Assessing the Finite-Time Performance of Local Search Algorithms

Darrall Henderson
Lieutenant Colonel, United States Army

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Industrial and Systems Engineering

Sheldon H. Jacobson, Co-Chair
C. Patrick Koelling, Co-Chair
Ebru Bish
Hanif Sherali
Ron R. Wakefield

April 18 2001
Blacksburg, Virginia

Keywords: combinatorial optimization, convergence, discrete optimization, heuristics, hill climbing algorithms, finite-time performance, simulated annealing, local search

Assessing the Finite-Time Performance of Local Search Algorithms

Darrall Henderson

(ABSTRACT)

Identifying a globally optimal solution for an intractable discrete optimization problem is often cost prohibitive. Therefore, solutions that are *within a predetermined threshold* are often *acceptable* in practice. This dissertation introduces the concept of β-acceptable solutions where β is a predetermined threshold for the objective function value.

It is difficult to assess a priori the effectiveness of local search algorithms, which makes the process of choosing parameters to improve their performance difficult. This dissertation introduces the β-acceptable solution probability in terms of β-acceptable solutions as a finite-time performance measure for local search algorithms. The β-acceptable solution probability reflects how effectively an algorithm has performed to date and how effectively an algorithm can be expected to perform in the future. The β-acceptable solution probability is also used to obtain necessary asymptotic convergence (with probability one) conditions. Upper and lower bounds for the β-acceptable solution probability are presented. These expressions assume particularly simple forms when applied to specific local search strategies such as Monte Carlo search and threshold accepting. Moreover, these expressions provide guidelines on how to manage the execution of local search algorithm runs. Computational experiments are reported to estimate the probability of reaching a β-acceptable solution for a fixed number of iterations. Logistic regression is applied as a tool to estimate the probability of reaching a β-acceptable solution for values of β close to the objective function value of a globally optimal solution as well as to estimate this objective function value. Computational experiments are reported with logistic regression for pure local search, simulated annealing and threshold accepting applied to instances of the TSP with known optimal solutions.

Subject Classification: Analysis of Algorithms, Efficiency, Finite-time Performance, Simulated Annealing, Sub-optimal Algorithms, Probability, Stochastic Model Applications.

Acknowledgements

I wish to acknowledge my advisors Dr. Sheldon H. Jacobson and Dr. C. Patrick Koelling, for their expertise, guidance, time and support throughout my education process. I also wish to acknowledge and thank my other committee members Dr. Hanif Sherali, Dr. Ron Wakefield and Dr. Ebru Bish for their time and support.

I would like to thank the Industrial and Systems Engineering Department for their support of and dedication to graduate students. In particular, I thank Ms. Lovedia Cole, Mrs. Kathy Buchanan, and Professor Mike Deisenroth for their patience, kindness and commitment to their job. I appreciate the support provided by the Virginia Tech Simulation and Optimization Laboratory. A special thanks goes to Will Vest and Brian Burgess for supporting my hardware and software needs.

I would like to thank the Statistical Consulting Center of Virginia Tech for their ideas and assistance with data analysis. In particular, Dr. George Terrell and Ms. Alyaa Zahran were most helpful with ideas and discussions regarding logistic regression and data analysis. I also wish to acknowledge and thank Dr. Jeffery Birch for his suggestions and help with regression analysis.

I would like to thank the Department of Mathematical Sciences of the United States Military Academy (West Point, New York) and the United States Special Operations Command at MacDill Air Force Base (Tampa, Florida) for supporting my educational efforts. In particular, I would like to thank Colonel Henry (Chip) Cobb, and Lieutenant Colonel Marc Alderman for their enthusiastic support through this entire process. I would also like to thank Lieutenant Colonel (Dr.) Patrick J. Driscoll for his support and encouragement over the years.

I would like to thank Dr. Neal D. Glassman of the Air Force Office of Scientific Research for supporting the research contained in this dissertation (through AFSOR grants F49620-98-1-0111, F49620-98-1-0432, and F49620-01-1-0007). I also would like to thank Dr. W. Garth Frazier, formerly of the Materials Process Design Branch of the Air Force Research Laboratory, Wright Patterson Air Force Base (Dayton, Ohio) for his assistance with this research and to the Generalized Hill Climbing Algorithm Group

A special thanks goes to Dr. Diane Vaughan for her assistance through this whole process. I like to think of her as the “unofficial” sixth member of my committee and a valued member of the “GHC” team. Finally, I would like to thank the rest of the “GHC” team: Dr. Sheldon Jacobson, Lieutenant Colonel (Dr.) Alan Johnson, Dr. Kelly Sullivan, Derek E. Armstrong, and Tevfik Aytemiz, for making my return to academia an enjoyable experience.

Dedication

This work is dedicated to Carol.

Contents

List of Figures

List of Tables

Chapter 1:

Introduction and Motivation

1.1 Motivation

Discrete optimization problems are defined by a set of solutions and an objective function associated with each solution (Garey and Johnson 1979, page 123). The goal when addressing a discrete optimization problem is to find solutions that globally optimize the objective function value. Unfortunately, many discrete optimization problems are in the class NP-hard (Garey and Johnson 1979), hence a polynomial time algorithm to solve the problem does not exist. Complete enumeration of the solution space for large discrete optimization problems is inefficient, if not impossible with current technology. Therefore, much effort has been directed toward developing efficient heuristic algorithms to find solutions that are "*good enough*" in a "*reasonable*" amount of time.

1.2 Research Goals

For NP-hard discrete optimization problems, heuristic procedures are formulated with the hope of finding good or near-optimal solutions. Local search algorithms such as simulated annealing (Fleischer 1995) and threshold accepting (Dueck and Scheuer 1990) are a class of algorithms that offer a means to find reasonable solutions to a wide variety of NP-hard discrete optimization problems. The objective of these algorithms is to find the best possible solution using a limited amount of computing resources. A further challenge is to construct

algorithms that find near-optimal solutions for all instances of a particular problem, since the effectiveness of many algorithms tends to be problem-specific, as they exploit particular characteristics of problem instances (e.g., Lin and Kernighan (1973) for the traveling salesman problem). It is therefore important to assess the performance of algorithms and devise strategies to improve their effectiveness in solving NP-hard discrete optimization problems.

There are several results in the literature concerning the asymptotic performance of simulated annealing algorithms. For simulated annealing algorithms with an exponential acceptance probability function, Mitra et al. (1986) and Hajek (1988) develop conditions for three convergence properties: asymptotic independence of the starting conditions, convergence in distribution of the solutions generated, and convergence to a global optimum. Mitra et al. (1986) and Hajek (1988) also characterize the convergence rate for simulated annealing algorithms with an exponential acceptance probability function. Anily and Federgruen (1987) extends these results to simulated annealing algorithms with general acceptance probabilities by developing necessary and sufficient conditions for convergence. In addition, they provide conditions for the reachability of the set of global optima. Yao and Li (1991) and Yao (1995) also discuss simulated annealing algorithms with general acceptance probabilities, though their primary contribution is with respect to general neighborhood generation distributions. Schuur (1997) provides a description of acceptance functions ensuring the convergence of the associated simulated annealing algorithm to the set of global optima.

The current literature focuses mainly on asymptotic convergence properties of local search algorithms, however, there have been a limited number of results reported on finite-time performance measures for simulated annealing. Mitra et al. (1986) presents bounds for the objective function over a finite horizon. Chiang and Chow (1989) and Mazza (1992) investigate the statistical properties of the first visit time to a global optima which provides insight into the time-asymptotic properties of the local search algorithm as the outer loop counter, $k \rightarrow +\infty$. Cantoni and Cerf (1997) investigates optimizing a finite-horizon cooling schedule to maximize the number of visits to a global optimum after a finite number of

iterations. Desai (1999) focuses on finite-time performance of local search algorithms by incorporating size-asymptotic information supplied by certain eigenvalues associated with the transition matrix. Desai provides some quantitative and qualitative information about the performance of simulated annealing after a finite number of steps by observing the quality of solutions related to the number of steps that the algorithm has taken.

Srichander (1995) examines the finite-time performance of simulated annealing using spectral decomposition of matrices. Srichander (1995) shows that for the final solution of the simulated annealing algorithm to converge to the global minimum with probability one, an annealing schedule on the temperature is not necessary. Furthermore, Srichander (1995) shows that annealing schedules on the temperature produce an inefficient algorithm in that the number of function evaluations required to reach a global minimum is extremely large. Srichander (1995) presents a modified simulated annealing algorithm with an iterative schedule on the size of the neighborhood sets that leads to a more efficient algorithm. The performance of this algorithm on a real-world example is reported.

Fleischer and Jacobson (1999) presents an entropy measure of the Markov chain model for simulated annealing, and then uses this measure to compare several finite-time simulated annealing implementations, with different cooling schedules. They show that higher values for this entropy measure correspond to better finite-time performance, as measured by the likelihood of terminating in a globally optimal solution. Nolte and Schrader (2000) provides a proof of convergence for simulated annealing by applying results on rapidly mixing Markov chains. Nolte and Schrader (2000) uses this proof technique to obtain better theoretical bounds for the finite-time performance of simulated annealing than those previously reported in the literature. However, these theoretical bounds are not practical to compute.

The current literature on asymptotic convergence properties and finite-time performance measures focuses primarily on visiting a globally optimal solution. However, in practice, solutions with objective function values that are *close enough* to the objective function value of the globally optimal solution are often acceptable. Without loss of generality, unless

otherwise noted, assume that all discrete optimization problems are minimization problems. Define solutions that have objective function value no greater than β as β-*acceptable solutions*, where β denotes the maximum acceptable objective function value (necessarily greater than or equal to the objective function value of a globally minimal solution).

This dissertation analyzes the finite-time behavior of local search algorithms in reaching β-acceptable solutions. In particular, the generalized hill climbing (GHC) algorithm framework (Johnson 1996, Johnson and Jacobson 2001a, 2001b) is used to study the finite-time behavior of local search algorithms for which convergence cannot be guaranteed. The goal of this research is to provide guidelines on how to design effective implementation strategies for local search algorithms applied to intractable discrete optimization problems. In particular, the research focuses on determining reasonable termination conditions for non-convergent local search algorithms. Ideally, this dissertation answers the question: For a given local search algorithm, what is a "*reasonable*" amount of time to use to search for solutions that are "*good enough*"?

1.3 Research Questions

The goal of this research is to address the finite-time behavior of local search algorithms applied to intractable discrete optimization problems where convergence cannot be guaranteed, hence provides mathematical expressions that allow practitioners to compare various local search algorithms as well as to choose run strategies, including termination conditions.

Let $\mathbf{A}$ be a local search algorithm applied to an instance of an intractable discrete optimization problem. Without loss of generality, assume that the discrete optimization problem is a minimization problem. Define β as a value greater than or equal to the objective function value of the optimal solution for the instance of the intractable discrete optimization problem.

1. Given that algorithm **A** has executed a fixed number of iterations and a value for β, is there a mathematical expression for the probability that **A** has visited a solution with an objective function value $\leq \beta$?
2. Is there a mathematical expression for the expected number of iterations that algorithm **A** must execute to visit a solution with objective function value $\leq \beta$?
3. Can the expression from question 1 be used to formulate conditions for algorithm **A** to converge to a solution with objective function value $\leq \beta$?
4. How can the answers to questions 1 and 2 be used to model the behavior of algorithm **A**? In particular,
    a. after a fixed number of iterations, can the probability of algorithm **A** visiting a solution with a sufficiently small objective function value be estimated?
    b. can the probability of visiting a solution with objective function value close to the objective function value of a globally optimal solution be estimated?
    c. can the objective function value of the globally optimal solution be estimated without actually visiting such a solution?

Chapter 2:

Local Search Algorithms for Addressing Intractable Discrete Optimization Problems

Many heuristic algorithms have been developed in recent years to address intractable discrete optimization problems. Some of the most popular such algorithms fall into the category of local search algorithms. A brief description of several local search algorithms is presented in this chapter. A more in-depth discussion of local search algorithms can be found in Aarts and Lenstra (1997).

2.1 Simulated Annealing

Simulated annealing is a local search algorithm (meta-heuristic) capable of escaping from local optima. Its ease of implementation, convergence properties, and its use of hill climbing moves to escape local optima have made it a popular technique over the past two decades. It is typically used to address discrete and, to a lesser extent, continuous optimization problems. Eglese (1990), Fleischer (1995), Henderson et al. (2001), Koulamas et al. (1994), and Romeo and Sangiovanni-Vincentelli (1991) present surveys that provide a good overview of simulated annealing's theoretical development and domains of application. Aarts and Korst

(1989) and van Laarhoven and Aarts (1987) devote entire books to the subject. Aarts and Lenstra (1997) dedicates an entire chapter to simulated annealing in their book on local search algorithms for discrete optimization problems.

Simulated annealing is so named because of its analogy to the process of physical annealing with solids, in which a crystalline solid is heated and then allowed to cool very slowly until it achieves its most regular possible crystal lattice configuration (i.e., its minimum lattice energy state), and thus is free of crystal defects. If the cooling schedule is sufficiently slow, the final configuration results in a solid with superior structural integrity. Simulated annealing mimics this type of thermodynamic behavior in searching for global minima for discrete optimization problems.

At each iteration of a simulated annealing algorithm applied to a discrete optimization problem, the objective function values for two solutions (the current solution and a newly generated neighboring solution) are compared. Improving solutions are always accepted, while a fraction of non-improving (inferior) solutions are accepted in the hope of escaping local optima in search of global optima. The probability of accepting non-improving solutions depends on a temperature parameter, which is non-increasing with each iteration of the algorithm.

The key algorithmic feature of simulated annealing is that it provides a means to escape local optima by allowing *hill climbing* moves (i.e., moves to a solution that corresponds to a worse objective function value). As the temperature parameter is decreased to zero, hill climbing moves occur less frequently, and the solution distribution associated with the inhomogeneous Markov chain that models the behavior of the algorithm converges to a distribution in which all the probability is concentrated on the set of globally optimal solutions (provided that the algorithm is convergent; otherwise the algorithm will converge to a local optimum, which may or not be globally optimal).

To formally describe the specific features of a simulated annealing algorithm for discrete optimization problems, several definitions are needed. Let Ω be the *solution space* (i.e., the

set of all possible solutions). Let f: $\Omega \rightarrow$ R be an *objective function* defined on the solution space. The goal is to find a global minima, ω^*, (i.e., $\omega^* \in \Omega$ such that $f(\omega) \geq f(\omega^*)$ for all $\omega \in \Omega$). Define $\eta(\omega)$ to be the *neighborhood function* for $\omega \in \Omega$. Therefore, associated with every solution, $\omega \in \Omega$, are neighboring solutions, $\eta(\omega)$, that can be reached in a single iteration of a local search algorithm.

Simulated annealing starts with an initial solution $\omega \in \Omega$. A neighboring solution $\omega' \in \eta(\omega)$ is then generated (either randomly or using some pre-specified rule). Simulated annealing is based on the Metropolis acceptance criterion (Metropolis et al. 1953), which models how a thermodynamic system moves from its current solution (state) $\omega \in \Omega$ to a candidate solution $\omega' \in \eta(\omega)$, in which the energy content is being minimized. The candidate solution, ω', is accepted as the current solution based on the acceptance probability

$$P\{\text{Accept } \omega' \text{ as next solution}\} = \begin{cases} \exp[-(f(\omega') - f(\omega))/t_k] & \text{if } f(\omega') - f(\omega) > 0 \\ 1 & \text{if } f(\omega') - f(\omega) \leq 0 \end{cases} \quad (1)$$

Define t_k as the temperature parameter at iteration *k*, such that

$$t_k > 0 \text{ for all k and } \lim_{k \rightarrow \infty} t_k = 0. \quad (2)$$

This acceptance probability is the basic element of the search mechanism in simulated annealing. If the temperature is reduced sufficiently slowly, then the system can reach a steady state at each iteration k. Let $f(\omega_i)$ and $f(\omega_j)$ denote the energy levels (i.e., objective function values) associated with solutions $\omega_i \in \Omega$ and $\omega_j \in \eta(\omega_i)$, respectively. Let the random variable X denote the state of the steady state system. The steady state is characterized by the Boltzmann distribution which can be described as the probability of the system being in state $\omega_i \in \Omega$ with energy $f(\omega_i)$ at temperature T. Therefore,

$$P_T\{X = \omega_i\} = \frac{\exp(-f(\omega_i)/t_k)}{\sum_{j \in \Omega} \exp(-f(\omega_j)/t_k)}. \quad (3)$$

Define $g_{\omega_i,\omega_j}(k)$ to be the probability of generating a candidate solution ω_j from the neighbors of solution $\omega_i \in \Omega$, where

$$\sum_{\omega_j \in \eta(\omega_i)} g_{\omega_i,\omega_j}(k) = 1, \quad \text{for all } \omega_i \in \Omega,\ k = 1, 2, \ldots. \tag{4}$$

Therefore, a nonnegative square stochastic matrix **P**(k) can be defined with transition probabilities

$$P_{\omega_i,\omega_j}(k) = \begin{cases} g_{\omega_i,\omega_j}(k)\exp(-\delta_{i,j}/t_k) & \omega_j \in \eta(\omega_i),\ j \neq i \\ 0 & \omega_j \notin \eta(\omega_i),\ j \neq i \\ 1 - \sum_{\substack{j \in \eta(\omega_i) \\ j \neq i}} P_{\omega_i,\omega_j}(k) & j = i \end{cases} \tag{5}$$

for all solutions $\omega_i \in \Omega$ and all iterations k = 1,2,… and $\delta_{i,j} \equiv f(\omega_j) - f(\omega_i)$. These transition probabilities define a sequence of solutions generated from an inhomogeneous Markov chain (Romeo and Sangiovanni-Vincentelli 1991).

The hill climbing strategy inherent in simulated annealing inspired the formulation of similar algorithms like threshold accepting (Dueck and Scheuer 1990, Moscato and Fontanari 1990) and the noising method (Charon and Hudry 1993). Moreover, though different in how they traverse the solution space, both tabu search and genetic algorithms share with simulated annealing the objective of using local information to find global optima over solution spaces contaminated with multiple local optima.

2.2 Threshold Accepting

Questioning the very need for a randomized acceptance function, Dueck and Scheuer (1990) and, independently, Moscato and Fontanari (1990) proposed the concept of threshold accepting, where the acceptance function is defined as

$$a_{i,j}(k, \delta_{i,j}) = \begin{cases} 1 & \text{if } t_k \geq \delta_{i,j} \\ 0 & \text{otherwise} \end{cases}$$

with t_k defined as the threshold value at iteration k and $\delta_{i,j} \equiv f(\omega_j) - f(\omega_i)$. The threshold value t_k is typically set to be a deterministic, non-increasing step function in k. Dueck and Scheuer (1990) reports computational results that suggest dramatic improvements in traveling salesman problem solution quality and algorithm run-time over basic simulated annealing. Moscato and Fontanari (1990) reports more conservative results that suggest simulated annealing's probabilistic acceptance function does not play a major role in the search for near-optimal solutions.

Althofer and Koschnick (1991) develops a convergence theory for threshold accepting based on the concept that simulated annealing belongs to the convex hull of threshold accepting. Althofer and Koschnick (1991) notes that (for a finite threshold sequence $\{t_k\}$) there can exist only finitely many threshold accepting transition matrices, while simulated annealing can have infinitely many transition matrices because its acceptance function is a random variable. However, every simulated annealing transition matrix for a given problem can be represented as a convex combination of the finitely many threshold accepting transition matrices. Althofer and Koschnick (1991) is unable to provide a proof that threshold accepting will asymptotically reach a global minimum, but are able to establish the existence of threshold schedules that provide convergence within an ε-neighborhood of the optimal solutions. Jacobson and Yücesan (2001) proves that if the threshold value approaches zero as k approaches infinity, then the algorithm does not converge with probability one to the set of globally optimal solutions.

Hu et al. (1995) modifies threshold accepting to include a non-monotonic, self-tuning threshold schedule in the hope of improving the algorithm's finite-time performance. Hu et al. (1995) allows the threshold t_k to change dynamically (either up or down), based on the perceived likelihood of being near a local minimum. These changes are accomplished using a principle called *dwindling expectation*--when the algorithm fails to move to neighboring solutions, the threshold t_k is gradually increased, in the hope of eventually escaping a local optimum. Conversely, when solution transitions are successful, the threshold is reduced, in order to explore local optima. The experimental results based on two traveling salesman

problems presented in Hu et al. (1995) shows that the proposed algorithm outperformed other hill climbing methods in terms of finding good solutions earlier in the optimization process.

The advantage of threshold accepting over simulated annealing lies in its ease of implementation and its generally faster execution time, due to the reduced computational effort in avoiding acceptance probability computations and generation of random numbers (Moscato and Fontanari 1990). However, compared to simulated annealing, few threshold accepting applications are reported in the literature (Lin et al. 1995, Scheermesser and Bryngdahl 1995, and Nissen and Paul 1995).

2.3 Noising Method

Charon and Hudry (1993) proposes a simple descent algorithm called the *noising method* as an alternative to simulate annealing. The algorithm first perturbs the solution space by adding random noise to the problem's objective function values. The noise is gradually reduced to zero during the algorithm's execution, allowing the original problem structure to reappear. Charon and Hudry (1993) provides computational results, but does not prove that the algorithm will asymptotically converge to the set of globally optimal solutions.

Storer et al. (1992) proposes an optimization strategy for sequencing problems by integrating fast, problem-specific heuristics with pure local search. Its key contribution is to base the definition of the search neighborhood on a heuristic problem pair (h, p), where h is a fast, known, problem-specific heuristic and p represents the problem data. By perturbing the heuristic, the problem, or both, a neighborhood of solutions is developed. This neighborhood then forms the basis for pure local search. The hope is that the perturbations will cluster good solutions close together, thus making it easier to perform pure local search.

2.4 Tabu Search

Tabu search (Glover 1989) is a general framework for a variety of iterative local search strategies for addressing discrete optimization problems. Tabu search uses the concept of *memory* by controlling the algorithm's execution via a dynamic list of forbidden moves. This allows the tabu search algorithm to intensify or diversify its search of a given problem's

solution space in an effort to avoid entrapment in local optima. Note that a criticism of simulated annealing is that it is completely memoryless (i.e., simulated annealing disregards all historical information gathered during the algorithm's execution). However, no proofs of convergence are reported in the literature for the general tabu search algorithm.

Faigle and Kern (1992) proposes a particular tabu search strategy called *probabilistic tabu search* (Glover 1989) as a meta-heuristic to help guide simulated annealing. Probabilistic tabu search attempts to capitalize on both the asymptotic convergence properties of simulated annealing and the memory feature of tabu search. In probabilistic tabu search, the probabilities of generating and accepting each candidate solution are set as functions of both a temperature parameter (as in simulated annealing) and information gained in previous iterations (as for a tabu search strategy). Faigle and Kern (1992) proves asymptotic convergence of a particular tabu search algorithm by using methods developed for simulated annealing (Faigle and Kern 1991, Faigle and Schraeder 1988). Note that the results of Faigle and Kern (1992) build upon Glover (1989) where probabilistic tabu search was first introduced and contrasted with simulated annealing.

2.5 Genetic Algorithms

Genetic algorithms (Liepins and Hilliard 1989) emulate the evolutionary behavior of biological systems. They generate a sequence of populations of candidate solutions for the underlying optimization problem by using a set of genetically inspired stochastic solution transition operators to transform each population of candidate solutions into a descendent population. The three most popular transition operators are reproduction, crossover, and mutation (Davis 1991). Davis (1991) and Rudolph (1994) attempt to use homogeneous finite Markov chain techniques to prove convergence of genetic algorithms (Cerf 1998), but are unable to develop a theory comparable in scope to that of simulated annealing.

Mühlenbein (1997) presents a theoretical analysis of genetic algorithms based on population genetics. Mühlenbein (1997) counters the popular notion that models that mimic natural phenomenon are superior to other models. The article argues that evolutionary algorithms can be inspired by nature, but do not necessarily have to copy a natural phenomenon.

Mühlenbein (1997) addresses the behavior of transition operators and designs new genetic operators that are not necessarily related to events in nature, yet still perform well in practice.

One criticism of simulated annealing is the slow speed at which it sometimes converges. Delport (1998) combines simulated annealing with evolutionary algorithms to improve performance in terms of speed and quality of solution. Delport (1998) improves this hybrid system of simulated annealing and evolutionary selection by improving the cooling schedule based on fast recognition of the thermal equilibrium in terms of selection intensity. This improved hybrid technique results in a much faster convergence of the algorithm.

