Combination and Comparison of Different Genetic Encodings for the Vehicle Routing Problem

Stefan Vonolfen[1], Andreas Beham[1], Michael Affenzeller[1], Stefan Wagner[1], and Andreas Mayr[2]

[1] School of Informatics, Communications and Media
Upper Austria University of Applied Sciences, Campus Hagenberg
Softwarepark 11, 4232 Hagenberg, Austria
{stefan.vonolfen,andreas.beham,michael.affenzeller,stefan.wagner}@fh-hagenberg.at

[2] mayr.andreas@gmail.com

Abstract. Unlike for other problems, such as the traveling salesman problem, no widely accepted encodings for the vehicle routing problem have been developed yet. In this work, we examine different encodings and operations for vehicle routing problems. We show, how different encodings can be combined in one algorithm run and compare the individual encodings in terms of runtime and solution quality. Based on those results, we perform extensive test cases on different benchmark instances and show how the combination of different encodings and operations can be beneficial and provide a balance between solution quality and runtime.

Keywords: genetic encoding, vehicle routing problem

1 Introduction

The vehicle routing problem (VRP) is a class of problems that frequently occurs in the field of transportation logistics. The original formulation of the problem has been defined in the late 1950ies and consists of a fleet of vehicles serving a set of customers with a certain demand from a single depot. Since then, many diverse variants of vehicle routing problems have been studied in the literature (for a taxonomic review see for example [6]) and many different solution strategies have been followed. Metaheuristics are frequently used to generate feasible and near-optimal solutions for large scale instances. For an overview of different metaheuristics for the VRP see for example [5] or [4].

Among other techniques, such as tabu search (TS) or variable neighborhood search (VNS), genetic algorithms (GA) have been used successfully to tackle large problem instances. However, unlike for other problems such as the traveling salesman problem (TSP), no standard encodings for VRPs have been established yet. In [2] several interesting encodings are identified which will be examined in this work. These different encodings, and combinations of them, are tested on several benchmark instances and compared to each other in terms of runtime and in terms of solution quality.

2 Genetic Encodings

As stated earlier, several interesting genetic encodings identified by [2] are examined in this work. All these encodings have been used in conjunction with different GA variants, to solve VRP benchmark instances by the corresponding authors and are outlined briefly in the following.

The operations described by Potvin [8] operate directly on the tours, are specific to the VRP and are tested using a standard GA and capacitated vehicle routing problem instances with time windows (CVRPTW).

Similar to that, the generic vehicle routing (GVR) concept proposed by Pereira [7] also encodes the tours directly and provides specific crossover and mutation operations, however in addition to that uses a repair function to avoid overload on the tours. Several capacitated problem instances (CVRP) are solved using a GA.

The encoding proposed by Prins [9] is based on a permutation encoding without trip delimiters in conjunction with standard permutation operations. It is based on a route-first, cluster second approach. The individual tours are determined using a specific split-procedure. Several CVRP instances are solved using a hybrid GA which is combined with local search mutation operations.

A permutation encoding without trip delimiters is also used by Zhu [13]. Specific crossover operations are used in conjunction with standard permutation operations. Multiple CVRPTW instances are solved by using a hybrid GA variant which is combined with hill climbing.

In contrast to that, Alba [3] uses a permutation encoding with trip delimiters and thus encodes the individual tours directly. Standard permutation operations are applied and several capacitated problem instances (CVRP) are solved using a cellular GA.

For all those encodings, the authors propose different operations suitable for the respective representation.

3 Analysis

We have implemented the different operations for the individual encodings in the HeuristicLab optimization environment [12] (`http://dev.heuristiclab.com`). Based on this implementation, we have compared the individual operations in terms of the solution quality of the produced offspring and also in terms of runtime. The detailed description of the individual operations can be found in the referenced literature listed in Section 2

3.1 Success Analysis

To analyze the success of the individual operations during an algorithm run, the offspring selection algorithm (OSGA) as proposed by [1] is used. The OSGA introduces a new generic scheme after reproduction which ensures that relevant genetic information is not lost during the search process e.g. due to bad operation design. This selection step discards children that are not better than their parents and only preserves the successful offspring.

