J Heuristics (2008) 14: 613–625
DOI 10.1007/s10732-007-9053-z

Experiments concerning sequential versus simultaneous maximization of objective function and distance

Peter Greistorfer · Arne Løkketangen · Stefan Voß · David L. Woodruff

Received: 27 September 2005 / Revised: 8 May 2006 / Accepted: 23 March 2007 /
Published online: 30 October 2007

Abstract Suppose two solution vectors are needed that have good objective function values and are different from each other. The following question has not yet been systematically researched: Should the two vectors be generated sequentially or simultaneously? We provide evidence that for broad ranges of practically achievable distances, sequential generation usually requires less computational effort and produces solutions that are at least as good as produced by simultaneous generation. This is done using experiments based on publicly available instances of the multi-constrained, zero-one knapsack problem, which are corroborated using experiments conducted with the linear assignment problem.

Keywords Distance · Variety · Diversity

P. Greistorfer
Institut für Industrie und Fertigungswirtschaft, Karl-Franzens-University Graz,
Universitätsstraße 15/G2, 8010 Graz, Austria
e-mail: peter.greistorfer@kfunigraz.ac.at

A. Løkketangen
Molde University College, Britvn. 2, 6400 Molde, Norway
e-mail: Arne.Lokketangen@hiMolde.no

S. Voß
Institute of Information Systems, University of Hamburg, Von-Melle-Park 5,
20146 Hamburg, Germany
e-mail: stefan.voss@uni-hamburg.de

D.L. Woodruff (✉)
Graduate School of Management, UC Davis, Davis, CA 95616, USA
e-mail: dlwoodruff@ucdavis.edu

1 Introduction

Suppose two solution vectors are needed that have good objective function values and are different from each other. The following question has not yet been systematically researched: Should the two vectors be generated sequentially or simultaneously?

The developers of population based metaheuristics posit the need for good, mutually distant solutions in a variety of ways. Examples include Genetic Algorithms (Whitley 1989), Memetic Algorithms (Moscato 1989), and Scatter Search (Glover 1995). Recently, more focused efforts have been undertaken to study metrics and their effects on solution pools. Fu et al. (2005) demonstrate the importance of diversity in a genetic algorithm searching for classification trees. Greistorfer and Voß (2005) provide extensive analysis and literature review concerning the effects of diversification and solution quality in population based metaheuristics. Sörensen and Sevaux have recently demonstrated the importance of population management for memetic algorithms (Sörensen and Sevaux 2006).

In the area of Multi-Criteria Optimization (MCO), the need for solution diversity is also well known (Ehrgott and Gandibleux 2002; Steuer 1986; Zeleny 1982). For continuous optimization problems, the set of undominated solutions can be infinite, so MCO researchers have long suggested that dissimilarity functions or metrics are needed to reduce the number of undominated solutions to be dealt with. It is in the same spirit of decision support that Løkketangen and Woodruff (2005) propose a distance for selection problems based on work in the psychology and computer science literature (Ryu and Eick 1998; Tversky 1977).

Consider the canonical form of an optimization problem:

$$
\begin{aligned}
(P) \quad & \max\nolimits_x f(x) \\
& \text{subject to} \\
& x \in X
\end{aligned}
$$

where we take x to be an n-vector with constraints summarized by X. A single, optimum solution to a problem, x^*, carries significant value. However, in many cases the form of the objective function and the constraints are a very rough approximation to the goals of the decision makers and stakeholders. In these situations, decision makers might prefer to see at least two decisions that are dissimilar, yet both good.

Such solutions can be generated sequentially or simultaneously. This paper explores the tradeoffs between these approaches. A straightforward mechanism for sequential generation is to first find x^* by solving (P) and then a second solution, y^*, by solving

$$
\begin{aligned}
(Q) \quad & \max\nolimits_y f(y) + \omega d(x^*, y) \\
& \text{subject to} \\
& y \in X
\end{aligned}
$$

where $d(\cdot)$ gives the dissimilarity between two solutions and ω is a parameter that determines the relative importance of diversity. Simultaneous generation can be accomplished by solving the single problem:

$$
\begin{aligned}
(R)\quad & \max_{a,b} f(a) + \beta f(b) + \gamma d(a,b) \\
& \text{subject to} \\
& a \in X \\
& b \in X
\end{aligned}
$$

where the parameters β and γ control the relative importance of quality and diversity. These procedures can be generalized to seek more than two solutions by using, e.g., the product of pairwise dissimilarity. However, we will focus on the two solution case because we are primarily interested in the basic approaches for generating these solutions.

