The Traveling Salesrep Problem, Edge Assembly Crossover, and 2-opt

J. Watson, C. Ross, V. Eisele, J. Denton, J. Bins, C. Guerra,
D. Whitley, A. Howe

Computer Science Department, Colorado State University, Fort Collins, CO 80523
{watsonj, rossc, eisele, bins, denton, guerra, whitley, howe}@cs.colostate.edu

Abstract. Optimal results for the Traveling Salesrep Problem have been reported on problems with up to 3038 cities using a GA with *Edge Assembly Crossover* (EAX). This paper first attempts to independently replicate these results on Padberg's 532 city problem. We then evaluate the performance contribution of the various algorithm components. The incorporation of 2-opt into the EAX GA is also explored. Finally, comparative results are presented for a population-based form of 2-opt that uses partial restarts.

1 Introduction

Nagata and Kobayashi [7] report optimal solutions on Traveling Salesrep Problems (TSPs) ranging in size from 101 to 3038 cities. They also report modest computation times on a 200MHz Pentium, ranging from approximately 13 minutes for the well known Padberg 532 city problem [8], to 2.5 hours for a 3038 city problem. These results are an important break-through for two reasons. First, they represent a dramatic improvement over previous evolutionary-based optimization methods for the Traveling Salesrep Problem; other researchers have reported good results for the Padberg 532 city problem, but rarely optimal solutions on problems of this size and larger. Second, these results are close enough to the state-of-the-art that evolutionary-based optimization methods could have a very real potential to provide the basis for new state-of-the-art approaches to the Traveling Salesrep Problem.

Although not explored in this paper, Freisleben and Merz [3] also report impressive performance on problems of similar complexity. Their algorithm, Genetic Local Search (GLS), exploits the 'big-valley' structure of TSP fitness landscapes documented in [1]. To exploit this structure, crossover in GLS is implemented using a computationally intensive search algorithm. Furthermore, GLS maintains an extremely small (≤ 40) population of individuals. In constrast, the EAX GA uses relatively weak search operators and a large population to obtain similar performance results.

This paper explores three questions. Two of these questions relate directly to the work of Nagata and Kobayashi. First, can the results be independently replicated from their description of the algorithm? We can answer this question in the affirmative; our implementation replicates the original results to within,

on average, 0.054% of the optimal tour cost; we document our implementation of their algorithm in Section 2 to enable resolution of the remaining discrepancies. Given this success, we then ask the question 'What components of the Nagata and Kobayashi algorithm are critical to performance?' We cannot definitively answer this question, but we provide some partial answers by testing the effects of removing or replacing various components of their algorithm and measuring the resulting performance impact.

The third question relates to the use of 2-opt. For the past ten years, researchers have reported near-optimal results on Padberg's 532 city problem using evolutionary algorithms [2] [5] [6]. Yet all of these approaches have used 2-opt as a local search algorithm within the genetic algorithm. Given this observation, we ask: 'To what degree does the success of these algorithms depend on the use of 2-opt and to what degree do the "evolution-based" features in the various algorithms contribute to producing good results?'

A distinguishing characteristic of the Nagata and Kobayashi algorithm is that it does not use 2-opt. Their recombination operator, *Edge Assembly Crossover* (EAX), uses the edges from the two parents to construct disjoint subtours. Then, using a construction analogous to a minimal spanning tree, the subtours are connected in a greedy fashion to produce the child tour. Thus, the EAX operator is not "blind;" local information is exploited in determining which edges to use to connect subtours. Thus, there would seem to be no reason not to also use 2-opt in conjunction with the algorithm.

Another important trait of the EAX operator is that it will introduce new edges into the child when connecting subtours. Edges not in the parents, or perhaps not even in the population, are introduced into offspring. We are now convinced that a good operator for the TSP must be able to introduce new edges into the offspring. This is contrary to the tenets put forward by Radcliffe [9] [10] and contrary to the goals behind the construction of operators such as Edge-3 [12] [5] which attempt to inherit as many edges from parents as possible.

The argument as to why good new edges must be introduced during recombination (or by mutation, or local search) is simple. Mathias and Whitley [5] point out that the complete graph of all possible edges for a symmetric TSP has $(N^2 - N)/2$ edges, where N is the number of cities. Each tour samples N of these edges, so a population must be of size at least (N-1)/2 in order to sample each edge exactly once. Assume population size is proportional to the number of the cities (as in the Nagata and Kobayashi algorithm and the algorithms presented here). Then each edge occurs twice in expectation in an initial random population. Selection can therefore quickly eliminate edges from the population. Good edges can also be lost if they occur in poor tours. Thus it is important for operators to intelligently introduce new good edges. Unlike other crossover operators, EAX makes the introduction of new edges an integral part of recombination–which may contribute to its effectiveness.

