INTERNATIONAL TRANSACTIONS IN OPERATIONAL RESEARCH

WILEY

INTERNATIONAL TRANSACTIONS IN OPERATIONAL RESEARCH

Intl. Trans. in Op. Res. 00 (2017) 1–10
DOI: 10.1111/itor.12443

A critical analysis of the "improved Clarke and Wright savings algorithm"

Kenneth Sörensen, Florian Arnold and Daniel Palhazi Cuervo

ANT/OR – Operations Research Group, Department of Engineering Management, University of Antwerp, Belgium
E-mail: kenneth.sorensen@uantwerpen.be [Sörensen]; florian.arnold@uantwerpen.be [Arnold]; daniel.palhazicuervo@uantwerpen.be [Palhazi Cuervo]

Received 16 February 2017; accepted 23 June 2017

Abstract

In their paper "An improved Clarke and Wright savings algorithm for the capacitated vehicle routing problem," published in *ScienceAsia* (38, 3, 307–318, 2012), Pichpibul and Kawtummachai developed a simple stochastic extension of the well-known Clarke and Wright savings heuristic for the capacitated vehicle routing problem. Notwithstanding the simplicity of the heuristic, which they call the "improved Clarke and Wright savings algorithm" (ICW), the reported results are among the best heuristics ever developed for this problem. Through a careful reimplementation, we demonstrate that the results published in the paper could not have been produced by the ICW heuristic. Studying the reasons how this paper could have passed the peer review process to be published in an ISI-ranked journal, we have to conclude that the necessary conditions for a thorough examination of a typical paper in the field of optimization are generally lacking. We investigate how this can be improved and come to the conclusion that disclosing source code to reviewers should become a prerequisite for publication.

Keywords: optimization; combinatorial optimization; heuristics

1. Introduction

The capacitated vehicle routing problem (CVRP) is defined on a complete, undirected graph with $n + 1$ nodes, one node representing the depot and the n remaining nodes a set of customers with known demand. The cost of traveling between any pair of customers, or between any customer and the depot, is also given in a distance matrix, as is the (uniform) capacity of a fleet of vehicles. The objective of the CVRP is to define a set of routes with minimal total cost such that each vehicle performs at most one route, the total demand in each route does not exceed the vehicle capacity, and all customers are visited exactly once.

The CVRP is among the most studied problems in the field of Operations Research, and a very large number of algorithms have been developed to solve it (see, e.g., Toth and Vigo, 2014, for

an overview). Exact algorithms—that guarantee to find the optimal solution—can reliably solve instances with up to 200 customers in a reasonable amount of computing time (Poggi and Uchoa, 2014). For larger instances, a large number of heuristics are available.

The Clarke and Wright savings algorithm (Clarke and Wright, 1964) for the CVRP is one of the most widely used heuristics for this problem, and arguably one of the best-known heuristics in the entire field of Operations Research. The simplicity of the algorithm, its intuitive appeal, and the quality of the solutions it produces contribute to the algorithm's widespread acceptance in the research community. The algorithm has been used as the basis for several variants and several modifications have been proposed (e.g., Altınel and Öncan, 2005; Doyuran and Çatay, 2011).

The Clarke and Wright heuristic starts from a solution in which each of the n customers is visited in a separate tour. The cost of this solution is equal to twice the sum of the travel costs between the depot and all customers. For each customer pair, the algorithm then determines the *saving* that would result from connecting these customers directly. The algorithm then creates a *savings list* by sorting these $n(n-1)/2$ savings in decreasing order.

Two versions of the Clarke and Wright algorithm exist: a sequential version in which one route at a time is built and a parallel version in which all routes are built simultaneously. The parallel Clarke and Wright (PCW) heuristic is more common.

The PCW iteratively merges two routes by connecting a pair of customers. For two customers to be connectable, they must be in different routes, they both need to be connected directly to the depot, and the sum of the total demand of the two routes that contain them must not be larger than the vehicle capacity. Each time a pair of customers is connected, the cost of the solution is decreased with the saving incurred by this customer pair. The algorithm proceeds down the savings list until no more customer pairs can be connected.