For the purpose of discussion, we will assume that a^*, b^*, x^*, and y^* are the solutions to the respective problems and that $f(a) \geq f(b)$. One remark is immediate:

Remark 1 $f(x^*) \geq f(a^*)$.

In other words: it is clear that for bounded values of the parameters, the simultaneous generation of vectors will not necessarily generate an optimal solution to (P). This alone might be so troubling to some people that they would never consider simultaneous generation. Others, however, may have slightly less preference for an optimal solution to a precisely stated, mathematical problem. Maybe there is some comfort in the fact that (R) can be parameterized to match (P). To wit:

Remark 2 For small enough β and γ, $a^* = x^*$.

Under some other circumstances, it is also clear that simultaneous generation of the solutions will result in more diversity. For example:

Remark 3 For any fixed values of ω and β and for large enough γ, $d(a^*, b^*) \geq d(x^*, y^*)$.

The proof is simply that a^* could take the value of x^* if that resulted in maximized distance, otherwise a better vector will be used.

These remarks are immediate consequences of the problem statements and are not particularly useful in guiding practice. Other issues must be explored using computational experiments. For experimental consideration of these issues, we use the multi-constrained, zero-one knapsack problem (MCKP) to define the structure of $f(\cdot)$ and X, and the Hamming distances to define $d(\cdot)$. Consequently, solutions can be found using commercially available MIP solvers. Furthermore, many instances of this problem are in the OR-LIB. These results are corroborated using experiments conducted with the linear assignment problem (LAP).

The MCKP is a well-known special case of mixed integer linear programming with a great variety of applications. In the MCKP one has to decide on how to use a knapsack with multiple resource constraints. For a comprehensive survey on the

MCKP see Fréville (2004) and the references given therein. The LAP is a classic special case of the transportation problem, for a review, see Burkard and Çela (1999).

In this paper, we will draw samples from a collection of instances and a variety of parameter settings. A priori, one would expect that for a given distance target, the computational effort associated with sequential generation would be less than the simultaneous because the problems have a rate of growth in the problem size that is generally of higher order than linear. We will verify this experimentally. The other issue is solution quality. As noted in Remark 1, comparing pairs of solutions found by each method by the maximum can never favor simultaneous generation. However, given its ability to balance quality and distance in one optimization the relative quality of the average solution is not clear. We will investigate this issue experimentally.

2 Computational experiments

2.1 Algorithms

In order to have heuristic algorithms that are consistent and reproducible, we made use of the well-known CPLEX solver from ILOG. The important parameters for our purposes form a four element tuple: version number, tolerance, maximum nodes, and maximum time in seconds. We made use of two different tuples, configuration A was (4.0.8, 0.0001, 500000, ∞), configuration B was (9.0, 0.01, ∞, 72000). Experiments using configuration A were done on a Pentium M725 1.6 GHz CPU with 1 GB RAM, while configuration B was done on a Pentium IV, running at 3.2 GHz, with 3.7 GB RAM. In our experiments, heuristic configuration A often terminated due to the node limit. The time limit was occasionally exceeded for configuration B, although the optimality gap was very close to the limit at that point in all such cases. Our interest here, though, is not in optimal solutions but in a consistent and reproducible comparison of two methods of generating diverse solutions. Although we cannot hope to provide evidence that our conclusions hold up over the space of all algorithms, the use of two configurations does provide some assurance that the results do not depend entirely on the heuristic used.

