Journal of Heuristics, 7: 77–97 (2000)
© 2001 Kluwer Academic Publishers

Using Experimental Design to Find Effective Parameter Settings for Heuristics

STEVEN P. COY
Continental Airlines, HQSRT, 1600 Smith Street, Houston, TX 77002, USA

BRUCE L. GOLDEN
Robert H. Smith School of Business, University of Maryland, College Park, MD 20742, USA

GEORGE C. RUNGER
College of Engineering and Applied Sciences, Arizona State University, Tempe, AZ 85287, USA

EDWARD A. WASIL*
Kogod School of Business, American University, Washington, DC 20016, USA
email: ewasil@american.edu

Abstract

In this paper, we propose a procedure, based on statistical design of experiments and gradient descent, that finds effective settings for parameters found in heuristics. We develop our procedure using four experiments. We use our procedure and a small subset of problems to find parameter settings for two new vehicle routing heuristics. We then set the parameters of each heuristic and solve 19 capacity-constrained and 15 capacity-constrained and route-length-constrained vehicle routing problems ranging in size from 50 to 483 customers. We conclude that our procedure is an effective method that deserves serious consideration by both researchers and operations research practitioners.

Key Words: statistical design of experiments, heuristics, vehicle routing

1. Introduction

Over the last 10 years or so, researchers have devoted an enormous effort to tailoring general-purpose metaheuristics such as simulated annealing, genetic algorithms, neural networks, and tabu search to solve difficult combinatorial optimization problems including the traveling salesman problem (TSP) and the vehicle routing problem (VRP). The volume edited by Reeves (1993) and the paper by Osman and Kelly (1996) provide comprehensive, accessible overviews of metaheuristics for combinatorial optimization problems.

The efforts of researchers in developing effective metaheuristics have already met with some success. For example, best-known solutions to the well-studied 14 benchmark VRPs of Christofides, Mingozzi, and Toth (1979) have been generated by tabu search procedures including those of Taillard (1993), Gendreau, Hertz, and Laporte (1994), and Xu and Kelly (1996). However, most of the efforts to develop effective metaheuristics have been computationally burdensome and very time consuming.

*Author to whom all correspondence should be addressed.

All of the metaheuristics that have been used to solve the VRP contain several parameters (roughly anywhere from five parameters to more than 25 parameters) whose values need to be set before the metaheuristic is run (Golden et al. (1998) provide a comprehensive survey of metaheuristics for the VRP). For example, the network flow-based tabu search heuristic of Xu and Kelly (1996) has 28 penalty, time-related, and control parameters and four parameters (such as the number of elite solutions to store) that determine how their heuristic solves a problem.

Our review of the literature indicates that it is not an easy task to determine appropriate values for parameters found in VRP metaheuristics. The procedures used to set a parameter's value have ranged from simple trial-and-error to sophisticated sensitivity analysis. For example, Van Breedam (1995) states: "The values that have to be assigned to all these technical parameters are for the greater part determined by trial-and-error-like experiments." Gendreau, Hertz, and Laporte (1991) state: "This algorithm contains several parameters and a number of variants can easily be envisaged. Several tests and a considerable amount of fine tuning were carried out in order to arrive at the current version." Once appropriate values for parameters have been identified, it is standard practice in the literature to report results generated by running a heuristic with the parameters fixed at these values (for example, the single pass version of TABUROUTE reported by Gendreau, Hertz, and Laporte (1994)). However, it is also common practice to report the best solution found during the course of performing sensitivity analysis (for example, the multiple passes of TABUROUTE reported by Gendreau, Hertz, and Laporte (1994)).

While, for the most part, researchers have set values of parameters in an ad hoc way, there are a few researchers who have developed systematic ways of identifying effective values of parameters found in VRP heuristics. Xu and Kelly (1996) try to identify the relative contributions of five different components of their tabu search heuristic (network flow moves, swap moves, tabu short-term memory, restart/recovery strategy, a simple tabu search procedure (TSTSP) to find the best sequence of customers on a route). They disable each component one at a time, execute their algorithm on seven VRPs, and compare the solutions of the five different strategies. Xu and Kelly conclude: ". . . the TS [tabu search] memory and restart/recovery strategy effectively help to locate extremely good solutions and TSTSP provides an effective enhancement over 3-opt . . . " Furthermore, the authors run their heuristic with different frequencies for the swap moves—every two, three, five, and six iterations—and conclude: ". . . the performance of our algorithm is sensitive to the frequency . . . [and it] performs best using a medium frequency."

Van Breedam (1996) tries to determine the significant effects of parameters for a genetic algorithm (GA) procedure and a simulated annealing (SA) procedure for the VRP. He attempts to discover the structure of the relation between total travel time and seven GA parameters (including population size, number of generations, type of local improvement operator, and quality of initial solution) and eight SA parameters (including cooling rate and type of move) by applying the Automatic Interaction Detection technique (AID) of Morgan and Sonquist (1963). AID is a tree-based classification method that uses analysis of variance to summarize the relationship between predictor and response variables.

Van Breedam applies GA and SA to 15 test problems with 100 customers each (four problems have time windows, two have pickups, and three have heterogeneous demand).

He concludes that certain parameters have "consistent significant effect for all problems." For example, in the case of GA, not using a local improvement operator gives worse solutions and using good initial solutions produces better final solutions.

The development of systematic procedures that determine appropriate values for parameters is not limited to VRP heuristics. Robertson, Golden, and Wasil (1998) use a fractional factorial experiment to set parameter values in neural network models for a finance application. Park and Kim (1998) use a nonlinear response surface optimization method based on a simplex design to find parameter settings in several applications of simulated annealing. Parsons and Johnson (1997) use statistical design of experiments to set the values of four parameters in a genetic algorithm. Xu, Chiu, and Glover (1996), in the context of the Steiner Tree-Star problem, develop a procedure for fine-tuning five key factors in a tabu search heuristic. Using two statistical tests in a very small number of experiments, they are able to find a set of values for the five factors that generates improved results in nearly three-quarters of their test problems. (For background information on designing computational experiments, see the article by Barr et al. (1995). The book by Montgomery (1991) is an excellent introduction to the design and analysis of experiments.)

In this paper, we show how statistical design of experiments can be used to find effective settings for parameters found in heuristics. In Section 2, we give an overview of our procedure. In Section 3, we illustrate our parameter setting procedure with a case study of the VRP. In Section 4, we give our conclusions.

2. Procedure for setting parameter values

Our procedure takes a small number of the problems from the entire problem set, finds high-quality parameter settings for each problem, and then combines the parameter settings to determine good parameter settings for the entire set of problems. Our procedure has four steps, which are outlined in figure 1.

In Step 1, we select a subset of problems to analyze (analysis set) from the entire set of problems (see figure 2). We select the problems so that most of the structural differences (for example, demand distribution and customer distribution) found in the problem set are represented in the analysis set. Since the time it takes to perform our procedure is directly related to the size of the analysis set, we select as few problems as possible.