Sullivan and Jacobson (2000) links genetic algorithms with simulated annealing using the generalized hill climbing algorithm framework (Jacobson et al. 1998). Sullivan and Jacobson (2000) first links genetic algorithms to ordinal hill climbing algorithms, which can then be used, through its formulation within the generalized hill climbing algorithm framework, to form a bridge with simulated annealing. Though genetic algorithms have proven to be effective for addressing intractable discrete optimization problems, and can be classified as a type of hill climbing approach, its link with generalized hill climbing algorithms (through the ordinal hill climbing formulation) provides a means to establish well-defined relationships with other generalized hill climbing algorithms (like simulated annealing and threshold accepting). Sullivan and Jacobson (2000) also presents two genetic algorithm formulations that provide a first step toward developing bridges between genetic algorithms and other local search strategies like simulated annealing.

Chapter 3:

Background and Definitions

The first section of this chapter contains definitions needed to model local search algorithms using the generalized hill climbing (GHC) algorithm framework (Johnson 1996, Johnson and Jacobson 2001a, 2001b). Current convergent and finite-time performance results for GHC algorithms found in Johnson and Jacobson (2001b), Sullivan and Jacobson (2001), and Jacobson and Yücesan (2001a,b) are described in Section 3.2.

3.1 Generalized Hill Climbing Algorithm Definitions and Notation

The GHC algorithm framework provides a structure for using local search algorithms to address intractable discrete optimization problems. The GHC algorithm framework contains many local search algorithms that seek to find optimal solutions for discrete optimization problems by allowing the local search algorithm to visit inferior solutions enroute to an optimal/near optimal solution. Figure 1 depicts the GHC pseudo-code.

Recall the following definitions from Chapter 2. For a discrete optimization problem, the solution space, Ω, is a finite set. A real-valued objective function $f: \Omega \rightarrow [0,+\infty]$ assigns a value to each element of the solution space. A neighborhood function, $\eta:\Omega \rightarrow 2^{\Omega}$, where $\eta(\omega) \subseteq \Omega$ for all $\omega \in \Omega$, provides the connection between the elements in the solution space Ω. Assume that the neighbors are generated uniformly at each iteration of a GHC algorithm

(i.e., P{ω' is selected as the neighbor of ω at a given iteration of a GHC algorithm} = $1/|\eta(\omega)|$). The initial solution ω(0) can either be selected or randomly generated from the set of solutions in Ω.

Figure 1: Generalized Hill Climbing Algorithm Pseudo-code

Define a neighborhood function $\eta:\Omega \rightarrow 2^{\Omega}$
Set the iteration counter bound K
Define a set of hill climbing (random) variables $R_k(\omega(k-1),\omega) : \Omega\times\Omega\rightarrow\Re \cup \{-\infty,+\infty\}$, k=1,2,…K
Set the iteration indices k = 1
Select an initial solution $\omega(0) \in \Omega$
    Repeat while $k \leq K$
        Generate a solution $\omega\in\eta(\omega(k-1))$
        Calculate $\delta(\omega(k-1),\omega) = f(\omega)-f(\omega(k-1))$
        If $R_k(\omega(k-1),\omega) \geq \delta(\omega(k-1),\omega)$, then $\omega(k)\leftarrow\omega$
        If $R_k(\omega(k-1),\omega) < \delta(\omega(k-1),\omega)$, then $\omega(k)\leftarrow\omega(k-1)$
        $k \leftarrow k+1$
    Until k = K

This dissertation considers local search algorithms with hill climbing random variables, $R_k(\omega(k),\omega)$, that have finite means and finite variances for all k and for all possible pairs of elements in the solution space (i.e., $E[R_k(\omega(k),\omega)] < +\infty$ and $Var[R_k(\omega(k),\omega)] < +\infty$ for all $\omega(k) \in \Omega$, $\omega \in \eta(\omega(k))$, and for all k = 1, 2,…, K.

The neighborhood function establishes relationships between the solutions in the solution space, hence allows the solution space to be traversed or searched by moving between solutions. To ensure that the solution space is connected, assume that all the solutions in the solution space (with neighborhood function η) are *reachable* (i.e., for all $\omega',\omega''\in\Omega$, there exists a set of solutions $\omega_{i_1},\omega_{i_2},...,\omega_{i_m} \in \Omega$ such that $\omega_{i_r} \in \eta(\omega_{i_{r-1}})$, r = 1, 2, ..., m+1, where ω' ≡ ω_{i_0} and ω'' ≡ $\omega_{i_{m+1}}$). Note that if all solutions in a solution space are reachable, then the solution space (with neighborhood function η) is said to be reachable.

The objective function, f, and the neighborhood function, η, allow the solution space, Ω, to be decomposed into three mutually exclusive and exhaustive sets (Johnson and Jacobson 2001b):

1. the set of *globally optimal solutions*, $G = \{\omega^* \in \Omega: f(\omega^*) \leq f(\omega) \text{ for all } \omega \in \Omega\}$,
2. the set of *locally optimal solutions*, $L = L(\eta) = \{\omega' \in \Omega: f(\omega') \leq f(\omega) \text{ for all } \omega \in \eta(\omega')\}$, (i.e., local optima that are not global optima),
3. the set of *all other* solutions, H.

Identifying a globally optimal solution for an intractable discrete optimization problem is often cost prohibitive. Therefore, solutions that are *within a predetermined threshold* for an intractable discrete optimization problem are often *acceptable* in practice. To describe solutions that are within such a threshold, define D_β as the set of β-*acceptable solutions* where $f(\omega^*) \leq \beta < \max_{\omega \in L} f(\omega)$, $\omega^* \in G$,

$$D_\beta = \{\omega \in \Omega: f(\omega) \leq \beta\}$$

Note that if $\beta < f(\omega^*)$, $\omega^* \in G$, then $D_\beta = \varnothing$.

Moreover,

$$\lim_{\beta \to f(\omega^*)^+} D_\beta = G, \text{ the set of globally optimal solutions.}$$

Each execution of a local search algorithm generates a sequence (sample) of $K \in Z^+$ solutions. In practice, the best solution visited over the entire local search algorithm run, not just the final solution, is reported. This allows the algorithm to aggressively traverse the solution space visiting many inferior solutions *en route* to a β-acceptable solution, while retaining the best solution visited throughout the entire run. Without loss of generality, assume that the algorithm run is initialized (either deterministically or stochastically) at a solution not in D_β (i.e., $P\{\omega(0) \in D_\beta^c\} = 1$). Each sequence of solutions is a function of the

initial solution and the two sets of independent and identically distributed U(0,1) random variables,

1. $\{\xi_k\}$, that generate the neighbors, $\omega \in \eta(\omega(k))$ at each iteration k = 1,2, …,
2. $\{\nu_k\}$, that generate values for hill climbing random variable $R_k(\omega(k),\omega)$, when $\delta(\omega(k),\omega) > 0$, to determine whether the generated neighbor is accepted or rejected (i.e., $R_k(\omega(k),\omega) \geq \delta(\omega(k),\omega)$ or $R_k(\omega(k),\omega) < \delta(\omega(k),\omega)$, respectively).

Note that the random variables $\{\xi_k\}$ and $\{\nu_k\}$ are independent of each other. Assume that when comparing different local search algorithms, the same initial solution is used for each algorithm. Therefore, $(\{\xi_k\},\{\nu_k\})$ completely define the probability of each sequence of K solutions generated by the particular formulation of the algorithm. More specifically, let $\Sigma \equiv \Sigma_{\omega(0)}$ denote all possible sequences of K solutions generated by a local search algorithm, where each of the K solutions is in Ω, $\Im$ denotes the sigma field of events on Σ, and $P \equiv P_{\xi,\nu}$ denote the probability measure. Therefore, $(\Sigma, \Im, P)$ defines a probability space on the sequences of K solutions generated by a local search algorithm, where this probability space is characterized by the initial solution, $\omega(0)$, and the set of independent and identically distributed U(0,1) random variables, $(\{\xi_k\},\{\nu_k\})$, that determine the sequence of K solutions generated by a local search algorithm. For simplicity and ease of notation, $\omega(0)$ and $(\{\xi_k\},\{\nu_k\})$ will be suppressed, unless they are needed to avoid ambiguities.

The iterations of a local search algorithm within the GHC algorithm framework can be modeled as an inhomogeneous (i.e., nonstationary) discrete-time Markov chain with a $|\Omega| \times |\Omega|$ (one step) transition matrix. Since the neighbors are generated uniformly, a nonnegative square stochastic matrix $\mathbf{P}(k)$ can be defined with transition probabilities

$$p_{j,l}(k) = \begin{cases} \dfrac{P\{R_k(\omega_j,\omega_l) \geq \delta(\omega_j,\omega_l)\}}{\left|\eta(\omega_j)\right|} & \omega_l \in \eta(\omega_j),\ l \neq j \\ 0 & \omega_l \notin \eta(\omega_j),\ l \neq j \\ 1 - \sum\limits_{\substack{\omega_l \in \eta(\omega_j) \\ l \neq j}} p_{j,l}(k) & l = j \end{cases}$$

for all solutions $\omega_j \in \Omega$ and iterations k = 1, 2,…, K. Section 4.1 uses this transition matrix to obtain closed form expressions for the β-acceptable solution probability as well as the expected number of iterations to reach a β-acceptable solution. The β-acceptable solution probability and the expected number of iterations to reach a β-acceptable solution are then used to assess the effectiveness of local search algorithms and to determine when to terminate their execution.

3.2 Performance Results for Generalized Hill Climbing Algorithms

Asymptotic convergence conditions for GHC algorithms can be used to obtain a theoretical understanding of the performance of local search algorithms. There is a large body of literature on the convergence properties and convergence conditions for simulated annealing (e.g., Hajek 1988, Mitra et al. 1986, Schuur 1997). To date, there are no results that establish the convergence of general tabu search. The convergence theory for simulated annealing depends heavily on its exponential hill climbing random variable, and the Markov chain structure inherent in the algorithm's design (i.e., reversibility or detailed balance). Johnson and Jacobson (2001a) exploits this structure by providing sufficient convergence conditions for a class of GHC algorithms. In particular, if the acceptance probability $P\{R_k(\omega(k),\omega') \geq f(\omega') - f(\omega(k))\}$ can be written as

$$\text{Min}[1, P\{\check{R}_k(\omega^*, \omega') \geq f(\omega') - f(\omega^*)\} / P\{\check{R}_k(\omega^*, \omega(k)) \geq f(\omega(k)) - f(\omega^*)\}], \quad (6)$$

where ω^* is a globally optimal solution, and $\check{R}_k$ are non-negative (random) variables, then the resulting GHC algorithm converges with probability one to the set of globally optimal solutions. Johnson and Jacobson (2001a) identifies several GHC algorithms (including simulated annealing) that satisfy (6). They also give a general expression that defines the acceptance probability in terms of the distribution functions for $\check{R}_k(\omega^*,\omega(k))$ and $\check{R}_k(\omega^*,\omega')$. The major limitation of their approach is that the globally optimal objective function value must be known *a priori*. For practical implementations, knowing a bound for this value is sufficient to effectively implement these convergent GHC algorithms.

Johnson and Jacobson (2001b) circumvents the restrictions imposed by Markov chain theory to develop general *sufficient* convergence conditions for GHC algorithms. Their conditions are based on a path argument that evaluates how difficult it is to traverse the solution space specifically between global and local optima. To describe these results, recall that the objective function, f, together with the neighborhood function, η, allow the solution space, Ω, to be decomposed into the three mutually exclusive and exhaustive sets (G, L, H) defined in Section 3.1, with $\Omega = G \cup L \cup H$, $G \cap L = \varnothing$, $G \cap H = \varnothing$, and $L \cap H = \varnothing$.

Johnson and Jacobson (2001b) classifies and group the iterations of a GHC algorithm based on the type of solutions visited. A *micro iteration* moves the current solution either to an immediate neighbor or back to itself. A *macro iteration* is a set of micro iterations that moves the algorithm from any element of $L \cup G$ to any element of $L \cup G$ (including itself), passing only through elements of H. The micro iterations between macro iterations define the *paths* between elements of $L \cup G$.

Using this structure, Johnson and Jacobson (2001b) obtains the following sufficient convergence condition.

THEOREM 1: For a given instance of a discrete optimization problem, with solution space Ω, objective function f, and neighborhood function η, and a particular GHC algorithm (with hill climbing random variable R_k), then under mild restrictions on the neighborhood function (see Johnson and Jacobson 2001b), if

i) $\sum_{k=1}^{+\infty} P_k(\text{Min\_Path}) = +\infty$,

ii) $\sum_{k=1}^{+\infty} P_k(\text{Max\_Path}) < +\infty$,

iii) $\sum_{k=1}^{+\infty} P_k\left(\text{Max\_Prod}\right) < +\infty$,

where $P_k(\text{Min\_Path}) \equiv \min\{P_k(j \to i) \mid j \in L,\ i \in L \cup G,\ P_k(j \to i) > 0\}$, $P_k(\text{Max\_Path}) \equiv \max\{P_k(i \to j) \mid i \in G,\ j \in L\}$, $P_k(\text{Max\_Prod}) \equiv \max\{\gamma_j(k) P_k(j \to q) \mid j,q \in L,\ q \neq j\}$, $P_k(j \to i)$ is the probability of moving from solution $j \in L \cup G$ to solution $i \in L \cup G$ along a path, k = 1, 2,…, and $\gamma_j(k)$ is the long run probability of being at solution j at macro iteration k, then the GHC algorithms converges with probability one to elements in G provided $R_k \to 0$ as $k \to +\infty$ (i.e., $\Sigma_{j \in G}\, \gamma_j(k) \to 1$ as $k \to +\infty$).

Johnson and Jacobson (2001b) relates the three conditions in Theorem 1 to properties of R_k (such as the rate at which R_k approaches zero) to provide convergent GHC algorithm formulations. One limitation of this result is that the sufficient conditions are based solely on the rates (how quickly or how slowly) at which the three different path probabilities approach zero as the number of iterations approaches infinity, without taking into account any interactions between the paths. For example, these conditions may be violated, yet the *relative rates* at which these probabilities approach zero may be sufficient to guarantee convergence.

Sullivan and Jacobson (2001) exploits this observation by developing necessary and sufficient convergence conditions based on ratios of probabilities. To obtain their results, they define a homogeneous discrete time Markov chain, with *micro iteration transition matrix*

$$P_m^k = \begin{bmatrix} P_m^k(G,G) & P_m^k(G,L) & P_m^k(G,H) \\ P_m^k(L,G) & P_m^k(L,L) & P_m^k(L,H) \\ P_m^k(H,G) & P_m^k(H,L) & P_m^k(H,H) \end{bmatrix}$$

at macro iteration k, where $P_m^k(U,V)$, $U,V \in \{G,L,H\}$ are $|U| \times |V|$ matrices representing the micro iteration transition probabilities from the elements in set U to the elements in set V.

Note that if the micro iteration transition matrix is aperiodic and irreducible, then at macro iteration k, the macro iterations define an inhomogeneous Markov chain, with *macro iteration transition matrix*

$$P_M^k = \begin{bmatrix} P_m^k(G,H)(I-P_m^k(H,H))^{-1}P_m^k(H,G)+P_m^k(G,G) & P_m^k(G,H)(I-P_m^k(G,H))^{-1}P_m^k(H,L)+P_m^k(G,L) \\ P_m^k(L,H)(I-P_m^k(H,H))^{-1}P_m^k(H,G)+P_m^k(L,G) & P_m^k(L,H)(I-P_m^k(H,H))^{-1}P_m^k(H,L)+P_m^k(L,L) \end{bmatrix}$$

To obtain the necessary and sufficient convergence conditions for a GHC algorithm to converge, define π_ω^k to be the long run stationary distribution of a GHC algorithm being at solution $\omega \in G \cup L$ at macro iteration k. The following lemma provides upper and lower bounds on the sum of the long run stationary distribution of being at a solution in L at macro iteration k.

LEMMA 1 (Sullivan and Jacobson 2002): For a GHC algorithm at macro iteration k (fixed), with macro iteration transition matrix P_M^k,

$$\min_{\omega\in G} \sum_{\sigma\in L} P_M^k(\omega,\omega') / [\min_{\omega\in G} \sum_{\sigma\in L} P_M^k(\omega,\omega') + \max_{\sigma\in L} \sum_{\omega\in G} P_M^k(\omega',\omega)] \le \sum_{\omega\in L} \pi_\omega^k$$

and

$$\sum_{\omega\in L} \pi_\omega^k \le \max_{\omega\in G} \sum_{\sigma\in L} P_M^k(\omega,\omega') / [\max_{\omega\in G} \sum_{\sigma\in L} P_M^k(\omega,\omega') + \min_{\sigma\in L} \sum_{\omega\in G} P_M^k(\omega',\omega)]$$

These bounds are used to obtain the necessary and sufficient convergence conditions for a GHC algorithm to converge with probability one to G.

THEOREM 2: (Sullivan and Jacobson 2002) For a GHC algorithm at macro iteration k (fixed), with macro iteration transition matrix P_M^k,

(i) (Necessary Condition) If $\pi_\omega^k \to 0$ as $k \to +\infty$, for all $\omega \in L$, then

$$[\min_{\omega\in G} \sum_{\sigma\in L} P_M^k(\omega,\omega')] / [\max_{\sigma\in L} \sum_{\omega\in G} P_M^k(\omega',\omega)] \to 0 \text{ as } k \to +\infty,$$

(ii) (Sufficient Condition) If $[\max_{\omega\in G} \sum_{\sigma\in L} P_M^k(\omega,\omega')] / [\max_{\omega\in G} \sum_{\sigma\in L} P_M^k(\omega,\omega') +$

$\min_{\sigma\in L} \sum_{\omega\in G} P_M^k(\omega',\omega)] \to 0$ as $k \to +\infty$, then $\pi_\omega^k \to 0$ as $k \to +\infty$, for all $\omega \in L$.

Theorem 2 uses the macro iteration structure and the macro iteration transition matrix to obtain conditions that can be used to determine when a GHC algorithm does or does not converge.

Convergence results provide theoretical support for applying local search algorithms to intractable discrete optimization problems. Practitioners, however, are interested in an algorithm's *finite-time* performance. The finite-time performance of GHC algorithms provides useful information into how the hill climbing (random) variables $R_k(\omega(i),\omega')$ should be selected for classes of discrete optimization problems when a limited computing budget is available.

Jacobson and Yucesan (2001a) addresses both the asymptotic and finite-time behavior of GHC algorithms for which convergence cannot be guaranteed. They use the concepts of micro and macro iterations to obtain necessary (asymptotic) convergence condition for GHC algorithms. They also introduce the *global visit probability* and the *false negative probability* as two new finite-time performance measures for GHC algorithms. Jacobson and Yucesan (2001b) uses these performance measures to provide a sufficient (asymptotic) convergence condition for GHC algorithms. They also use these measures to evaluate and compare GHC algorithms with random restart local search. Such a comparison provides insights into both the asymptotic and the finite-time performance of GHC algorithms. This work provides guidelines on when random restart local search is an effective alternative to GHC algorithms that are either convergent or nonconvergent.

Chapter 4:

The β - Acceptable Solution

Identifying a globally optimal solution for an intractable discrete optimization problem is often cost prohibitive. Therefore, solutions that are *within a predetermined threshold* for an intractable discrete optimization problem are often *acceptable* in practice. Recall from Section 3.1 that D_β is the set of β-acceptable solutions, where β denotes the maximum acceptable objective function value (necessarily larger than the global minimum objective function value). Assume that the local search algorithm run is initialized at a solution not in D_β (i.e., $P\{\omega(0) \in D_\beta^c\} = 1$).

Section 4.1 introduces expressions for the β-acceptable solution probability as a measure of effectiveness for local search algorithms. Section 4.2 addresses asymptotic convergence to a β-acceptable solution in terms of the β-acceptable solution probability. Section 4.3 presents expressions for the expected number of iterations required to reach a β-acceptable solution, $\omega \in D_\beta$. The β-acceptable solution probability and the expected number of iterations to reach a β-acceptable solution are then used to assess the effectiveness of a local search algorithm and to determine conditions for terminating the algorithm's execution. Properties of local search algorithms with different β-acceptable values are addressed in Section 4.4. The results from Sections 4.1 through 4.4 are applied to specific local search algorithms in Section 4.5.

4.1 The β-Acceptable Solution Probability

Define **A** to be a local search algorithm applied to an instance of a discrete optimization problem, where k = 1, 2,…, K represents the iterations executed by the algorithm. Assume that for **A**, $R_k(\omega(i),\omega) \geq 0$, $\omega(i) \in \Omega$, $\omega \in \eta(\omega(i))$, i = 1,2,…,K. At each iteration k, define the events on the probability space $(\Sigma, \Im, P)$,

$$D(k,\beta) = \{(\omega(1),\omega(2),\ldots,\omega(k)): \ f(\omega(i)) \leq \beta \text{ for some } \omega(i) \in \Omega,\ i = 1,2,\ldots,k\}$$
$$\equiv \{\mathbf{A} \text{ visits at least one element of } D_\beta \text{ over the first k iterations}\} \quad (7)$$

and

$$D(\beta) = \{(\omega(1),\omega(2),\ldots): \ f(\omega(i)) \leq \beta \text{ for some } \omega(i) \in \Omega,\ i = 1,2,\ldots\}$$
$$\equiv \{\mathbf{A} \text{ visits at least one element of } D_\beta\}. \quad (8)$$

These two events are distinct in that D(k,β) is over a finite number of iterations k, while D(β) assumes the algorithm does not terminate. The complementary events, $D^c(k,\beta)$ and $D^c(\beta)$, are defined as

$$D^c(k,\beta) = \{(\omega(1),\omega(2),\ldots,\omega(k)): \ f(\omega(i)) > \beta \text{ for all } \omega(i) \in \Omega,\ i = 1,2,\ldots,k\}$$
$$\equiv \{\mathbf{A} \text{ does not visit any element of } D_\beta \text{ over the first k iterations}\} \quad (9)$$

and

$$D^c(\beta) = \{(\omega(1),\omega(2),\ldots): \ f(\omega(i)) > \beta \text{ for all } \omega(i) \in \Omega,\ i = 1,2,\ldots\}$$
$$\equiv \{\mathbf{A} \text{ does not visit any element of } D_\beta\} \quad (10)$$

Assume that $P\{D^c(k,\beta)\} > 0$ for all iterations k = 1,2, … (i.e., finite-time convergence to a β-acceptable solution is not guaranteed). The definition of $D^c(k,\beta)$ in (9) establishes the relationship $D^c(k–1,\beta) \supseteq D^c(k,\beta)$ (i.e., the algorithm has yet to visit an element of D_β over the first k iterations) which implies $P\{D^c(k–1,\beta)\} \geq P\{D^c(k,\beta)\}$, for all iterations k = 1,2,… . Therefore, $\{D^c(k,\beta)\}$ is a telescoping, sequence of events in k, hence since the probability function is a continuous set function, then by the Monotone Convergence Theorem (Ross 1988, p.44),

$$P\{D^c(k,\beta)\} \rightarrow P\{D^c(\beta)\} \text{ as } k \rightarrow +\infty,$$

where

$$D^c(\beta) = \bigcap_{k=1}^{+\infty} D^c(k,\beta).$$

After k iterations, algorithm **A** yields k solutions, $\omega(1), \omega(2),\ldots, \omega(k)$, where $\omega(i) \in \Omega$ for all i = 1, 2,…, k. Define f^k to be the minimum objective function value among these k solutions and ω^k to be the associated solution (i.e., $f^k = f(\omega^k)$ with $\omega^k = \text{argmin}\{f(\omega(i)), i = 1, 2,\ldots,k\}$). In practice, the best solution to date (i.e., ω^k) is reported. If the goal is to find a solution within a β-acceptable threshold (i.e., $\omega \in \Omega$ such that $f(\omega) \le \beta$), then the key issue is whether $\omega^k \in D_\beta$. If $\omega^k \in D_\beta$ (i.e., $f(\omega^k) \le \beta$), then algorithm **A** would be terminated at (no later than) iteration k. The probability that ω^k is an element of D_β is given by $P\{D(k,\beta)\}$. If this probability is sufficiently close to one, then algorithm **A** may be terminated. Therefore, $P\{D(k,\beta)\}$ provides a quality measure for the solutions visited after k iterations.

In practice, if algorithm **A** has not found a β-acceptable solution by iteration k (i.e., $f(\omega^k) > \beta$), it may be desirable to determine whether algorithm **A** will, at some future iteration, visit a solution in D_β. If a β-acceptable solution is reachable at some future iteration, then it is useful to determine the number of additional iterations required to visit such a solution.

To establish the relationship between the asymptotic convergence of a local search algorithm and the event D(β), the following definition is needed.

DEFINITION 1: A local search algorithm **A** *converges with probability one* to an element of D_β if $P\{C(k,\beta)\} \to 1$ as $k \to +\infty$, where $C(k,\beta) = P\{f(\omega(k)) \le \beta, \omega(k) \in \Omega\} \equiv \{$**A** is at an element of D_β at iteration k$\}$.