2.2 MCKP experiments

Using the MCKP to define the structure of $f(\cdot)$ and X, and using the Hamming distances to define $d(\cdot)$, yields the following formulations. For the base problem, that we called (P) in the general case, the MCKP is:

$$
\begin{array}{ll}
(\mathcal{P}) & \max_x \sum_{i=1}^n c_i x_i \\
& \text{subject to} \\
& \sum_{i=1}^n W_{ij} x_i \le r_j, \quad j = 1, \ldots, m \\
& x_i \in \{0, 1\}, \quad i = 1, \ldots, n
\end{array}
$$

where the n-vector of "contributions," c, the m by n matrix of "weights," W, and the m-vector of "weight restrictions," r, are given as data. So for the MCKP, $f(x) =$

$\sum_{i=1}^{n} c_i x_i$. With the same input data for the MCKP plus a solution to $(\mathcal{P})$ given as x^*, the general problem (Q) becomes the MCKP version:

$$
\begin{array}{ll}
(\mathcal{Q}) & \max_y \sum_{i=1}^{n} \left(c_i y_i + \omega |y_i - x_i^*|\right) \\
& \text{subject to} \\
& \sum_{i=1}^{n} W_{ij} y_i \leq r_j, \quad j = 1, \ldots, m \\
& y_i \in \{0, 1\}, \quad i = 1, \ldots, n
\end{array}
$$

where by $|y_i - x_i^*|$ we mean the absolute value of the difference in the ith vector elements, which will be binary since the elements of x and y are constrained to be binary. In similar fashion, the general problem (R) becomes:

$$
\begin{array}{ll}
(\mathcal{R}) & \max_{a,b} \sum_{i=1}^{n} (c_i a_i + \beta c_i b_i + \gamma |a_i - b_i|) \\
& \text{subject to} \\
& \sum_{i=1}^{n} W_{ij} a_i \leq r_j, \quad j = 1, \ldots, m \\
& \sum_{i=1}^{n} W_{ij} b_i \leq r_j, \quad j = 1, \ldots, m \\
& a_i \in \{0, 1\}, \quad i = 1, \ldots, n \\
& b_i \in \{0, 1\}, \quad i = 1, \ldots, n
\end{array}
$$

The implementation of the absolute value can be done in a variety of ways. The simplest is to take advantage of the fact that it is the difference of binary values and use the square. That is, for binary values a_i and b_i, $|a_i - b_i| = (a_i - b_i)^2$.

But to use a linear solver, we linearize by adding a distance variable that will indicate the difference. This variable, d, must be bounded from below and above using the following constraints:

$$
\begin{array}{l}
d_i \geq a_i - b_i \\
d_i \geq -a_i + b_i \\
d_i \leq a_i + b_i \\
d_i \leq 2 - a_i - b_i \\
d_i \in \{0, 1\}, \quad i = 1, \ldots, n
\end{array}
$$

Using this approach, problem $(\mathcal{Q})$ is implemented as

$$
\begin{array}{l}
\max_y \sum_{i=1}^{n} (c_i y_i + \omega d_i) \\
\text{subject to} \\
\sum_{i=1}^{n} W_{ij} y_i \leq r_j, \quad j = 1, \ldots, m \\
y_i \in \{0, 1\}, \quad i = 1, \ldots, n \\
d_i \geq x_i^* - y_i, \quad i = 1, \ldots, n \\
d_i \geq -x_i^* + y_i, \quad i = 1, \ldots, n \\
d_i \leq x_i^* + y_i, \quad i = 1, \ldots, n \\
d_i \leq 2 - x_i^* - y_i, \quad i = 1, \ldots, n \\
d_i \in \{0, 1\}, \quad i = 1, \ldots, n
\end{array}
$$

and $(\mathcal{R})$ for $\gamma > 0$ is implemented as

$$
\begin{aligned}
&\max_{a,b} \textstyle\sum_{i=1}^{n} (c_i a_i + \beta c_i b_i + \gamma d_i) \\
&\text{subject to} \\
&\textstyle\sum_{i=1}^{n} W_{ij} a_i \le r_j, \quad j = 1, \ldots, m \\
&\textstyle\sum_{i=1}^{n} W_{ij} b_i \le r_j, \quad j = 1, \ldots, m \\
&a_i \in \{0, 1\}, \quad i = 1, \ldots, n \\
&b_i \in \{0, 1\}, \quad i = 1, \ldots, n \\
&d_i \ge a_i - b_i, \quad i = 1, \ldots, n \\
&d_i \ge -a_i + b_i, \quad i = 1, \ldots, n \\
&d_i \le a_i + b_i, \quad i = 1, \ldots, n \\
&d_i \le 2 - a_i - b_i, \quad i = 1, \ldots, n \\
&d_i \in \{0, 1\}, \quad i = 1, \ldots, n
\end{aligned}
$$