The Clarke and Wright algorithm is a purely greedy algorithm: at each iteration it selects the route merger that yields the largest saving. The deterministic nature of the Clarke and Wright algorithm results in the algorithm producing the same solution every time it is run on the same instance. A straightforward extension of such a greedy algorithm is to add a controlled randomization in the greedy selection rule, to allow the algorithm to generate a different solution at each iteration. This idea has been popularized in the GRASP metaheuristic (greedy randomized adaptive search procedure; Feo and Resende 1995), but is in fact older than that (Hart and Shogan, 1987). In most cases, the randomized heuristic is executed several times and the best solution is reported. Solutions constructed in this way may also be used as a starting point for a local search heuristic.

In Pichpibul and Kawtummachai (2012) (which we refer to as "P&K" henceforth), the authors develop a straightforward randomized extension of the Clarke and Wright heuristic, which they call the improved Clarke and Wright heuristic, or ICW. The reported results of this heuristic are remarkable and put the heuristic among the best-performing heuristics for the CVRP. In this paper, we demonstrate that the results reported in P&K could not have been produced by the ICW heuristic. An independent reimplementation of the ICW yields results that are worse for almost all instances (and never better), even when a generous 100-fold extension of the number of iterations is allowed.

The question therefore poses itself how this paper could have passed the peer review process, and forces us to conclude that the field of Operations Research urgently needs to improve its peer review standards to avoid publication of results that cannot be replicated.

The reviewing process of new algorithmic or computational contributions has been subject to strong criticism, and the field of Operations Research is no exception. Ignizio (1971) was one of the

first authors to single out the lack of guidelines for comparing different algorithms. Already in the early 1970s, he suggested that a more scientific comparison would have the following effects: "(a) upgrade the status of our profession, (b) eliminate many marginal publications of dubious value, and (c) provide a means by which algorithms may be selected more objectively by the practitioner." Almost 10 years later, Crowder et al. (1979) raised the issue about the reproducibility of the results reported in scientific papers. They suggested that "authors should be willing to reproduce their experiments for the referees," and, "when necessary, referees should exercise this right." Since then, many researchers have advocated higher standards for publication in the field (see, e.g., the papers by Jackson et al., 1990; Hooker, 1995; Kendall et al., 2015; Sörensen, 2015). Worried about the replicability of scientific results, Ince et al. (2012) argued that "anything less than the release of source programs is intolerable for results that depend on computation."

The remainder of this paper is organized as follows. In Section 2, we discuss in detail the functioning of the ICW. In Section 3, the results reported in P&K are compared to the results of our reimplementation of the ICW. Section 4 makes some recommendations on how the field of heuristic optimization could improve its scientific standards and procedures.

2. The improved Clarke and Wright savings algorithm of Pichpibul and Kawtummachai

The PCW, when implemented efficiently, can compute solutions for typical CVRP instances in fractions of a second on a standard PC. These extremely short computing times leave room for extensions in which the heuristic is partially randomized so as to produce a different solution each run. This allows the construction process to be iterated many times, after which the best solution found during all iterations is reported. As mentioned before, the ICW heuristic described in P&K follows this simple principle.

The ICW uses the same savings list as the PCW as its first *current savings list*. Before each run, the algorithm randomizes the current savings list using a process the authors call "a combination of tournament and roulette wheel selection." The *randomized savings list* is formed by iteratively choosing the next customer pair using tournament selection (i.e., proportionally to the savings of this customer pair) among the first T elements in the savings list. Without motivating this choice, the ICW randomly chooses T between 3 and 9 at each iteration. When a customer pair is chosen, it is removed from the current savings list and added to the randomized savings list. When all customer pairs have been selected, a new solution is constructed by the PCW using this randomized savings list.

If the solution found using the randomized savings list is both valid and better than the best solution found so far, the randomized savings list replaces the current savings list. A solution is defined as valid if the number of routes does not exceed the number of available vehicles. This process of randomizing the savings list and constructing a new solution using PCW is repeated 10,000 times, after which the best solution found is reported.

For a more elaborate description, including pseudocode, of the ICW, we refer to Pichpibul and Kawtummachai (2012). We reimplemented the ICW algorithm based on the description in P&K. To validate the correctness of the implementation and the results obtained, the ICW algorithm was independently implemented twice by two of the authors, with two different programming languages

and tested on two different computers. The full source code of our reimplementations of the ICW is available online (http://antor.uantwerpen.be/ICW).

