probabilistic models for permutation problems using edge histogram models. This algorithm is called the edge histogram based sampling algorithm (EHBSA). In EHBSA, new solutions are created by combining partial solutions which exist in the current population, and partial solutions newly generated based on the edge histogram model of the current population. The EHBSA worked well on several benchmark instances of the traveling salesman problem (TSP). Nonetheless, the methods proposed are not limited to TSP, like most other TSP solvers and specialized variation operators. As a result, this approach provides a promising direction for solutions of other problems that can be formulated within the domain of fixed-length permutations; flow shop scheduling is an example of such a problem as described in [9].

In this paper, we focus our attention on the effects of using partial solutions in EHBSA. The basic sampling algorithms in EHBSAs are very similar to the sampling algorithms that are used in ant colony optimization (ACO) [10, 11] and thus this method can be applied to ACO [12]. We also discuss these results.

In the remainder of this paper, Section 2 gives a general scheme for using partial solutions in EDAs. In Section 3, we discuss the effect of using partial solutions in EHBSA and show how using partial solutions can maintain the diversity of the population, resulting in a successful search. The effectiveness of using partial solution in ACO is discussed in Section 4. Finally, Section 5 concludes this paper.

2 Using Partial Solutions in EDAs

Figure 1 shows a typical scheme of EDAs. EDAs evolve a population of candidate solutions to the given problem by building and sampling a probabilistic model of promising solutions. EDAs start with a random population of candidate solutions (individuals). Each iteration of EDAs starts by selecting better individuals from the current population. Next, the probability distribution M of the selected population of individuals (P^{sel}) is estimated. New individuals are then generated according to this estimate, forming the population of candidate solutions for the next generation. The process is repeated until the termination conditions are satisfied.

The scheme of EDAs with partial solutions is shown in Figure 2. In the figure, new solutions are obtained by combining partial solutions which exist in the current population, and partial solutions newly generated according to the model M of P^{sel}. What kind of effect can we be expected with this scheme in Figure 2? In a typical EDA scheme in Figure 1, new solutions are directly reflected from the distribution of P^{sel}. If the selection operator is not designed properly, the repetition of iterations within the scheme may cause strong, but incomplete positive feedback to model M, resulting in a premature convergence of the population and a failed search. On the other hand, with the scheme described in Figure 2, new solutions are generated not only according to model M but also using partial solutions from the existing solutions in P. This reduces the rapid change of the population. Thus, we can expect that using partial solutions as shown in Figure 2 has the effect of maintaining diversity and preventing premature convergence of the population.

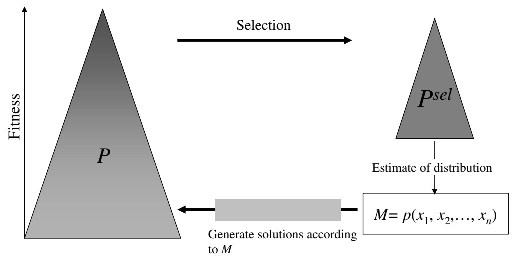

Selection
Fitness
P
P^{sel}
Estimate of distribution
$M= p(x_1, x_2,\ldots, x_n)$
Generate solutions according to M

Figure 1: General Scheme of EDAs

3 EHBSA

3.1 Edge Histogram Model

An *edge* is a link or connection between two nodes. The basic idea of the edge histogram based sampling algorithm (EHBSA) is to use the edge distribution of the whole population in generating new strings. The algorithm starts by generating a random permutation string for each individual population of candidate solutions. Promising solutions are then selected using any popular selection scheme. An *edge histogram matrix (EHM)* for the selected solutions is constructed and new solutions are generated by sampling, based on the edge histogram matrix. New solutions replace some of the old ones and the process is repeated until the termination criteria are met. An example of EHM at t $EHM^t = (e^t_{i,j})$ is shown in Fig 3. The integer value of each $(e^t_{i,j})$ represents number of edges from node i to node j in the population. The fractional value represents minimum value to give a *bias* to control pressure in sampling nodes.