Given a fixed initial solution ω(0), if algorithm **A** converges with probability one to D_β (as $k \to +\infty$), then $P\{D(\beta)\} = 1$. Equivalently, if $P\{D(\beta)\} < 1$, then algorithm **A** does not converge with probability one to D_β (i.e., for all $\varepsilon > 0$ there exists some iteration k_0 such that $P\{C(k,D_\beta)\} \le 1-\varepsilon$ for all $k \ge k_0$).

The *β-acceptable solution problem* asks whether a local search algorithm **A** will eventually visit an element of D_β, given that the algorithm, after executing a finite number of iterations, has yet to visit an element of D_β. This question can be quantified by considering the *β-acceptable solution probability* at iteration k, defined as

$$P\{D(\beta) \mid D^c(k,\beta)\} \text{ for all } k = 1,2,\ldots . \tag{11}$$

The β-acceptable solution probability at iteration k provides a measure for the effectiveness of a local search algorithm **A**, namely whether algorithm **A** can reach an element in D_β given that **A** has yet to visit an element at D_β in the first k iterations. In particular, if $P\{D(\beta) \mid D^c(k,\beta)\}$ is sufficiently close to zero, then one can use the β-acceptable solution probability to assess whether a local search algorithm will eventually visit an element of D_β, hence determine when the algorithm can be terminated.

4.2 Asymptotic Convergence to a β-Acceptable Solution

In this section, the β-acceptable solution probability is used to derive necessary conditions for a local search algorithm to converge asymptotically to a β-acceptable solution. The β-acceptable solution probability is also used to measure and compare the effectiveness of non-convergent (with probability one) local search algorithms.

Recall that $P\{\omega(0) \in D_\beta^c\} = 1$ (i.e., all local search algorithm runs are initialized at a solution not in D_β). Unless otherwise stated, assume that $P\{D(k,\beta)\} < 1$ for all finite iterations k (i.e., finite-time convergence cannot be guaranteed for the algorithm). For each iteration k = 1, 2,…, K, define the event

$$R(k,\beta) \equiv \{\textbf{A} \text{ visits an element of } D_\beta \text{ after k iterations, given that } \textbf{A} \text{ has not visited any element of } D_\beta \text{ after k–1 iterations}\}, \tag{12}$$

where

$$r(k,\beta) \equiv P\{R(k,\beta)\} = P\{D(k,\beta) \mid D^c(k-1,\beta)\} = P\{C(k,\beta) \mid D^c(k-1,\beta)\}. \tag{13}$$

This probability can be used to quantify the β-acceptable solution probability. Lemma 2 expresses the relationship between (13) and (9).

LEMMA 2: Consider a local search algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. Then

(i) $P\{D^c(k,\beta)\} = \prod_{i=1}^{k} [1 - r(i,\beta)]$ for all iterations k = 1,2,….

(ii) $P\{D^c(\beta)\} = \prod_{i=1}^{+\infty} [1 - r(i,\beta)]$.

Proof: By the definition of r(k,β) in (13),

$1 - r(k,\beta) = P\{D^c(k,\beta) \mid D^c(k-1,\beta)\} = P\{D^c(k,\beta) \cap D^c(k-1,\beta)\} / P\{D^c(k-1,\beta)\}$.

Since $D^c(k,\beta) \subseteq D^c(k-1,\beta)$, then $P\{D^c(k,\beta) \cap D^c(k-1,\beta)\} = P\{D^c(k,\beta)\}$.

Therefore,

$1 - r(k,\beta) = P\{D^c(k,\beta)\} / P\{D^c(k-1,\beta)\}$.

Since $P\{D^c(0,\beta)\} = 1$, then $P\{D^c(k,\beta)\} = \prod_{i=1}^{k} [1 - r(i,\beta)]$.

Taking the limit as $k \to +\infty$ establishes (ii). □

The *finite β-acceptable problem* asks whether a local search algorithm **A** will visit an element of D_β at iteration k_2, given that the algorithm, after executing a finite number of iterations k_1, has not visited an element of D_β, $k_2 > k_1$. This question can be quantified by considering the finite β-acceptable solution probability at iteration k_2.

For all iterations $k_2 > k_1$, define the *finite β-acceptable solution probability* as

$$P\{D(k_2,\beta) \mid D^c(k_1,\beta)\} \tag{14}$$

Lemma 3 provides an expression for the finite β-acceptable solution probability for all iterations $k_2 > k_1$.

LEMMA 3: Consider a local search algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. Then for all iterations $k_2 > k_1$,

$$P\{D(k_2,\beta) \mid D^c(k_1,\beta)\} = 1 - \prod_{i=k_1+1}^{k_2} [1 - r(i,\beta)].$$

Proof: $P\{D(k_2,\beta) \mid D^c(k_1,\beta)\} = 1 - P\{D^c(k_2,\beta) \mid D^c(k_1,\beta)\}$

$$= 1 - [P\{D^c(k_2,\beta) \cap D^c(k_1,\beta)\} / P\{D^c(k_1,\beta)\}]$$

$$= 1 - [P\{D^c(k_2,\beta)\} / P\{D^c(k_1,\beta)\}]$$

$$= 1 - [\prod_{i=1}^{k_2} [1 - r(i,\beta)] \; / \prod_{i=1}^{k_1} [1 - r(i,\beta)]]$$

$$= 1 - \prod_{i=k_1+1}^{k_2} [1 - r(i,\beta)] \; . \qquad \square$$

Theorem 3 provides an expression for the β-acceptable solution probability.

THEOREM 3: Consider a local search algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. Then for all iterations $k = 1, 2, \ldots,$

$$P\{D(\beta) \mid D^c(k,\beta)\} = 1 - \prod_{i=k+1}^{+\infty} [1 - r(i,\beta)].$$

Proof: Taking the limit as $k_2 \to +\infty$ in Lemma 3 establishes the result. $\square$

Corollary 1 shows that the β-acceptable solution probability is non-increasing in k.

COROLLARY 1: Consider a local search algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. Then for all iterations $k_2 \geq k_1$, $P\{D(\beta) \mid D^c(k_1,\beta)\} \geq P\{D(\beta) \mid D^c(k_2,\beta)\}$.

Proof: Since $\prod_{i=k+1}^{+\infty} [1 - r(i,\beta)] \leq \prod_{i=k+2}^{+\infty} [1 - r(i,\beta)]$ the results follow from Theorem 3. $\square$

Corollary 1 establishes that the probability of visiting any element of D_β never increases with each successive iteration that the algorithm has not visited any element of D_β. This result is used as a guideline for determining execution termination in Section 4.3. For example, consider two local search algorithms $\mathbf{A}_1$ and $\mathbf{A}_2$ that are not guaranteed to converge with probability one to D_β. If after k iterations, suppose that algorithm $\mathbf{A}_1$ has a smaller β-acceptable solution probability than algorithm $\mathbf{A}_2$, even though the best solution visited thus

far by algorithm $\mathbf{A}_1$ has lower objective function value than the best solution visited thus far by algorithm $\mathbf{A}_2$. If one of the algorithms must be terminated after k iterations, then algorithm $\mathbf{A}_2$ would be a likely candidate since its performance over k iterations has been inferior to that of algorithm $\mathbf{A}_1$. However, since the β-acceptable solution probability is a *measure of future performance*, then algorithm $\mathbf{A}_1$ should be terminated, because algorithm $\mathbf{A}_1$ is less likely than algorithm $\mathbf{A}_2$ to visit any element of D_β in subsequent iterations, given that neither algorithm has visited any element of D_β after k iterations. More intuitively, suppose that $\mathbf{A}_1$ and $\mathbf{A}_2$ are cars with the same fuel capacity. $\mathbf{A}_1$ is a sports car that gets 15 MPG while $\mathbf{A}_2$ is an economy car that gets 30 MPG. Both vehicles begin a 20 mile trip with one gallon of gas. If performance is measured at the 10 mile mark, $\mathbf{A}_1$ appears to perform better, yet $\mathbf{A}_2$ has the best *potential* for finishing the 20 mile trip.

Theorem 4 gives an expression for the probability that a local search algorithm has not visited any element of D_β (after k_1 iterations), given that it *eventually* did visit an element of D_β in $k_2 > k_1$ iterations (where k_2 could be infinite). This result provides practitioners with a marginal value measure for executing additional iterations.

THEOREM 4: Consider a local search algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. Then for all iterations $k_2 > k_1$, and $P\{D(k_2,\beta)\} > 0$,

$$P\{D^c(k_1,\beta) \mid D(k_2,\beta)\} = [\prod_{i=1}^{k_1} [1-r(i,\beta)] - \prod_{i=1}^{k_2} [1-r(i,\beta)]] /$$

$$[[\prod_{i=1}^{k_1} [1-r(i,\beta)] - \prod_{i=1}^{k_2} [1-r(i,\beta)]] + [1 - \prod_{i=1}^{k_1} [1-r(i,\beta)]]]$$

Proof: By the definition of conditional probability,

$$P\{D^c(k_1,\beta) \mid D(k_2,\beta)\} = P\{D^c(k_1,\beta) \cap D^c(k_2,\beta)\} / P\{D(k_2,\beta)\}.$$

Applying Bayes theorem,

$$P\{D^c(k_1,\beta) \mid D(k_2,\beta)\} = P\{D(k_2,\beta) \mid D^c(k_1,\beta)\} P\{D^c(k_1,\beta)\} /$$
$$P\{D(k_2,\beta) \mid D^c(k_1,\beta)\}P\{D^c(k_1,\beta)\}+P\{D(k_2,\beta) \mid D(k_1,\beta)\}\{D(k_1,\beta)\}$$

$$P\{D^c(k_1,\beta) \mid D(k_2,\beta)\} = P\{D(k_2,\beta) \mid D^c(k_1,\beta)\} P\{D^c(k_1,\beta)\} /$$
$$P\{D(k_2,\beta) \mid D^c(k_1,\beta)\}P\{D^c(k_1,\beta)\}+P\{D(k_1,\beta)\}.$$

From Lemma 3,

$$P\{D(k_2,\beta) \mid D^c(k_1,\beta)\} = 1 - \prod_{i=k_1+1}^{k_2} [1 - r(i,\beta)].$$

From Lemma 2,

$$P\{D(k_2,\beta)\} = 1 - \prod_{i=1}^{k_2} [1-r(i,\beta)]$$

and

$$P\{D(k_1,\beta)\} = 1 - \prod_{i=1}^{k_1} [1-r(i,\beta)].$$

Therefore,

$$P\{D^c(k_1,\beta) \mid D(k_2,\beta)\} = [[1 - \prod_{i=k_1+1}^{k_2} [1 - r(i,\beta)]] \prod_{i=1}^{k_1} [1-r(i,\beta)]] /$$

$$[[[1 - \prod_{i=k_1+1}^{k_2} [1-r(i,\beta)]] \prod_{i=1}^{k_1} [1-r(i,\beta)]] + [1 - \prod_{i=1}^{k_1} [1-r(i,\beta)]]]$$

$$= [\prod_{i=1}^{k_1} [1-r(i,\beta)] - \prod_{i=1}^{k_2} [1-r(i,\beta)]] /$$

$$[[\prod_{i=1}^{k_1} [1-r(i,\beta)] - \prod_{i=1}^{k_2} [1-r(i,\beta)]] + [1 - \prod_{i=1}^{k_1} [1-r(i,\beta)]]] \quad \square$$

Corollary 2 provides an expression for the probability that a local search algorithm has not visited any element of D_β (after k iterations) given that it will eventually visit an element of D_β. This provides practitioners with a means to assess finite-time performance of a local search algorithm that has a positive probability of reaching a solution in D_β (i.e., a convergent local search algorithm).

COROLLARY 2: Consider a local search algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. If $P\{D(\beta)\} > 0$ (i.e., there is a positive probability of reaching a solution in D_β) then for all iterations k = 1, 2,...,

$$P\{D^c(k,\beta) \mid D(\beta)\} = [\prod_{i=1}^{k} [1-r(i,\beta)] - \prod_{i=1}^{+\infty} [1-r(i,\beta)]] /$$

$$[[\prod_{i=1}^{k} [1-r(i,\beta)] - \prod_{i=1}^{+\infty} [1-r(i,\beta)]] + [1 - \prod_{i=1}^{k} [1-r(i,\beta)]]].$$

Proof: Letting $k = k_1$ and taking the limit as $k_2 \rightarrow +\infty$ in Theorem 4 establishes the result. □

Note that the $\lim_{k\rightarrow+\infty} P\{D^c(k,\beta) \mid D(\beta)\} = 0$. Therefore, for all $\varepsilon > 0$, there exists an iteration k_0, such that, for all iterations $k \geq k_0$, $P\{D^c(k,\beta) \mid D(\beta)\} \leq \varepsilon$. This can be used to terminate a local search algorithm once $P\{D^c(k,\beta) \mid D(\beta)\}$ is sufficiently small. Moreover, since the probability that an element of D_β has been visited over the first k_0 iterations given that an element of D_β is eventually visited is at least $1-\varepsilon$, then it is unlikely that a premature termination has occurred.

Theorem 5 provides upper and lower bounds for the finite β-acceptable solution probability.

THEOREM 5: Consider a local search algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. Then for all iterations $k_2 \geq k_1$, if $r(k,\beta) < 1$, then

$$1 - \exp\{-\sum_{i=k_1+1}^{k_2} r(i,\beta)\} \leq P\{D(k_2,\beta) \mid D^c(k_1,\beta)\} \leq 1 - \exp\{-\sum_{i=k_1+1}^{k_2} [r(i,\beta)] / [1-r(i,\beta)]\}.$$

Proof: From Lemma 3, for all iterations $k_2 \geq k_1$,

$$P\{D(k_2,\beta) \mid D^c(k_1,\beta)\} = 1 - \prod_{i=k_1+1}^{k_2} [1-r(i,\beta)] .$$

Therefore, $P\{D^c(k_2,\beta) \mid D^c(k_1,\beta)\} = \prod_{i=k_1+1}^{k_2} [1-r(i,\beta)]$, which implies that

$$\ln[P\{D^c(k_2,\beta) \mid D^c(k_1,\beta)\}] = \sum_{i=k_1+1}^{k_2} \ln [1-r(i,\beta)].$$

For $0 < r(i,\beta) < 1$, $i = k_1+1, k_1+2, \ldots, k_2$,

$$-r(i,\beta) / [1-r(i,\beta)] \leq \ln[1-r(i,\beta)] \leq -r(i,\beta).$$

Therefore,

$$-\sum_{i=k_1+1}^{k_2} [r(i,\beta)] / [1-r(i,\beta)] \le \ln[P\{D^c(k_2,\beta) \mid D^c(k_1,\beta)\}] \le -\sum_{i=k_1+1}^{k_2} r(i,\beta).$$

Taking the exponential function yields the result. □

Corollary 3 provides upper and lower bounds for the β-acceptable solution probability.

COROLLARY 3: Consider a local search algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. Then for all iterations k = 1, 2,...,

$$1 - \exp\{-\sum_{i=k+1}^{+\infty} r(i,\beta)\} \le P\{D(\beta) \mid D^c(k,\beta)\} \le 1 - \exp\{-\sum_{i=k+1}^{+\infty} [r(i,\beta)] / [1 - r(i,\beta)]\}.$$

Proof: Letting $k = k_1$ and Taking the limit as $k_2 \to +\infty$ in Theorem 5 establishes the result. □

Proposition 1 establishes the relationship between convergence with probability one to D_β and the β-acceptable solution probability.

PROPOSITION 1: If a local search algorithm **A** converges with probability one to D_β, then local search algorithm **A** *visits* D_β *with probability one* (i.e., $P\{D(\beta) \mid D^c(k,\beta)\} = 1$ for all iterations k = 1,2,…).

Proof: By the definition of conditional probability,

$$P\{D^c(\beta) \mid D^c(k,\beta)\} = P\{D^c(\beta) \cap D^c(k,\beta)\} / P\{D^c(k,\beta)\}$$
$$= P\{D^c(\beta)\} / P\{D^c(k,\beta)\} \text{ for all iterations } k = 1,2,\ldots.$$

Note that $C(j,\beta) \subseteq D(j,\beta)$, hence, $D^c(j,\beta) \subseteq C^c(j,\beta)$, for all iterations $j \ge k$, j = 1,2,…. Therefore, for all iterations k = 1,2,…,

$$P\{D^c(\beta)\} / P\{D^c(k,\beta)\} \le \lim_{j\to+\infty} P\{C^c(j,\beta)\} / P\{D^c(k,\beta)\} = 0.$$

Hence $P\{D(\beta) \mid D^c(k,\beta)\} = 1$ for all iterations k = 1,2,… . □

From Proposition 1, if a local search algorithm converges with probability one to D_β, then the local search algorithm **A** visits D_β with probability one. Proposition 2 provides necessary and sufficient conditions for the β-acceptable solution probability to be one for all iterations.

PROPOSITION 2: A local search algorithm **A** visits D_β with probability one if and only if $\sum_{i=k+1}^{+\infty} r(k,\beta) = +\infty$ for all iterations $k = 1,2,\ldots$.

Proof: If $\sum_{i=k+1}^{+\infty} r(i,\beta) = +\infty$ for all iterations k = 1, 2,..., then, from the lower bound of Corollary 3, $P\{D(\beta) \mid D^c(k,\beta)\} = 1$ for all iterations $k = 1,2,\ldots$. Conversely, if $P\{D(\beta) \mid D^c(k,\beta)\} = 1$ for all iterations k = 1, 2,..., then from the upper bound in Corollary 3, $\sum_{i=k+1}^{+\infty} r(i,\beta) / [1-r(i,\beta)] = +\infty$ for all iterations $k = 1,2,\ldots$. To show that $\sum_{i=1}^{+\infty} r(i,\beta) = +\infty$, by contradiction, if $\sum_{i=1}^{+\infty} r(i,\beta) < +\infty$, then for all $\varepsilon > 0$, there exists $k(\varepsilon) \in Z^+$ such that $r(i,\beta) \le \varepsilon$ for all $i \ge k(\varepsilon)$. Therefore,

$$r(i,\beta) / (1-r(i,\beta)) \le r(i,\beta) / (1-\varepsilon)$$

for all $i \ge k(\varepsilon)$, which implies that

$$\sum_{i=k(\varepsilon)}^{+\infty} r(i,\beta) / (1-r(i,\beta)) \le \sum_{i=k(\varepsilon)}^{+\infty} r(i,\beta) / (1-\varepsilon).$$

However, if the right hand side summation is finite, then the left hand side summation must also be finite, which contradicts that the left hand side summation diverges. □

Proposition 3 establishes the relationship between the β-acceptable solution probability and $P\{D(\beta)\}$. Recall that $f(\omega^*) \le \beta < \max_{\omega \in L} f(\omega)$, $\omega^* \in G$.

PROPOSITION 3: A local search algorithm **A** visits D_β with probability one if and only if $P\{D(\beta)\} = 1$.

Proof: The result follows from the law of total probability, since

$$P\{D(\beta)\} = P\{D(\beta) \mid D^c(k,\beta)\}\, P\{D^c(k,\beta)\} + P\{D(\beta) \mid D(k,\beta)\}\, P\{D(k,\beta)\}.$$ □

Theorem 6 summarizes the relationship between $P\{D(\beta)\}$, the β-acceptable probabilities, $r(k,\beta)$, and convergence with probability one to D_β.

THEOREM 6: For a local search algorithm **A** with a fixed β-acceptable value, where $f(\omega^*) \leq \beta < \max_{\omega \in L} f(\omega)$, $\omega^* \in G$, consider the expressions

(D1) $P\{C(k,\beta)\} \rightarrow 1$ as $k \rightarrow +\infty$ (converges with probability one to D_β).

(D2) $P\{D(\beta) \mid D^c(k,\beta)\} = 1$ for all iterations $k = 1,2,\ldots$ (visits D_β with probability one).

(D3) $P\{D(\beta)\} = 1$ (visits D_β with probability one).

(D4) $\sum_{i=k+1}^{+\infty} r(i,\beta) = +\infty$ for all iterations $k = 1,2,\ldots$.

Then, (D1) $\Rightarrow$ (D2) $\Leftrightarrow$ (D3) $\Leftrightarrow$ (D4).

Proof: Follows from Propositions 1, 2, and 3. □

Theorem 6 provides three necessary conditions for a local search algorithm's convergence (with probability one) to the set D_β. The only restriction on how the local search algorithm traverses the solution space is that $P\{D^c(k,\beta)\} > 0$ for all iterations $k = 1,2,\ldots$. This restriction means that the local search algorithm is never guaranteed (i.e., with probability one) to visit any element of D_β for k finite (i.e., finite convergence is not guaranteed since $P\{D(k,\beta)\} < 1$ for all iterations $k = 1,2,\ldots$). Note that From Lemma 2, if $P\{D^c(\beta)\} = \prod_{i=1}^{+\infty} [1-r(i,\beta)] > 0$, then, from Theorem 6, the local search algorithm does not converge with probability one to D_β.

Theorem 7 shows that for a local search algorithm with $\sum_{i=1}^{+\infty} r(i,\beta) < +\infty$, then with probability one, only a finite number of the events $R(i,\beta)$, $i = 1,2,\ldots$, occur simultaneously, or equivalently, with probability zero, the $R(i,\beta)$, $i = 1,2,\ldots$, occur infinitely often.

THEOREM 7: Suppose that for a local search algorithm **A**, $\sum_{i=1}^{+\infty} r(i,\beta) < +\infty$ (hence, the algorithm does not converge with probability one to D_β). Then $R(i,\beta)$, $i = 1,2,\ldots$, occur finitely often with probability one.

Proof: Define R' to be the event that infinitely many (over i) $R(i,\beta)$ occur. Then,

$R' \subset \bigcup_{i=k}^{+\infty} R(i,\beta)$, for all iterations k = 1,2,….

Hence,

$$P\{R'\} \le P\{\bigcup_{i=k}^{+\infty} R(i,\beta)\} \le \sum_{i=k}^{+\infty} r(i,\beta).$$

Taking the limit (as $k \to +\infty$) leads to $P\{R'\} \le 0$ since $\sum_{i=1}^{+\infty} r(i,\beta) < +\infty$. Therefore, $P\{R'\} = 0$, hence $P\{(R')^c\} = 1$. This means that, with probability one, only a finite number (over i) of the $R(i,\beta)$ occur. □

Theorem 7 shows that if $\sum_{i=1}^{+\infty} r(i,\beta) < +\infty$, hence the local search algorithm does not converge with probability one to D_β, then for all $\varepsilon > 0$, there exists a $k(\varepsilon) \in Z^+$ such that $P\{\bigcup_{i=k(\varepsilon)}^{+\infty} R(i,\beta)\} \le \sum_{i=k(\varepsilon)}^{+\infty} r(i,\beta) \le \varepsilon$. This means that for some non-convergent (with probability one) local search algorithms, the probability that the algorithm will visit an element of D_β for the first time (at or beyond iteration k = 1,2,…) can be made arbitrarily small if by setting the iteration k sufficiently large. This result can be used to define a stopping condition for a local search algorithm run. For example, if an improved solution has not been observed for some pre-specified number of iterations, it may be feasible to terminate the local search algorithm run.

4.3 Determination of Algorithm Run Lengths

The results developed in the previous sections suggest that a local search algorithm execution should be terminated at iteration k if the β-acceptable solution probability is sufficiently small. Such conditions are useful in that they save computational effort when the marginal value of each additional iteration (i.e., $P\{D(\beta) \mid D^c(k+1,\beta)\} - P\{D(\beta) \mid D^c(k,\beta)\}$) becomes negligible. On the other hand, one should not terminate an algorithm prematurely, that is, without giving the algorithm a reasonable opportunity to explore the solution space and attempt to visit an element of D_β.

This section develops an expression for the expected minimum number of iterations to move from an element in D_β^c to an element in D_β for the first time. This expression can be used to determine the minimum number of iterations needed for a local search algorithm to reach a β-acceptable solution.