3. Results

The reimplemented ICW was tested on the same 84 problem instances reported in P&K. Among those, 70 instances are from Augerat (denoted A, B, and P), 11 are from Christofides and Eilon (E), and 3 are from Fisher (F). We remark that P&K do not report results on all instances from those authors. Results on B-n51-k7, B-n57-k7, B-n63-k10, E-n30-k4, E-n76-k15, P-n55-k15, for example, are missing. No reason is reported to explain these missing results. All the instances are provided by the NEO Research Group (http://neo.lcc.uma.es/vrp/vrp-instances/capacitated-vrpinstances/).

The reimplemented ICW was executed 100 times for each instance. We report the arithmetic mean and median of the results as well as the best result over those 100 runs. These metrics are necessary to measure the performance of ICW accurately, since, due to the stochastic components in the algorithm, the results can vary between different executions. P&K acknowledge this, but still only report the result of a single run. Detailed results for all instances are listed in Tables 1–4. An asterisk in the column "P&K" indicates that the best-known result for the instance was found by the ICW of P&K. Values highlighted in gray indicate that the obtained results with our reimplemented ICW differ from those reported in P&K.

To ensure the validity of the results presented in this paper, the ICW was implemented independently by two authors of this paper. In Tables 1–4, we show only the results corresponding to one of the re-implementations of the ICW algorithm. This is because both implementations have a similar behavior and produce virtually the same results. This was corroborated by means of sign tests and Wilcoxon signed-rank tests (Gibbons and Chakraborti, 2011) carried out with the outcome of both source codes. These nonparametric statistical tests are used to determine if the average performance of two algorithms, over a set of instances, is different (in other words, to assess if the mean ranks and medians of two related samples are different). They have been widely used in the artificial intelligence community since the recommendation made by Demšar (2006). The results of all tests performed showed that there is no (statistically significant) evidence to believe that the implementations should be considered different. More specifically, all tests failed to reject the null hypotheses with p-values larger than 0.10.

Since we run the reimplemented ICW 100 times per instance, in comparison to only a single run in P&K, we would expect to find a similar or even better solution at least once. On the contrary, the reimplemented ICW is able to reproduce the results in P&K for only 25 of 84 instances, while for 59 instances a worse solution is obtained in each of the 100 repetitions. The 25 instances for which the best result out of 100 runs of the reimplemented ICW matches the (single) result of the original ICW are all relatively small instances, that is, with 50 or less customers. In total, the average gap over all instances between our best found solution and the reported results amounts to 0.68%, and is even higher if we only consider instances of large size. To summarize, our reimplementation of the ICW *consistently finds worse solutions than those reported in P&K, even when allowed to run 100 times longer.*

Unsurprisingly, comparing the average result out of our 100 runs rather than the best result to the result in P&K reveals an even larger discrepancy. In this case, we can replicate none of the reported

Table 1
Results on the Augerat "A" instances. Values highlighted in grey indicate that the results of the reimplemented ICW differ from those reported in P&K