2.3 A small example

To illustrate some of these concepts, consider a particular instance of $(\mathcal{P})$:

$$
\begin{aligned}
&\max\ 80x_1 + 20x_2 + 60x_3 + 50x_4 + 70x_5 + 70x_6 + 50x_7 + 70x_8 + 80x_9 + 20x_{10} \\
&\text{subject to} \\
&43x_1 + 89x_2 + 65x_3 + 42x_4 + 12x_5 + 64x_6 + 72x_7 + 34x_8 + 25x_9 + 35x_{10} \le 247 \\
&64x_1 + 267x_2 + 305x_3 + 10x_4 + 311x_5 + 36x_6 + 379x_7 + 26x_8 \\
&\quad + 76x_9 + 50x_{10} \le 956 \\
&x_i \in \{0, 1\}, \quad i = 1, \ldots, 10
\end{aligned}
$$

The optimal value of the objective function for this problem is 430. Since $n = 10$ for this instance, 10 is the maximum Hamming distance between any two solution vectors. Table 1 is really two tables combined to save space. The left table shows results for $\mathcal{Q}$ with varying values of ω. The right side shows $\mathcal{R}$ with β set at one and γ varying as shown. It is not surprising that for this small instance, the performance of the two methods is similar. Both methods allow predictable trade-offs between diversity and quality as controlled by the parameters.

2.4 Experiments with OR-lib instances of the MCKP

To obtain more substantial data on the differences between simultaneous and sequential generation, we made use of fifteen instances from the OR-LIB library (Beasley 2005): from MKNAP1.TXT we used the files 1 and 3–7 (not 2 because of real valued coefficients), from MKNAPCB1.TXT the files 1–6, and from MKNAPCB4.TXT the files 1–3 . These instances range in size from ($n = 6, m = 10$) to ($n = 100, m = 10$). To facilitate discussion across instances, it is helpful to scale distances and objective function values. We accomplish that by using the maximum obtainable values as the scale factor: we divide distances between two solutions by n and the sum of the two

Table 1 Results for the example instance of the MCKP

$\mathcal{Q}$				$\mathcal{R}$			
ω	$f(x^*)$	$f(y^*)$	$d(x^*, y^*)$	γ	$f(a^*)$	$f(b^*)$	$d(a^*, b^*)$
100	430	140	10	100	310	260	10
50	430	350	6	50	350	430	6
25	430	400	4	25	400	430	4
15	430	400	4	15	400	430	4
10	430	400	4	10	420	430	2
7	430	420	2	7	420	430	2
6	430	420	2	6	420	430	2
5	430	430	0	5	430	430	0

objective function values by $2f(x^*)$. This establishes data between zero and one for both of the performance measures.

2.4.1 Exploratory experiments

For these experiments, we fix β at one and vary the other parameters across ten values for each instance. To obtain a reasonable range of parameter values for each instance, we varied γ and ω over intervals that are roughly comparable: ten evenly spaced values between 1 and $2f(x^*)/n$. This endpoint was chosen because the maximum distance between solutions is n and the maximum value of $f(x)$ occurs at x^*. This range of parameters results in a spread of distances and objective function values for both methods over the 15 instances.