In Step 2, we determine a starting level for each parameter and the range over which the parameter can vary. These decisions require some *a priori* knowledge of the behavior of the heuristic on the specific problem class. If we do not have the necessary computational experience, we conduct a pilot study. This study can be performed in a few trials on a small

Step 1.	Select a subset of problems to analyze (analysis set) from the entire set of problems.
Step 2.	Select the starting level of each parameter, the range over which each parameter will be varied, and the amount to change each parameter.
Step 3.	Select good parameter settings for each problem in the analysis set using design of experiments.
Step 4.	Combine the settings obtained in Step 3 to obtain high-quality parameter values.

Figure 1. Outline of procedure used to set parameter values.

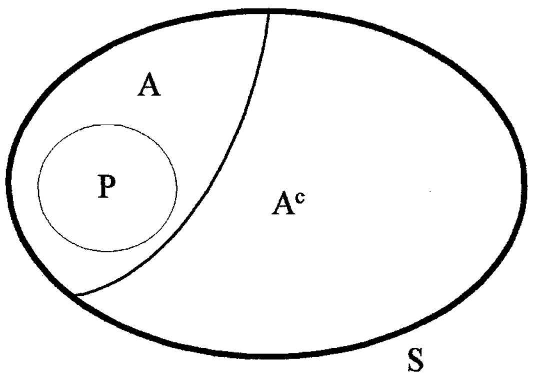


Figure 2. The pilot study problem set (P) is a proper subset of the analysis problem set (A), that is $P \subset A$. S is the entire problem set.

number of problems (taken from the analysis set). The pilot study has three objectives. The first objective is to obtain a rough approximation of the best setting for each parameter—this acts as a starting point. The second objective is to identify the experimental ranges for each parameter. In many cases, the range of a parameter is dictated by the heuristic. For example, the heuristic may not converge if a particular parameter is less than zero. In this case, we would set the minimum of this parameter to zero. The third objective is to identify the amount to change each parameter during the experimental design phase. We will discuss the third objective in more detail later in this section.

Our procedure uses a two-level factorial design. In a two-level factorial design, each parameter is tested at a low level and high level. The low and high settings are usually denoted as -1 and $+1$, respectively. This indicates that the tests are performed at design center $-\Delta$ and design center $+\Delta$. A two-level full factorial design is often abbreviated 2^k, where k is the number of parameters. For example, a full factorial design consisting of two parameters has 2^2 combinations of the two levels of each parameter. We point out that a partial two-level design such as a Taguchi design might provide quality results more efficiently.

When there are more than a few parameters, it is usually necessary to use a fractional factorial design. A fractional factorial design is often abbreviated as a 2^{k-p} where $1/2^p$ is the size of the fraction. For example, a half-fraction of a three parameter factorial design, 2^{3-1}, requires four runs (see Table 1). We plot an example of a 2^{3-1} design in figure 3. The choice of a fractional factorial design (for example, a half fraction or a quarter fraction) consists of a trade-off between the desire to conduct as few experimental runs as possible with the need to conduct enough runs to identify the significant effects.

The triples in figure 3 are called the *coded variables* associated with the factorial design. Statisticians typically convert the natural variables (value of the parameter after adding or subtracting Δ) to a matrix of $+1$s and -1s (matrix of coded variables). In figure 3, we see that the design center is located at (0, 0, 0). This is often called a *zero point*. An experimental design is typically augmented with one or more zero points to estimate the process behavior at the design center (for further details, see Montgomery (1991)).

Table 1. A 2^{3-1} fractional factorial design.

Run	A	B	C
1	$-$	$-$	$+$
2	$+$	$-$	$-$
3	$-$	$+$	$-$
4	$+$	$+$	$+$

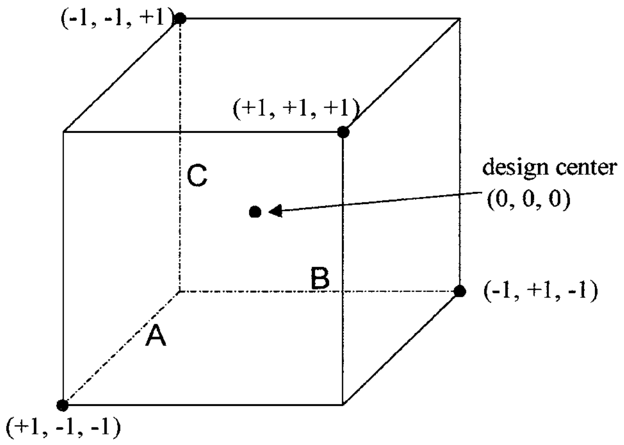


Figure 3. Geometric interpretation of a 2^{3-1} factorial design where $\Delta_1 = \Delta_2 = \Delta_3 = 1$. The triples are the coded variables associated with the entries in Table 1.

We use the factorial experimental design to determine the parameter settings for each experimental run. After we complete all of the experimental runs, we apply linear regression to the results obtained from each run to find a linear approximation of the response surface (the response is the quantity that we are trying to optimize with the heuristic). Next, we calculate the path of steepest descent on the response surface (if the objective is to minimize), using the starting point identified in the pilot study, and we make small steps along this path by changing the parameter values. At each step, we conduct one or more trials (a trial is one execution of a heuristic from a single initial solution). We continue until we reach the limit of the experimental region or the best solution found has not changed for a specified number of steps. At this point, we save the settings of each parameter (parameter vector) associated with the minimum result.

In Step 4, we determine the final parameter settings for the heuristic by taking the average of the parameter vectors that we obtained in Step 3 for each problem in the analysis set.

We point out that the response surface of a particular problem might not be convex. Thus, moving down the path of steepest descent will probably not provide an optimal solution. In fact, we do not perform exact optimization on any single problem (for example, by fitting a quadratic model after the descent step; see Montgomery (1991) for details), so that it is unlikely we will determine the exact local minimum. However, since we want to determine parameter settings that work well over a number of similar problems with different response surfaces, searching for the global minimum or exactly identifying the local minimum of each problem in the analysis set would only add complexity to the procedure without a substantial contribution to our overall objective.

3. Case study

To illustrate our procedure, we use two local search heuristics to solve a total of 34 VRPs. We report on four experiments. In each experiment, we apply a variant of Lagrangean relaxation and an edge exchange procedure to the VRPs. In the first experiment, we solve 19 capacity-constrained VRPs with a Lagrangean-relaxed version of two-opt (LT) and in the second experiment, we solve the same 19 VRPs with Lagrangean-relaxed sequential smoothing (LS) (see Coy (1998) and Coy et al. (1997)) for a description of sequential smoothing). In the third and fourth experiments, we solve 15 capacity-constrained and route-length-constrained VRPs with Lagrangean relaxed two-opt and Lagrangean-relaxed sequential smoothing, respectively. The Lagrangean relaxation procedure used by both heuristics requires several parameters. In each experiment, we use statistical design of experiments to set the parameter values.