Instance	Best known	P&K	S&A&P (100 runs)					
			Average	Gap	Median	Gap	Best	Gap
A-n32-k5	784	784*	792.9	1.14	789.5	0.70	784	0.00
A-n33-k5	661	661*	670.3	1.41	672	1.66	661	0.00
A-n33-k6	742	742*	744.7	0.37	744	0.27	742	0.00
A-n34-k5	778	778*	793.9	2.04	793	1.93	778	0.00
A-n36-k5	799	799*	805.8	0.84	805	0.75	799	0.00
A-n37-k5	669	669*	687.6	2.77	691	3.29	669	0.00
A-n37-k6	949	949*	955.4	0.67	955	0.63	949	0.00
A-n38-k5	730	730*	753.3	3.19	753	3.15	732	0.27
A-n39-k5	822	822*	835.9	1.69	838	1.95	823	0.12
A-n39-k6	831	831*	836.2	0.63	835	0.48	831	0.00
A-n44-k7	937	937*	961.4	2.61	962	2.67	946	0.96
A-n45-k6	944	944*	978.3	3.63	980	3.81	952	0.85
A-n45-k7	1146	1146*	1158.7	1.11	1159	1.13	1148	0.17
A-n46-k7	914	914*	924.6	1.16	922	0.88	914	0.00
A-n48-k7	1073	1073*	1099.5	2.47	1099	2.42	1096	2.14
A-n53-k7	1010	1010*	1040.0	2.97	1036	2.57	1026	1.58
A-n54-k7	1167	1167*	1177.7	0.92	1175.5	0.73	1168	0.09
A-n55-k9	1073	1073*	1085.0	1.12	1084	1.03	1076	0.28
A-n60-k9	1354	1354*	1367.0	0.96	1366	0.89	1360	0.44
A-n61-k9	1034	1034*	1101.7	6.55	1100	6.38	1056	2.13
A-n62-k8	1288	1298	1319.0	1.62	1318	1.54	1303	0.39
A-n63-k9	1616	1616*	1666.2	3.10	1665	3.03	1631	0.93
A-n63-k10	1314	1314*	1325.6	0.88	1324	0.76	1318	0.30
A-n64-k9	1401	1415	1442.4	1.93	1442	1.91	1423	0.57
A-n65-k9	1174	1174*	1204.2	2.57	1206	2.73	1184	0.85
A-n69-k9	1159	1159*	1176.9	1.54	1177	1.55	1167	0.69
A-n80-k10	1763	1772	1802.2	1.70	1801	1.64	1782	0.56

results. Moreover, the average gap over all instances between the results of our reimplementation and the reported results of the original ICW increases to 2.4%. To validate that these results are not subject to outliers, we also compute the median and obtain a similar gap of 2.2% on average over all instances. For instances like A-n48-k7, A-n61-k9, P-n76-k5, and F-n72-k4, the gaps between our best results (found after 100 runs) and the reported results are larger than 2%. The detailed results of the 100 runs for A-n48-k7 and A-n61-k9 are presented in Fig. 1. Gaps of this size cannot be explained by stochastic effects or differences in implementation, but clearly demonstrate that both results stem from two completely different algorithms. Indeed, a paired rank test highlights that the differences between the reported results in P&K and both our average and best results over 100 repetitions are highly significant (with p-values smaller than 0.01).

Additionally, for the tightly constrained instances E-n30-k3 and P-n50-k8, our reimplementation was not able to generate any feasible solution in 35 and 72 of 100 runs, respectively. While P&K report the optimal solution to those instances, the best solutions we find have a gap of 6.93% and 4.75%, respectively.

Table 2
Results on the Augerat "B" and Christofides and Eilon instances. Values highlighted in grey indicate that the results of the reimplemented ICW differ from those reported in P&K

Instance	Best known	P&K	S&A&P (100 runs)					
			Average	Gap	Median	Gap	Best	Gap
B-n31-k5	672	672*	672.7	0.10	672	0.00	672	0.00
B-n34-k5	788	788*	791.0	0.39	792	0.51	788	0.00
B-n35-k5	955	955*	966.4	1.20	965	1.05	958	0.31
B-n38-k6	805	805*	822.4	2.17	823	2.24	808	0.37
B-n39-k5	549	549*	552.8	0.70	553	0.73	550	0.18
B-n41-k6	829	829*	852.6	2.85	850.5	2.59	835	0.72
B-n43-k6	742	742*	750.6	1.16	751	1.21	742	0.00
B-n44-k7	909	909*	925.6	1.82	928	2.09	909	0.00
B-n45-k5	751	751*	751.7	0.09	752	0.13	751	0.00
B-n45-k6	678	678*	712.9	5.14	712	5.01	690	1.77
B-n50-k7	741	741*	744.1	0.41	744	0.40	741	0.00
B-n50-k8	1312	1312*	1334.8	1.74	1334	1.68	1321	0.69
B-n52-k7	747	751	754.2	0.42	754	0.40	752	0.13
B-n56-k7	707	707*	719.2	1.73	719.5	1.77	714	0.99
B-n57-k9	1598	1598*	1604.8	0.42	1604	0.38	1599	0.06
B-n64-k9	861	861*	908.9	5.56	910.5	5.75	870	1.05
B-n66-k9	1316	1320	1338.7	1.42	1340.5	1.55	1324	0.30
B-n67-k10	1032	1032*	1072.9	3.96	1069.5	3.63	1044	1.16
B-n68-k9	1272	1281	1300.3	1.50	1301	1.56	1284	0.23
B-n78-k10	1221	1238	1251.5	1.09	1252	1.13	1246	0.65
E-n22-k4	375	375*	377.6	0.69	375	0.00	375	0.00
E-n23-k3	569	569*	569.9	0.15	569	0.00	569	0.00
E-n30-k3	534	534*	–	–	–	–	571	6.93
E-n33-k4	835	835*	840.9	0.70	841	0.72	839	0.48
E-n51-k5	521	521*	548.0	5.18	551.5	5.85	526	0.96
E-n76-k7	682	686	700.5	2.11	699	1.90	692	0.87
E-n76-k8	735	742	764.9	3.08	764	2.96	747	0.67
E-n76-k10	830	839	869.7	3.66	869.5	3.64	845	0.72
E-n76-k14	1021	1027	1067.2	3.92	1067	3.89	1033	0.58
E-n101-k8	817	821	845.4	2.97	847	3.17	824	0.37
E-n101-k14	1071	1084	1105.6	2.00	1107	2.12	1094	0.92