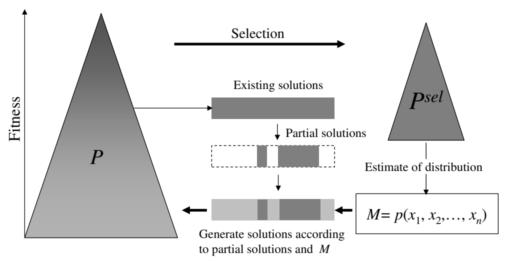

Selection
Fitness
P
P^{sel}
Existing solutions
Partial solutions
Estimate of distribution
$M= p(x_1, x_2,\ldots, x_n)$
Generate solutions according to partial solutions and M

Figure 2: Scheme of EDAs with Partial Solutions

$$\begin{array}{l} s^t_1 = (0,\ 1,\ 2,\ 3,\ 4) \\ s^t_2 = (1,\ 3,\ 4,\ 2,\ 0) \\ s^t_3 = (3,\ 4,\ 2,\ 1,\ 0) \\ s^t_4 = (4,\ 0,\ 3,\ 1,\ 2) \\ s^t_5 = (2,\ 1,\ 3,\ 4,\ 0) \end{array} \qquad \begin{array}{cc} & \text{node } j \\ \text{node } i & \begin{pmatrix} 0 & 3.1 & 2.1 & 2.1 & 3.1 \\ 3.1 & 0 & 4.1 & 3.1 & 0.1 \\ 2.1 & 4.1 & 0 & 1.1 & 3.1 \\ 2.1 & 3.1 & 1.1 & 0 & 4.1 \\ 3.1 & 0.1 & 3.1 & 4.1 & 0 \end{pmatrix} \end{array}$$

(a) $P(t)$ (b) EHM^t

Figure 3: An Example of EHM

3.2 Using Partial Solutions in EHBSA

In EHBSA, a *template* individual, from which a partial solution is obtained, is chosen from $P(t)$ (normally, randomly). A crucial question when we create a new solution $c[]$ is how to determine which part of the partial solution the $c[]$ will borrow from the template. To ensure robustness across a wide spectrum of problems, it is advantageous to introduce variation both in the portion and the number of nodes of the partial solution that is borrowed from the template. First we choose the starting node position of the partial solution randomly. Thereafter, the number of nodes of the partial solution must be determined. Let us represent the number of nodes that are constructed based on EHM by l_s. Then, l_c, the number of nodes of the partial solution, which $c[]$ borrows from the template, is $l_c = L – l_s$. Here, let us introduce a control parameter γ which can define $E(l_s)$ (the average of l_s) by $E(l_s) = L \times \gamma$. To determine the sampling portion in a string, we used the n cut-point approach in previous study [8]. However with the approach, $E(l_s)$ is $L/2$, $L/3$, . . . for n= 2, 3, and so on, and, γ corresponds to $1/n$, i.e., γ can take only the values of 0.5, 0.333, and 0.25, corresponding to n= 2, 3, 4 and so on. In the current research, we extend this elementary method to a more flexible technique which allows for γ taking values in the rage [0.0, 1.0]. The probability density functions for l_s in this research are:

$$f_s(l) = \frac{1-\gamma}{L\gamma}\left(1-\frac{l}{n}\right)^{\frac{1-2\gamma}{\gamma}}, \text{ for } 0 < \gamma \leq 0.5. \quad (1)$$

$$f_s(l) = \frac{\gamma}{L(1-\gamma)}\left(\frac{l}{n}\right)^{\frac{2\gamma-1}{1-\gamma}}, \text{ for } 0 < \gamma \leq 0.5. \quad (2)$$

Using partial solutions in EHBSA is summarized in Figure 4. For a given γ value, l_s is generated according to Eqs. 1 or 2, and c_{top}, the first position of partial solution for $c[]$ to borrow from the template, is sampled randomly. Then, the partial solution of length $l_c = L – l_s$, and which starts from c_{top} is copied to $c[]$ from the template. Then the remaining sequence of nodes of length l_s in $c[]$ is sampled according to EHM probabilistically.