3.1. Background

In the capacity-constrained vehicle routing problem, a homogeneous fleet of vehicles with limited capacity delivers items from a single depot to a set of customers with known demands. The objective of the VRP is to determine routes for the vehicles so that the sum of the route lengths is minimized subject to the following constraints. Each customer can be visited only once, each customer's demand must be satisfied, the total demand on each route may not exceed vehicle capacity, and each route must begin and end at the depot. Additional constraints may include a route-length restriction and delivery time windows. In this paper, we concentrate on the capacity-constrained VRP and on the capacity-constrained VRP with route-length restrictions.

The articles by Bodin et al. (1983) and Laporte (1992), the edited volume by Golden and Assad (1988) and the first five chapters in Ball et al. (1995) provide a survey of optimal and heuristic methods for the capacity-constrained VRP and its variants. Gendreau, Laporte, and Potvin (1997) and Golden et al. (1998) provide extensive reviews of metaheuristics for the VRP.

3.2. Problem sets

We use two problem sets in our experiments—the 14 problems of Christofides, Mingozzi, and Toth (1979) (denoted CMT) and the 20 large-scale problems of Golden et al. (1998). The characteristics of each problem set are described in Table 2. The problems range in size from $N = 50$ to $N = 483$ customers. Fifteen problems have route-length restrictions.

Table 2. Problem sets used in our experiments. Problems 1 to 14 are from CMT and problems 15 to 34 are from Golden et al. (1998).

Problem	N	Vehicle capacity	Max route length	Service Time
1	50	160	∞	0
2	75	140	∞	0
3	100	200	∞	0
4	150	200	∞	0
5	199	200	∞	0
6	50	160	200	10
7	75	140	160	10
8	100	200	230	10
9	150	200	200	10
10	199	200	200	10
11	120	200	∞	0
12	100	200	∞	0
13	120	200	720	50
14	100	200	1040	90
15	240	550	650	0
16	320	700	900	0
17	400	900	1200	0
18	480	1000	1600	0
19	200	900	1800	0
20	280	900	1500	0
21	360	900	1300	0
22	440	900	1200	0
23	255	1000	∞	0
24	323	1000	∞	0
25	399	1000	∞	0
26	483	1000	∞	0
27	252	1000	∞	0
28	320	1000	∞	0
29	396	1000	∞	0
30	480	1000	∞	0
31	240	200	∞	0
32	300	200	∞	0
33	360	200	∞	0
34	420	200	∞	0

3.3. Parameters used by each heuristic

Both heuristics in our experiments use a variant of Lagrangean relaxation. In a Lagrangean-relaxed VRP, the capacity constraints are relaxed and a penalty term is added to the objective function. The penalty term is formulated so that if an infeasible condition exists, the objective function is penalized. For example, if a route exceeds the capacity restriction, the penalty term will add length to the objective function value. Infeasible moves are possible under these conditions as long as the total route length reduction exceeds the penalty incurred. The penalty term has an adjustable constant, λ_C. If λ_C is too small, the edge exchange procedure will converge without finding a feasible solution. If λ_C is too large, the edge exchange procedure may converge to a poor local minimum. Thus, a key feature of an implementation of Lagrangean relaxation is the search for the right λ_C. Each of the parameters that we set with our procedure is used to control this search. The function of each parameter is described in Table 3. For further details, see Stewart and Golden (1984) and Coy (1998).

3.4. Experiments 1 and 2

In this section, we illustrate the procedure outlined in figure 1. In Experiment 1, we give a detailed description of each step of the procedure. In Experiment 2, we summarize the results of the four steps (a full description of Experiment 2 is provided by Coy (1998)). Since the two heuristics use the same Lagrangean relaxation procedure, we perform Step 1 and Step 2 only once and use the decisions made in these steps for both experiments.

3.4.1 Step 1. In Step 1, we need to select a subset of the 19 capacity-constrained problems to form our analysis set. We want to select problems that are representative of the characteristics (problem size, distribution of customer location, and distribution of demand) found

Table 3. Parameters set in Experiments 1 and 2 with design of experiments.

Parameter name	Description
Initial capacity lambda (λ_{C0})	First and smallest λ_C used in search
Large factor (LF)	Until the first feasible result has been found, increase λ_C and λ_D by LF (e.g., $\lambda_{C1} = \lambda_{C0} \times \text{LF}$)
Small factor (SF)	After a feasible result has been found, increase λ_C and λ_D by $1 + \text{SF} \times (1 - \text{LF})$
Excess load upper bound (LUB)	Used to determine whether solution is “close” to feasible; if total excess load/number of routes $\leq$ LUB and the solution is either feasible or close to feasible with respect to the route-length constraint, attempt to force feasibility
Temporary lambda factor (λ_T)	When attempting to force feasibility, temporarily increase λ_C by λ_T if the solution is infeasible with respect to vehicle capacity and temporarily increase λ_D by λ_T if the solution is infeasible with respect to the maximum route-length constraints
Number of feasible solutions (NFS)	Number of feasible solutions found on each trial

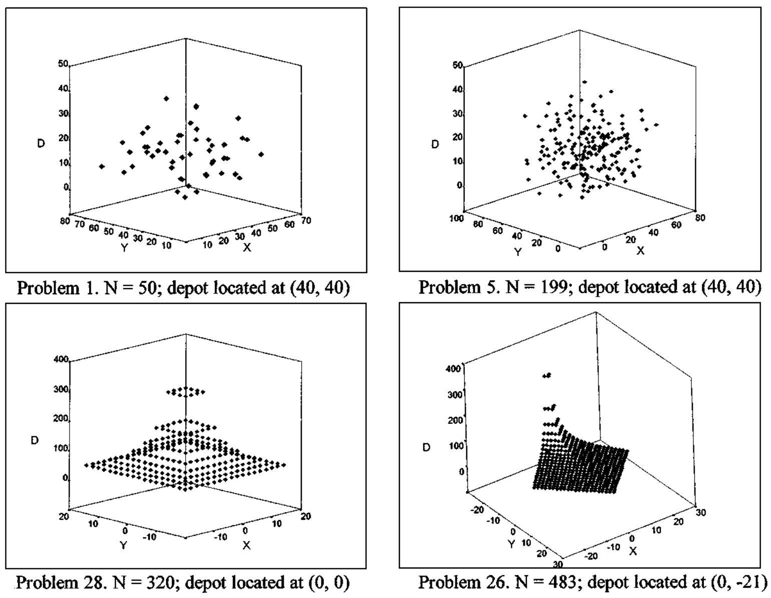

Problem 1. N = 50; depot located at (40, 40)
Problem 5. N = 199; depot located at (40, 40)
Problem 28. N = 320; depot located at (0, 0)
Problem 26. N = 483; depot located at (0, -21)

Figure 4. Three-dimensional plots of the four problems in the analysis set for Experiment 1 and Experiment 2. Demand at each customer is shown on the vertical axis.

in the entire set of problems. In addition, since the time it takes to run our experiments is related to the number of problems in the analysis set—the larger the number of problems, the more time required to run our experiments—we select a small number of problems. We require an analysis set that can be analyzed in a reasonable amount of time and produce good average parameter values. We select four problems: 1, 5, 26, and 28. This analysis set contains the smallest problem, the largest problem, and two mid-size problems. Two problems have a random distribution of customers and random distribution of demand and two problems have a symmetric arrangement of customers and clustered demand.