To this end, consider the local search algorithm **A** defined in Section 4.1. Assume that the initial solution is not in D_β (i.e., $P\{\omega(0) \in D_\beta^c\} = 1$). Define the random variable

$$T_\beta \equiv \min \{i \geq 1 : f(\omega(i)) \leq \beta, \omega(i) \in \Omega\}.$$

The random variable T_β defines the number of iterations needed for algorithm **A** to visit an element of D_β for the first time. Then,

$$P\{T_\beta > k\} = P\{T_\beta \geq k+1\} = P\{\bigcap_{i=1}^{k} \{\omega(i) \in D_\beta^c\}\} = P\{D^c(k,\beta)\} = \prod_{i=1}^{k} [1-r(i,\beta)]. \qquad (15)$$

Moreover,

$$\begin{aligned} P\{T_\beta = k\} &= [P\{T_\beta > k-1\}] - [P\{T_\beta > k\}] \\ &= \prod_{i=1}^{k-1} [1 - r(i,\beta)] - \prod_{i=1}^{k} [1 - r(i,\beta)] \\ &= r(k,\beta) \prod_{i=1}^{k-1} [1 - r(i,\beta)] = r(k,\beta)\, P\{D^c(k-1,\beta\}. \end{aligned} \qquad (16)$$

Using (15),

$$\begin{aligned} E[T_\beta] &= \sum_{k=0}^{+\infty} P\{T_\beta > k\} = \sum_{k=0}^{+\infty} P\{D^c(k,\beta)\} \\ &= \sum_{k=0}^{+\infty} \prod_{i=1}^{k} [1-r(i,\beta)] = 1 + \sum_{k=1}^{+\infty} \prod_{i=1}^{k} [1-r(i,\beta)]. \end{aligned} \qquad (17)$$

Alternatively,

$$E[T_\beta] = \sum_{k=1}^{+\infty} k\, P\{T_\beta = k\} = \sum_{k=1}^{+\infty} k\, r(k,\beta)\, P\{D^c(k-1,\beta)\} = \sum_{k=1}^{+\infty} k\, r(k,\beta) \prod_{i=0}^{k-1} [1-r(i,\beta)].$$

Moreover, by the variance definition,

$$Var[T_\beta] = E[(T_\beta)^2] - E^2[T_\beta].$$

From (17),

$$E^2[T_\beta] = [1 + \sum_{k=1}^{+\infty} \prod_{i=1}^{k} [1-r(i,\beta)]]^2.$$

Assuming $E[(T_\beta)^3] < +\infty$, hence $\sum_{k=1}^{+\infty} k^2 [P\{T_\beta > k-1\} < +\infty$, then by definition,

$$E[(T_\beta)^2] = \sum_{k=1}^{+\infty} k^2 P\{T_\beta = k\}$$

$$= \sum_{k=1}^{+\infty} k^2 [P\{T_\beta > k-1\} - P\{T_\beta > k\}]$$

$$= \sum_{k=1}^{+\infty} k^2 P\{T_\beta > k-1\} - \sum_{k=1}^{+\infty} k^2 P\{T_\beta > k\}$$

$$= \sum_{k=0}^{+\infty} (k+1)^2 P\{T_\beta > k\} - \sum_{k=0}^{+\infty} k^2 P\{T_\beta > k\}$$

$$= \sum_{k=0}^{+\infty} [(k+1)^2 - k^2] P\{T_\beta > k\}$$

$$= \sum_{k=0}^{+\infty} (2k+1) P\{T_\beta > k\}.$$

From (15),

$$E[(T_\beta)^2] = \sum_{k=0}^{+\infty} (2k+1) P\{D^c(k,\beta\}$$

$$= \sum_{k=0}^{+\infty} (2k+1) \prod_{i=1}^{k} [1-r(i,\beta)]$$

$$=1 + \sum_{k=1}^{+\infty} [(2k+1) \prod_{i=1}^{k} [1-r(i,\beta)]].$$

Note that this expression is valid when $\sum_{k=1}^{+\infty} k^2 P\{T_\beta > k-1\} < +\infty$. Therefore,

$$\begin{aligned}
Var[T_\beta] &= 1 + \sum_{k=1}^{+\infty} [(2k+1)\prod_{i=1}^{k} [1-r(i,\beta)]] - [1 + \sum_{k=1}^{+\infty} \prod_{i=1}^{k} [1-r(i,\beta)]]^2 \\
&= 1 + \sum_{k=1}^{+\infty} [(2k+1)P\{D^c(k,\beta\}] - [1 + \sum_{k=1}^{+\infty} P\{D^c(k,\beta\}]^2 \\
&= 1 + \sum_{k=1}^{+\infty} [2kP\{D^c(k,\beta)\} + P\{D^c(k,\beta\}] - 1 - 2\sum_{k=1}^{+\infty} P\{D^c(k,\beta\} \\
&\quad - [\sum_{k=1}^{+\infty} P\{D^c(k,\beta\}]^2 \\
&= \sum_{k=1}^{+\infty} [2kP\{D^c(k,\beta)\} - P\{D^c(k,\beta\}] - [\sum_{k=1}^{+\infty} P\{D^c(k,\beta\}]^2 \\
&= \sum_{k=1}^{+\infty} [(2k-1)P\{D^c(k,\beta\}] - [\sum_{k=1}^{+\infty} P\{D^c(k,\beta\}]^2
\end{aligned}$$

This analysis establishes the following theorem.

THEOREM 8: Consider a local search algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. Then,

(a) $P\{T_\beta > k\} = P\{D^c(k,\beta\} = \prod_{i=1}^{k} [1 - r(i,\beta)]$

(b) $P\{T_\beta = k\} = r(k,\beta)] \prod_{i=1}^{k-1} [1 - r(i,\beta)]$

(c) $E[T_\beta] = 1 + \sum_{k=1}^{+\infty} \prod_{i=1}^{k} [1 - r(i,\beta)]$

(d) $Var[T_\beta] = \sum_{k=1}^{+\infty} [(2k-1)P\{D^c(k,\beta\}] - [\sum_{k=1}^{+\infty} P\{D^c(k,\beta\}]^2$.

Note that since T_β is a random variable that takes on only nonnegative values, then for any $k > 0$, Markov's inequality results in

$$P\{T_\beta \geq k\} \leq E[T_\beta] / k \leq [1 + \sum_{k=1}^{+\infty} \prod_{i=1}^{k} [1-r(i,\beta)]] / k .$$

Similarly, since T_β is a random variable with finite mean, μ, and variance, σ^2, then for any value $k > 0$, Chebyshev's inequality results in

$$P\{|T_\beta - \mu| \geq k\} \leq \sigma^2 / k^2$$

$$\leq [\sum_{k=1}^{+\infty} [(2k-1)P\{D^c(k,\beta\}] - [\sum_{k=1}^{+\infty} P\{D^c(k,\beta\}]^2] / k^2 .$$

These inequalities provide relationships between probabilities associated with T_β and the mean and variance of T_β.

4.4 Properties of Local Search Algorithms with Different β-Acceptable Values

The results presented in the previous sections focused on the convergence and finite-time performance with respect to β of local search algorithm **A** based on the total number of iterations executed. Such conditions are important when considering convergence and finite-time performance for a fixed β-acceptable value. However, multiple β-acceptable values are often encountered in practice. This section extends the results for a fixed value of β to cases with two different values of β. Recall that $P\{D^c(0,\beta)\} = 1$ (i.e., all local search algorithm runs are initialized at an element not in D_β). Furthermore, unless otherwise stated, assume that $P\{D(k,\beta)\} < 1$ for all iterations $k = 1,2,\ldots$.

Lemma 4 establishes the relationship between $P\{D(k,\beta_1)\}$ and $P\{D(k,\beta_2)\}$ when $\beta_2 > \beta_1$ for iterations $k = 1,2,\ldots$.

LEMMA 4: Consider a local search algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. Then for β-acceptable values, $\beta_2 > \beta_1$ and iterations $k = 1, 2,...,$

$$P\{D(k,\beta_1)\} \leq P\{D(k,\beta_2)\}.$$

Proof: The definition in (7) establishes the relationship,

$$D(k,\beta_1) \subseteq D(k,\beta_2)$$

and

$$D(k,\beta_2) = D(k,\beta_1) \cup [D^c(k,\beta_1) \cap D(k,\beta_2)].$$

Therefore,

$$P\{D(k,\beta_2)\} = P\{D(k,\beta_1)\} + P\{D^c(k,\beta_1) \cap D(k,\beta_2)\},$$

with

$$P\{D^c(k,\beta_1) \cap D(k,\beta_2)\} \geq 0.$$

Then,

$$P\{D(k,\beta_1)\} \leq P\{D(k,\beta_2)\}.$$ □

Lemma 5 establishes the relationship between $P\{D^c(k_1,\beta_1)\}$ and $P\{D^c(k_2,\beta_2)\}$ when $\beta_2 > \beta_1$ and iterations $k_1 \leq k_2$.

LEMMA 5: Consider a local search algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. Then for β-acceptable values $\beta_2 > \beta_1$, and iterations $k_1 \leq k_2$,

$$P\{D^c(k_2,\beta_2)\} \leq P\{D^c(k_1,\beta_1)\}.$$

Proof: The definition in (9) establishes the relationship,

$$D^c(k_2,\beta_2)\} \subseteq D^c(k_1,\beta_2)\} \subseteq D^c(k_1,\beta_1)\}$$

Therefore,

$$P\{D^c(k_1,\beta_2)\} \leq P\{D^c(k_1,\beta_1)\}.$$ □

Theorem 9 provides an expression for the finite β-acceptable solution probability, $P\{D(k,\beta_2) \mid D^c(k,\beta_1)\}$, $\beta_2 > \beta_1$. This is the probability that a local search algorithm will visit an element of D_{β_2} after k iterations, given that it has visited an element of D_{β_1}, $\beta_2 > \beta_1$, after k iterations, hence provides insights into the possibility that solutions D_{β_2} are available even though the algorithm has yet to visit a solution in D_{β_1} after k iterations.

THEOREM 9: Consider a local search algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. Then

(i) for all iterations k = 1, 2,..., if $\beta_2 > \beta_1$ and $r(i,\beta) < 1$ for all i = 1,2,…,k, then

$$P\{D(k,\beta_2) \mid D^c(k,\beta_1)\} = 1 - [\prod_{i=1}^{k} [1-r(i,\beta_2)] \; / \; \prod_{i=1}^{k} [1-r(i,\beta_1)]],$$

(ii) for iterations k = 1, 2,..., if $\beta_2 > \beta_1$ and $r(i,\beta_1) < 1$ for all i = 1,2,…, then

$$P\{D(\beta_2) \mid D^c(\beta_1)\} = 1 - [\prod_{i=1}^{+\infty} [1-r(i,\beta_2)] \; / \; \prod_{i=1}^{+\infty} [1-r(i,\beta_1)]],$$

(iii) for iterations $k_1 \le k_2$, if $\beta_2 > \beta_1$ and $r(i,\beta_1) < 1$ for all i =1,2,…, then

$$P\{D(\beta_2) \mid D^c(k_1,\beta_1)\} = 1 - [\prod_{i=1}^{+\infty} [1-r(i,\beta_2)] \; / \; \prod_{i=1}^{k_1} [1-r(i,\beta_1)]].$$

Proof: $P\{D(k,\beta_2) \mid D^c(k,\beta_1)\} = 1 - P\{D^c(k,\beta_2) \mid D^c(k,\beta_1)\}$

$$P\{D^c(k,\beta_2) \mid D^c(k,\beta_1)\} = P\{D^c(k,\beta_2)\} \cap D^c(k,\beta_1)\} / P\{D^c(k,\beta_1)\}$$
$$= P\{D^c(k,\beta_2)\} / P\{D^c(k,\beta_1)\}$$
$$= [\prod_{i=1}^{k} [1-r(i,\beta_2)] \; / \; \prod_{i=1}^{k} [1-r(i,\beta_1)]].$$

Therefore,

$$P\{D(k,\beta_2) \mid D^c(k,\beta_1)\} = 1 - [\prod_{i=1}^{k} [1-r(i,\beta_2)] \; / \; \prod_{i=1}^{k} [1-r(i,\beta_1)]]$$

Taking the limit as $k \to +\infty$ in (i) establishes (ii). Rewriting $P\{D(k,\beta_2) \mid D^c(k,\beta_1)\}$ as $P\{D(k_2,\beta_2) \mid D^c(k_1,\beta_1)\}$, $k_1 \le k_2$, and taking the limit as $k_2 \to +\infty$ establishes (iii) □

Theorem 10 provides an expression for the probability that a local search algorithm has not visited an element of D_{β_1} after k iterations, given that it has visited an element of D_{β_2}, $\beta_2 > \beta_1$ after k iterations. This result provides insights into the possibility that better solutions may be available in k iterations.

THEOREM 10: Consider a local search algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. Then

(i) for all iterations k = 1, 2,..., if $\beta_2 > \beta_1$, $r(i,\beta_1) < 1$, and $P\{D(k,\beta_2)\} \ge 0$ for all i = 1,2,…,k,

then $P\{D^c(k,\beta_1) \mid D(k,\beta_2)\} = [\prod_{i=1}^{k} [1-r(i,\beta_1)] - \prod_{i=1}^{k} [1-r(i,\beta_2)]] \; / \; [1 - \prod_{i=1}^{k} [1-r(i,\beta_2)]]$,

(ii) for all iterations k = 1, 2,..., if $\beta_2 > \beta_1$, $r(i,\beta_1) < 1$, and $P\{D(k,\beta_2)\} \geq 0$ for all i = 1,2,…,

$$\text{then } P\{D^c(\beta_1) \mid D(\beta_2)\} = [\prod_{i=1}^{+\infty} [1-r(i,\beta_1)] - \prod_{i=1}^{+\infty} [1-r(i,\beta_2)]] / [1 - \prod_{i=1}^{+\infty} [1-r(i,\beta_2)]],$$

(iii) for all iterations $k_1 \leq k_2$, if $\beta_2 > \beta_1$, $P\{D(\beta_2)\} \geq 0$, and $r(i,\beta_1) < 1$ for all i = 1,2,…, then

$$P\{D^c(k_1,\beta_1) \mid D(\beta_2)\} = [\prod_{i=1}^{k_1} [1-r(i,\beta_1)] - \prod_{i=1}^{+\infty} [1-r(i,\beta_2)]] / [1 - \prod_{i=1}^{+\infty} [1-r(i,\beta_2)]].$$

Proof:

$$\begin{aligned} P\{D^c(k,\beta_1) \mid D(k,\beta_2)\} &= 1 - P\{D(k,\beta_1) \mid D(k,\beta_2)\} \\ &= 1 - P\{D(k,\beta_1)\} \cap D(k,\beta_2)\} / P\{D(k,\beta_2)\} \\ &= 1 - P\{D(k,\beta_1)\} / P\{D(k,\beta_2)\} \\ &= 1 - [1-\prod_{i=1}^{j} [1-r(i,\beta_1)] / [1 - \prod_{i=1}^{j} [1-r(i,\beta_2)]] \\ &= [\prod_{i=1}^{k} [1-r(i,\beta_1)] - \prod_{i=1}^{k} [1-r(i,\beta_2)]] / [1-\prod_{i=1}^{k} [1-r(i,\beta_2)]]. \end{aligned}$$

Taking the limit as $k \to +\infty$ establishes (ii). Rewriting $P\{D^c(k,\beta_2) \mid D(k,\beta_1)\}$ as $P\{D^c(k_2,\beta_2) \mid D(k_1,\beta_1)\}$, $k_1 \leq k_2$, and taking the limit as $k_2 \to +\infty$ establishes (iii). □

4.5 Application to Specific Local Search Algorithms

The results presented in this chapter can be used to assess the performance of various local search algorithms using the GHC framework. In this section, the performance of four such algorithms, Monte Carlo search, pure local search, random restart local search, and threshold accepting, are evaluated.

4.5.1 Monte Carlo Search

Theorem 6 implies that the β-acceptable solution probability $P\{D(\beta) \mid D^c(k,\beta)\} = 1$ for all k for Monte Carlo search. To see this, Monte Carlo search can be described as a GHC algorithm by setting $\eta(\omega) = \Omega$ for all $\omega \in \Omega$, and $R_k = +\infty$ for all iterations k = 1,2,… . Assume that the neighbors are generated uniformly at each iteration of a GHC algorithm. Define

$$p(\beta) \equiv r(k,\beta) = |D_\beta| / |\Omega| \equiv \{\text{Monte Carlo search is at an element of } D_\beta \text{ at iteration k}\}.$$

Therefore, $P\{D^c(k,\beta)\} = [1-p(\beta)]^k$. From Lemma 3, for iterations $k \leq k'$,

$$P\{D(k',\beta) \mid D^c(k,\beta)\} = 1 - [1-p(\beta)]^{(k'-k)}.$$

Hence, the finite β-acceptable solution probability approaches one as k' approaches infinity. Moreover, from Theorem 4, $\sum_{i=k+1}^{+\infty} r(i,\beta) = \sum_{i=k+1}^{+\infty} p(\beta) = +\infty$ for all iterations $k = 1, 2, \ldots$. This means that Monte Carlo search visits D_β with probability one. However, $P\{C(k,\beta)\} = p(\beta)$ for all iterations k, hence, Monte Carlo search does not converge with probability one to D_β. More specifically, from Theorem 4, (D2), (D3) and (D4) hold, but (D1) is not satisfied.

From Theorem 8, $E[T_\beta] = \sum_{k=0}^{+\infty} \prod_{i=1}^{k} [1-p(\beta)]$ is the expected minimum number of iterations to reach an element of D_β from an element of D_β^c, and $P\{T_\beta > k\} = P\{D^c(k,\beta\} = \prod_{i=1}^{k} [1-r(i,\beta)] = [1-p(\beta)]^k$ is the probability that the minimum number of iterations needed to reach an element of D_β from an element of D_β^c is at least k. Therefore, the minimum number of iterations to reach an element of D_β from an element of D_β^c is distributed geometric with probability $p(\beta)$, hence $E[T_\beta] = 1/p(\beta)$. Note that although $P\{D(k,\beta)\}$ approaches one as k approaches infinity, the rate at which is does so is too slow to be of any practical value.

4.5.1 Pure Local Search

Pure local search is a particular GHC algorithm with $R_k(\omega(i),\omega) = 0$, $\omega(i) \in \Omega$, $\omega \in \eta(\omega(i))$, for iteration $k = 1, 2, \ldots$. Therefore, $P\{R_k(\omega(i),\omega) \geq \delta(\omega(i),\omega)\} = 0$ for all $\omega(i) \in \Omega$, $\omega \in \eta(\omega(i))$, such that $\delta(\omega(i),\omega) > 0$ (i.e., non improving solutions). Since the solution space is finite, if the discrete optimization problem has more than one locally optimal solution, then there exists an iteration k_0 such that $r(k,\beta) = 0$ for all $k \geq k_0$. Therefore, (D2), (D3) and (D4) in Theorem 6 cannot hold. This implies that pure local search does not converge with probability one to D_β if the solution space contains more than one locally optimal solution.

Assuming that the particular instance of the discrete optimization problem has more than one locally optimal solution and that there exists an iteration k_0 such that $r(k,\beta) = 0$ for all $k \geq k_0$, then from Theorem 8, $E[T_\beta] = \sum_{k=0}^{+\infty} \prod_{i=1}^{k} [1-r(i,\beta)] = +\infty$, and $P\{T_\beta = k\} = r(k,\beta) \prod_{i=1}^{k} [1 - r(i,\beta)] = 0$ for all iterations $k \geq k_0$. Note that while pure local search does not converge with probability one to D_β, it remains a popular algorithm and, in practice, often yields satisfactory results.

4.5.2 Random Restart Local Search

Random-restart local search combines Monte Carlo search and pure local search, by randomly selecting a new initial solution (i.e., random restart) whenever the algorithm has reached a local optimum. Assume that each initial solution in uniformly generated (i.e., $P\{\omega' \in \Omega$ is selected as the initial solution$\} = 1/|\Omega|$). Define $h = 1,2,\ldots$, as the number of times the random-restart local search algorithm selects a new initial solution (i.e., the number of random restarts) over a single algorithm run with K total iterations. Define the random variable K_h as the total number of iterations that the random-restart local search algorithm has executed after h restarts. Since each of the h random restarts is a replication of a pure local search algorithm, define

$$p(\omega|\omega') \equiv P\{\text{local search terminates at } \omega \in L \cup G \mid \omega' \in \Omega \text{ is selected as the initial solution}\},$$

$$p(\omega) = \sum_{\omega' \in \Omega} p(\omega|\omega')\, P\{\omega' \in \Omega \text{ is selected as the initial solution}\} = \left[\sum_{\omega' \in \Omega} p(\omega|\omega')\right] / |\Omega|$$

$$\equiv P\{\text{local search terminates at } \omega \in L \cup G\},$$

$$q(\beta) = \sum_{\omega \in L \cup G \cup D_\beta} p(\omega)$$

$$\equiv P\{\text{local search terminates at } \omega \in L \cup G \text{ with } f(\omega) \leq \beta\},$$

Therefore, for $K_h = k$, $P\{D^c(k,\beta)\} = [1 - q(\beta)]^h$. Note that by the definition of K_h, if $h \leq h'$, then $P\{K_h \leq K_{h'}\} = 1$.

Lemma 6 provides an expression for the finite β-acceptable solution probability in terms of q(β).

LEMMA 6: For $h \le h'$, $K_h = k$ and $K_{h'} = k'$, with $k \le k'$, $P\{D(k',\beta) \mid D^c(k,\beta)\} = 1-[1-q(\beta)]^{h'-h}$.

Proof:

$$\begin{aligned} P\{D(k',\beta) \mid D^c(k,\beta)\} &= 1 - P\{D^c(k',\beta) \mid D^c(k,\beta)\} \\ &= 1 - [P\{D^c(k',\beta) \cap D^c(k,\beta)\} / P\{D^c(k,\beta)\}] \\ &= 1 - [P\{D^c(k',\beta)\} / P\{D^c(k,\beta)\}] \\ &= 1 - [[1-q(\beta)]^{h'} / [1-q(\beta)]^{h}] = 1 - [1-q(\beta)]^{h'-h}. \end{aligned}$$

□

Therefore, the finite β-acceptable solution probability approaches one as the number of restarts h' approaches infinity. Moreover, random-restart local search does not converge with probability one to D_β (i.e., from Theorem 6, (D2), (D3), and (D4) all hold, but (D1) is not satisfied).

Define ω_j as the locally optimal solution found by the random-restart local search algorithm at restart j = 1,2,…,h. Define the random variable

$$T'_\beta \equiv \min \{j \ge 1 : f(\omega_j) \le \beta, \omega_j \in L \cup G\}$$

as the minimum number of restarts needed to visit an element of D_β for the first time, where $P\{T'_\beta > h\} = [1-q(\beta)]^h$ is the probability that the minimum number of restarts needed to reach an element of D_β for the first time is greater than h. Therefore, the minimum number of restarts to visit an element of D_β for the first time is distributed geometric with probability q(β), hence $E[T'_\beta] = 1/q(\beta)$.

4.5.3 Threshold Accepting

Threshold accepting is a particular GHC algorithm with $R_k(\omega(i),\omega) = t_k$, $\omega(i) \in \Omega$, $\omega \in \eta(\omega(i))$, for iteration k, where t_k typically approaches zero as k approaches infinity (i.e., $\lim_{k \to +\infty} t_k = 0$). Therefore, for all ε > 0, there exists an iteration k_0 such that $|t_k| < \varepsilon$ and $t_k < |\delta(\omega(i),\omega)|$ for all $\omega(i) \in D_\beta^c$, $\omega \in \eta(\omega(i))$, and all $k \ge k_0$ (i.e., threshold accepting behaves like pure local search for all $k \ge k_0$), hence (D4) in Theorem 6 does not hold. This implies

that this formulation of threshold accepting does not converge with probability one to D_β. However, if $\{t_k\}$ is a sequence that does not approach zero and $t_k \geq \delta(\omega(i),\omega)$ for some $\delta(\omega(i),\omega) > 0$, with $\omega(i) \in \Omega$, $\omega \in \eta(\omega(i))$ and for all iterations k, then (D4) in Theorem 6 may hold and the probability of a β-acceptable solution is one at all iterations k. However, setting t_k in this way may not be feasible in practice, since it requires full knowledge of the solution space.

If t_k approaches zero as iteration k approaches infinity such that there exists an iteration k_0 with $r(k,\beta) = 0$ for all $k \geq k_0$, then from Lemma 2 and Theorem 8, $E[T_\beta] = \sum_{k=0}^{+\infty} \prod_{i=0}^{k} [1 - r(i,\beta)] = +\infty$, and $P\{T_\beta = k\} = r(k,\beta) \prod_{i=0}^{k} [1 - r(i,\beta)] = 0$ for all $k \geq k_0$. Note that while this formulation of threshold accepting does not converge with probability one to D_β, it may yield satisfactory results in practice.

Chapter 5:

Estimating the Performance of Local Search Algorithms

The goal of this research is to provide guidelines on how to design effective run strategies for local search algorithms. This chapter focuses on developing techniques for determining termination conditions for non-convergent local search algorithms by formulating procedures for estimating the probability of visiting a β-acceptable solution in finite-time. Section 5.1 develops a procedure to estimate $P\{T_\beta \leq K\}$ for values of β that can be easily reached in K iterations. Section 5.2 introduces logistic regression as a tool to estimate $P\{T_\beta \leq K\}$ for smaller values of β over K iterations that would otherwise be too difficult to determine computationally (i.e., values of β close to $f(\omega^*)$).