To get a general sense of the data, we make use of scatter plots for the results using configuration A that display all 300 observations (two methods times ten parameter values times fifteen instances). Figure 1 shows a scatter plot of CPU time versus the distance ratio for the observations. The '+' symbols show CPU time versus $d(x^*, y^*)/n$, while the '$\circ$' symbols show CPU time versus $d(a^*, b^*)/n$. Figure 2 shows a scatter plot of CPU time versus the objective function ratio for the observations. The '+' symbols show the sum of CPU time for problems ($\mathcal{P}$) and ($\mathcal{Q}$) versus $(f(x^*) + f(y^*))/2f(x^*)$, while the '$\circ$' symbols show CPU time for problem ($\mathcal{R}$) versus $(f(a^*) + f(b^*))/2f(x^*)$. As expected, one gets the sense that simultaneous generation, problem ($\mathcal{R}$), requires more computational effort. Figure 3 shows a scatter plot of distance ratio versus the objective function ratio for the observations. The '+' symbols show $d(x^*, y^*)/n$ versus $(f(x^*) + f(y^*))/2f(x^*)$, while the '$\circ$' symbols show $d(a^*, b^*)/n$ versus $(f(a^*) + f(b^*))/2f(x^*)$. These points are well mixed, so additional sorting is required to analyze these relationships. A side effect of the scatter plots is that they demonstrate that the parameter ranges result in a variety of distances and objective function values. That is, both methods result in overlapping observations throughout the range of possible distances.

For a more refined analysis, we make use of paired observations. Formulations for the MCKP are described in the next subsection. Results are summarized in Sect. 2.6.

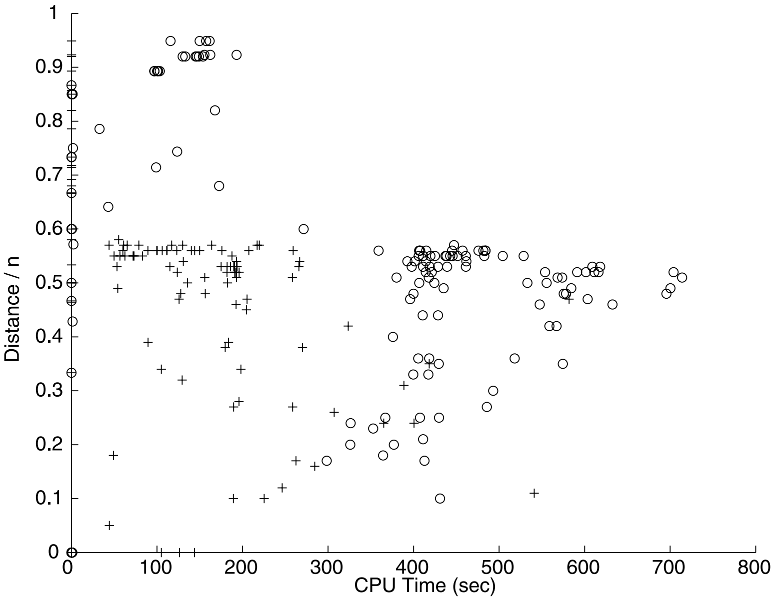

Fig. 1 Scatter plot of CPU time versus distance ratio using algorithm configuration A. The symbol '+' shows sequential generation instances, and '∘' shows simultaneous

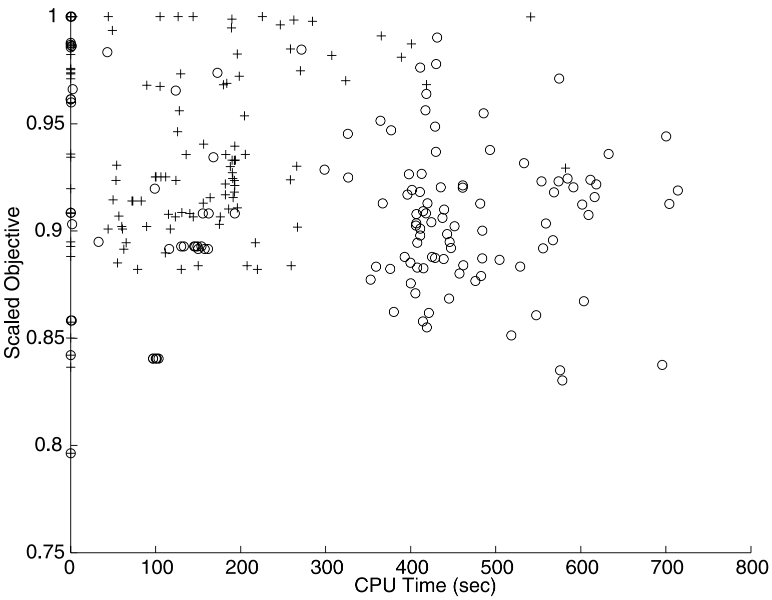

Fig. 2 Scatter plot of CPU time versus objective function ratio. The symbol '+' shows sequential generation instances, and '∘' shows simultaneous using algorithm configuration A