In figure 4, we illustrate the four problems in the analysis set. The *x* and *y* axes are located on the bottom of each plot. The quantity of demand at each location is located on the vertical axis. From these plots, we can determine how the customers are distributed and how customer demand is distributed. For example, the plot of Problem 28 indicates that the customers are placed at the corners of a grid and the locations with the highest demands are located close to the depot.

3.4.2 Step 2. In Step 2, we make several decisions in order to initialize our procedure. We choose an initial set of parameter values (design center), the size of the incremental change of each parameter (Δ), and the limits of the experimental region. In preliminary tests, we gained enough computational experience to make these choices for each of the four experiments. In Table 4, we show the values that we use to initialize our procedure. Based on the behavior of the two heuristics and the fact that our procedure relies on gradient descent, we do not need to place upper limits on the parameters.

Table 4. Minimum value, design center, and incremental change for each parameter used in Experiment 1 and Experiment 2.

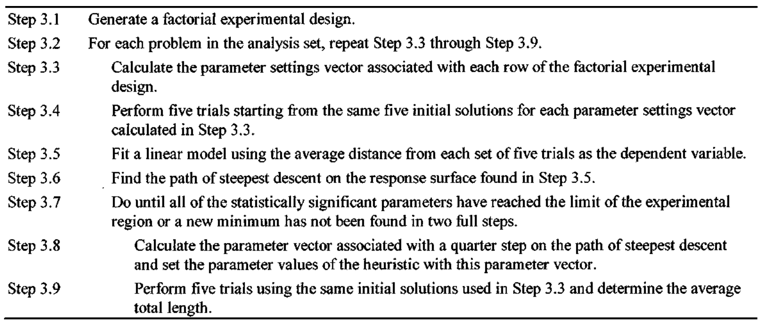
Parameter	Min value	Design center	Δ
λ_{C0}	0.01	0.6000	0.3000
LF	1.01	1.2500	0.1500
SF	0.01	0.5000	0.2500
LUB	0.00	0.1500	0.1000
λ_T	1.01	1.7500	0.5000
NFS	1.00	6.0000	3.0000

If we did not have sufficient computational experience to initialize our procedure, we could perform a pilot study using problems from the analysis set. The pilot study could be used to determine rough approximations for the experimental region, the design center, and the Δ for each parameter. We sketch a pilot study in the next paragraph.

To begin the pilot study, we use a trial-and-error approach with the problems in the pilot study problem set to find a set of parameter values in which the heuristic performs reasonably well. We call this set of parameter values the design center. Next, we identify extreme values for each parameter. Briefly, we choose one parameter, denoted by P, and hold the remaining parameters values constant at the design center. We increase the value of P. After each increase, we conduct a trial on each of the problems in the pilot study problem set and we evaluate how well the heuristic performs. When the heuristic stops performing well (for example, the heuristic is taking far too long to process or the solutions are very poor), we stop increasing P and call the current value of P the maximum value for P. We then decrease the value of P in order to identify its minimum value. We repeat this procedure with the other parameters. In the process of identifying the extreme values for each parameter, we vary the incremental change of the parameter value. Using this experience, we estimate how large a change in the parameter value is necessary to make a significant difference in the performance of the heuristic and we call this value the Δ for the parameter.

3.4.3 Step 3. In Step 3, we perform the nine steps shown in figure 5. We begin by choosing a fractional factorial design. For Experiment 1 and Experiment 2, we choose a fractional factorial experimental design with $2^{6-1} = 32$ runs. This allows us to conduct each experiment with half of the runs of a full factorial design. We transform the fractional factorial design to a matrix of coded variables and we augment our experimental design with one zero point. The augmented matrix of coded variables for Experiment 1 and Experiment 2 is shown in Table 5.

We calculate the parameter values for each run by taking the value of a parameter at the design center and either adding or subtracting Δ. For example, to find the value for λ_{C0} on the first run, we take the value of λ_{C0} at the design center and the size of the corresponding Δ (0.6 and 0.3, respectively from Table 4). We then add or subtract Δ depending on the sign of the coded variable in Table 5 to obtain $0.6 - 0.3 = 0.3$. The entire parameter vector



```
Step 3.1  Generate a factorial experimental design.
Step 3.2  For each problem in the analysis set, repeat Step 3.3 through Step 3.9.
Step 3.3      Calculate the parameter settings vector associated with each row of the factorial experimental
              design.
Step 3.4      Perform five trials starting from the same five initial solutions for each parameter settings vector
              calculated in Step 3.3.
Step 3.5      Fit a linear model using the average distance from each set of five trials as the dependent variable.
Step 3.6      Find the path of steepest descent on the response surface found in Step 3.5.
Step 3.7      Do until all of the statistically significant parameters have reached the limit of the experimental
              region or a new minimum has not been found in two full steps.
Step 3.8          Calculate the parameter vector associated with a quarter step on the path of steepest descent
                  and set the parameter values of the heuristic with this parameter vector.
Step 3.9          Perform five trials using the same initial solutions used in Step 3.3 and determine the average
                  total length.
```

Figure 5. Design of experiments procedure used for setting the parameters of a VRP heuristic.

for the first run is (0.3, 0.1, 0.25, 0.25, 2.25, 3). After computing the parameter vectors associated with each row in the fractional factorial design, we conduct a set of five trials for each parameter vector. We begin each trial from one of five solutions to the traveling salesman problem (an infeasible solution to the VRP having only one route). We generate each solution with the randomized greedy heuristic described by Johnson and McGeoch (1997).

For each experiment, we construct a data set that has an augmented matrix of coded variables (independent variables) and the average result from each run (dependent variable). We then fit a linear model to the data set using linear regression to obtain an estimate of the response surface. We point out that, since we use the same number of trials for all parameter settings, the fitted linear regression model from the raw trial data is the same as the model we fit from the averaged data. Consequently, the calculated gradient is the same. The significance tests and R^2 values are more significant for the averaged data. Another researcher with the same problem and parameters settings would obtain different solutions from the randomized greedy heuristic used for initial solutions. We average over the runs to provide an analysis that is somewhat less sensitive to this randomization—one that another researcher could, at least on the average, duplicate.

For Experiments 1 and 2, we find that the linear regression models are significant at the 0.01 level and have adjusted R^2 values that range from 0.42 to 0.96. Each parameter that we set with our procedure is statistically significant in at least one of the linear models. The linear models that we use for each problem are described in Table 6.

It is critical to choose the Δ of each parameter so that the behavior of the process will vary enough to allow a significant linear fit (using the F test). If a linear model does not fit (F test fails), we recommend that the Δ of each parameter should be increased by as much as 50% to 100% and that a new replication of the experiment should be performed. We use the linear estimate of the response surface to determine how to set the parameter values. Since we are minimizing, we find the path of steepest descent on the response surface.