This ratio between the count of the unsuccessful and the successful offspring has been analyzed for each operation by performing multiple independent test runs on different problem instances, namely the Taillard385, Taillard150a, Solomon R211 and Solomon RC208 instances as proposed by Taillard [11] and Solomon [10]. As parameter settings of the OSGA, a population size of 1,000 was used, with 10,000,000 maximum evaluated solutions, proportional selection and a success ration and comparison factor of 1.

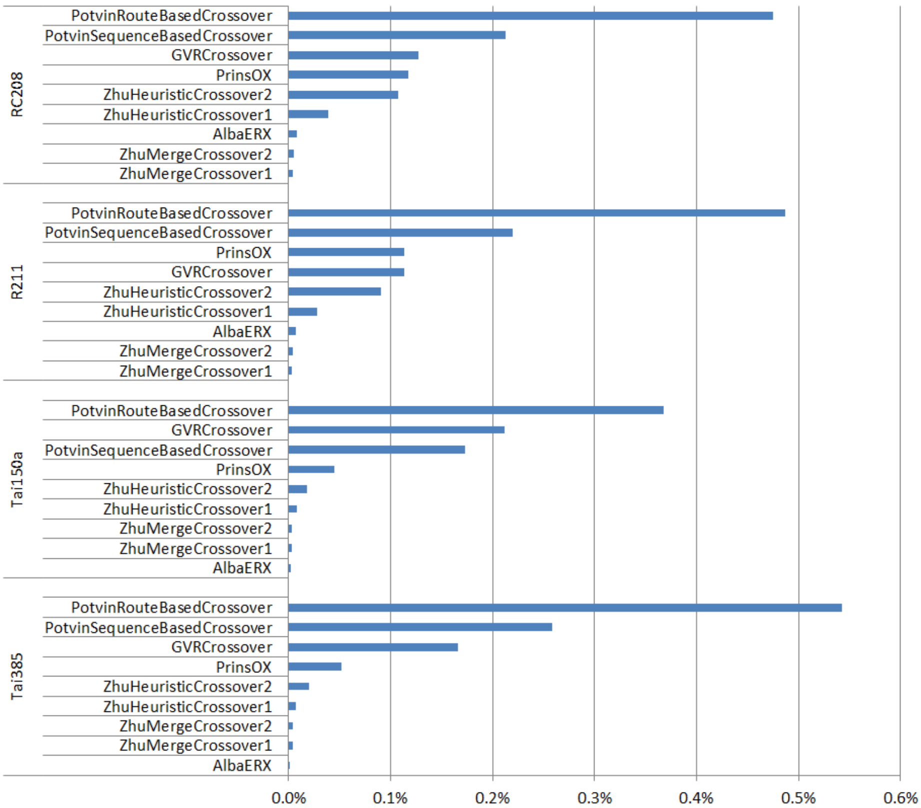

Fig. 1. Relative success of crossover operations

Figure 1 illustrates the relative operation success of the different crossover operations, Figure 2 the success of the examined mutation operations. The prefix of the operation denotes the respective encoding (Alba, GVR, Potvin, Prins and Zhu). All success ratios are normalized according to the success of the best operation.