On the basis of these observations, we have to conclude that the results reported in P&K cannot have been obtained using only the ICW algorithm. We also tried several small variations on the ICW, changing slightly the way in which the randomized savings list is constructed, in which the randomized savings list replaces the current savings list, running our algorithm for more than 100 runs, etc. Needless to say that none of these variations yielded a significant improvement over our original implementation of the ICW.

4. Discussion

We do not wish to speculate on whether the errors in the results reported in P&K are the result of an unintentional mistake, or whether they were intentionally "polished" to window dress an otherwise

Table 3
Results on the Augerat "P" instances. Values highlighted in grey indicate that the results of the reimplemented ICW differ from those reported in P&K

Instance	Best known	P&K	S&A&P (100 runs)					
			Average	Gap	Median	Gap	Best	Gap
P-n16-k8	450	450*	452.3	0.50	450	0.00	450	0.00
P-n19-k2	212	212*	223.9	5.62	223	5.19	212	0.00
P-n20-k2	216	216*	219.6	1.67	219	1.39	216	0.00
P-n21-k2	211	211*	214.5	1.67	215	1.90	211	0.00
P-n22-k2	216	216*	219.3	1.50	219	1.39	216	0.00
P-n22-k8	603	603*	619.8	2.79	623	3.32	608	0.83
P-n23-k8	529	529*	537.0	1.52	534.5	1.04	529	0.00
P-n40-k5	458	458*	468.8	2.36	467.5	2.07	459	0.22
P-n45-k5	510	510*	521.6	2.28	521	2.16	511	0.20
P-n50-k7	554	554*	564.0	1.80	563	1.62	554	0.00
P-n50-k8	631	631*	–	–	–	–	661	4.75
P-n50-k10	696	696*	718.5	3.24	718	3.16	701	0.72
P-n51-k10	741	741*	762.3	2.87	760	2.56	744	0.40
P-n55-k7	568	568*	582.2	2.50	582	2.46	575	1.23
P-n55-k8	576	576*	590.9	2.58	589	2.26	576	0.00
P-n55-k10	694	697	705.9	1.27	705	1.15	700	0.43
P-n60-k10	744	744*	768.8	3.33	770	3.49	751	0.94
P-n60-k15	968	968*	991.5	2.43	992	2.48	969	0.10
P-n65-k10	792	792*	812.1	2.54	809	2.15	805	1.64
P-n70-k10	827	827*	857.0	3.62	857	3.63	830	0.36
P-n76-k4	593	597	625.1	4.71	624	4.52	607	1.68
P-n76-k5	627	627*	654.9	4.44	653	4.15	640	2.07
P-n101-k4	681	684	704.4	2.98	704	2.92	689	0.73

Table 4
Results for the Fisher "F" instances. Values highlighted in grey indicate that the results of the reimplemented ICW differ from those reported in P&K

Instance	Best known	P&K	S&A&P (100 runs)					
			Average	Gap	Median	Gap	Best	Gap
F-n45-k4	721	721*	729.4	1.17	729	1.11	728	0.97
F-n72-k4	237	237*	256.4	8.19	257	8.44	246	3.80
F-n135-k7	1162	1162*	1197.2	3.03	1198	3.10	1178	1.38

unexceptional heuristic, for example, to improve the probability of its publication. Instead, we focus on the (arguably more important) question of how this paper could have passed the peer review process.