The next section describes our implementation of the Nagata and Kobayashi algorithm. Section three describes our comparative experiments. Our study focuses exclusively on Padberg's symmetric 532 city problem for several reasons.

First, it has been widely studied and is notoriously difficult to solve to optimality. Seocnd, Nagata and Kobayashi also found this problem to be harder than any of the larger problems they investigated. Finally, Section four discusses the role of 2-opt in hybrid evolutionary algorithms.

2 Nagata and Kobayashi's Algorithm

Once two parents have been selected for crossover, the EAX operator merges these two individuals into a single graph denoted by R. The two parents are denoted by A and B, respectively. Each edge in R is annotated with the parent to which it belongs. R may contain two instances of the same edge, if both parents contain the edge. R is next divided into a set of disjoint subtours.

2.1 Edge Assembly Crossover (EAX): AB-Cycles

Let v_i represent a vertex from R and let $(v_i, v_j), i \neq j$, represent an edge. Suppose (v_i, v_j) represents an edge randomly chosen from parent A. (Note that A and B are just randomly assigned labels; the choice of A is arbitrary.) Choose one vertex (either v_i or v_j) as the origin. If v_i is the origin, then choose an edge which leads from the second vertex, v_j, to any other vertex in R. However, this edge must come from parent B. If more than one such edge exists, a random selection is made. The algorithm continues to traverse R, at each step alternately picking edges from parent A and parent B.

After each edge is traversed, the algorithm checks to see if adding this new edge to the set of previously selected edges will result in what Nagata and Kobayashi term an *AB-cycle*. An AB-cycle is a even-length sub-cycle of R with edges that alternately come from A and B. An AB-cycle may repeat cities, but not edges. While there can be two edges between a pair of cities, they are uniquely identified as an A or B edge, and thus distinct.

Once an AB-cycle has been found it is stored and the edges making up that cycle are removed from R. The algorithm repeats this procedure until R contains no more edges, having been completely decomposed into a set of AB-cycles.

The first several edges used in the construction of the AB-cycle may not appear in the final AB-cycle. This occurs when the final edge connects back onto the subgraph at some city x other than the origin city, and the induced subcycle is an AB-cycle. In this case the 'extraneous' edges are left in the R graph, to eventually be used in forming another cycle. In this situation Nagata and Kobayashi choose an edge incident with x from R to begin construction of the next AB-cycle. As we can find no reason to prefer this method over random selection, our algorithm always selects the starting location of a new AB-cycle at random from R.

When R is undirected, as is the case for the symmetric TSP, the set of AB-cycles is not uniquely determined by the algorithm. Furthermore, a number of "ineffective" AB-cycles may be formed by the algorithm. Such cycles consist solely of two edges between the same pair of cities. Any ineffective AB-cycles are removed from consideration by the remaining phases of the algorithm.

2.2 Edge Assembly Crossover (EAX): E-set to Offspring

After construction of the set of AB-cycles, a subset of AB-cycles is chosen to be used in the generation of an intermediate child. This subset is called an *E-set*. Two methods for selecting AB-cycles for inclusion into the E-set are defined by Nagata and Kobayashi. The first, denoted by $EAX(rand)$, simply selects each AB-cycle for inclusion into the E-set with a probability of 0.5. The second method, denoted by $EAX(heuristic)$, makes use of a heuristic metric to determine the inclusion of an AB-cycle into the E-set. The metric balances the need for maintaining population diversity against the need to reduce overall tour cost. We implemented the heuristic metric as described in [7]. As described in the next subsection, the EAX GA uses these two methods in different contexts.

Construction of an intermediate child, C, begins with a copy of parent A. Then each edge of each subtour in the E-set is examined, with the following actions taken on C. If the edge from the E-set is a member of parent A, the edge is deleted from C. If the edge is a member of parent B, the edge is added to C. The result is a set of disjoint subtours which comprise the intermediate child.