The gradient of the linear model is the vector $\mathbf{b} = (b_1, b_2, \ldots, b_k)$ where b_j is an estimated regression coefficient. The path of steepest descent is the negative gradient of the linear model ($-\mathbf{b}$). To move along the path of steepest descent, a change of one unit in the coded variable, x_j, corresponds to a change of b_i/b_j units (coded) in variable x_i. In order to make

Table 5. Augmented matrix of coded variables used in Experiments 1 and 2.

	Parameters					
Run	λ_{C0}	LF	SF	LUB	λ_T	NFS
1	−1	−1	−1	+1	+1	−1
2	+1	−1	−1	+1	+1	+1
3	−1	+1	−1	+1	+1	+1
4	+1	+1	−1	+1	+1	−1
5	−1	−1	+1	+1	+1	+1
6	+1	−1	+1	+1	+1	−1
7	−1	+1	+1	+1	+1	−1
8	+1	+1	+1	+1	+1	+1
9	−1	−1	−1	−1	+1	+1
10	+1	−1	−1	−1	+1	−1
11	−1	+1	−1	−1	+1	−1
12	+1	+1	−1	−1	+1	+1
13	−1	−1	+1	−1	+1	−1
14	+1	−1	+1	−1	+1	+1
15	−1	+1	+1	−1	+1	+1
16	+1	+1	+1	−1	+1	−1
17	−1	−1	−1	+1	−1	+1
18	+1	−1	−1	+1	−1	−1
19	−1	+1	−1	+1	−1	−1
20	+1	+1	−1	+1	−1	+1
21	−1	−1	+1	+1	−1	−1
22	+1	−1	+1	+1	−1	+1
23	−1	+1	+1	+1	−1	+1
24	+1	+1	+1	+1	−1	−1
25	−1	−1	−1	−1	−1	−1
26	+1	−1	−1	−1	−1	+1
27	−1	+1	−1	−1	−1	+1
28	+1	+1	−1	−1	−1	−1
29	−1	−1	+1	−1	−1	+1
30	+1	−1	+1	−1	−1	−1
31	−1	+1	+1	−1	−1	−1
32	+1	+1	+1	−1	−1	+1
33	0	0	0	0	0	0

Table 6. Coefficients of linear models from Experiments 1 and 2. All models are significant at the 0.01 level. A zero indicates that the parameter is not significant at the 0.05 level.

Problem	Adj.R^2	Intercept	λ_{C0}	LF	SF	LUB	λ_T	NFS
Experiment 1 (LT)								
1	0.581	550.276	3.14865	0.00000	0.00000	0.00000	0.00000	−3.35173
5	0.677	1397.097	7.96334	2.59741	3.09920	0.00000	0.00000	−3.68940
28	0.447	1191.016	3.22946	3.42525	0.00000	0.00000	0.00000	0.00000
26	0.440	1247.358	2.82791	2.89954	0.00000	0.00000	0.00000	−3.42412
Experiment 2 (LS)								
1	0.966	551.621	12.93426	0.00000	0.00000	−1.46257	0.00000	0.00000
5	0.758	1360.410	3.79195	1.41637	0.00000	0.00000	0.00000	−2.93175
28	0.596	1164.822	3.81332	0.00000	0.00000	0.00000	0.00000	−1.92033
26	0.420	1202.462	1.86986	1.71696	1.87747	0.00000	−1.62750	−1.53584

a one unit (coded) change in the variable with the maximum coefficient, we divide each coefficient by the absolute value of the maximum coefficient in the model (b_m). To calculate the step size (uncoded), we multiply each ratio b_j/b_m by Δ_j.

To illustrate this technique, we demonstrate how to calculate a step along the path of steepest descent using Problem 1 in Experiment 1 (see Table 6). We begin by finding the regression coefficient with the largest absolute value. The largest absolute coefficient is 3.35173 (the coefficient of NFS). We then divide each parameter's coefficient by this quantity and multiply each of these results by the parameter's Δ. The result is a full step for each parameter. The step size for λ_{C0}, is $(3.14865/3.35173) \times 0.3 = .2818$, the step size for NFS is $(3.35173/3.35173) \times 3 = 3$, and the step sizes for the remaining parameters are 0 (these parameters are not statistically significant).

In the last three steps of our procedure (Steps 3.7 to 3.9), we make steps along the path of steepest descent. Starting at the design center, we subtract the step size of each parameter from the previous level of each parameter. If the next step along the path will cause one of the parameters to go outside the experimental region, we hold that parameter constant and continue making steps with the other parameters. After calculating a step, we perform a set of five trials to determine the performance of the heuristic at this point (using the same five initial solutions used earlier). We continue making steps along the path until we fail to improve the response for two full steps or when all of the parameters reach the limit of the experimental region. Finally, we select the parameter settings associated with the minimum response.

Table 7 shows our results for Problem 1 in Experiment 1. (In the remainder of the section, we use the term length to refer to the sum of the route lengths in the VRP.) Using the step sizes calculated above, we make 1/4 steps down the path of steepest descent (we make small steps to avoid stepping over potentially good local minima) by subtracting 1/4 $\times$ the step size from the value of each parameter in the previous parameter vector. For example, Step 0 in Table 7 represents the design center. We calculate the parameter vector for Step 1 in the following manner. The value for λ_{C0} is $0.6 - 1/4 \times 0.2818 = 0.6 - .07045 = 0.52955$,

Table 7. Setting parameters for Problem 1 in Experiment 1. The parameter vector for each set of trials is 1/4 step down the path of steepest descent from the parameter vector used in the previous set of trials.

	Parameters							
Step	λ_{C0}	LF	SF	LUB	λ_T	NFS	Average length	Average times (s)
0	0.6000	1.2500	0.5000	0.1500	1.7500	6.0000	553.54	0.36
1	0.5295	1.2500	0.5000	0.1500	1.7500	6.7500	544.45	0.41
2	0.4591	1.2500	0.5000	0.1500	1.7500	7.5000	546.06	0.42
3	0.3886	1.2500	0.5000	0.1500	1.7500	8.2500	546.42	0.50
4	0.3182	1.2500	0.5000	0.1500	1.7500	9.0000	542.16	0.54
5	0.2477	1.2500	0.5000	0.1500	1.7500	9.7500	542.61	0.61
6	0.1773	1.2500	0.5000	0.1500	1.7500	10.5000	543.35	0.68
7	0.1068	1.2500	0.5000	0.1500	1.7500	11.2500	539.39	0.74
8	0.0364	1.2500	0.5000	0.1500	1.7500	12.0000	540.47	0.85
9	0.0364	1.2500	0.5000	0.1500	1.7500	12.7500	540.47	0.90
10	0.0364	1.2500	0.5000	0.1500	1.7500	13.5000	540.47	0.94
11	0.0364	1.2500	0.5000	0.1500	1.7500	14.2500	540.47	0.93
12	0.0364	1.2500	0.5000	0.1500	1.7500	15.0000	540.47	0.96
13	0.0364	1.2500	0.5000	0.1500	1.7500	15.7500	540.47	1.03
14	0.0364	1.2500	0.5000	0.1500	1.7500	16.5000	540.47	1.08
15	0.0364	1.2500	0.5000	0.1500	1.7500	17.2500	540.47	1.07

the value for parameter NFS is $6 - 1/4 \times (-3) = 6.75$, and the values of LF, SF, LUB, and λ_T do not change.