Scientific research critically relies on the peer review process to catch all—or at the very least most—errors and guarantee the quality of the research that is published. Reviewers and editors have a critical role as gatekeepers and should ensure that they stop low-quality or fraudulent research from being published. From the previous analysis, it is clear that P&K's paper should not have been accepted for publication.

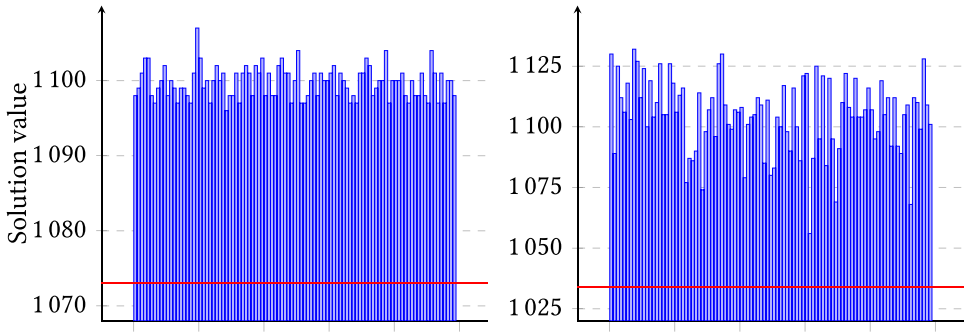

Fig. 1. Results of 100 different runs of our reimplementation of ICW on instances A-n48-k7 (left) and A-n61-k9 (right). In none of the runs are we able to reproduce the results in P&K, indicated by the red line. [Colour figure can be viewed at wileyonlinelibrary.com]

The results presented in the paper should have triggered an alarm in any researcher familiar with state-of-the-art algorithms for the CVRP: the results presented in P&K are simply too good to be true. Had the results been slightly less good, the ICW might not have stood out from the large number of mediocre algorithms for the CVRP in existence, and probably would not have prompted us to reimplement it.

Additionally, the ICW is very easy—almost trivial—to reimplement. Had the algorithm been more involved and included more complicated operators, then the challenge of reimplementing it would probably have not been worth the effort. Replication studies are typically not accepted for publication in the Operations Research literature, hence the incentive to reimplement an existing algorithm is generally very small.

These two observations—modest exaggerations of the quality of an algorithm are hard to spot, combined with the fact that a reimplementation of an existing algorithm is both difficult and unrewarding—force us to conclude that falsely reporting the performance of an algorithm is extremely hard to catch. If papers can pass the review process containing results that are so obviously exaggerated, then what guarantees are there that the review process will catch results that are exaggerated in a more sophisticated way. It is trivial, for example, to report grossly understated computing times, much smaller than the ones actually used to obtain certain results. For an algorithm that is sufficiently complicated to discourage other authors from replicating it, the risk of being caught is almost zero.

Moreover, not only is the risk of being caught extremely low, the gain is exceedingly high. The worldwide system of publish-or-perish forces researchers to publish as many papers as they can, knowing that the quality of the papers they publish has far less influence on their careers than the quantity.

Yet, the solution is exceedingly simple: editors should enforce that *source code is made available to reviewers*, to allow them to check whether the results presented in a paper correspond to the actual output of an algorithm. Needless to say, the extra work this imposes on the reviewer should be kept to a minimum by the authors, for example, by including detailed compilation instructions, makefiles, or other automated testing tools. Confidentiality problems can be solved by nondisclosure agreements. The fact that the author is not a professional programmer, that the code is not clean or difficult to

read, that it is just "research code" not meant for publication, etc., should not count as an excuse, as the only purpose of the code is to check that the results in the paper are a truthful representation of the output of the code. Some journals even have this requirement (e.g., *Mathematical Programming Computation*), but it should become the norm rather than the exception.