The last stage of the EAX operator involves transformation of the intermediate child into a single legal tour. The subtours are merged into a single tour using a greedy construction procedure. The smallest tour, in terms of number of edges, is selected from the set of subtours. A pair of edges, one from the smallest subtour and one from another distinct subtour, is then selected such that merging of the two subtours at those edges minimizes the change in overall tour cost. Let (v_q, v_{q+1}) be an edge in one subtour, and (v'_r, v'_{r+1}) be an edge in the other. The location of the verticies is arbitrary and addition on the indices for edges in v and v' is $mod(|v|)$ and $mod(|v'|)$. A greedy algorithm for connecting subtours uses the following metrics:

$$Cut(q,r) = (\$(v_q, v_{q+1}) + \$(v'_r, v'_{r+1}))$$

$$Link(q,r) = MIN\ ((\$(v_q, v'_r) + \$(v_{q+1}, v'_{r+1})), (\$(v_q, v'_{r+1}) + \$(v_{q+1}, v'_r)))$$

where $\$(v_i, v_j)$ is the cost of edge (v_i, v_j). Then we seek:

$$MIN\ [Link(q,r) - Cut(q,r)] \quad \forall r, q$$

This process of merging subtours is repeated until a single tour remains. Nagata and Kobayashi use a heuristic method to reduce the number of edge pairs considered. We use an exhaustive enumerative of all edge pairs, since the procedure did not contribute significantly to the overall runtime.

2.3 The EAX Genetic Algorithm

Nagata and Kobayashi introduce a variation on a traditional generational GA which employs a form of elitist tournament selection. Two parents are randomly selected, without replacement, from the population and recombined using

crossover. The two parents and the resulting child are then compared, and the individual with the best fitness is passed to the next generation. This procedure is repeated to produce all N members of the next generation. Nagata and Kobayahsi claim that the EAX GA is better able to maintain population diversity by giving a large number of parents the ability to pass children into the next generaton.

Finally, the EAX GA defines recombination as an iterative procedure. First, a child is produced using the $EAX(heuristic)$ E-set construction method. Should this fail to produce a child with better fitness than both parents, the $EAX(rand)$ E-set construction method is used to produce more children until either such an improved child is found or 100 children are produced. We refer to this method as *iterative child generation*, or ICG.

3 Empirical Results

In this section, we investigate the performance contribution of the various components of the Nagata and Kobayahsi algorithm. For convenience we will refer to their GA as the *EAX GA*, while *EAX operator* will refer to the actual recombination operator.

We first compared the EAX GA with GENITOR [1] [11], with both using the EAX operator. GENITOR is a steady-state GA, with a child always replacing the worst member of the population. Linear ranked selection was employed, with a bias of 1.25.

Next, we focused on the use of "offspring improving" operators such as ICG and 2-opt. ICG, in conjunction with the ability of the EAX operator to produce a variety of children (depending on the composition of the E-set), increases the probability of creating a child with higher fitness than either parent. Local search mechanisms such as 2-opt [5] also increase this probability. One of the issues we explore is the use of 2-opt in place of ICG.

Lastly, the performance of another crossover operator, Edge-3 [5], is explored in the context of both the 2-opt and ICG search operators.

As noted, the test problem is Padberg's 532 city problem. TSPLIB [2] reports 27686 as the optimal tour cost for this instance. A population size of 500 was used in both the EAX and GENITOR GA's; the sizing is identical to that reported in [7]. A GA population was considered converged when the best individual fitness equaled the average individual fitness; all runs were allowed to fully converge. The code was implemented in C on a SUN Ultrasparc-30. We used the UNIX rand48 family of random number generators.

Each experimental trial consisted of 30 runs of a particular combination of GA, crossover operator, and search operator. The final tour cost and number of evaluations required for convergence was recorded for each run. For purposes of comparison with GENITOR results, the number of evaluations required by

[1] GENITOR can be found at http://www.cs.colostate.edu/ genitor

[2] TSPLIB: www.iwr.uni-heidelberg.de/iwr/comopt/soft/TSPLIB95/TSPLIB.html

ALG	X-over Operator	Search Op	Mean	Percent Abv. Opt.	Best	Percent Abv. Opt.	Worst	Percent Abv. Opt.	Std. Dev.
EAX	EAX	None	27840	0.56	27713	0.09	28010	1.17	86.91
EAX	EAX	ICG	27709	0.08	27693	0.03	27739	0.19	9.32
EAX	EAX	2-opt	27742	0.20	27708	0.08	27838	0.55	29.12
GEN	EAX	None	28379	2.50	28065	1.37	28720	3.73	181.75
GEN	EAX	ICG	27830	0.52	27739	0.19	28004	1.15	77.40
GEN	EAX	2-opt	27861	0.63	27759	0.26	28002	1.14	70.80
GEN	EDGE-3	NONE	74049	167.46	69890	152.44	78966	185.22	2615.70
GEN	EDGE-3	2-OPT	27878	0.70	27781	0.34	27999	1.13	54.00

Table 1. Final tour costs.