We chose to make several small steps along the gradient of descent rather than two large steps. We did this so as not to step over a good local minimum along the path. Clearly, our method is a compromise between performance and complexity. A traditional statistical approach would make full steps down the path, stop after two steps, and fit a quadratic model to the response surface to find the local minimum. Exact optimization based on a quadratic model for one particular problem adds complexity to the procedure without substantially improving our ability to determine parameter settings that work reasonably well over a number of similar problems. Furthermore, smaller initial values for delta would no doubt improve the solution to a specific problem, but again contribute little to our overall objective.

After making a step, we conduct five trials using the same five initial solutions and the current set of parameter settings. We then average the results of the five trials. In this problem, we found the best average length on Step 7. In Step 9, λ_{C0} did not decrease, because the next step would have gone outside the experimental region. In this case, we hold λ_{C0} constant and we continue to make steps with NFS until we meet the stopping criterion at Step 15.

In figure 6, we summarize the response surface procedure for each of the four test problems. The panels on the left show the average length of each trial at each step. (We

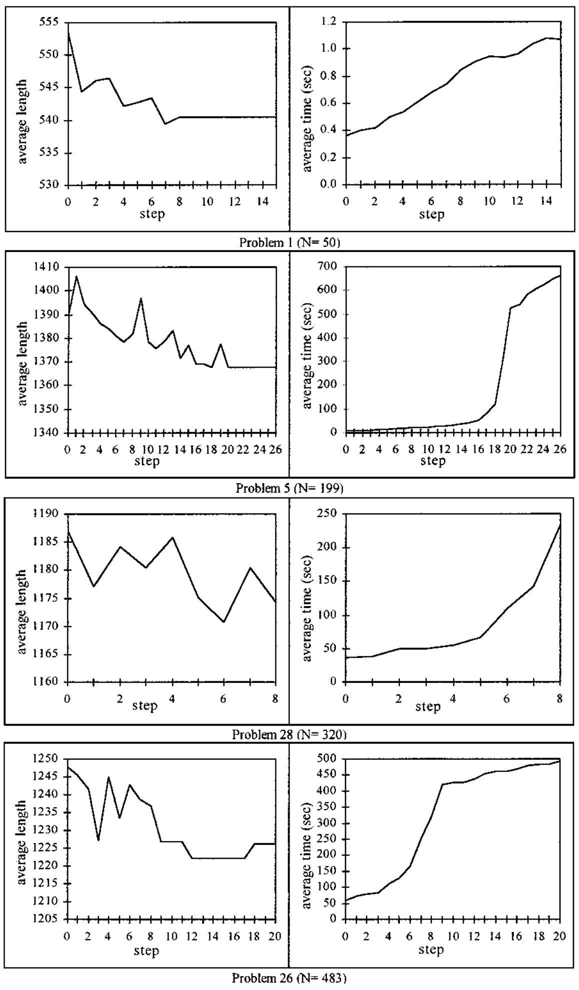


Figure 6. Response surface optimization for Experiment 1.

point out that some smoothing of the curve would assist in the location of a minimizing step. However, our objective is to make a recommendation for a class of problems and the additional accuracy is not necessary for any specific problem.) The panels on the right show the average processing time for each trial at each step (that is, the average CPU time it took to complete one trial). The top two panels (Problem 1) show the results given in Table 7. We find the minimum result at Step 7 and the average length plateaus at a slightly higher value until the search stops at Step 15. This indicates that decreasing λ_{C0} in Step 8 does not improve the average result and that the subsequent increase in NFS adds processing time, but does not change the average length.

Problem 5 has four significant parameters, λ_{C0}, LF, SF, and NFS (see Table 6). In this problem, reducing the values of LF and SF (each has a positive coefficient) causes processing time to increase rapidly for most of the steps. By Step 20, λ_{C0}, LF, and SF reach their minimum values. Increasing NFS (the only parameter that is still within the experimental region) adds processing time at a slower rate, but does not reduce the average length.

In Problem 28, NFS is not statistically significant. In this case, we find a minimum at Step 6. The procedure stops at Step 9, since any further changes would cause the values of λ_{C0} and LF to leave the experimental region.

In Problem 26, we find the minimum average length at Step 12 and stop testing at Step 20 since no additional improvement has been made for eight 1/4 steps. This problem has three statistically significant parameters (λ_{C0}, LF, NFS). By Step 9, λ_{C0} and LF reach the limit of the experimental region. For the remaining steps, we increase NFS. In contrast to Problems 1 and 5, increasing NFS, after the other parameters have reached their limits, does decrease the average length.

3.4.4 Step 4. In the final step of our procedure, we average the parameter values produced by our response surface procedure to obtain the final parameter values for the experiment. In Table 8, we show the parameter values for each problem and the average parameter values for both experiments.

3.4.5 Computational results. We solve all 19 capacity-constrained problems with LT and LS using the average parameter values. The computational results are given in Table 9. On average, LT generates solutions that are 4.81% above the best-known solutions on all 19 problems. On average, LS generates solutions that are 3.43% above the best-known solutions on all 19 problems.

Table 8. Parameter vectors for Experiments 1 and 2.

Problem	λ_C	LF	SF	LUB	λ_T	NFS
Experiment 1 (LT)						
1	0.1068	1.2500	0.5000	0.1500	1.7500	11.2500
5	0.0750	1.0176	0.0135	0.1500	1.7500	12.9495
28	0.1757	1.0250	0.5000	0.1500	1.7500	6.0000
26	0.0425	1.0277	0.5000	0.1500	1.7500	15.0000
Average	0.1000	1.0801	0.3784	0.1500	1.7500	11.2999
Experiment 2 (LS)						
1	0.0750	1.2500	0.5000	0.1698	1.7500	6.0000
5	0.0750	1.0819	0.2199	0.1500	1.7500	12.9583
28	0.3750	1.2500	0.5000	0.1500	1.7500	7.1331
26	0.3759	1.1472	0.3125	0.1500	2.0751	7.8406
Average	0.2252	1.1823	0.3831	0.1549	1.8313	8.4830

Table 9. Computational results from Experiments 1 and 2. Minimum result from 25 trials on each problem.