5.1 Estimating the Probability of Reaching β-Acceptable Solutions

Assume that $P\{D^c(k,\beta)\} > 0$ for all iterations k = 1,2, … (i.e., finite-time convergence to a β-acceptable solution is not guaranteed). Corollary 1 establishes that the probability of visiting any element of D_β is nonincreasing with each successive iteration that a local search algorithm has *not* visited any element of D_β (i.e., $P\{D(\beta) \mid D^c(k+1,\beta)\} \leq P\{D(\beta) \mid D^c(k,\beta)\}$). Define the *marginal value of additional iterations* for a local search algorithm as $P\{D(\beta) \mid D^c(k,\beta)\} - P\{D(\beta) \mid D^c(k+1,\beta)\}$. This suggests that a local search algorithm should

be terminated at iteration k if the marginal value of additional iterations is sufficiently small (i.e., when the marginal value of additional iterations becomes negligible). Hence, the β-acceptable solution probability may be used as a stopping condition for the local search algorithm.

This marginal value of additional iterations provides a measure of performance for a local search algorithm based on the number of iterations executed. However, in practice, computing budgets (in terms of time or memory) often determine the upper bound on the number of iterations for a given algorithm and practitioners must choose a β-acceptable threshold based on the probability of successfully reaching a predetermined threshold in a given number of iterations.

This section introduces a procedure for estimating $P\{T_\beta \le K\}$ (i.e., the probability that the algorithm has visited an element of D_β after K iterations) with $f(\omega^*) \le \beta < \max_{\omega \in L} f(\omega)$, $\omega^* \in G$. Define algorithm **A** to be a local search algorithm applied to an instance of an intractable discrete optimization problem. For a given instance of an intractable discrete optimization problem, $P\{T_\beta \le K\}$ is the probability of visiting a β-acceptable solution after K iterations, where $\omega(0) \notin D_\beta$ (i.e., $P\{D^c(0,\beta)\} = 1$). Suppose that a single K iteration run of algorithm **A** results in the K solutions, $\omega(1), \omega(2),\ldots,\omega(K)$, where $\omega(i) \in \Omega$, i = 1,2,…, K. Define $f(\omega^K)$ to be the minimum objective function value among these K solutions, with associated solution $\omega^K = \text{argmin}\{f(\omega(i)), i = 1,2,\ldots,K\}$. Define the Bernoulli random variable

$$X_{K,\beta} = \begin{cases} 1 & \text{if } f(\omega^K) \le \beta \\ 0 & \text{if } f(\omega^K) > \beta \end{cases}, \tag{18}$$

with parameter $P\{T_\beta \le K\}$, hence $E[X_{K,\beta}] = P\{T_\beta \le K\}$.

The probability $P\{T_\beta \le k\}$ can be estimated by fixing the total number of iterations K and executing h = 1, 2,…, H independent replications of algorithm **A**, where each such replication is initialized with a randomly generated solution $\omega(0) \notin D_\beta$ (i.e., $P\{D^c(0,\beta)\} = 1$).

For replication h, algorithm **A** yields K solutions, $\omega^h(1)$, $\omega^h(2)$,…, $\omega^h(K)$, where $\omega^h(i) \in \Omega$, i = 1, 2,…, K, h = 1, 2,…, H. Define $f(\omega^{h,K})$, h = 1, 2,…, H, to be the minimum objective function value among these K solutions for replication h, with associated solution $\omega^{h,K}$ = argmin{$f(\omega^h(i))$, i = 1, 2,…, K}. Therefore, for each h = 1, 2,…, H, define the Bernoulli random variable

$$X_{K,\beta}(h) = \begin{cases} 1 & \text{if } f(\omega^{h,K}) \le \beta \\ 0 & \text{if } f(\omega^{h,K}) > \beta \end{cases}, \tag{19}$$

with parameter $P\{T_\beta \le K\}$. These Bernoulli random variables can be used to define the estimator for $P\{T_\beta \le K\}$,

$$\hat{P}\{T_\beta \le K\} = [\sum_{h=1}^{H} X_{K,\beta}(h)] / H. \tag{20}$$

This estimator is the sample mean of H independent and identically distributed Bernoulli trials. For practical purposes, this estimator is useful for those values of β that are sufficiently large such that solutions with objective function value at or below these values can be visited within K iterations. However, it may not be practical to estimate the probability of visiting solutions with objective function values close to $f(\omega^*)$, $\omega^* \in G$, since such events may be very rare. To circumvent this problem, suppose that a model can be defined to estimate such probabilities. To this effect, consider the model

$$E[X_{K,\beta}] = P\{T_\beta \le K\} = g(K,\beta) \tag{21}$$

where

$$g: Z^+ \times \Re \rightarrow [0,1] \text{ and } \lim_{\substack{\beta \to \max f(\omega)^- \\ \omega \in \Omega}} g(K,\beta) = 1, \quad \lim_{\substack{\beta \to \min f(\omega)^+ \\ \omega \in \Omega}} g(K,\beta) = 0$$

Since $X_{K,\beta}$ is a Bernoulli random variable, then the function $g(K,\beta)$ can be estimated by fitting a curve to the estimated values for $P\{T_\beta \le K\}$ using logistic regression (Hosmer and

Lemeshow 1989). The function, $g(K,\beta)$, can then be used to estimate $P\{T_\beta \le K\}$ for values of β close to $f(\omega^*)$. Moreover, $g(K,\beta)$ can be used to compare different local search algorithms applied to the same discrete optimization problem. Therefore, the function $g(K,\beta)$ can aid practitioners in making decisions about which local search algorithm will provide the best finite-time results for a particular instance of a discrete optimization problem. The resulting procedures are described in detail in Section 5.2.

5.2 Modeling $P\{T_\beta \le k\}$ Using Logistic Regression

Section 5.1 presented a procedure for estimating $P\{T_\beta \le K\}$ (the probability of visiting a β-acceptable solution for a given number of iterations K). The function $g(K,\beta)$ was also presented as a model to estimate the probability of visiting solutions with objective function values close to the objective function value of a globally optimal solution. This section describes logistic regression as the model for such estimation.

The random variable $X_{K,\beta}$ defined in Section 5.1 is a Bernoulli random variable with parameter $P\{T_\beta \le K\}$, where $E[X_{K,\beta}] = P\{T_\beta \le K\}$ (i.e., the local search algorithm either visits or does not visit a solution with objective function value less than or equal to β in K or fewer iterations). The Bernoulli nature of $X_{K,\beta}$ suggests that a plot of $\hat{P}\{T_\beta \le K\}$ in (20) for several values of β defines an S-shaped nonlinear response between the values of β and $\hat{P}\{T_\beta \le K\}$(see Figures 5-10 of Section 6.3). This observation suggests that the response function $g(K,\beta)$ proposed in Section 5.1 should be S-shaped with asymptotes at zero and one. Montgomery and Peck (1982) suggests that for such random variables, logistic regression can be used to estimate parameters for $g(K,\beta)$. The goal of logistic regression is to find the best fitting model to describe the relationship between a predictor and a response variable (i.e., the relationship between the values of β and $P\{T_\beta \le K\}$). The principle difference between logistic regression and linear regression is that the outcome variable in logistic regression is *binary* or *dichotomous* (Hosmer and Lemeshow 1989).

In regression analysis, the mean of the response variable (as it relates to the predictor variable) is a critical value of interest. Define the *conditional mean* $E[X_{K,\beta} \mid \beta]$ where $X_{K,\beta}$ is

the response variable and β is the predictor variable. In linear regression, if the conditional mean is expressed as

$$E[X_{K,\beta} \mid \beta] = \delta_0 + \delta_1\beta + \delta_2\beta^2 + \delta_3\beta^3, \tag{22}$$

then $E[X_{K,\beta} \mid \beta]$ can take on any value (hence is not restricted to values between zero and one) for some values of β. Note that a special case of (22) is the linear model $E[X_{K,\beta} \mid \beta] = \delta_0 + \delta_1\beta$. However, with dichotomous data, $E[X_{K,\beta}(h) \mid \beta]$ must take on values between zero and one (i.e., $0 \le E[X_{K,\beta}(h) \mid \beta] \le 1$) (Hosmer and Lemeshow 1989). Note that the plot of $\hat{P}\{T_\beta \le K\}$ for several values of β is S-shaped with asymptotes at zero and one. Logistic regression uses the CDF of the logistic distribution to model $E[X_{K,\beta} \mid \beta]$. The logistic regression model is expressed as

$$g(K,\beta) = E[X_{K,\beta} \mid \beta] = \frac{e^{(\delta_0+\delta_1\beta+\delta_2\beta^2+\delta_3\beta^3)}}{1 + e^{(\delta_0+\delta_1\beta+\delta_2\beta^2+\delta_3\beta^3)}}. \tag{23}$$

The coefficients δ_0, δ_1, δ_2, and δ_3 in (23) can be estimated from the sample data. To obtain these estimates, (23) is transformed into a cubic model using the *logit transformation* (Hosmer and Lemeshow 1989)

$$f(K,\beta) = \ln\left[\frac{g(K,\beta)}{1\text{-}g(K,\beta)}\right] = \delta_0 + \delta_1\beta + \delta_2\beta^2 + \delta_3\beta^3 \tag{24}$$

One important difference between linear regression and logistic regression concerns the conditional distribution of the response variable (Hosemer and Lemeshow 1989). If a linear regression model is used, then the observation of the response variable can be expressed as

$$X_{K,\beta} = E[X_{K,\beta} \mid \beta] + \varepsilon, \tag{25}$$

where ε represents the error between the observation and the conditional mean. To be valid, the error term ε is typically required to follow a normal distribution with mean zero and a constant variance for all values of β. However, with dichotomous data, the error term ε takes on one of two values (Recall from (23) that $g(K,\beta) = E[X_{K,\beta} \mid \beta]$, the conditional mean of $X_{K,\beta}$). Therefore, when $X_{K,\beta} = 1$, then $\varepsilon = 1 - g(K,\beta)$ with probability $g(K,\beta)$, while if $X_{K,\beta} = 0$, then $\varepsilon = -g(K,\beta)$ with probability $1 - g(K,\beta)$. This means that ε is distributed with mean zero and variance $g(K,\beta)[1 - g(K,\beta)]$ (i.e., ε is a Bernoulli random variable with probability $g(K,\beta)$; see Hoel et al. 1971). Therefore, the linear regression model in (25) does not provide an adequate fit, hence is not appropriate to use.

Logistic regression uses the method of maximum likelihood to estimate the parameters in (24) (see Hosmer and Lemeshow 1989). The likelihood function expresses the probability of the observed data as a function of the parameters δ_0, δ_1, δ_2, and δ_3, where the maximum likelihood estimators for δ_0, δ_1, δ_2, and δ_3 are those values that maximize the likelihood function (see Rice 1987).

To find the maximum likelihood estimators for the logistic regression model (24), recall that the Bernoulli random variable $X_{K,\beta}$ defined in Section 5.1 can take on a value of either zero or one, depending on the value of $f(\omega^K)$. Therefore, $g(K,\beta) = P\{X_{K,\beta} = 1 \mid \beta\}$ with $P\{X_{K,\beta} = 0 \mid \beta\} = 1 - g(K,\beta)$. Define $S_\beta = \{\beta_1, \beta_{2,} \ldots \beta_n\}$ as the set of β values at which $P\{T_\beta \le K\}$ is estimated using (20). Therefore, for the predictor and response variable pair (β_i, x_i), where $x_i = X_{K,\beta_i}$ for all values of $\beta_i \in S_\beta$, i = 1, 2,…, n, the contribution to the likelihood function $\xi(\beta_i)$ is $g(K,\beta_i)$ when $x_i = 1$ and $1-g(K,\beta_i)$ when $x_i = 0$. Therefore, the likelihood function is

$$\xi(\beta_i) = g(K,\beta_i)^{x_i} [1-g(K,\beta_i)]^{1-x_i}. \qquad (26)$$

Assuming that the observations x_i, i = 1, 2, …, n, are independent, the likelihood function can be expressed as

$$l(\boldsymbol{\delta}) = \prod_{i=1}^{n} \xi(\beta_i), \tag{27}$$

with the parameters δ_0, δ_1, δ_2, and δ_3 represented by the vector $\boldsymbol{\delta} = [\ \delta_0\ \ \delta_1\ \ \delta_2\ \ \delta_3\ ]'$. The goal of maximum likelihood estimation is to choose values for $\boldsymbol{\delta}$ that maximize (27). Define the *log likelihood function* as

$$L(\boldsymbol{\delta}) = \ln[l(\boldsymbol{\delta})] = \sum_{i=1}^{n} [x_i \ln[g(K,\beta_i)] + (1 - x_i)\ln[1 - g(K,\beta_i)]]. \tag{28}$$

Using the log likelihood function versus the likelihood function simplifies the mathematics involved in finding a value for $\boldsymbol{\delta}$ that maximizes the likelihood function $l(\boldsymbol{\delta})$. Define the *likelihood equations* as the system of equations formed by differentiating $L(\boldsymbol{\delta})$ with respect to δ_0, δ_1, δ_2, and δ_3 and setting their values to zero. From (23) and (28), the resulting likelihood equations are

$$\sum_{i=1}^{n} [x_i - g(K,\beta_i)] = \sum_{i=1}^{n} \left[x_i - \frac{e^{(\delta_0+\delta_1\beta_i+\delta_1\beta_i^2+\delta_1\beta_i^3)}}{1 + e^{(\delta_0+\delta_1\beta_i+\delta_1\beta_i^2+\delta_1\beta_i^3)}}\right] = 0, \tag{29}$$

$$\sum_{i=1}^{n} \beta_i[x_i - g(K,\beta_i)] = \sum_{i=1}^{n} \beta_i\left[x_i - \frac{e^{(\delta_0+\delta_1\beta_i+\delta_1\beta_i^2+\delta_1\beta_i^3)}}{1 + e^{(\delta_0+\delta_1\beta_i+\delta_1\beta_i^2+\delta_1\beta_i^3)}}\right] = 0, \tag{30}$$

$$\sum_{i=1}^{n} \beta_i^2[x_i - g(K,\beta_i)] = \sum_{i=1}^{n} \beta_i^2\left[x_i - \frac{e^{(\delta_0+\delta_1\beta_i+\delta_1\beta_i^2+\delta_1\beta_i^3)}}{1 + e^{(\delta_0+\delta_1\beta_i+\delta_1\beta_i^2+\delta_1\beta_i^3)}}\right] = 0, \tag{31}$$

and

$$\sum_{i=1}^{n} \beta_i^3[x_i - g(K,\beta_i)] = \sum_{i=1}^{n} \beta_i^3\left[x_i - \frac{e^{(\delta_0+\delta_1\beta_i+\delta_1\beta_i^2+\delta_1\beta_i^3)}}{1 + e^{(\delta_0+\delta_1\beta_i+\delta_1\beta_i^2+\delta_1\beta_i^3)}}\right] = 0. \tag{32}$$

For logistic regression, (29), (30), (31), and (32) are non-linear equations in $\boldsymbol{\delta}$, hence require iterative methods using generalized weighted least squares to solve for a value for $\boldsymbol{\delta}$. These iterative methods are available in many commercial statistical software packages; see McCullagh and Nelder (1989) for a complete discussion of these iterative methods.

Let the *maximum likelihood estimator* for $\boldsymbol{\delta}$ be denoted by $\hat{\boldsymbol{\delta}}$. Note that these estimators are asymptotically unbiased (as n approaches infinity), but not consistent (see McCullagh and Nelder 1989, p.119). Using this maximum likelihood estimator, the estimated logit function is

$$\hat{f}(K,\beta) = \hat{\delta}_0 + \hat{\delta}_1\beta + \hat{\delta}_2\beta^2 + \hat{\delta}_3\beta^3 \tag{33}$$

and the estimated logistic regression model is

$$\hat{g}(K,\beta) = \frac{e^{(\hat{\delta}_0 + \hat{\delta}_1\beta + \hat{\delta}_2\beta^2 + \hat{\delta}_3\beta^3)}}{1 + e^{(\hat{\delta}_0 + \hat{\delta}_1\beta + \hat{\delta}_2\beta^2 + \hat{\delta}_3\beta^3)}}. \tag{34}$$

Expression (34) can be used to obtain estimators for $E[X_{K,\beta} \mid \beta]$. In particular, a point estimator for $E[X_{K,\beta} \mid \beta_0]$ is

$$\hat{g}(K,\beta_0) = \frac{e^{(\hat{\delta}_0 + \hat{\delta}_1\beta_0 + \hat{\delta}_1\beta_0^2 + \hat{\delta}_1\beta_0^3)}}{1 + e^{(\hat{\delta}_0 + \hat{\delta}_1\beta_0 + \hat{\delta}_1\beta_0^2 + \hat{\delta}_1\beta_0^3)}}. \tag{35}$$

Expression (34) can also be used to obtain a confidence interval estimator for $E[X_{K,\beta} \mid \beta]$. To see this, a $(1-\alpha)100\%$ confidence interval estimator for $f(K,\beta_0) = \delta_0 + \delta_1\beta + \delta_2\beta_0^2 + \delta_3\beta_0^3$ is

$$(\hat{f}(K,\beta_0) - \sigma z_{(1-\alpha/2)100\%},\ \hat{f}(K,\beta_0) + \sigma z_{(1-\alpha/2)100\%}), \tag{36}$$

where $\sigma^2 = \text{var}(\hat{\delta}_0 + \hat{\delta}_1\beta_0 + \hat{\delta}_1\beta_0^2 + \hat{\delta}_1\beta_0^3)$ (see Ross 1988) and $z_{(1-\alpha/2)}$ is the $1-\alpha/2$ fractile of the standard normal distribution. Note that for moderate to large sample sizes (replications), confidence interval estimators based on the normal approximation to the sampling distribution are appropriate (Hosmer and Lemeshow 1989, p.44). Theorem 11 uses this confidence interval estimator to obtain a $(1-\alpha)100\%$ confidence interval estimator for $E[X_{K,\beta} \mid \beta_0]$.

THEOREM 11: Given the $(1-\alpha)100\%$ confidence interval estimator for $f(K,\beta_0) = \delta_0 + \delta_1\beta + \delta_2\beta_0^2 + \delta_3\beta_0^3$ in (36), a $(1-\alpha)100\%$ confidence interval estimator for $E[X_{K,\beta} \mid \beta_0]$ is

$$\left(\frac{e^{(\hat{f}(K,\beta_0) - \sigma z_{(1-\alpha/2)100\%})}}{1 + e^{(\hat{f}(K,\beta_0) - \sigma z_{(1-\alpha/2)100\%})}}, \frac{e^{(\hat{f}(K,\beta_0) + \sigma z_{(1-\alpha/2)100\%})}}{1 + e^{(\hat{f}(K,\beta_0) + \sigma z_{(1-\alpha/2)100\%})}}\right). \tag{37}$$

Proof: From (36)

$$P\{\hat{f}(K,\beta_0) - \sigma z_{(1-\alpha/2)100\%} \le f(K,\beta) = \delta_0 + \delta_1\beta \le \hat{f}(K,\beta_0) + \sigma z_{(1-\alpha/2)100\%}\} = (1-\alpha)100\% .$$

Using (24),

$$\Leftrightarrow P\{\hat{f}(K,\beta_0) - \sigma z_{(1-\alpha/2)100\%} \le \ln\left[\frac{g(K,\beta_0)}{1-g(K,\beta_0)}\right] \le \hat{f}(K,\beta_0) + \sigma z_{(1-\alpha/2)100\%}\} = (1-\alpha)100\%$$

$$\Leftrightarrow P\{e^{\hat{f}(K,\beta_0) - \sigma z_{(1-\alpha/2)100\%}} \le \frac{g(K,\beta_0)}{1-g(K,\beta_0)} \le e^{\hat{f}(K,\beta_0) + \sigma z_{(1-\alpha/2)100\%}}\} = (1-\alpha)100\%.$$

$$\Leftrightarrow P\{(1-g(K,\beta_0)e^{\hat{f}(K,\beta_0) - \sigma z_{(1-\alpha/2)100\%}} \le g(K,\beta_0) \le (1-g(K,\beta_0))e^{\hat{f}(K,\beta_0) + \sigma z_{(1-\alpha/2)100\%}}\} = (1-\alpha)100\%.$$

Therefore,

$$P\{ e^{\hat{f}(K,\beta_0) - \sigma z_{(1-\alpha/2)100\%}} \le g(K,\beta_0)\,(1+ e^{\hat{f}(K,\beta_0) - \sigma z_{(1-\alpha/2)100\%}})$$

$$\text{and } g(K,\beta_0)(1+ e^{\hat{f}(K,\beta_0) + \sigma z_{(1-\alpha/2)100\%}}) \le e^{\hat{f}(K,\beta_0) + \sigma z_{(1-\alpha/2)100\%}} \} = (1-\alpha)100\%$$

$$\Leftrightarrow P\{ \frac{e^{\hat{f}(K,\beta_0) - \sigma z_{(1-\alpha/2)100\%}}}{1 + e^{\hat{f}(K,\beta_0) - \sigma z_{(1-\alpha/2)100\%}}} \le g(K,\beta_0) \le \frac{e^{\hat{f}(K,\beta_0) + \sigma z_{(1-\alpha/2)100\%}}}{1 + e^{\hat{f}(K,\beta_0) + \sigma z_{(1-\alpha/2)100\%}}} \} = (1-\alpha)100\% .$$

Therefore, from (23),

$$P\{ \frac{e^{\hat{f}(K,\beta_0) - \sigma z_{(1-\alpha/2)100\%}}}{1 + e^{\hat{f}(K,\beta_0) - \sigma z_{(1-\alpha/2)100\%}}} \le E[X_{K,\beta} \mid \beta_0] \le \frac{e^{\hat{f}(K,\beta_0) + \sigma z_{(1-\alpha/2)100\%}}}{1 + e^{\hat{f}(K,\beta_0) + \sigma z_{(1-\alpha/2)100\%}}} \} = (1-\alpha)100\% ,$$

hence $\left(\frac{e^{(\hat{f}(K,\beta_0) - \sigma z_{(1-\alpha/2)100\%})}}{1 + e^{(\hat{f}(K,\beta_0) - \sigma z_{(1-\alpha/2)100\%})}}, \frac{e^{(\hat{f}(K,\beta_0) + \sigma z_{(1-\alpha/2)100\%})}}{1 + e^{(\hat{f}(K,\beta_0) + \sigma z_{(1-\alpha/2)100\%})}} \right)$ is a $(1-\alpha)100\%$ confidence

interval estimator for $E[X_{K,\beta} \mid \beta_0]$. □

The function $g(K,\beta)$ was presented in Section 5.1 as a model to estimate the probability of visiting solutions with objective function values close to the objective function value of a globally optimal solution. Logistic regression is used to estimate the model coefficients in (23) that capture the relationship between a predictor and a response variable (i.e., the relationship between the values of β and $P\{T_\beta \le K\}$).

Chapter 6:

Computational Experiments

Chapter 4 presents theoretical results on the performance of local search algorithms in terms of the β-acceptable solution probability. Chapter 5 presents procedures for estimating the probability of visiting a β-acceptable solution in finite-time and for estimating $P\{T_\beta \leq K\}$ for values of β close to $f(\omega^*)$. This chapter reports computational experiments using six instances of the traveling salesman problem to validate the results presented in Chapter 4 and Chapter 5. The traveling salesman problem (TSP) (Lawler et al. 1985) is an NP-hard discrete optimization problem described in Section 6.1 and is used to validate the expression for $P\{D^c(k,\beta)\}$ presented in Lemma 2(i). Section 6.2 describes the design of experiment used to validate the expression for $P\{D^c(K,\beta)\}$ and to estimate $P\{T_\beta \leq K\}$ for a fixed number of iterations K and over a range of values for β; the complete results are reported in Section 6.3. Section 6.4 describes how data collected from the experiments described in Section 6.3 can be used to estimate the parameters for $g(K,\beta)$ for the six TSP instances described in Section 6.2; these results are reported in Section 6.4. The function $g(K,\beta)$, with corresponding confidence interval estimator, provides insights into the finite-time behavior of local search algorithms for a fixed number of iterations K, as well as for estimating the globally optimal solution objective function value for a discrete optimization problem, as described in Section 6.5.

6.1 The Traveling Salesman Problem

This section formally describes the traveling salesman problem (Lawler et al. 1985), which is used to computationally illustrate the results presented in Chapters 4 and 5.