ALG	X-over Operator	Search Op	Mean	Best	Worst	Std. Dev.
EAX	EAX	None	72367	60500	85500	5755
EAX	EAX	ICG	51433	45500	56000	2605.5
EAX	EAX	2-opt	36317	31000	43500	2978.3
GEN	EAX	None	19883	15737	22664	1812.3
GEN	EAX	ICG	14680	12359	16734	1145.7
GEN	EAX	2-opt	10525	7981	12251	1088.6
GEN	Edge-3	None	288187.10	242744	333388	20151.41
GEN	Edge-3	2-opt	145369.83	109216	243464	23955.66

Table 2. Number of evaluations required for convergence.

the EAX GA is taken as the product of the population size and number of generations required for convergence.

Tables 1 and 2 summarize the results from all experimental trials. Table 1 reports tour costs; Table 2 reports the number of evaluations required for convergence. The number of evaluations represents the number of times a single pair of parents was used to produce a child. This number does not include any additional tour evaluations required by the search operators. The first three columns of each table represent the algorithm components used in a given trial. The remaining columns report summary statistics for the various trials.

3.1 Influence of Genetic Algorithm on Performance

A substitution experiment was used to determine the relative contribution of the EAX GA and GENITOR to search performance. We performed two one-tailed T-tests (GA as the independent variable, final tour cost and number of evaluations as the dependent variables) on the data from all experimental trials using the EAX crossover operator. The T-test with tour cost as the dependent variable indicated a significant difference ($t(178) = 3.42, p < 0.01$), with the EAX GA outperforming GENITOR. Similarly, the T-test with number of evaluations

as the dependent variable indicated a significant difference ($t(178) = 12.12, p < 0.01$), with GENITOR converging faster on average than the EAX GA.

The reduction in tour cost obtained by the EAX GA comes at the expense of roughly tripling the number of tour evaluations in comparison to GENITOR. However, the reduction is significant, and enables the EAX GA to find solutions within a fraction of the optimal tour cost. Furthermore, as shown in Table 1, the EAX GA also offers the benefit of lower variance in final tour cost than that provided by GENITOR.

3.2 Influence of Iterative Child Generation on Performance

As described in Section 2, ICG enables the EAX operator to generate multiple potential offspring from the same set of parents, increasing the probability of finding an improved child. In constrast, previous work on the TSP [4] [2] [5] [6] has focused on using variants of 2-opt to increase the fitness of a child produced by some crossover operator. In either case, the goal is equivalent: to find offspring similar to or superior to the parents in fitness. Thus, we analyze which method has the higher payoff. Finally, we also evaluated the performance of the EAX operator without ICG or 2-opt.

For completeness, we evaluated the performance of the EAX operator both with and without ICG. We performed two one-tailed T-tests (selection of ICG as the independent variable, final tour cost and number of evaluations as the dependent variables) on the data from experimental trials using both GA's.

Both T-tests indicated significant differences in both tour cost ($t(118) = 3.42, p < 0.01$) and number of evaluations ($t(118) = 2.51, p < 0.01$), with ICG substantially decreasing final tour costs. ICG has the additional apparent benefit of decreasing the number of evaluations required for convergence, a side-effect of improving the probability of finding a child better than both parents. However, this benefit is actually detrimental to run-time; the number of evaluations does not count the considerable number of tour evaluations consumed in the search for a better child.

To determine whether various offspring improvement operators (none, 2-opt, or ICG) led to different performance, we ran a pair of two-way ANOVA's (GA and search operator as the independent variables, final tour cost and number of evaluations as the dependent variables) on data from experimental trials using the EAX operator and both GA's.

The ANOVA's with tour cost as the dependent variable indicated significant main effects in both the GA ($F(1) = 346.5, p < 0.01$) and the search operator ($F(2) = 241.3, p < 0.01$). Similarly, a significant interaction effect between the GA and search operator was detected ($F(2) = 100.4, p < 0.01$). The ANOVA's with number of evaluations as the dependent variable provided nearly identical results, with main and interaction effects detected at $p < 0.01$.