			Experiment 1		Experiment 2	
Problem	*N*	Best known	LT	Time (min)	LS	Time (min)
1	50	524.61	524.61	0.61	524.61	0.95
2	75	835.26	850.88	1.95	849.84	2.27
3	100	826.14	837.22	2.07	836.36	2.36
4	150	1028.42	1054.23	5.68	1054.17	7.01
5	199	1291.45	1350.03	12.06	1340.53	16.48
11	100	1042.11	1046.91	3.15	1047.73	5.05
12	120	819.56	819.56	2.38	819.56	3.03
23	255	587.09	630.31	17.03	614.27	28.99
24	323	746.56	799.23	28.26	782.48	54.38
25	399	932.68	996.43	48.91	973.54	94.23
26	483	1137.18	1214.66	78.29	1177.14	147.38
27	252	881.04	913.41	27.80	902.76	31.83
28	320	1103.69	1168.24	46.28	1146.17	53.26
29	396	1364.23	1447.97	76.10	1416.65	86.26
30	480	1656.66	1744.97	121.73	1714.70	132.36
31	240	666.84	733.87	17.02	724.68	15.55
32	300	973.60	1058.85	30.34	1040.33	31.70
33	360	1338.78	1446.30	48.60	1409.63	51.47
34	420	1831.62	1918.26	73.01	1896.72	87.96

3.5. *Capacity-constrained and distance-constrained problems*

Using the procedure described above to set the parameter values, we solve 15 capacity and route-length-constrained problems with Lagrangean-relaxed two-opt and Lagrangean-relaxed sequential smoothing. To accommodate the route-length constraint, the two heuristics require two additional parameters. We use a 2^{8-2} fractional factorial experimental design for the procedure.

We solve all 15 capacity-constrained and route-length-constrained problems with LT and LS using the average parameter values. The computational results are given in Table 10.

On average, LT generates solutions that are 1.63% above the best-known solutions on all 15 problems. On average, LS generates solutions that are 0.80% above the best-known solutions on all 15 problems. In addition, LT generates two new best-known solutions and LS generates five new best-known solutions. The routes for all of the new best-known solutions as well as a detailed presentation of Experiments 3 and 4 are provided by Coy (1998).

Table 10. Final results from Experiments 3 and 4. Each result is the minimum from 25 trials on each problem. Bold print indicates a new best-known solution.

			Experiment 3		Experiment 4	
Problem	*N*	Best known	LT	Time (min)	LS	Time (min)
6	50	555.43	561.77	0.95	560.89	1.70
7	75	909.68	937.33	3.48	929.19	8.69
8	100	865.94	879.07	3.27	866.87	5.20
9	150	1162.55	1189.08	14.18	1186.70	38.81
10	199	1395.85	1480.08	25.41	1443.95	69.95
13	100	1541.14	1561.45	9.02	1552.90	26.73
14	120	866.37	869.20	2.96	866.53	6.53
15	240	5646.46	5862.58	31.54	5761.64	49.92
16	320	8566.04	8606.01	62.18	**8512.72**	65.00
17	400	11649.06	**11200.38**	74.41	11242.54	84.43
18	480	14639.32	14031.06	53.80	**13782.41**	43.49
19	200	6702.73	6508.26	3.90	**6466.68**	12.38
20	280	9016.93	**8540.19**	10.36	8540.74	11.84
21	360	11047.69	10415.84	37.14	**10334.90**	36.24
22	440	12250.06	12042.50	111.29	**11957.15**	99.04

3.6. *Summary of computational results*

In this section, we compare the quality of the solutions generated by LT and LS to the solutions generated by the tabu search heuristic (TS) of Xu and Kelly (1996) and the record-to-record travel heuristic (RTR) of Golden et al. (1998).

In Tables 11 and 12, we summarize the performance of the four heuristics. Over all 34 problems, we make the following observations. LS is first in solution quality and third in running time. It averages 2.27% above the best-known solutions and its running time is approximately 1,412 minutes (23.5 hours). TS is a close second in solution quality and fourth in running time. It averages 2.44% above the best-known solutions and its running time is approximately 54,232 minutes (903.9 hours). RTR is third in solution quality and first in running time. It averages 3.04% above the best-known solutions and its running time is approximately 935 minutes (15.6 hours). LT is fourth in solution quality and second in running time. It averages 3.41% above the best-known solutions and its running time is approximately 1,085 minutes (18.1 hours).

Finally, we conduct the following experiment in order to examine how results produced by our tuned parameters compare to results produced by random parameters. We generate 10 random parameter vectors where the values of the six parameters (λ_{C0}, LF, SF, LUB, λ_T, NFS) are selected from a uniform distribution on the interval (Minimum parameter value, 2 $\times$ Design center $-$ Minimum parameter value). We then run LT and LS using the 10 random parameter vectors on each of the 19 capacity-constrained problems. On each

Table 11. Percent above the best-known solutions for the LT, LS, TS, and RTR heuristics. Bold print indicates the best result for a particular category.

Problem set	LT	LS	TS	RTR
Capacity-constrained				
CMT	1.53	1.40	**0.09**	2.10
Large-scale	6.72	4.61	**0.25**	1.79
Overall	4.81	3.43	**0.19**	1.91
Capacity-constrained and route-length-constrained				
CMT	2.24	**1.36**	3.62	3.34
Large-scale	1.11	**0.30**	6.92	5.47
Overall	1.63	**0.80**	5.50	4.48
All problems				
CMT	1.88	**1.38**	1.72	2.72
Large-scale	4.47	**2.89**	2.91	3.26
Overall	3.41	**2.27**	2.44	3.04

Table 12. Running times (in minutes) for the LT, LS, TS, and RTR heuristics. Bold print indicates the best result for a particular category.

Problem set	LT	LS	TS	RTR
Capacity-constrained				
CMT	**27.90**	37.15	927.80	76.84
Large-scale	613.37	815.37	42145.56	**390.10**
Overall	641.27	852.52	43073.36	**466.94**
Capacity-constrained and route-length-constrained				
CMT	**59.27**	157.60	1122.13	115.46
Large-scale	384.63	402.35	10037.12	**352.94**
Overall	**443.89**	559.96	11159.25	468.40
All problems				
CMT	**87.17**	194.75	2049.93	192.30
Large-scale	998.00	1217.72	52182.68	**743.04**
Overall	1085.16	1412.48	54232.61	**935.34**

problem, we start each heuristic from the same 25 initial solutions used in our previous experiments.

In Table 13, we give the average percent above the best-known solutions for LT and LS using tuned parameter vectors and the 10 random parameter vectors on the 19 capacity-constrained problems. We observe that the average of 6.47% produced by LT with tuned parameters is smaller than all 10 averages produced by LT with random parameters, and

Table 13. Average percent above the best-known solutions for LT and LS heuristics using tuned and randomly generated parameter vectors on the 19 capacity-constrained problems.

	Tuned	Random vector									
	Vector	1	2	3	4	5	6	7	8	9	10
LT	6.47	7.79	8.04	6.73	9.13	7.76	8.78	8.17	8.16	7.23	7.53
LS	4.61	6.18	6.21	4.71	6.79	6.15	6.51	5.94	6.05	4.72	4.82

the average of 4.61% produced by LS with tuned parameters is also smaller than all 10 averages produced by LS with random parameters.