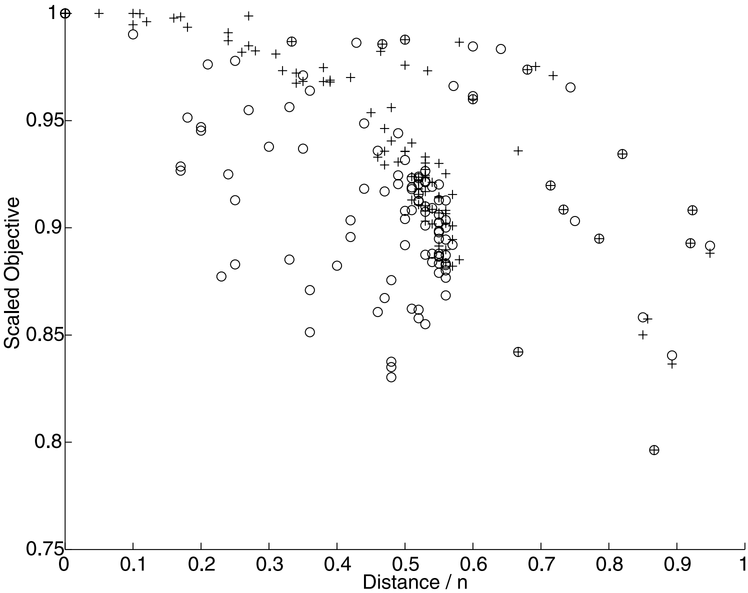

Fig. 3 Scatter plot of distance ratio versus objective function ratio. The symbol '+' shows sequential generation instances, and '∘' shows simultaneous using algorithm configuration A

2.4.2 Experiments paired by distance

To obtain a pairing based on distances, we move the distance criterion from the objective function to the constraints. Problem (Q) becomes:

$$(\hat{Q}) \quad \begin{array}{l} \max_y f(y) \\ \text{subject to} \\ d(x^*, y) \geq D \\ y \in X. \end{array}$$

Simultaneous generation can be accomplished by solving the single problem:

$$(\hat{R}) \quad \begin{array}{l} \max_{a,b} f(a) + \beta f(b) \\ \text{subject to} \\ d(a,b) \geq D \\ a \in X \\ b \in X \end{array}$$

For the MCKP, the requirement that $d(\cdot) \geq D$ is implemented as

$$\sum_{i=1}^{n} d_i \geq D$$

For each value of D for each instance, we get a pair of observations: one for sequential and one for simultaneous generation. For each of the 15 instances, we use three rounded values of D: $n/6, n/3, n/2$. This results in a total of 45 paired observations.

2.4.3 Experiments paired by average quality

We also make use of observations paired by average quality. To obtain a pairing, we move the quality criterion from the objective function to the constraints. Problem (Q) becomes:

$$
\begin{array}{ll}
(\tilde{Q}) & \max_y d(x^*, y) \\
& \text{subject to} \\
& f(y) + f(x^*) \geq A \\
& y \in X
\end{array}
$$

Simultaneous generation can be accomplished by solving a single problem:

$$
\begin{array}{ll}
(\tilde{R}) & \max_{a,b} d(a, b) \\
& \text{subject to} \\
& f(a) + \beta f(b) \geq A \\
& a \in X \\
& b \in X
\end{array}
$$

We continue with β set to one in the interest of symmetry with sequential generation. For each value of A for each instance, we get a pair of observations: one for sequential and one for simultaneous generation. To obtaining values for A, we make use of the exploratory experiments described in Sect. 2.4.1. Index the MCKP instances by $j = 1, \ldots, 15$. For each instance j, define $A^j_{\min}$ to be the lowest average objective function observed during the 20 experiments done on the instance during the exploratory experiments and let $A^j_{\max}$ be the maximum. For each of the 15 instances, we use three in rounded values of A: $(A^j_{\min} + A^j_{\max})/7$, $2(A^j_{\min} + A^j_{\max})/7$, $3(A^j_{\min} + A^j_{\max})/7$. This results again in a total of 45 paired observations.