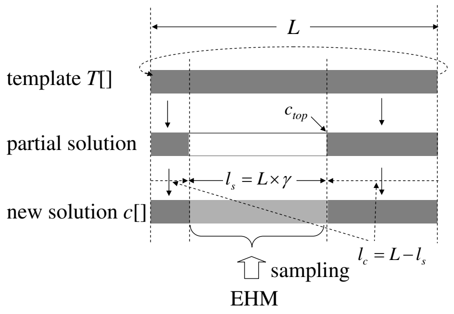


Figure 4: Using partial solutions in EHBSA

3.3 Results

In this section, we show the results using the scheme described in Section 3.2 on TSP instances of gr48 and pr76. The γ values were tested from 0.1 to 1 with step size 0.1. Here note that $\gamma = 1$ corresponds to the case where no partial solutions are used. The following control parameter values were used: population size $N = L \times 2$ (L is the number of cities), maximum number of tour constructions $E_{max} = L \times 20,000$. Results are summarized in Table 1 where $\#OPT$ indicates the number of runs which found the best-known solution, MNE indicates the mean number of tour constructions to find the best-known solution in those runs where it did find the solution, and $Error$ indicates the average excess value from the best-known solution over 25 independent runs.

From these results, we can see that good results of $\#OPT$, MNE, and $Error$ are found with γ values in [0.3, 0.4] for both gr48 and pr96. In the case in which we do not use partial solutions ($\gamma = 1$), the second worst results were obtained on both instances, i.e., $\#OPT = 0$ for both instances, and a $Error = 1.159\%$ for gr48 and $Error = 9.747\%$ for pr76, respectively.

Table 1: Results of EHBSA on gr48 and pr76

γ	gr48			pr76		
	#OPT	*MNE*	*Error*	*#OPT*	*MNE*	*Error*
0.1	0	-	1.753%	0	-	14.895%
0.2	25	185458.3	0.000%	10	1276794.0	0.175%
0.3	25	115680.5	0.000%	24	735272.6	0.005%
0.4	24	113575.6	0.022%	23	595011.3	0.010%
0.5	23	135540.6	0.029%	21	622464.3	0.045%
0.6	16	182385.0	0.099%	17	687602.9	0.092%
0.7	4	256192.0	0.278%	13	960202.4	0.115%
0.8	0	-	0.682%	2	1400801.5	1.076%
0.9	0	-	0.912%	0	-	5.336%
1	0	-	1.159%	0	-	9.747%

Figure 5 shows the change of the diversity of the population. Here, the diversity was measured by the standard deviation (STD) of tour length of individuals in the population and was averaged over 25 runs. From this figure, we can see that for $\gamma = 0.1$ the change of the STD is very slow. This is because only 10% of new edges are sampled in generating a new string on average. Thus, the population did not converge in the defined maximum number of tour constructions E_{max}. With γ values of 0.3, and 0.4, each STD gradually becomes smaller as the number of tour constructions increases, resulting in successful searches. However, in the case in which we do not use partial solutions ($\gamma = 1$), the population loses diversity rapidly, resulting in a failed search. Thus, we can see the effectiveness of using partial solutions with appropriate γ values.

4 Using Partial Solutions in ACO

As a bio-inspired computational paradigm, ACO has been applied with great success to a large number

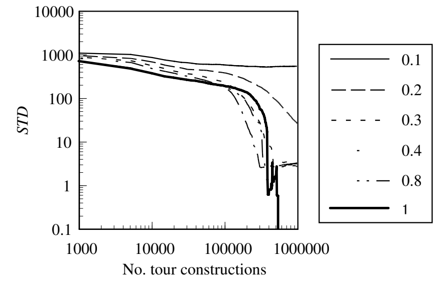


Figure 5: Change of population diversity on gr48

of hard problems in permutation domains [11]. The first ACO algorithm was called the Ant System (AS), and was applied to the TSP. Since then, many variant ACO algorithms have been proposed as extensions of AS [11]. On a TSP, AS works as follows: Let m be the total number of ants. Initially, each ant is put on a randomly chosen city k(k=1, ..., n). Let $\tau_{ij}(t)$ be the trail density on edge (i, j) at iteration t. Each of m ants at iteration t makes a tour by choosing a sequence of cities. When all ants complete their tours, the trail density of each edge on their tours is updated. We can note here that the $\tau_{ij}(t)$ corresponds to $EHM^t = (e^t_{i,j})$ of EHBSA and thus ACO has a tight relation with EDAs.