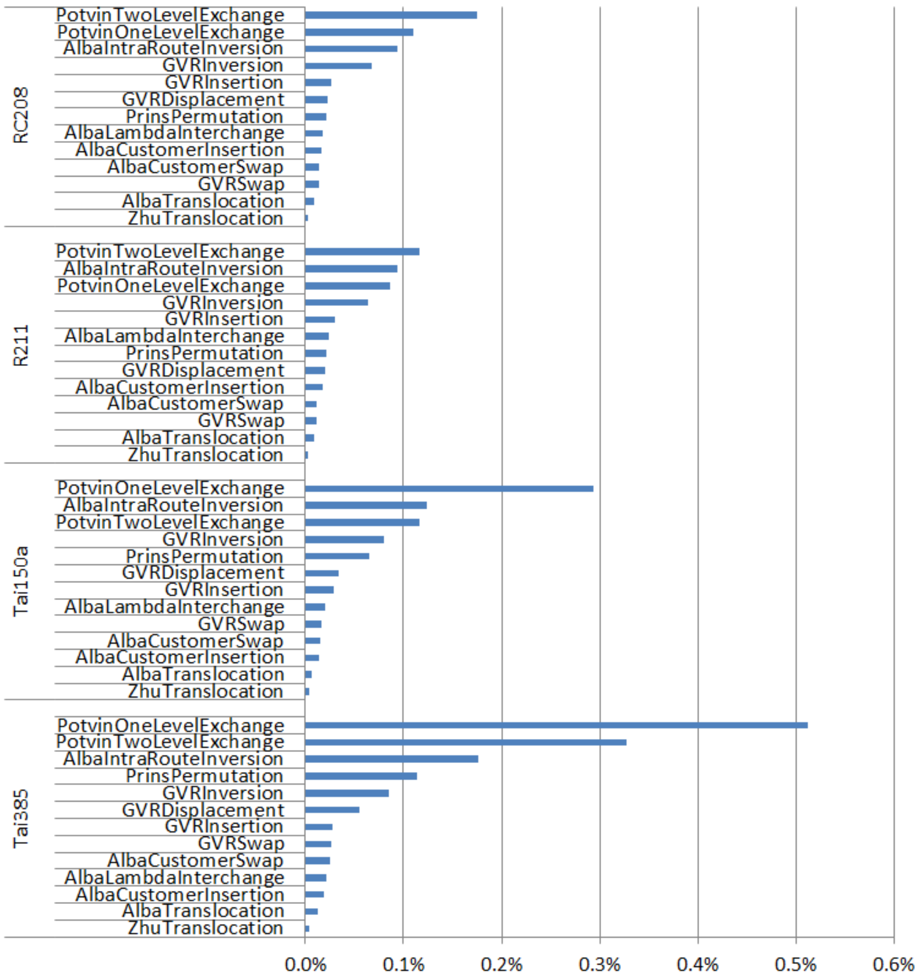

Fig. 2. Relative success of mutation operations

3.2 Runtime Analysis

When comparing the performance of the individual operations, it is also interesting to examine their runtime. To examine the runtime of the operations, a population of 1,000 individuals has been initialized randomly for the Taillard385 instance and each of the operations has been applied to this population. If an encoding includes a repair function, it is included in the runtime analysis as well.

Figure 3 compares the runtime of the implemented crossover operations, and Figure 4 illustrates the runtime of the individual mutation operations. All values have been normalized according to the runtime of the most time-consuming operation.