4. Conclusions

In this paper, we propose a procedure based on statistical design of experiments that systematically selects high-quality parameter values. Our parameter setting procedure has four steps. In the first step, we select a subset of problems to analyze from the entire set of problems. In the second step, we use computational experience to select the starting level of each parameter, the range over which each parameter will be varied, and the amount to change each parameter. In the third step, we select good parameter settings for each problem in the analysis set using statistical design of experiments and response surface optimization. In the fourth step, we average the parameter values obtained in the third step to obtain high-quality parameter values. Using our procedure, we set the values of parameters and run our heuristics on 34 test problems that range in size from 50 to 483 customers. Our computational results show that LT and LS are reasonably effective in terms of solution quality. The accuracy of LS is comparable to tabu search and record-to-record travel. Furthermore, LS is much faster than tabu search.

Perhaps, most importantly, our procedure is a single pass procedure. The fact that complex heuristics such as LT and LS work so well under this restriction attests to the robustness of the approach.

In our computational experiments, we found that our procedure worked well on both the capacity-constrained and capacity-constrained and route-length-constrained problems. This may not always be the case with other combinatorial optimization problems or heuristics. Poor performance may indicate that the class of problems being studied is too broad for one set of parameter values. Thus, it may be necessary to divide the class into two subclasses. If the heuristic does not perform well using the average settings and different problems require very different parameters settings, the problem class is probably too broadly specified. To divide the class into subclasses, we would determine how the problems in the analysis set differ and which of these differences is significant (for example, depot location or distribution of demand). Then we would divide the problems into two or more classes based on these characteristics. Finally, we would apply our procedure again on each of the new subclasses.

References

Ball, M., T. Magnanti, C. Monma, and G. Nemhauser (eds.). (1995). *Network Models*. Amsterdam: North-Holland.

Barr, R., B. Golden, J. Kelly, M. Resende, and W. Stewart. (1995). "Designing and Reporting on Computational Experiments with Heuristic Methods." *Journal of Heuristics* 1, 9–32.

Bodin, L., B. Golden, A. Assad, and M. Ball. (1983). "Routing and Scheduling of Vehicles and Crews." *Computers & Operations Research* 10(2), 63–211.

Christofides, N., A. Mingozzi, and P. Toth. (1979). "The Vehicle Routing Problem." In N. Christofides, A. Mignozzi, P. Toth, and C. Sandi (eds.), *Combinatorial Optimization*. Chichester, UK: John Wiley & Sons, pp. 315–338.

Coy, S. (1998). "Fine-Tuned Learning: A New Approach to Improving the Performance of Local Search Heuristics." Ph.D. Dissertation, University of Maryland, College Park, Maryland.

Coy, S., B. Golden, G. Runger, and E. Wasil. (1997). "Solving the TSP with Sequential Smoothing." In *Proceedings of the 2nd International Conference on Computational Intelligence and Neuroscience*. Research Triangle Park, North Carolina, pp. 280–283.

Gendreau, M., A. Hertz, and G. Laporte. (1991). "A Tabu Search Heuristic for the Vehicle Routing Problem." CRT-777, Centre de Recherche sur les Transports, Université de Montréal, Montréal, Canada.

Gendreau, M., A. Hertz, and G. Laporte. (1994). "A Tabu Search Heuristic for the VRP." *Management Science* 40, 1276–1290.

Gendreau, M., G. Laporte, and J.-Y. Potvin. (1997). "Vehicle Routing: Modern Heuristics." In E. Aarts and J.K. Lenstra (eds.), *Local Search in Combinatorial Optimization*. London, England: John Wiley & Sons, Ltd., pp. 311–336.

Golden, B. and A. Assad (ed.). (1988). *Vehicle Routing: Methods and Studies, Studies in Management Science and Systems*, *Vol. 16*. Amsterdam, The Netherlands: North Holland.

Golden, B., E. Wasil, J. Kelly, and I-M. Chao. (1998). "The Impact of Metaheuristics on Solving the Vehicle Routing Problem: Algorithms, Problems Sets, and Computational Results." In T. Crainic and G. Laporte (eds.), *Fleet Management and Logistics.* Boston, MA: Kluwer Academic Publishers, pp. 33–56.

Johnson, D. and L. McGeoch. (1997). "The Traveling Salesman Problem: A Case Study in Optimization." In E. Aarts and J.K. Lenstra (eds.), *Local Search in Combinatorial Optimization*. London, England: John Wiley & Sons, Ltd., pp. 215–310.

Laporte, G. (1992). "The Vehicle Routing Problem: An Overview of Exact and Approximate Algorithms." *European Journal of Operational Research* 59, 345–358.

Montgomery, D. (1991). *Design and Analysis of Experiments*, John Wiley & Sons, New York.

Morgan, J. and J. Sonquist. (1963). "Problems in the Analysis of Survey Data and a Proposal." *Journal of the American Statistical Association* 58, 415–434.

Osman, I. and J. Kelly. (1996). "Metaheuristics: An Overview." In I. Osman and J. Kelly (eds.), *Metaheuristics: Theory and Applications*. Boston, MA: Kluwer Academic Publishers, pp. 1–21.

Park, M.-W. and Y.-D. Kim. (1998). "A Systematic Procedure for Setting Parameters in Simulated Annealing Algorithms." *Computers & Operations Research* 24(3), 207–217.

Parsons, R. and M. Johnson. (1997). "A Case Study in Experimental Design Applied to Genetic Algorithms with Applications to DNA Sequence Assembly." *American Journal of Mathematical and Management Sciences* 17(3/4), 369–396.

Reeves, C. (ed.). (1993). *Modern Heuristic Techniques for Combinatorial Problems*. New York: Halstead Press.

Robertson, S., B. Golden, and E. Wasil. (1998). "Neural Network Models for Initial Public Offerings." *Neurocomputing* 18, 165–182.

Stewart, W. and B. Golden. (1984). "A Lagrangean Relaxation Heuristic for Vehicle Routing." *European Journal of Operational Research* 15, 84–88.

Taillard, E. (1993). "Parallel Iterative Search Methods for Vehicle Routing Problems." *Networks* 23, 661–673.

Van Breedam, A. (1996). "An Analysis of the Effect of Local Improvement Operators in Genetic Algorithms and Simulated Annealing for the Vehicle Routing Problem." RUCA Working Paper 96/14, Faculty of Applied Economics, University of Antwerp, Antwerp, Belgium.

Van Breedam, A. (1995). "Improvement Heuristics for the Vehicle Routing Problem Based on Simulated Annealing." *European Journal of Operational Research* 86, 480–490.

Xu, J., S. Chiu, and F. Glover. (1996). "Fine-tuning a Tabu Search Algorithm with Statistical Tests." Working Paper, Graduate School of Business, University of Colorado, Boulder, Colorado.

Xu, J. and J. Kelly. (1996). "A Network Flow-based Tabu Search Heuristic for the Vehicle Routing Problem." *Transportation Science* 30, 379–393.