In this section, we present the results of a new ACO which uses partial solutions in generating new solutions. Please see [12] for more detail. The algorithm is called the *c*AS (cunning AS). In traditional ACO algorithms, each ant generates a solution probabilistically or pseudo-probabilistically based on the current pheromone trail $\tau_{ij}(t)$. In *c*AS, an agent called *cunning ant* (*c-ant*) is introduced. The *c-ant* differs from traditional ants in the manner of solution construction. It constructs a solution by borrowing a part of existing solutions. The remainder of the solution is constructed based on $\tau_{ij}(t)$ probabilistically as usual. An agent which has a solution borrowed by a *c-ant* is called a *donor ant* (*d-ant*). Using *c-ant* in this way, we can prevent premature stagnation the of search, because only a partial solutions are newly generated, and this can prevent over exploitation caused by strong positive feedback to $\tau\ ij(t)$ as discussed in Section 2.

Figure 6 illustrate the convergence process by the change of $Error$ on kroA100 TSP instance for γ values of 0.1, 0.3, 0.5, 0.7, and 0.9. Early stagnations of search can be observed with γ vales of 0.7 and 0.9. With γ values of 0.3 and 0.5, stagnations of search occur much later in the search. With a γ value of 0.1, no stagnation can be observed. But the convergence process is very slow. Thus we can see that using appropriate small values of γ can prevent over exploitation.

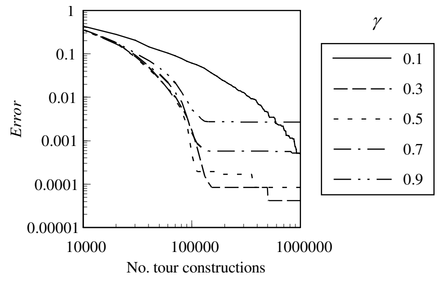


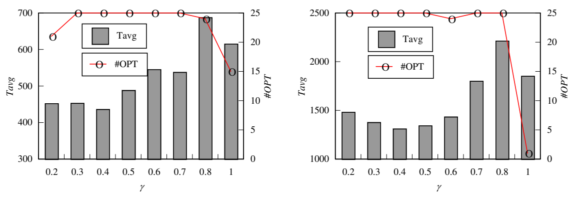
Figure 6: Convergence process on kroA100

Here we show *c*AS with a local search on TSP. One of the best performing local searches for TSP is the well-known Lin-Kernighan algorithm (LK) [13]. We used a Chained LK which applies the basic LK repeatedly. For γ value, γ =0.4 was used.

Table 2: Results of *c*AS with LK on TSP. T_{avg} is average time in second to find optimal in successful runs. The machine we used had two Opteron 280 (2.4GHz) processors with Java code.

TSP	*c* AS						MMAS			Chained LK			T_{max}
	c AS (γ=0.4)			*non-c* AS (γ=1)									
	#OPT	*Error* (%)	T_{avg}	*#OPT*	*Error* (%)	T_{avg}	*#OPT*	*Error* (%)	T_{avg}	*#OPT*	*Error* (%)	T_{avg}	
pr2392	25	0.00	104.9	24	0.00	137.2	12	0.00	211.3	4	0.17	122.4	240
fl3795	25	0.00	435.1	15	0.00	615.9	17	0.00	770.7	0	0.57	-	1400
rl5934	25	0.00	1336.1	1	0.00	1854.6	10	0.00	2533.6	0	0.27	-	3300

To confirm the effectiveness of combining *c*AS with LK, we also tested the following three algorithms: *non-c*AS with LK (i.e., γ=1; no partial solutions are used), MMAS with LK, and Chained LK alone. The results are shown in Table 2. We can see all algorithms of *c*AS, *non-c*AS, and MMAS showed very small values of *Error* by combining LK and thus the advantage of combining these algorithms with LK is very clear. However, when we focus our attention on the results of $\#OPT$, all algorithms except for *c*AS could not attain $\#OPT$ = 25. In contrast to this, *c*AS could attain $\#OPT$ = 25 within the allowed run time T_{max} showing the smallest T_{avg} (average time in seconds to find optimal in successful runs) among algorithms tested. Thus, we can see that using partial solutions is useful when the approach is combined with local search. Figure 7 shows the variations of T_{avg} and $\#OPT$ for various γ values.