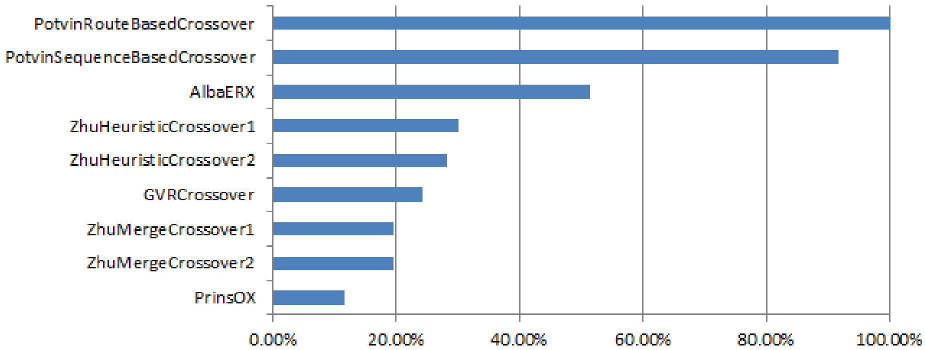

Fig. 3. Runtime of crossover operations

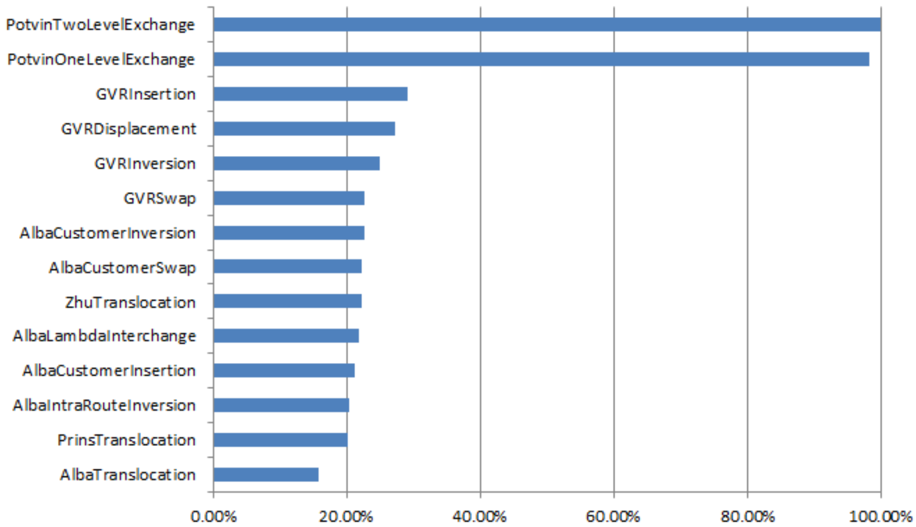

Fig. 4. Runtime of mutation operations

Basically it can be stated, that operations designed specifically for the VRP (such as the operations proposed by Potvin) outperform the other operations, however they are also quite runtime consuming. Other operations that are based on existing default representations (such as the operations that operate on permutations) and are not designed specifically for the VRP are quite runtime efficient and can be implemented easily in existing frameworks, however they are not able to produce comparably good results.

4 Results

The results of the operation analysis show that some operations are able to produce good offspring throughout the search process, however are very runtime consuming. Other operations are quite efficient in terms of runtime, however they produce less successful offspring. Thus, by combining different encodings and operations in one algorithm run, one can achieve a balance between solution quality and runtime.

Different genotypes can be combined in one single algorithm run, by providing a conversion functionality for each encoding. If an operation is executed on an individual which is represented in a different encoding, the routes are extracted from that individual and then converted to the respective encoding. In other words, each individual can be converted from its genotype to its phenotype and then be converted back to another genotype.

To achieve a balance between runtime and solution quality, the individual operations are combined and executed with a probability proportional to their success. This probability is based on the results of the operation success analysis presented in Section 3.

To test our approach, we compared the combination of different encodings to an algorithm only using the best encoding in terms of solution quality identified in our analysis, namely the encoding proposed by Potvin. We executed 10 independent test runs on the extended Solomon benchmark instances with 200 customers[1] using an OSGA with a population size of 100, a comparison factor and success ratio of 1, a mutation probability of 10% and a maximum of 3,000,000 evaluated solutions.

The results are listed in Table 1 and indicate, that using a combination of different encodings, a speedup of 1.52 can be achieved while maintaining a comparable solution quality (0.2% worse).

5 Conclusion

Concluding, we have examined different interesting encodings and operations for the VRP that have been previously proposed in the literature in terms of solution quality and runtime. Some encodings are designed specifically for the VRP and apply complex heuristics in their operations or apply specific repair functions and can thus constantly produce successful offspring. Other operations are based on standard representations (such as the permutation representation), apply generic problem-independent operations and can thus be integrated easily into existing frameworks and require less runtime.

To balance runtime and solution quality, we combined different encodings in one single algorithm run and tested our approach on several large-scale benchmark instances. The execution probability of the individual operations is based the relative success identified in a success analysis we performed in this work.

[1] http://www.fernuni-hagen.de/WINF/touren/inhalte/probinst.htm

Table 1. Test runs

	Potvin Encoding		**Multiple Encodings**	
Instance	Distance	Time	Distance	Time
C121	2708.619832	00:26:32.67	2706.730453	00:14:45.47
C122	2703.480206	00:53:19.64	2701.937712	00:33:51.37
C123	2715.62376	01:21:18.83	2699.171919	00:59:48.40
C124	2709.707198	02:42:26.62	2694.025678	01:36:21.10
C125	2725.407901	00:39:11.17	2720.501962	00:18:34.38
C126	2721.47175	00:41:08.02	2708.945979	00:29:53.61
C127	2723.28039	01:03:03.47	2744.924038	00:34:17.87
C128	2714.775288	01:13:48.18	2729.621202	00:43:24.98
C129	2713.073696	01:44:00.13	2726.745322	01:00:58.27
C1210	2706.428582	01:50:40.06	2718.962129	01:15:44.51
R121	4812.901954	01:02:08.40	4820.493169	00:41:26.62
R122	4078.221774	01:34:44.27	4113.581515	01:23:50.64
R123	3488.101078	01:42:58.26	3547.343408	01:05:30.06
R124	3221.410561	04:40:23.62	3231.882271	02:51:04.99
R125	4242.133204	01:26:28.67	4235.020105	01:02:48.69
R126	3761.25046	01:55:54.58	3716.745634	01:41:09.48
R127	3319.105583	02:29:55.44	3323.783617	01:42:19.46
R128	3116.862251	05:15:31.78	3159.575218	02:14:40.41
R129	3943.073464	01:46:09.28	3971.118208	01:00:52.79
R1210	3517.606513	02:11:44.99	3521.048604	01:36:19.83
RC121	3618.423861	01:08:24.96	3624.986628	00:52:22.82
RC122	3354.955299	01:34:52.82	3359.678325	01:05:42.69
RC123	3157.009295	02:15:44.18	3163.744872	01:33:08.76
RC124	2997.736948	02:54:10.46	3011.080543	02:10:37.18
RC125	3505.777025	01:21:10.33	3504.913716	00:59:06.77
RC126	3496.010864	01:06:03.08	3495.618001	00:52:02.60
RC127	3367.516003	01:40:11.36	3338.838531	01:08:15.40
RC128	3254.704917	01:47:19.10	3258.787123	01:21:13.44
RC129	3222.971162	01:46:15.59	3258.071847	01:06:18.05
RC1210	3129.302473	02:06:15.80	3154.434277	01:41:13.88
Average	**3258.231443**	**01:48:43.86**	**3265.4104**	**01:11:55.48**