A final two-tailed T-test used tour cost as the dependent variable and the choice of either the ICG or 2-opt as the independent variable indicated no significant difference ($t(118) = 0.93, p < 0.0325$) in mean tour cost. Given roughly equal mean tour costs, 2-opt offers significant advantages over the ICG search

operator. First, each additional tour evaluation performed by 2-opt can be done in constant time, in contrast to the full linear-time evaluation required by ICG. Second, inspection of Table 1 indicates 2-opt converges nearly twice as fast as ICG (verified by a one-tailed t-test, $p < 0.01$).

3.3 Influence of ICG and 2-opt on Edge-3 recombination

The previous section focused on the impact of various search operators on performance. Both the 2-opt and ICG substantially enhanced the performance of the 'plain' EAX operator. However, the complexity of the EAX operator is high in comparison to other operators such as Edge-3 [5] and MPX [6]. In this section, we examine the performance impact of the 2-opt and ICG on the Edge-3 crossover operator. In addition, the performance is compared with that obtained using the EAX operator.

We performed a simple experiment to measure the performance of the hybridization of the Edge-3 crossover and ICG search operators. Under Edge-3, children inherit a high (95-99%) fraction of tour edges directly from their parents. The remaining edges are chosen at random such that a legal tour is constructed. The hybridization of Edge-3 and ICG was implemented simply by iterating the Edge-3 operator on identical parents until either 100 iterations were performed or a child better than both parents was found.

The Edge-3/ICG hybridization was used in conjunction with the GENITOR GA. Thirty runs were performed, resulting in an average tour cost of 46818.49, which is substantially worse than any of the results obtained using the EAX operator. In spite of the poor relative performance, the ICG search operator substantially improved the performance obtained by the Edge-3 operator when used in isolation; as shown in Table 1, the mean tour cost obtained using Edge-3 in isolation was 74049.67.

Finally, we compared the performance of the GENITOR/Edge-3/2-opt hybrid with that of the EAX/EAX/2-opt hybrid using two one-tailed T-tests. Both T-tests indicated significant differences in both tour cost ($t(29) = 3.02, p < 0.01$) and number of evaluations ($t(29) = 33.16, p < 0.01$). In both cases, the EAX/EAX/2-opt hybrid outperformed the GENITOR/Edge-3/2-opt hybrid.

It should be noted that the difference in mean tour cost between the two variants is only 136; the statistical significance stems primarily from the low variance in tour cost obtained with 2-opt. In spite of a significant increase in evaluations, the 2-opt operator was able to reduce differences in tour costs between the EAX and Edge-3 operators to nearly identical levels. This 'performance leveling' prompted us to investigate the performance of 2-opt when used in relative isolation.

4 The Impact of 2-opt

As shown in the previous section, 2-opt is extremely effective at improving the performance of the EAX and GENITOR GA's for the TSP. This lead us to ask

ALG	X-over Operator	Search Op	Mean	Percent Abv. Opt.	Best	Percent Abv. Opt.	Worst	Percent Abv. Opt.	Std. Dev.
EAX	EAX	ICG	27709	0.08	27693	0.03	27739	0.19	9.32
EAX	EAX	2-opt	27742	0.20	27708	0.08	27838	0.55	29.12
GEN	EDGE-3	2-opt	27878	0.70	27781	0.34	27999	1.13	54.00
2-opt	N.A.	N.A.	27985	1.08	27841	0.55	28211	1.89	90.11

Table 3. A comparison of the best algorithms with 2-opt using partial restarts.

the question 'What if 2-opt were the only operator?' Of course, 2-opt must be applied to distinct starting tours to produce different results. So to create different starting points, we used an idea which has connections to both evolutionary algorithms and local search.

A small population of solutions is used (30 in this case). 2-opt is applied to each solution in the population until all of the solutions are locally optimal. The best solution in the population is then used to re-seed the entire population.

Starting at a random city, the best solution is broken into segments composed of two adjacent edges. These fragments are then randomly reconnected. The entire population is regenerated in this way and then 2-opt is applied to all of the new solutions. The idea is that this both provides a type of partial restart to local search, and also preserves good "building blocks" from the previous best solution.

There are many variations on this idea that could be explored: the best two or three solutions could be used to reseed the population and the best solutions could be broken into fragments that preserve segments composed of three or four edges. However, we wished to keep this search strategy simple.