2.5 LAP experiments

The LAP shares some characteristics of the MCKP that are helpful for our study: it has binary decision variables so that Hamming distances are sensible and it is an important problem with many applications. In other ways, though, it offers some contrast to the MCKP: it is a much easier problem with very simple constraints. For the base problem, that we called (P) in the general case, the LAP is:

$$
\begin{array}{ll}
(\mathcal{P}') & \max_x \sum_{i=1}^n \sum_{j=1}^n c_{ij} x_{ij} \\
& \text{subject to} \\
& \sum_{i=1}^n x_{ij} = 1, \quad j = 1, \ldots, n \\
& \sum_{j=1}^n x_{ij} = 1, \quad i = 1, \ldots, n \\
& x_{ij} \in \{0, 1\}, \quad i = 1, \ldots, n, \; j = 1, \ldots, n
\end{array}
$$

where the $n \times n$-matrix of "costs," c, is given as data that is negative since generally LAP data corresponds to minimization. The extension to all other formulations involving Hamming distances is done in a fashion that is analogous to the extensions to the formulation for the MCKP.

Nine instances for the LAP were obtained by varying n systematically from five to eighty. Cost coefficients where generated as a random integer between 1 and $10n$.

2.6 Summary of paired results

Table 2 shows the results of the partial experimental design varying across two algorithm configurations, two problems and two pairings. For the column labeled "Paired by" the rows labeled "Distance" correspond to formulation $(\hat{Q})$ and $(\hat{R})$ for the MCKP and the analogous formulations for the LAP because these are the experimental instances that are paired by distance. Meanwhile, the rows labeled "$f(\cdot)$" correspond to $\tilde{\mathrm{Q}}$ and $\tilde{\mathrm{R}}$, which are paired by objective function value. The columns labeled "Obj. Diff." has values computed depending on the row. For rows paired by distance, this value is computed using the results of the paired experiments as

$$100 \frac{(f(x^*) + f(y^*)) - (f(a^*) + f(b^*))}{f(a^*) + f(b^*)}.$$

For rows paired by $f(\cdot)$, this has the average percent difference in realized difference:

$$100 \frac{d(x^*, y^*) - d(a^*, b^*)}{d(a^*, b^*)}.$$

The column labeled "Time Ratio" simply has the ratio of the time for the sequential method with the time for the simultaneous method averaged over the pairs for each

Table 2 Summary of average values over the paired experiments

Problem	Paired by	Configuration A		Configuration B	
		Obj. diff. (%)	Time ratio	Obj. diff. (%)	Time ratio
MCKP	Distance	0.46%	7.11	−0.0035%	801.14
MCKP	$f(\cdot)$	4.91%	7.80	−0.9101%	721.32
LAP	Distance			0.10%	6.35
LAP	$f(\cdot)$			−2.76%	10.31

problem instance and method; however, instances that required times below 0.1 second were not included because the timing precision on our computers was too low for meaningful comparisons.

A priori, one would expect that the CPU time ratio would be larger than one and that it would vary with the problem and algorithm. The table confirms this and gives some sense of how large the difference can be. The two rows paired by average quality give a sense that for fixed quality there can be measurement differences in distance, although the differences are not large as a percentage. The averages give a good sense of the degree of the time differences; furthermore, the sequential method was faster for all runs with measurable time.

The (MCKP, Configuration A) result that sequential generation had an average of 4.91% better differences than simultaneous is a little bit surprising. This is not a large difference, but the fact that sequential generation actually was slightly better was not expected. However, the results for the LAP, where sequential generation resulted in slightly better distances provides a counterexample. In both cases the ratios for the pairs are mixed in sign and neither average difference is large.