In the future, additional research could be performed in the direction of self-steering parameter tuning. This means, that the execution probability is adapted dynamically based on the operator success in the current stage of the search process and not set statically as in our approach. Additionally, it would be interesting to examine the different encodings and operations using fitness landscape analysis techniques.

Acknowledgments The work described in this paper was done within the Regio 13 program sponsored by the European Regional Development Fund and by Upper Austrian public funds.

References

1. Affenzeller, M., Wagner, S.: Offspring selection: A new self-adaptive selection scheme for genetic algorithms. In: Ribeiro, B., Albrecht, R.F., Dobnikar, A., Pearson, D.W., Steele, N.C. (eds.) Adaptive and Natural Computing Algorithms. pp. 218–221. Springer Computer Series, Springer (2005)
2. Affenzeller, M., Winkler, S., Wagner, S., Beham, A.: Genetic Algorithms and Genetic Programming: Modern Concepts and Practical Applications (Numerical Insights). Chapman & Hall, 1 edn. (April 2009)
3. Alba, E., Dorronsoro, B.: Solving the vehicle routing problem by using cellular genetic algorithms. In: Evolutionary Computation in Combinatorial Optimization, pp. 11–20. Lecture Notes in Computer Science (2004)
4. Bräysy, O., Gendreau, M.: Vehicle routing problem with time windows part ii: Metaheuristics. Transportation Science 39, 119–139 (2005)
5. Cordeau, J.F., Gendreau, M., Hertz, A., Laporte, G., Sormany, J.S.: New heuristics for the vehicle routing problem. In: Logistics Systems: Design and Optimization, chap. 9, pp. 279–297. New York (2005)
6. Eksioglu, B., Vural, A.V., Reisman, A.: The vehicle routing problem: A taxonomic review. Computers & Industrial Engineering 57(4), 1472 – 1483 (2009)
7. Pereira, F., Tavares, J., Machado, P., Costa, E.: Gvr: A new genetic representation for the vehicle routing problem. In: Artificial Intelligence and Cognitive Science, pp. 285–320. Lecture Notes in Computer Science (2002)
8. Potvin, J.Y., Bengio, S.: The vehicle routing problem with time windows -. part ii: Genetic search. INFORMS Journal on Computing 8, 165–172 (1996)
9. Prins, C.: A simple and effective evolutionary algorithm for the vehicle routing problem. Computers & Operations Research 31(12), 1985–2002 (October 2004)
10. Solomon, M.: Algorithms for the Vehicle Routing and Scheduling Problem with Time Window Constraints. Operations Research 35(2), 254–265 (1987)
11. Taillard, E.D.: Benchmarks for basic scheduling problems. European Journal of Operational Research 64, 278–285 (1993)
12. Wagner, S.: Heuristic Optimization Software Systems - Modeling of Heuristic Optimization Algorithms in the HeuristicLab Software Environment. Ph.D. thesis, Johannes Kepler University, Linz, Austria (2009)
13. Zhu, K.Q.: A new genetic algorithm for vrptw. In: Proceedings of the International Conference on Artificial Intelligence. p. 311264 (2000)