The *traveling salesman (optimization) problem* (TSP) is a well-studied NP-hard discrete optimization problem (Lawler et al. 1985). The diversity of applications for the TSP makes it a frequent choice for testing and evaluating the efficiency and effectiveness of algorithms and heuristics for intractable discrete optimization problems. Traditional applications of the TSP can be found in numerous domains, including vehicle routing and scheduling problems. More recently, it has been applied to the manipulation of robotics (Balaguer et al. 2000), the cutting of industrial components (Foerster and Wäscher 1998), and circuit board design (Kobayashi et al. 1999).

To formally describe the TSP, several definitions are needed (Lawler et al. 1985). Define a *graph* to be a finite set of *vertices*, some pairs of which are joined by *edges*. A *cycle* in a graph is a set of vertices of the graph, such that it is possible to move between vertices so that all vertices are encountered exactly once, finishing at the start. If a cycle contains all the vertices of the graph, it is called a *Hamiltonian cycle (or tour)*. The usage of the terms *cities* and *vertices* is interchangeable, hence if there exists a direct path between two cities it is equivalent to the existence of an *edge* between the two cities. Using this terminology, the TSP can now be formally stated (Garey and Johnson 1979).

TRAVELING SALESMAN PROBLEM

INSTANCE: Given a set of k cities, $L=\{l_1, l_2,\ldots., l_k\}$, and a symmetric distance matrix D that represents the distance between cities in L.

QUESTION: Find a *Hamiltonian cycle*, $H = (l_1,l_2,\ldots l_k)$, such that

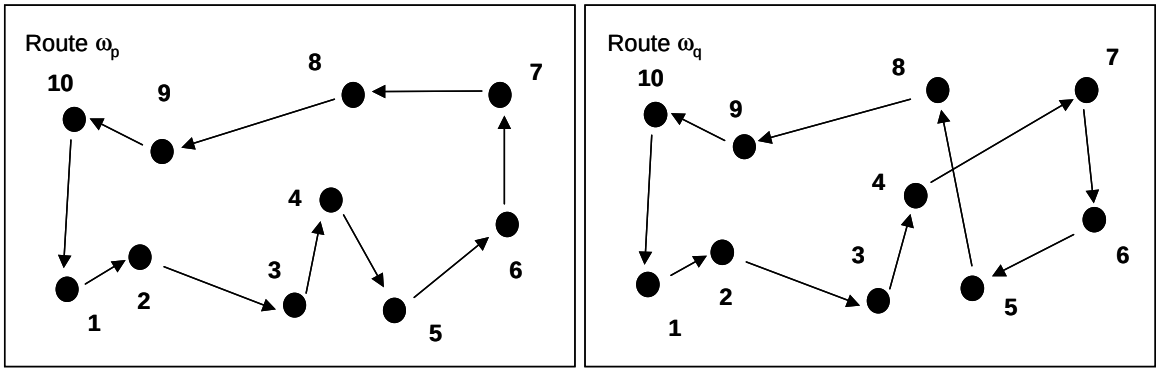
$$f(H) = \sum_{i=1}^{k-1} \mathbf{D}(l_i,l_{i+1}) + \mathbf{D}(l_k,l_1) \text{ is minimized.}$$

An instance of a TSP is a discrete optimization problem, where the solution space is the set of all possible Hamiltonian cycles (with each cycle consisting of n cities), $\Omega = \{\omega_1, \omega_2, \ldots,$

$\omega_{(n-1)!/2}\}$, where (n–1)! / 2 is the number of unique Hamiltonian cycles, provided the distance matrix is symmetric. The objective function value for each solution $\omega_i \in \Omega$ is the sum of the distances along the cycle. The optimal objective function value corresponds to the shortest Hamiltonian cycle.

To apply a local search algorithm to an instance of the TSP, a neighborhood function must be defined. There are numerous neighborhood functions that have been devised for the TSP. One such neighborhood function is the *2-Opt neighborhood function* (Croes 1958). The 2-Opt neighborhood function is a specific version of a more general neighborhood function termed λ-Opt (Helsgaun 2000). The 2-Opt neighborhood function moves from one solution to another solution by the exchange of two cities. For example, consider the finite set of 10 cities $L = \{l_1, l_2, l_3, l_4, l_5, l_6, l_7, l_8, l_9, l_{10}\}$, and the corresponding solution space $\Omega = \{\omega_1, \omega_2, \ldots, \omega_{181,440}\}$. If the current solution is $\omega_p = (l_1, l_2, l_3, l_4, l_5, l_6, l_7, l_8, l_9, l_{10})$, then one possible neighboring solution is $\omega_q = (l_1, l_2, l_3, l_4, l_7, l_6, l_5, l_8, l_9, l_{10})$, which is obtained by exchanging the cities l_5 and l_7 (Figure 3). For a complete discussion of the 2-Opt neighborhood function see Aarts and Lenstra (1997).

Figure 2: The 2-Opt Neighborhood Function



For all experiments using the TSP, the 2-Opt neighborhood function is used.

6.2 Experiments for Estimating P{D(k,β)}

Several computational experiments were conducted with the TSP to validate results presented in Chapter 4 and Chapter 5. These experiments were not designed to present new

local search algorithms to solve the TSP, but rather to develop methods to analyze different local search algorithms applied to the TSP. This section describes how these experiments were designed, while the computational results are presented in Section 6.3.

Lemma 2 in Section 4.2 provides an expression for $P\{D^c(k,\beta)\}$. Six instances of the TSP were used to validate this expression for $P\{D^c(k,\beta)\}$. Moreover, three local search algorithms were implemented for each instance of the TSP: pure local search, simulated annealing, and threshold accepting. The hill climbing random variables, R_k, for each local search algorithm are

i) Pure local search $R_k = 0$

ii) Simulated annealing $R_k = -t_k \ln(U(0,1))$ where t_k is the temperature value at iteration k defined in Section 2.1.

iii) Threshold accepting $R_k = t_k$ where t_k is the threshold value at iteration k defined in Section 2.2.

For simulated annealing and threshold accepting, the temperature value t_k is updated by multiplying the previous temperature value t_{k-1} by the increment multiplier ϕ, where $0 \le \phi \le 1$ (i.e., $t_k = \phi t_{k-1}$). Four instances of the TSP with a known optimal solution were selected from the Traveling Salesman Problem Library (Reinelt 1991) and two instances of the TSP were randomly generated. A best-known-solution for each randomly generated instance of the TSP was generated using random restart local search (RRLS). This RRLS algorithm applied to the two randomly generated instances of the TSP consisted of 10,000 restarts with each restart being initialized with a unique randomly generated initial solution. Note that the best-known-solution cannot be guaranteed to be the globally optimal solution, but rather, provides an upper bound on the globally optimal solution objective function value. A range of values for β was selected for each instance of the TSP based on the known optimal solution or the best-known-solution found (with RRLS). Each local search algorithm experiment involved $H = 500$ independent replications, where each replication was executed for K iterations and was initialized with a unique randomly generated initial solution.

Two instances of the TSP consisting of 50 and 100 cities respectively were generated by randomly locating each city on a 1000 by 1000 unit grid. The other four instances of the TSP with known optimal solutions were selected from the Traveling Salesman Problem Library (Reinelt 1991). The (Euclidean) distance $\mathbf{D}(l_i,l_j)$ between each pair of cities $l_i,l_j \in L$ for each instance of the TSP was computed based on the location of each city on the unit grid and placed in a symmetric distance matrix $\mathbf{D}$ (i.e., $\mathbf{D}(l_i,l_j) = \mathbf{D}(l_j,l_i)$). The three local search algorithms, pure local search, simulated annealing and threshold accepting, were applied to each instance of the TSP with K total iterations. See Table 1 for the size and optimal / best-known-solutions for these problems.

Table 1: Instances of the Traveling Salesman Problem

Name	Size of Problem Instance	Value of Best-Known-Solution
Random 50	50 Cities	5657.9
Random 100	100 Cities	7668.8
Berlin 52	52 Cities	7542*
ST70	70 Cities	675*
PR76	76 Cities	108159*
KROA100	100 Cities	21282*

*** known optimal solution**

The 2-Opt neighborhood function described in Section 6.1 was used for each local search algorithm applied to the TSP. Note that the purpose of all these experiments is not to identify better algorithms for the TSP, but rather, to demonstrate how local search algorithms can be analyzed using the tools introduced in Chapters 4 and 5. All experiments were performed in MATLAB on a Personal Computer using the Microsoft Windows operating system. The execution time for each experiment was between two and thirteen CPU hours. Execution time depended on the size of the discrete optimization problem instance (i.e., number of cities in the TSP) and number of iterations K executed for each independent replication of the local search algorithm. Recall that each experiment consisted of H = 500 independent replications of the local search algorithm. Results of these computational experiments are reported in Section 6.3.

6.3 Computational Results for Estimating $P\{T_\beta \leq K\}$

To validate the expression for $P\{D^c(k,\beta)\}$ presented in Section 4.2, computational results are reported for pure local search, simulated annealing, and threshold accepting algorithms applied to the six instances of the TSP. Results are reported in terms of $P\{T_\beta \leq K\} = 1 - P\{D^c(K,\beta)\}$ for K total iterations. For the simulated annealing and threshold accepting algorithms, the increment multiplier ϕ for the temperature value t_k was set at $\phi = \left(\frac{10}{t_0}\right)^{\frac{1}{K}}$, resulting in a final temperature $t_K = 10$ (i.e, $\phi^K t_0 = 10$). The initial temperature t_0 used in the TSP for simulated annealing and threshold accepting was initialized at $t_0 = LM\varphi$, where L = number of cities, $M = \text{Max}\{\mathbf{D}(l_i,l_j)\ \ i,j = 1,2,\ldots,m\}$ and $0 \leq \varphi \leq 1$. The value φ was fixed at $\varphi = .15$ for the six instances of the TSP.

A range of fifty β values, $\beta_i \in S_\beta$, i = 1, 2,…, 50 for each instance of the TSP was chosen with the minimum β value, β_1, set close to the objective function value of the known globally optimal solution or the best-known-solution found with RRLS. The maximum β value, β_{50}, for each instance of the TSP was determined based on the worst-known-solution for each instance of the TSP found with RRLS. The remaining $\beta_2, \beta_3,\ldots, \beta_{49}$ values of β were determined by incrementing each β value by the step size $\theta = (\beta_{50} - \beta_1)/50$ (i.e., $\beta_i = \beta_{i-1} + \theta$). Note that all β values, $\beta_i \in S_\beta$, i = 1, 2,…, 50 were rounded to the nearest whole number. See Table 2 for the range of β values used in each experiment.

Table 2: Range of β Values

Instance of the TSP	MIN β Value (β_1)	MAX β Value (β_{50})	Step Size θ
Random 50 TSP	5660	6885	25
Berlin 52 TSP	7550	8775	25
ST 70 TSP	678	825	3
PR 76 TSP	109700	126605	345
Random 100 TSP	7730	9200	30
KROA 100 TSP	21700	25620	80

To estimate $P\{T_{\beta i} \leq K\}$ for each value of $\beta_i \in S_\beta$, pure local search, simulated annealing, and threshold accepting were applied to the six instances of the TSP. Using the estimator (20) in

Section 5.1, H = 500 independent replications were executed for each local search algorithm applied to each instance of the TSP. Each algorithm replication (h = 1, 2,…, H) consisted of K iterations and was initialized with a randomly generated initial solution. This process resulted in a total of K solutions for each replication, $\omega^h(1)$, $\omega^h(2)$,…, $\omega^h(K)$, where $\omega^h(i) \in \Omega$, i = 1,2,…, K, h = 1, 2,…, H. Recall from Section 5.1 that $f(\omega^{h,K})$, h = 1, 2,…, H, is the minimum objective function value among these K solutions for replication h and $\omega^{h,K}$ is the associated solution.

The estimator $\hat{P}\{T_{\beta_i} \leq K\} = [\sum_{h=1}^{H} X_{K,\beta_i}(h)] / H$, i = 1, 2,…, 50 for $P\{T_{\beta_i} \leq K\}$ was computed over H = 500 independent replications of the local search algorithms. The resulting fifty estimates for $\hat{P}\{T_{\beta_i} \leq K\}$ (i.e., $\hat{P}\{T_{\beta_i} \leq K\}$, i = 1, 2,…, 50) were plotted for each local search algorithm applied to the six instances of the TSP. The results of these experiments are reported in Figures 3-8.

Two experiments consisting of different values for K total iterations were conducted for each local search algorithm applied to the six instances of the TSP, resulting in 36 experiments. These values were chosen to provide a sufficiently long algorithm run to obtain good solutions (though not necessarily the optimal solutions) to the six problems in a reasonable amount of computing time. See Table 3 for the number of iterations K implemented for each experiment.

Table 3: Number of Iterations K

Random 50	**Berlin 52**	**ST 70**	**PR 76**	**Random 100**	**KROA 100**
K = 5000	K = 5000	K = 10000	K = 10000	K = 20000	K = 20000
K = 10000	K = 10000	K = 20000	K = 20000	K = 30000	K = 30000

The minimum objective function value $f(\omega^{h,K})$ among the K solutions for replication h = 1, 2,…, H was recorded for each local search algorithm applied to the six instances of the TSP. The minimum, maximum, mean (μ) and standard deviation (σ) for $f(\omega^{h,K})$, h = 1, 2,…, H, are

all reported in Tables 4-9. The number of times that a local search algorithm reached the best-known-solution over H replications is reported as γ / H.

Figure 3: Estimate for P{$T_\beta \leq K$} for the Random 50 Traveling Salesman Problem

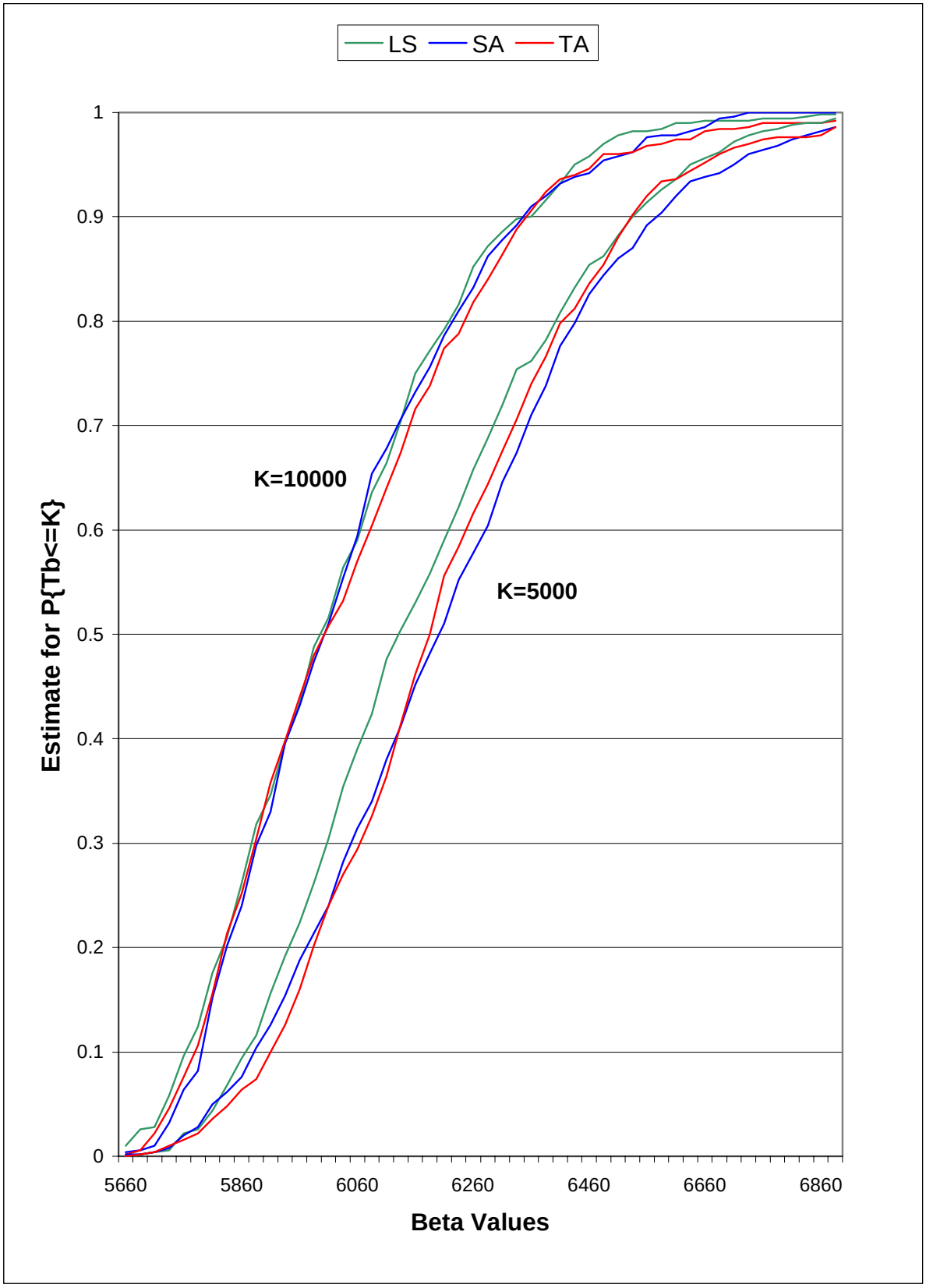


Figure 4: Estimate for $P\{T_\beta \leq K\}$ for the Berlin 52 Traveling Salesman Problem

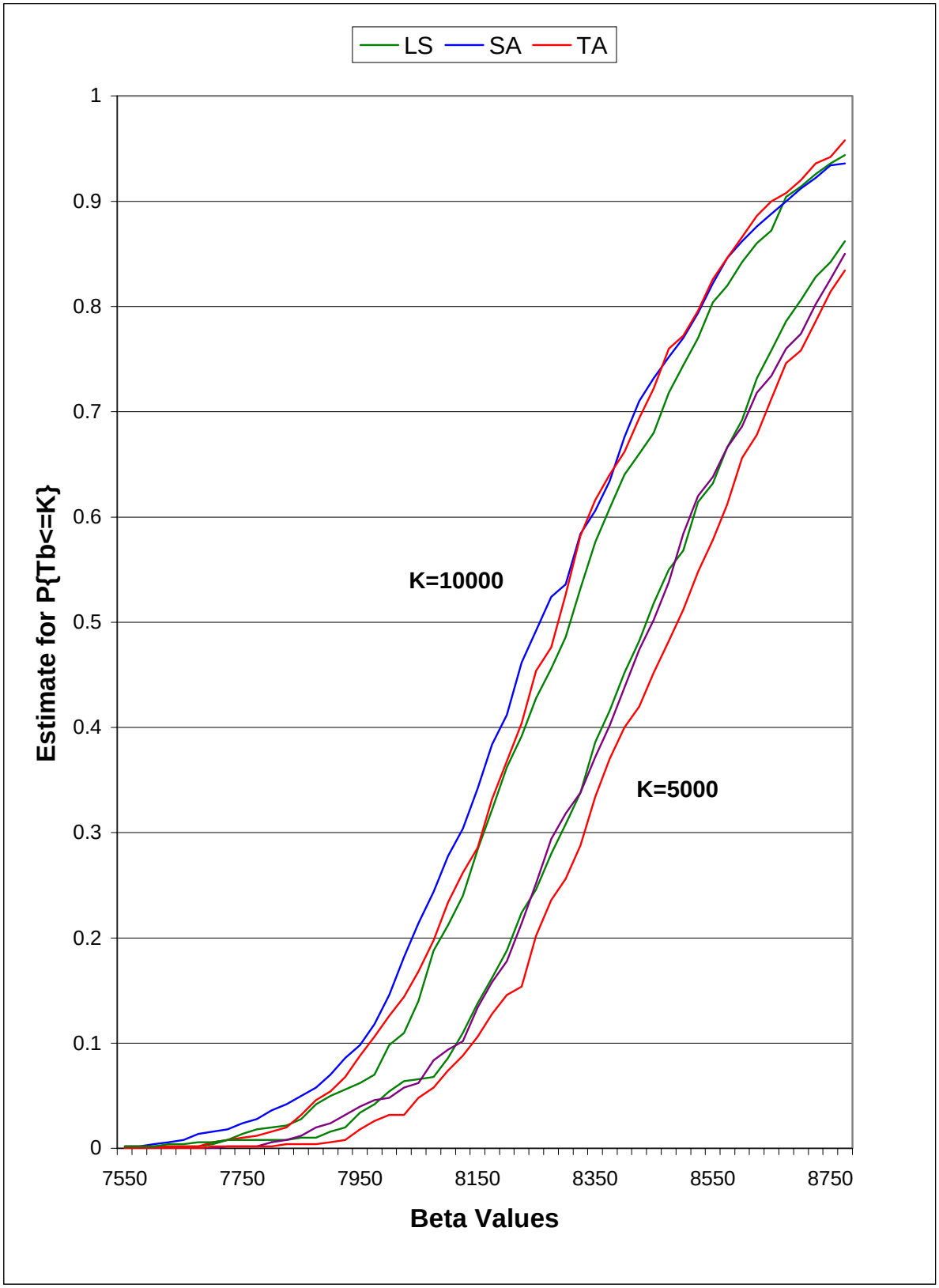


Figure 5: Estimate for $P\{T_\beta \leq K\}$ for the ST70 Traveling Salesman Problem

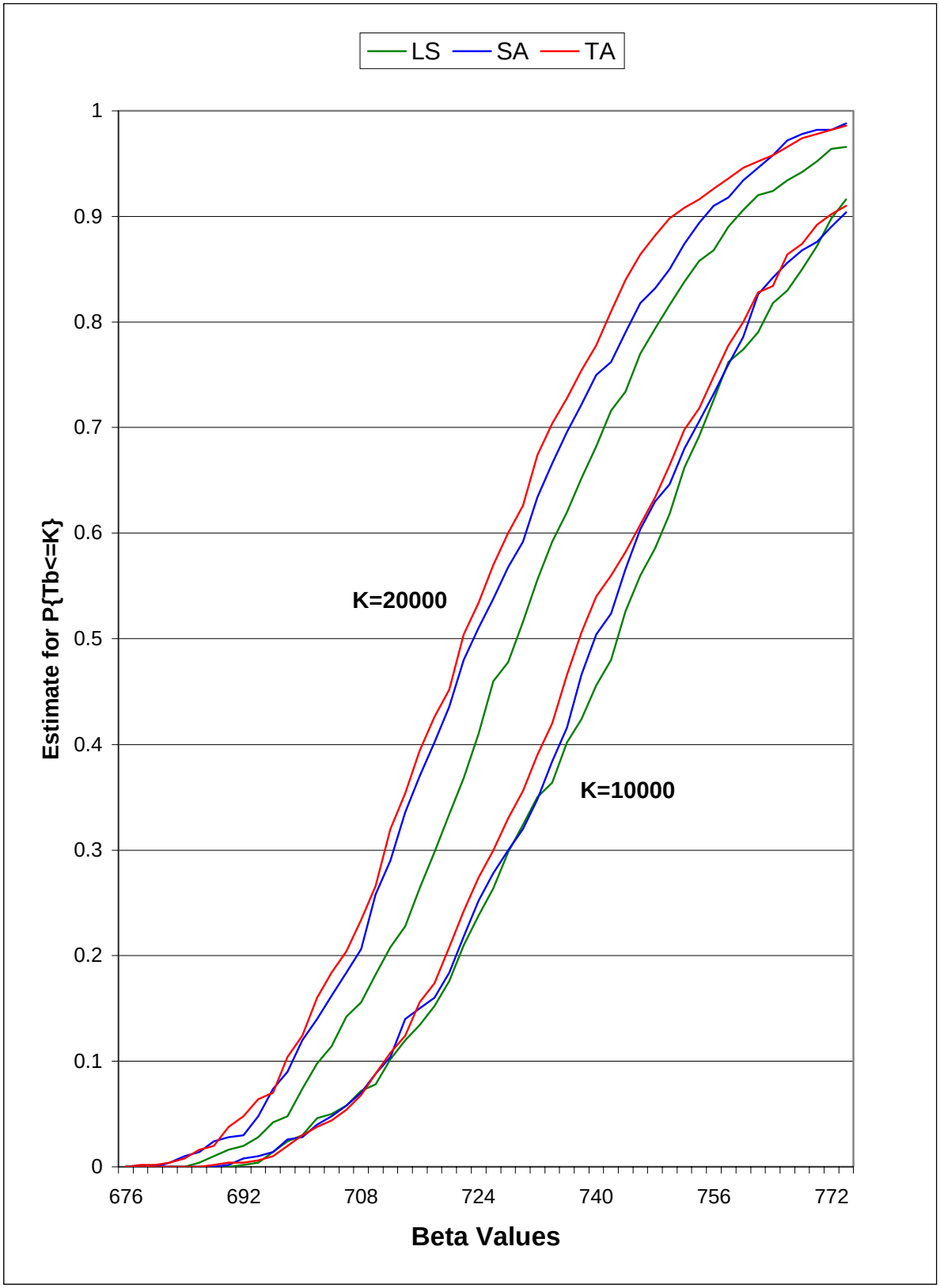


Figure 6: Estimate for $P\{T_\beta \leq K\}$ for the PR76 Traveling Salesman Problem