Table 3 presents results for the 2-opt with partial restarts algorithm. The other results in the table are those previously reported in this paper for the EAX GA using ICG and 2-opt, as well as the GENITOR/Edge-3/2-opt hybrid algorithm. A pair of one-tailed T-tests indicate that there is a significant difference in final tour costs between the 2-opt with partial restarts algorithm and both the original Nagata and Kobayahsi algorithm ($(t(58) = 3.57, p < 0.01)$) and the EAX GA/EAX operator/2-opt hybrid algorithm ($(t(58) = 2.93, p < 0.01)$). In spite of the poor relative performance, it is surprising how well a simple algorithm based on 2-opt works in this domain.

5 Conclusions

We replicated the results of Nagata and Kobayahsi with only a very slight error margin. We document our interpretation of their algorithm description in an effort to resolve the remaining discrepancies. More importantly, we provide results that raise important issues concerning what components of their algorithm are critical to performance. Our results suggest that 2-opt might be used as an

effective replacement for ICG. Since the EAX operator uses local information anyway, there is no reason not to use 2-opt to improve the resulting child.

The effectiveness of the selection mechanism in the EAX GA in comparison to the selection mechanism in GENITOR was also surprising. Allowing an improved offspring to replace one of the parents instead of the worst member of the population may have two important effects. It clearly results in lower selective pressure; but it is also likely that when children replace parents the children still retain some of the "genetic material" of the parents. Thus, this selection scheme may result in the population maintaining diversity for a longer period of time. This point of view is supported by the data which shows that the EAX GA converges much slower than GENITOR.

6 Acknowledgements

This effort was sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant number F49620-97-1-0271. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

References

1. K. Boese. Cost versus distance in the traveling salesman problem. Technical report, Computer Science Department, Univeristy of California, Los Angeles, 1995.
2. Larry Eshelman. The CHC Adaptive Search Algorithm. How to Have Safe Search When Engaging in Nontraditional Genetic Recombination. In G. Rawlins, editor, *FOGA -1*, pages 265–283. Morgan Kaufmann, 1991.
3. Bernd Freisleben and Peter Merz. New genetic local search operators for the traveling salesman problem. In H.M. Voigt, W. Ebeling, Ingo Rechenberg, and H.P. Schwefel, editors, *Parallel Problem Solving from Nature, 4*, pages 890–899. Springer/Verlag, 1996.
4. Martina Gorges-Schleuter. ASPARAGOS An Asynchronous Parallel Genetic Optimization Strategy. In J.D. Schaffer, editor, *Proc. of the 3rd Int'l. Conf. on GAs*, pages 422–433. Morgan Kaufmann, 1989.
5. Keith E. Mathias and L. Darrell Whitley. Genetic Operators, the Fitness Landscape and the Traveling Salesman Problem. In R. Männer and B. Manderick, editors, *Parallel Problem Solving from Nature, 2*, pages 219–228. Elsevier Science Publishers, 1992.
6. H. Mühlenbein. Evolution in Time and Space: The Parallel Genetic Algorithm. In G. Rawlins, editor, *FOGA -1*, pages 316–337. Morgan Kaufmann, 1991.
7. Yuichi Nagata and Shigenobu Kobayashi. Edge assembly crossover: A high-power genetic algorithm for the traveling salesman problem. In T. Bäck, editor, *Proc. of the 7th Int'l. Conf. on GAs*, pages 450–457. Morgan Kaufmann, 1997.
8. W. Padberg and G. Rinaldi. Optimization of a 532 City Symmetric TSP. *Optimization Research Letters*, 6(1):1–7, 1987.
9. Nicholas J. Radcliffe. The algebra of genetic algorithms. *Annals of Maths and Artificial Intelligence*, 10:339–384, 1994.

10. N.J. Radcliffe and P.D. Surry. Fitness variance of formae and performance predictions. In D. Whitley and M. Vose, editors, *FOGA - 3*, pages 51–72. Morgan Kaufmann, 1995.
11. Darrell Whitley and Joan Kauth. GENITOR: A Different Genetic Algorithm. In *Proceedings of the 1988 Rocky Mountain Conference on Artificial Intelligence*, 1988.
12. Darrell Whitley, Timothy Starkweather, and D'ann Fuquay. Scheduling Problems and Traveling Salesmen: The Genetic Edge Recombination Operator. In J. D. Schaffer, editor, *Proc. of the 3rd Int'l. Conf. on GAs*. Morgan Kaufmann, 1989.

This article was processed using the LaTeX macro package with LLNCS style