Of course, sharing code with the entire research community would be even better, as this would not only prevent erroneous or fraudulent results from reaching publication, but would benefit the community as a whole. It would help to pave the way for more standardized and openly accessible libraries, for instance the local search operators of Groër et al. (2010). Overall, this could lead to more comparable heuristics and would relieve the community of the burden of programming everything from scratch, time and time again. Therefore, disclosing source code to reviewers only should be seen as a bare minimum.

5. Conclusions

In this paper, we have conclusively demonstrated that the results presented in P&K could not have been generated by the "ICW algorithm." A careful reimplementation of that algorithm has yielded results that are an order of magnitude worse, downgrading the ICW from one of the best algorithms for the CVRP to a mediocre one. We have avoided the discussion on whether this publication is the result of fraud or of carelessness, but rather focused on what the research community could do to stop such obviously wrong results from being published. The solution, we found, is simple: source code should be disclosed to the reviewers.

References

Altınel, İ.K., Öncan, T., 2005. A new enhancement of the Clarke and Wright savings heuristic for the capacitated vehicle routing problem. *Journal of the Operational Research Society* 56, 8, 954–961.

Clarke, G.U., Wright, J.W., 1964. Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research* 12, 4, 568–581.

Crowder, H., Dembo, R.S., Mulvey, J.M., 1979. On reporting computational experiments with mathematical software. *ACM Transactions on Mathematical Software (TOMS)* 5, 2, 193–203.

Demšar, Janez., 2006. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research* 7, 1–30.

Doyuran, T., Çatay, B., 2011. A robust enhancement to the Clarke–Wright savings algorithm. *Journal of the Operational Research Society* 62, 1, 223–231.

Feo, T.A., Resende, M.G.C., 1995. Greedy randomized adaptive search procedures. *Journal of Global Optimization* 6, 2, 109–133.

Gibbons, J.D., Chakraborti, S., 2011. Nonparametric statistical inference. In Lovric, M. (ed.) *International Encyclopedia of Statistical Science*. Springer, Berlin, pp. 977–979.

Groër, C., Golden, B., Wasil, E., 2010. A library of local search heuristics for the vehicle routing problem. *Mathematical Programming Computation* 2, 2, 79–101.

Hart, J.P., Shogan, A.W., 1987. Semi-greedy heuristics: an empirical study. *Operations Research Letters* 6, 3, 107–114.

Hooker, J.N., 1995. Testing Heuristics: we have it all wrong. *Journal of Heuristics* 1, 1, 33–42.

Ignizio, J.P., 1971. On the establishment of standards for comparing algorithm performance. *Interfaces* 2, 1, 8–11.

Ince, D.C., Hatton, L., Graham-Cumming, J., 2012. The case for open computer programs. *Nature* 482, 7386, 485–488.

Jackson, R.H.F., Boggs, P.T., Nash, S.G., Powell, S., 1990. Guidelines for reporting results of computational experiments. Report of the ad hoc committee. *Mathematical Programming* 49, 1, 413–425.

Kendall, G., Bai, R., Błazewicz, J., De Causmaecker, P., Gendreau, M., John, R., Li, J., McCollum, B., Pesch, E., Qu, R., Sabar, N., Vanden Berghe, G., Yee, A., 2015. Good laboratory practice for optimization research. *Journal of the Operational Research Society* 67, 4, 676–689.

Pichpibul, T., Kawtummachai, R., 2012. An improved Clarke and Wright savings algorithm for the capacitated vehicle routing problem. *ScienceAsia* 38, 3, 307–318.

Poggi, M., Uchoa, E. 2014. New exact algorithms for the capacitated vehicle routing problem. In Toth, P., Vigo, D. (eds) *Vehicle Routing: Problems, Methods, and Applications* (2nd edn). SIAM, Philadelphia, PA, pp. 59–86.

Sörensen, K., 2015. Metaheuristics—the metaphor exposed. *International Transactions in Operational Research* 22, 1, 3–18.

Toth, P., Vigo, D., eds, 2014. *Vehicle Routing: Problems, Methods, and Applications* (2nd edn). SIAM, Philadelphia, PA.