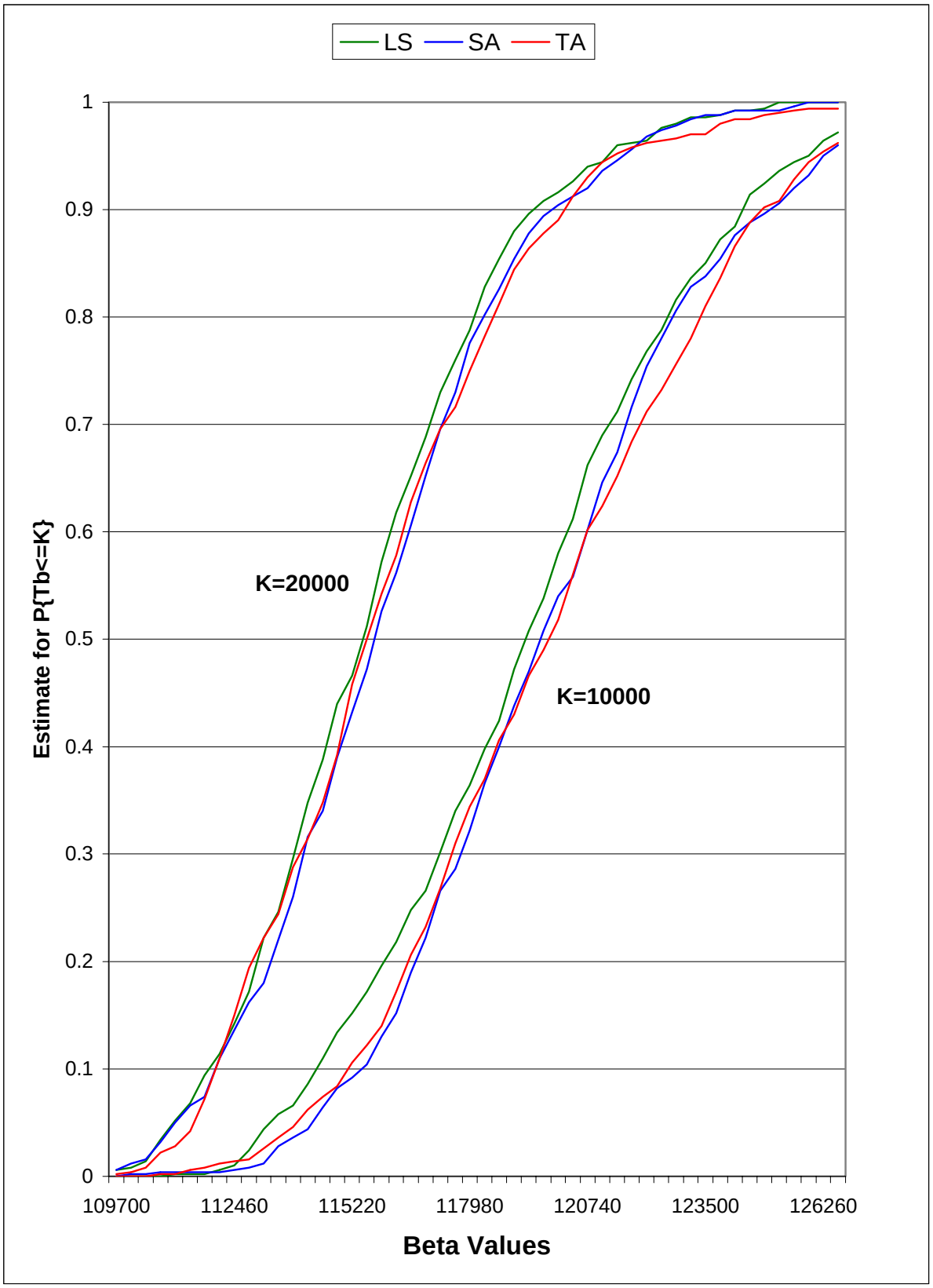


Figure 7: Estimate for $P\{T_\beta \leq K\}$ for the Random 100 Traveling Salesman Problem

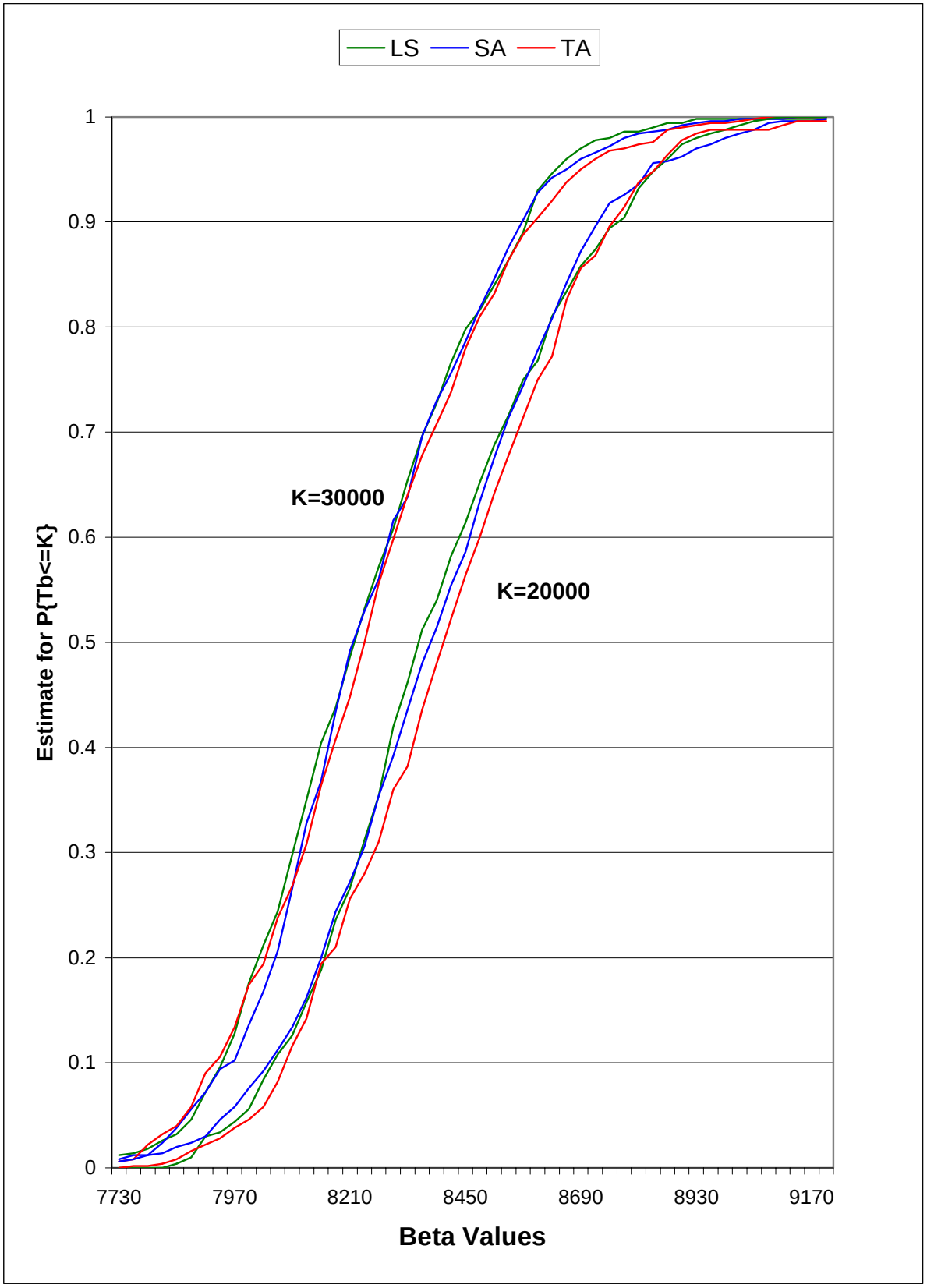


Figure 8: Estimate for $P\{T_\beta \leq K\}$ for the KROA100 Traveling Salesman Problem

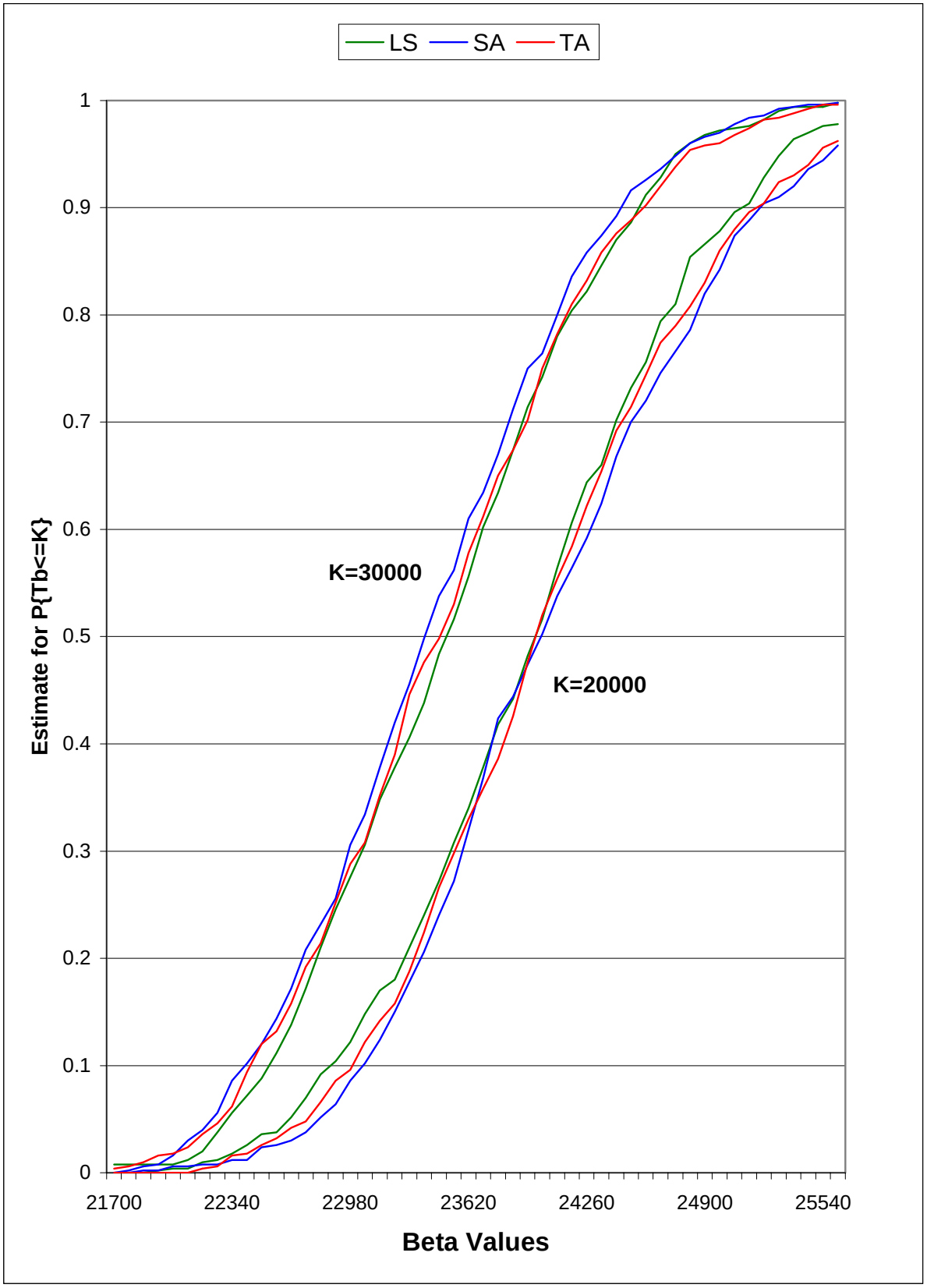


Table 4: Results for the Random 50 Traveling Salesman Problem

Algorithm	**K**	**γ/H**	**Min**	**Max**	**μ**	**σ**
Pure Local Search	5000	1/500	5692.8	6812.0	6195.9	220.4
Simulated Annealing	5000	1/500	5658.3	7301.2	6219.2	271.1
Threshold Accepting	5000	1/500	5677.5	7221.6	6201.7	254.2
Pure Local Search	10000	2/500	5657.9*	7094.1	6028.9	226.5
Simulated Annealing	10000	1/500	5657.9*	6727.5	6038.5	222.4
Threshold Accepting	10000	1/500	5658.3	7156.5	6045.7	240.8

***** The objective function value of the best-known-solution found with RRLS

Table 5: Results for the Berlin 52 Traveling Salesman Problem

Algorithm	**K**	**γ/H**	**Min**	**Max**	**μ**	**σ**
Pure Local Search	5000	1/500	7542*	9383	8456.5	290.2
Simulated Annealing	5000	1/500	7706	9377	8463.5	295.2
Threshold Accepting	5000	1/500	7694	9408	8500.9	287.0
Pure Local Search	10000	1/500	7618	9160	8321.3	266.5
Simulated Annealing	10000	1/500	7542*	9289	8287.3	289.9
Threshold Accepting	10000	1/500	7542*	9096	8299	261.5

* The objective function value of the global minimum

Table 6: Results for the ST70 Traveling Salesman Problem

Algorithm	**K**	**γ/H**	**Min**	**Max**	**μ**	**σ**
Pure Local Search	10000	1/500	692	840	743.1	21.8
Simulated Annealing	10000	1/500	690	834	741.8	23.4
Threshold Accepting	10000	1/500	688	809	740.6	23.1
Pure Local Search	20000	1/500	685	808	731.0	21.9
Simulated Annealing	20000	1/500	681	796	726.0	21.3
Threshold Accepting	20000	1/500	678	793	724.2	20.6

* The objective function value of the globally optimal solution = 675.

Table 7: Results for the PR76 Traveling Salesman Problem

Algorithm	**K**	**γ/H**	**Min**	**Max**	**μ**	**σ**
Pure Local Search	10000	1/500	110780	134000	119390	3752.4
Simulated Annealing	10000	1/500	109764	130937	119870	3629.6
Threshold Accepting	10000	1/500	111281	131726	119950	3642.7
Pure Local Search	20000	1/500	109311	125196	115640	3012.4
Simulated Annealing	20000	1/500	109577	125780	115910	3063.7
Threshold Accepting	20000	1/500	109104	127541	115930	3213.8

* The objective function value of the globally optimal solution = 108159.

Table 8: Results for the Random 100 Traveling Salesman Problem

Algorithm	**K**	**γ/H**	**Min**	**Max**	**μ**	**σ**
Pure Local Search	20000	1/500	7825.9	9086.7	8385.8	262.7
Simulated Annealing	20000	1/500	7691.5	9172.7	8364.2	284.1
Threshold Accepting	20000	1/500	7738.2	9267.4	8410.7	262.6
Pure Local Search	30000	1/500	7644.0	9222.1	8239.5	243.7
Simulated Annealing	30000	1/500	7639.2	9121.8	8249.5	241.7
Threshold Accepting	30000	1/500	7567.1*	9072.6	8254.2	255.9

* The objective function value of the best-known-solution found with RRLS

Table 9: Results for the KROA100 Traveling Salesman Problem

Algorithm	**K**	**γ/H**	**Min**	**Max**	**μ**	**σ**
Pure Local Search	20000	1/500	21865	26244	23978	823.21
Simulated Annealing	20000	1/500	21854	26551	24076	833.0
Threshold Accepting	20000	1/500	22105	27154	24046	850.7
Pure Local Search	30000	1/500	21721	22926	23520	747.11
Simulated Annealing	30000	1/500	21710	25849	23428	769.34
Threshold Accepting	30000	1/500	21680	25862	23484	793.47

* The objective function value of the globally optimal solution = 21285.

Figures 3-8 depict plots of the fifty estimates for $P\{T_\beta \le K\}$ (i.e., $\hat{P}\{T_{\beta i} \le K\}$, i = 1, 2,…, 50) for each local search algorithm applied to the six instances of the TSP. Note that for each local search algorithm, the plot of $\hat{P}\{T_{\beta i} \le K\}$ forms an S shaped curve with asymptotes at zero and one. The plots demonstrate that for the same local search algorithm applied to the same instance of the TSP, as the value of K increased, the estimates for $P\{T_{\beta i} \le K\}$ increased (i.e., $P\{T_{\beta i} \le 10000\} \le P\{T_{\beta i} \le 20000\}$, i = 1, 2,…, 50). It is also interesting to note that the difference between $\hat{P}\{T_{\beta i} \le K_1\}$ and $\hat{P}\{T_{\beta i} \le K_2\}$, $K_1 \le K_2$, for each local search algorithm was smallest at the tails of the S-shaped cure and largest for middle values of β. In particular, estimates for $P\{T_{\beta i} \le K_2\}$ for values of β near the mean (μ) for $f(\omega^{h,K})$, h = 1, 2,…, H, were 40% to 90% greater than the estimates for $P\{T_{\beta i} \le K_1\}$, $K_1 \le K_2$. Figures 3-8 suggest that for values of β significantly greater than $f(\omega^*)$, the number of iterations K can significantly impact the value for $P\{T_\beta \le K\}$, while for values of β close to $f(\omega^*)$, the number of iterations K has little impact on the difference between $\hat{P}\{T_{\beta i} \le K_1\}$ and $\hat{P}\{T_{\beta i} \le K_2\}$, $K_1 \le K_2$. This observation implies that for values of β close to $f(\omega^*)$, executing additional iterations of a local search algorithm may not lead to an increased probability of visiting solutions with lower objective function values.

The results presented in Tables 4-9 illustrate that the minimum objective function value $f(\omega^{h,K})$ among the K solutions for replication h = 1, 2,…, H improved by between 1% – 3% when the local search algorithm executed additional iterations $K_2 \ge K_1$. In addition, the mean (μ) and standard deviation (σ) values for $f(\omega^{h,K})$, h = 1, 2,…, H, reported in Tables 4-9 improve by between 2% – 3% with the additional iterations $K_2 \ge K_1$.

6.4 Computational Results for g(K,β)

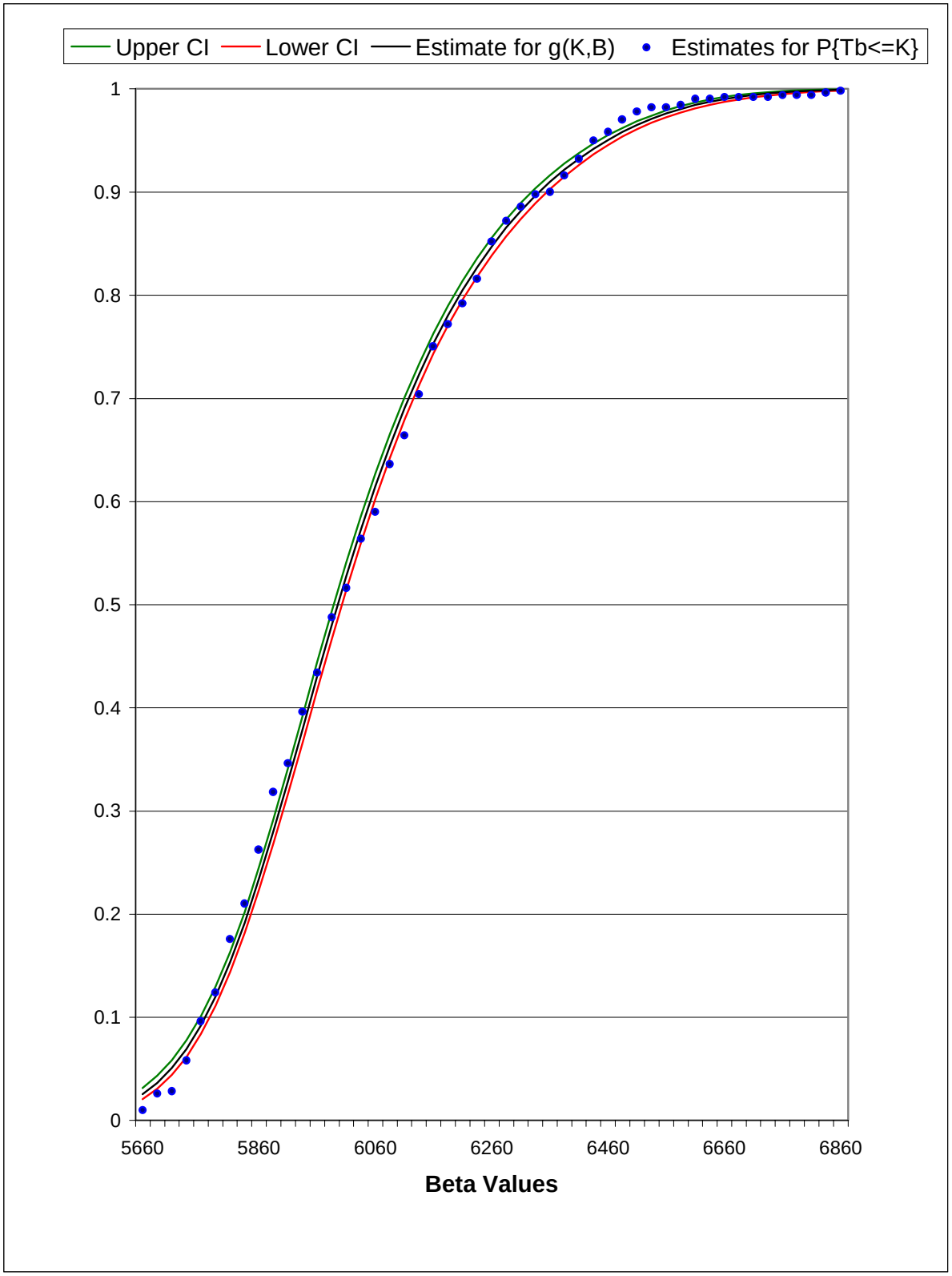
Section 6.3 presented point estimates for $P\{T_\beta \le K\}$ for values of β that are sufficiently large such that solutions with objective function value at or below these values can be visited within K iterations. However, the technique used in Section 6.3 may not be practical for estimating the probability of visiting solutions with objective function values close to $f(\omega^*)$, $\omega^* \in G$, since such events are likely to be rare. This section uses the logistic regression

model (23) presented in Section 5.2 and the data collected in Section 6.3 to estimate $P\{T_\beta \leq K\}$ for values of β close to $f(\omega^*)$, $\omega^* \in G$. Data generated from the computational experiments reported in Section 6.3 were used to estimate the parameters for the logit transformation presented in (24). The commercial statistical software package MINITAB was used to compute the estimates for the coefficients δ_0, δ_1, δ_2, and δ_3 in (24). Estimates for the coefficients δ_0, δ_1, δ_2, and δ_3 are reported in Table 10. Point estimates for $P\{T_{\beta_i} \leq K\}$ for all values of $\beta_i \in S_\beta$, i = 1,2,…,n (See Table 2) were computed using (35) and are plotted in Figures 9-20 along with a plot of $\hat{P}\{T_{\beta_i} \leq K\}$ for each instance of the TSP described in Section 6.2. The corresponding 95% confidence interval estimates for the point estimates for $P\{T_{\beta_i} \leq K\}$ were computed using (37) and are presented in Figures 9-20. Computational results are reported for pure local search and simulated annealing applied to the six instances of the TSP described in Section 6.2.