Overall, we can conclude that for our experiments, sequential generation was much faster and that the quality differences were small. The quality differences were mixed in sign and were always small. On the other hand, sequential generation was always faster and simultaneous took a large *multiple* of the time.

3 Conclusions

There are many examples of algorithms that are based on populations of good, mutually distant solution vectors. There are also important applications of the identification of mutually distant solutions in multi-criteria optimization and in the design of decision support systems. Our paper has provided empirical evidence concerning the fundamental issue of whether these solution vectors should be generated sequentially or simultaneously.

Our experiments are substantial, reproducible and provide the first data on this important question. It would not be possible to conduct experiments that are guaranteed to apply to all problems in all settings, so we must proceed as all experimental scientists by conducting careful experiments. By choosing the MCKP and LAP, we are able to provide standardized and reproducible methods: For these problems, Hamming distances are available as a solution vector metric and commercial LP solvers such as CPLEX provide a well-known method of seeking solutions.

Of course, if maximizing the distance is the overriding concern, simultaneous generation may be preferred as indicated by Remark 3. However, when there is some desire to provide both quality and diversity, we have presented evidence that for broad ranges of practically achievable distances, sequential generation usually requires less computational effort and produces solutions that are at least as good as simultaneous generation. This provides an experimental justification for algorithm design decisions favoring sequential of simultaneous generation of diverse solutions.

References

Beasley, J.E.: http://people.brunel.ac.uk/~mastjjb/jeb/orlib/mknapinfo.html (2005)

Burkard, R., Çela, E.: Linear assignment problems and extensions. In: Du, D.-Z., Pardalos, P. (eds.) Handbook of Combinatorial Optimization, pp. 75–149. Kluwer Academic, Dordrecht (1999)

Ehrgott, M., Gandibleux, X. (eds.): Multiple Criteria Optimization: State of the Art Bibliographic Surveys. Kluwer Academic, Boston (2002)

Fréville, A.: The multidimensional 0–1 knapsack problem: an overview. Eur. J. Oper. Res. **155**, 1–21 (2004)

Fu, Z., Golden, B., Lele, S., Raghavan, S., Wasil, E.: Genetically engineered decision trees: population diversity produces smarter trees. Oper. Res. **51**, 894–907 (2005)

Glover, F.: Scatter search and star paths: beyond the genetic metaphor. OR Spektrum **17**, 125–137 (1995)

Greistorfer, P., Voß, S.: Controlled pool maintenance for metaheuristics. In: Rego, C., Alidaee, B. (eds.) Metaheuristic Optimization via Memory and Evolution, pp. 387–424. Kluwer Academic, Boston (2005)

Løkketangen, A., Woodruff, D.L.: A distance function to support optimized selection decisions. Dec. Support Syst. **39**, 345–354 (2005)

Moscato, P.: On evolution, search, optimization, genetic algorithms and martial arts: towards memetic algorithms. Caltech Concurrent Computation Program, pp. 158–179. California Institute of Technology, Pasadena, CA 91125, U.S.A., C3P Report 826 (1989)

Ryu, T., Eick, C.F.: A unified similarity measure for attributes with set or bag of values for database clustering. In: Proc. Sixth International Workshop on Rough Sets, Data Mining and Granular Computing (RSDMGrC'98) (1998)

Sörensen, K., Sevaux, M.: MA|PM: memetic algorithms with population management. Comput. Oper. Res. **33**, 1214–1225 (2006)

Steuer, R.E.: Multiple Criteria Optimization: Theory, Computation and Application. Wiley, New York (1986)

Tversky, A.: Features of similarity. Psychol. Rev. **84**, 327–352 (1977)

Whitley, D.: The GENITOR algorithm and selection pressure: why rank-based allocation of reproductive trials is best. In: Proceedings of the Third International Conference on Genetic Algorithms, pp. 116–121. Morgan Kaufmann, San Mateo (1989)

Zeleny, M.: Multiple Criteria Decision Making. McGraw–Hill, New York (1982)