(1) fl3795 (2) rl5934

Figure 7: Results of cAS for variable γ values with LK on (1) fl3795 and (2) rl5934

5 Summary

In previous studies, we have proposed an algorithm referred to as the edge histogram based sampling algorithm (EHBSA) with EDA scheme for permutation domains. In EHBSA, new solutions are obtained by combining partial solutions which exist in the current population, and partial solutions newly generated according to an edge histogram model of the current population. In this paper, we showed that using partial solutions can maintain diversity of a population resulting in a successful search. We have also shown that using partial solutions in ACO can greatly enhance its performance. Thus, we can expect that a scheme using partial solutions in generating new solutions can be used in a wide range of EDAs. Appling this scheme to EDAs in other domains, such as real coding and binary coding remains for for future work.

Acknowledgements

This research is partially supported by the Ministry of Education, Culture, Sports, Science and Technology of Japan under Grant-in-Aid for Scientific Research number 16500143.

References

[1] D. E. Goldberg, *The design of innovation: Lessons from and for competent genetic algorithms* Vol. of Genetic and Evolutionary Computation, Kluwer, Boston, MA, 2002.

[2] M. Pelikan, *Bayesian optimization algorithm: From single level to hierarchy*, Doctoral dissertation, University of Illinois at Urbana-Champaign, Urbana, IL, 2002. (Also IlliGAL Report No. 2002023).

[3] M. Pelikan, D. E. Goldberg, and F. Lobo, "A survey of optimization by building and using probabilistic models," *Computational Optimization and Applications*, Vol. 20, No. 1, pp. 5-20, 2002.

[4] P. Larrañaga and J. A. Lozano, *Estimation of distribution algorithms: A new tool for evolutionary computation*, Kluwer, Boston, MA, 2002.

[5] H. Mühlenbein and G. Paaβ, "From recombination of genes to the estimation of distributions I. binary parameters," *Proc. of the 4th Parallel Problem Solving from Nature (PPSN IV)*, pp. 178-187, 1996.

[6] P. A. N. Bosman and D. Thierens, "Permutation optimization by iterated estimation of random keys marginal product factorizations," em Parallel Problem Solving From Nature (PPSN VII), pp. 331-340, 2002.

[7] V. Robles, P. D. Miguel, and P. Larrañaga, "Solving the traveling salesman problem with edas," *Estimation of Distribution Algorithms*, pp. 211-229, 2002.

[8] S. Tsutsui, "Probabilistic model-building genetic algorithms in permutation representation domain using edge histogram," *Proc. of the 7th Int. Conf. on Parallel Problem Solving from Nature (PPSN VII)*, pp. 224-233, 2002.

[9] S. Tsutsui and M. Miki, "Solving flow shop scheduling problems with probabilistic model-building genetic algorithms using edge histograms," *Proc. of the 4th Asia-Pacific Conference on Simulated Evolution And Learning (SEAL02)*, 2002.

[10] M. Dorigo and L. M. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem," *IEEE Trans. on Evolutionary Computation*, Vol.1, No. 1, pp. 53-66, 1997.

[11] M. Dorigo and T. Stützle, T., *Ant Colony Optimization*, MIT press, Massachusetts, 2004.

[12] S. Tsutsui, "cAS: Ant colony optimization with cunning ant," *Proc. of the 9th Int. Conf. on Parallel Problem Solving from Nature (PPSN IX)*, pp. 162-171, 2006.

[13] S. Lin, and B. W. Kernighan, "An effective heuristic algorithm for the traveling salesman problem," *Operations Research*, Vol. 21, pp. 498–516, 1973.