Table 10: Parameter Estimates

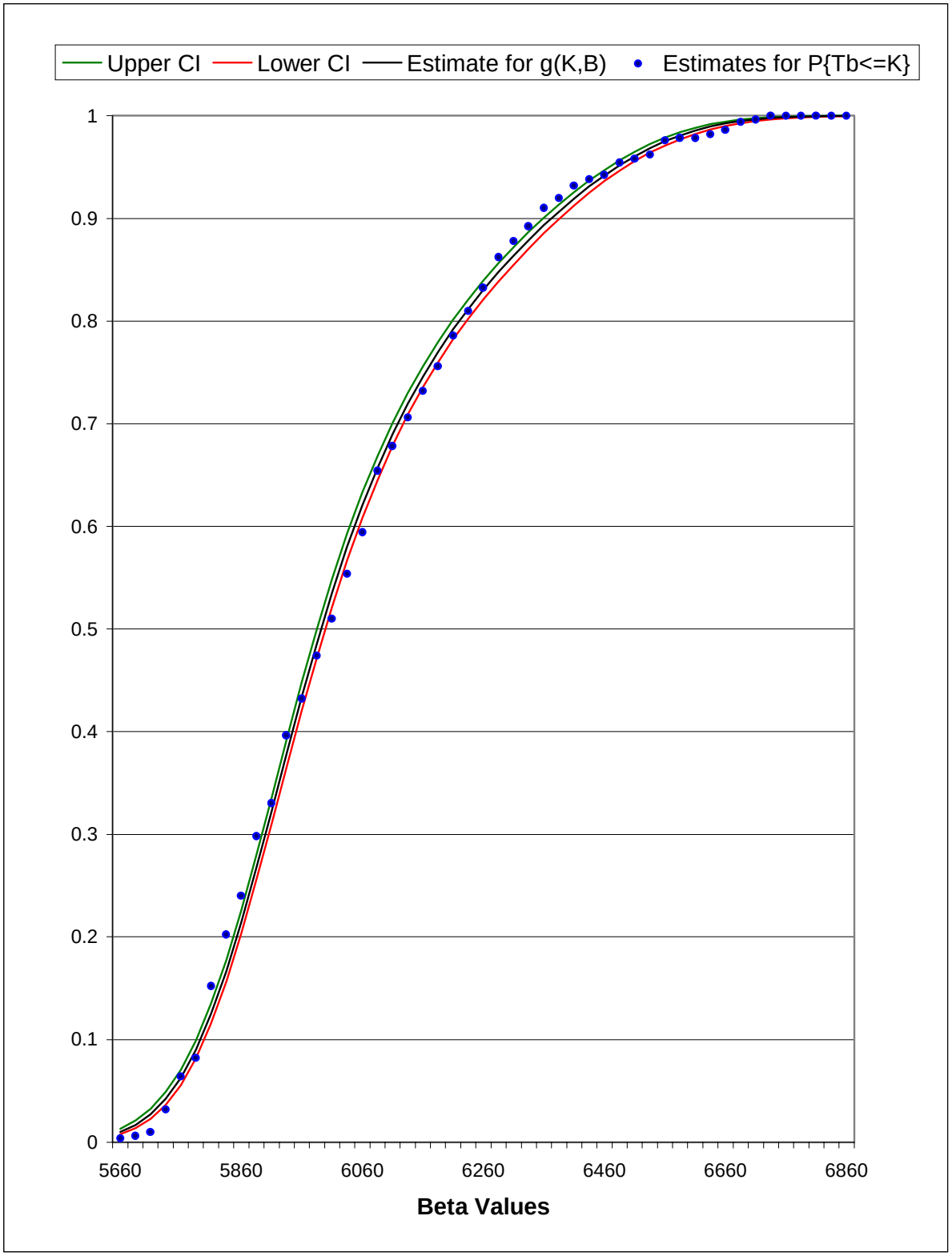
Problem (Algorithm,K)	$\hat{\delta}_0$	$\hat{\delta}_1$	$\hat{\delta}_2$	$\hat{\delta}_3$
Random 50 (LS,10000)	-2103.84	0.997153	-1.5833E-04	8.4297E-09
Random 50 (SA,10000)	-4040.68	1.934941	-3.0955E-04	1.6550E-08
Random 50 (TA,10000)	-1789.24	0.834785	-1.3046E-04	6.8377E-09
Berlin 52 (LS,10000)	-3033.95	1.067285	-1.2577E-04	4.9660E-09
Berlin 52 (SA,10000)	-1689.30	0.584894	-6.8036E-05	2.6617E-09
Berline 52 (TA,10000)	-2791.41	0.981397	-1.1564E-04	4.5687E-09
ST 70 (LS,20000)	-1911.85	7.331651	-9.4231E-03	4.0612E-06
ST 70 (SA,20000)	-2649.56	10.337106	-1.3494E-02	5.8948E-06
ST 70 (TA,20000)	-1891.46	7.316291	-9.4864E-03	4.1256E-06
PR 76 (LS,20000)	-5261.95	0.132768	-1.1203E-06	3.1627E-12
PR 76 (SA,20000)	-4120.94	0.103683	-8.7336E-07	2.4638E-12
PR 76 (TA,20000)	-4363.51	0.108837	-9.0803E-07	2.5352E-12
Random 100 (LS,30000)	-3860.41	1.372667	-1.6338E-04	6.5106E-09
Random 100 (SA,30000)	-3385.70	1.196493	-1.4163E-04	5.6174E-09
Random 100 (TA,30000)	-3385.43	1.199757	-1.4237E-04	5.6589E-09
KROA 100 (LS,30000)	-4012.27	0.504143	-2.1191E-05	2.9803E-10
KROA 100 (SA,30000)	-4025.21	0.506512	-2.1320E-05	3.0021E-10
KROA 100 (TA,30000)	-3282.24	0.411214	-1.7246E-05	2.4216E-10

Figure 9: Estimators for the Random 50 Traveling Salesman Problem (LS)



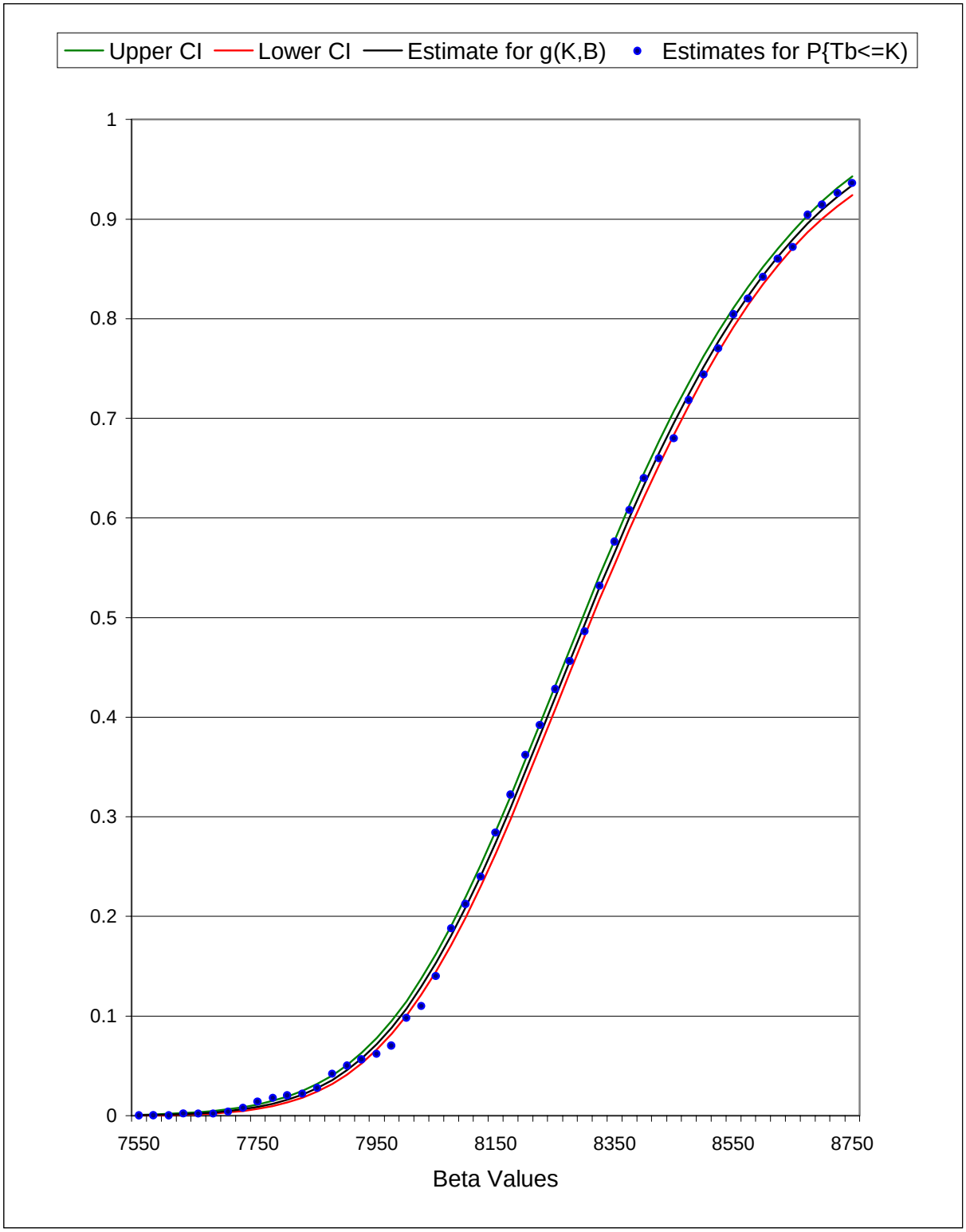
$$\hat{f}(K,\beta) = -2103.8 + 0.9972\beta - 1.583E\text{-}04\beta^2 + 8.4297E\text{-}09\beta^3$$

Figure 10: Estimators for the Random 50 Traveling Salesman Problem (SA)



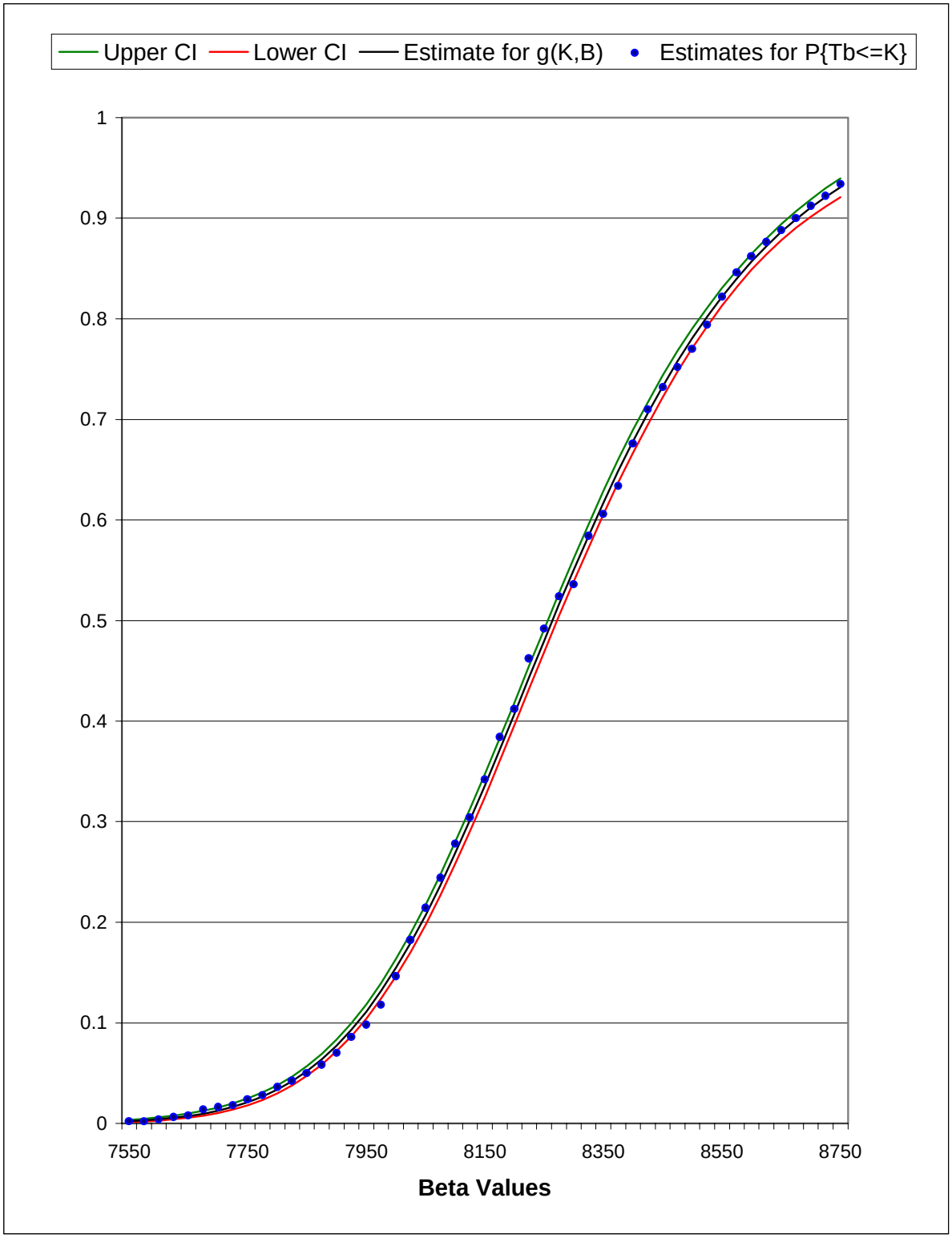
$$\hat{f}(K,\beta) = -4040.7 + 1.9349\beta - 3.095E\text{-}04\beta^2 + 1.6550E\text{-}08\beta^3$$

Figure 11: Estimators for the Berlin 52 Traveling Salesman Problem (LS)



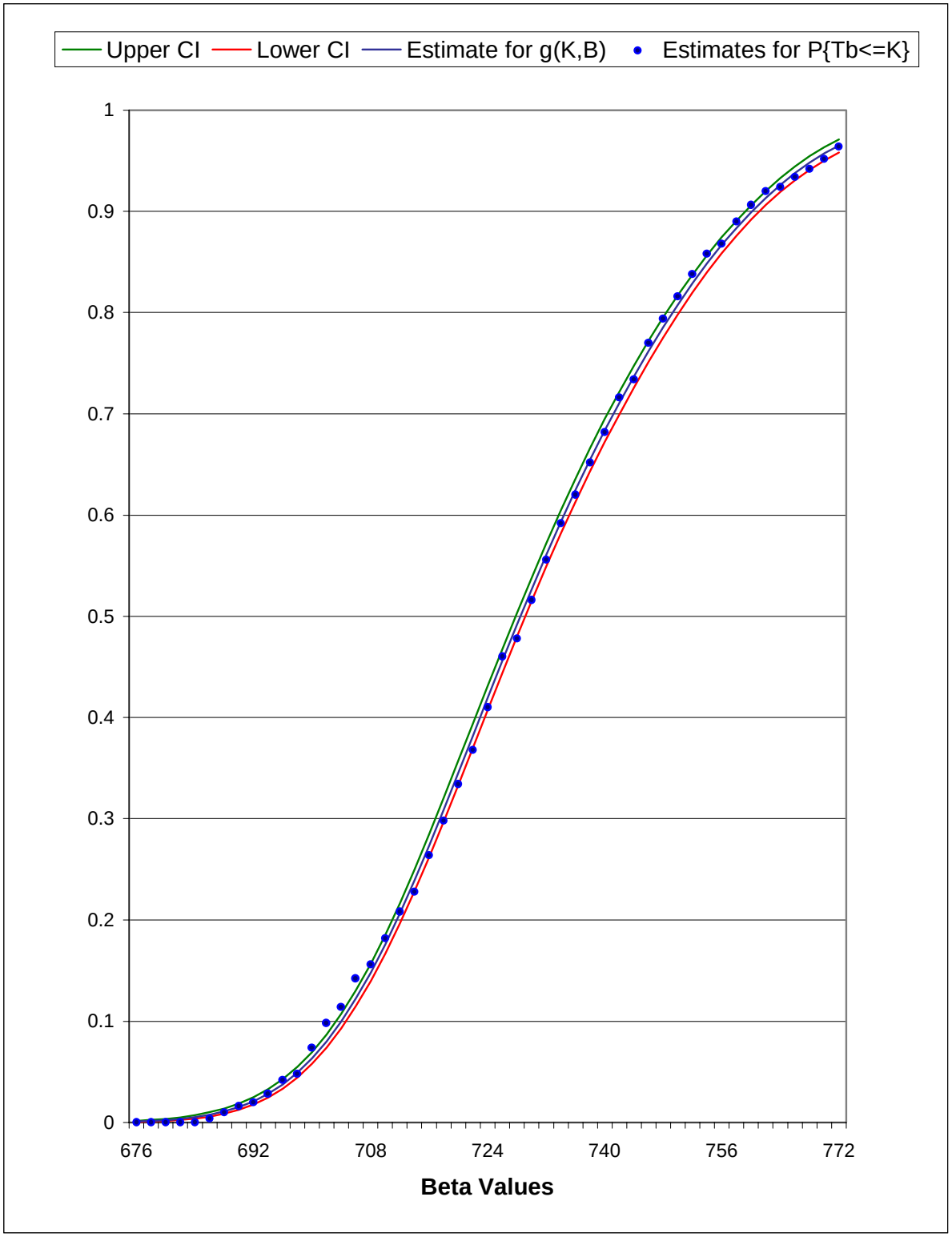
$$\hat{f}(K,\beta) = -3033.9 + 1.0673\beta - 1.258E\text{-}04\beta^2 + 4.966E\text{-}09\beta^3$$

Figure 12: Estimators for the Berlin 52 Traveling Salesman Problem (SA)



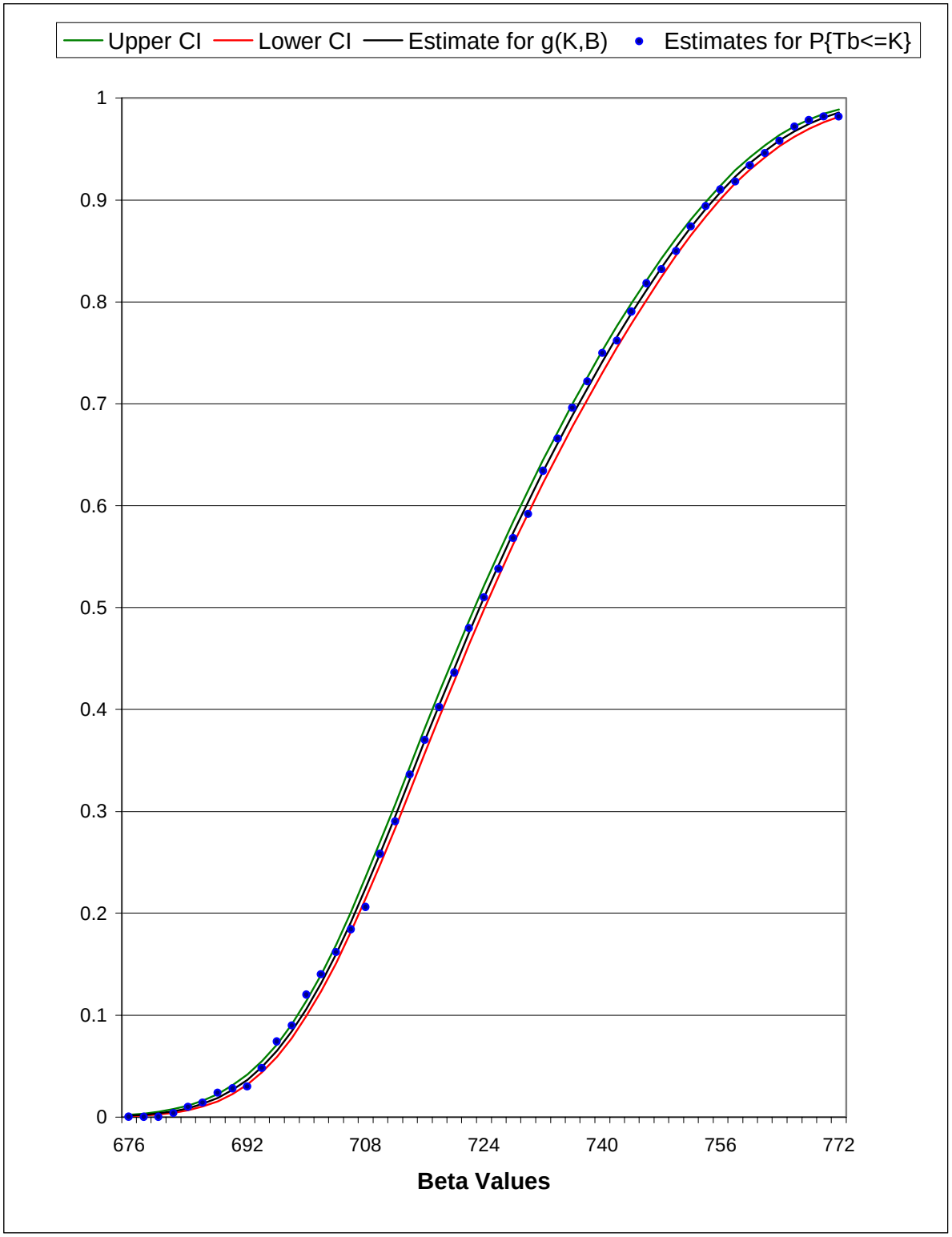
$\hat{f}(K,\beta) = -1689.3 + 0.5849\beta - 6.8E\text{-}05\beta^2 + 2.6617E\text{-}09\beta^3$

Figure 13: Estimators for the ST70 Traveling Salesman Problem (LS)



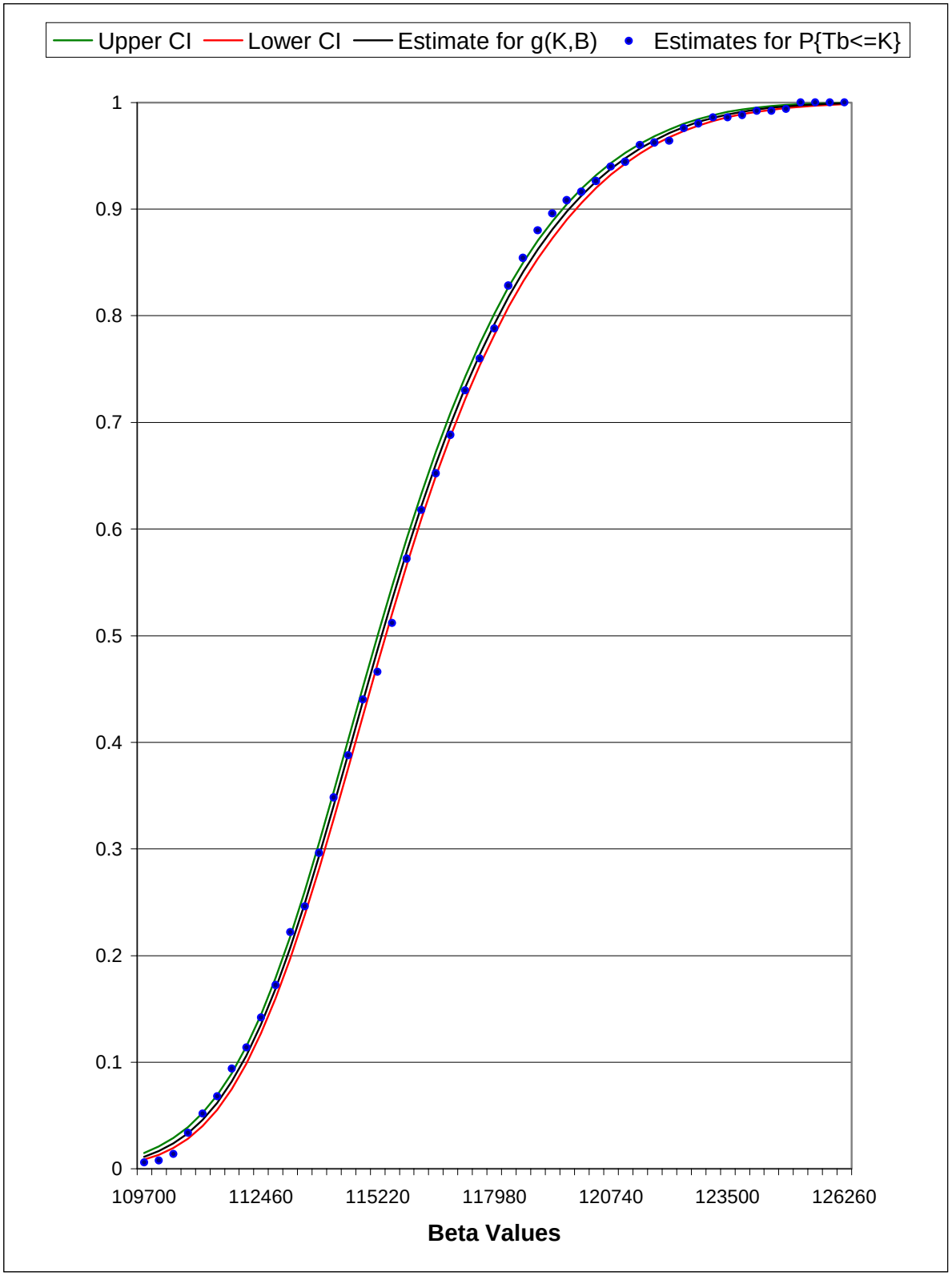
$$\hat{f}(K,\beta) = -1911.9 + 7.3317\beta - 0.009423\beta^2 + 4.0612E\text{-}06\beta^3$$

Figure 14: Estimators for the ST70 Traveling Salesman Problem (SA)



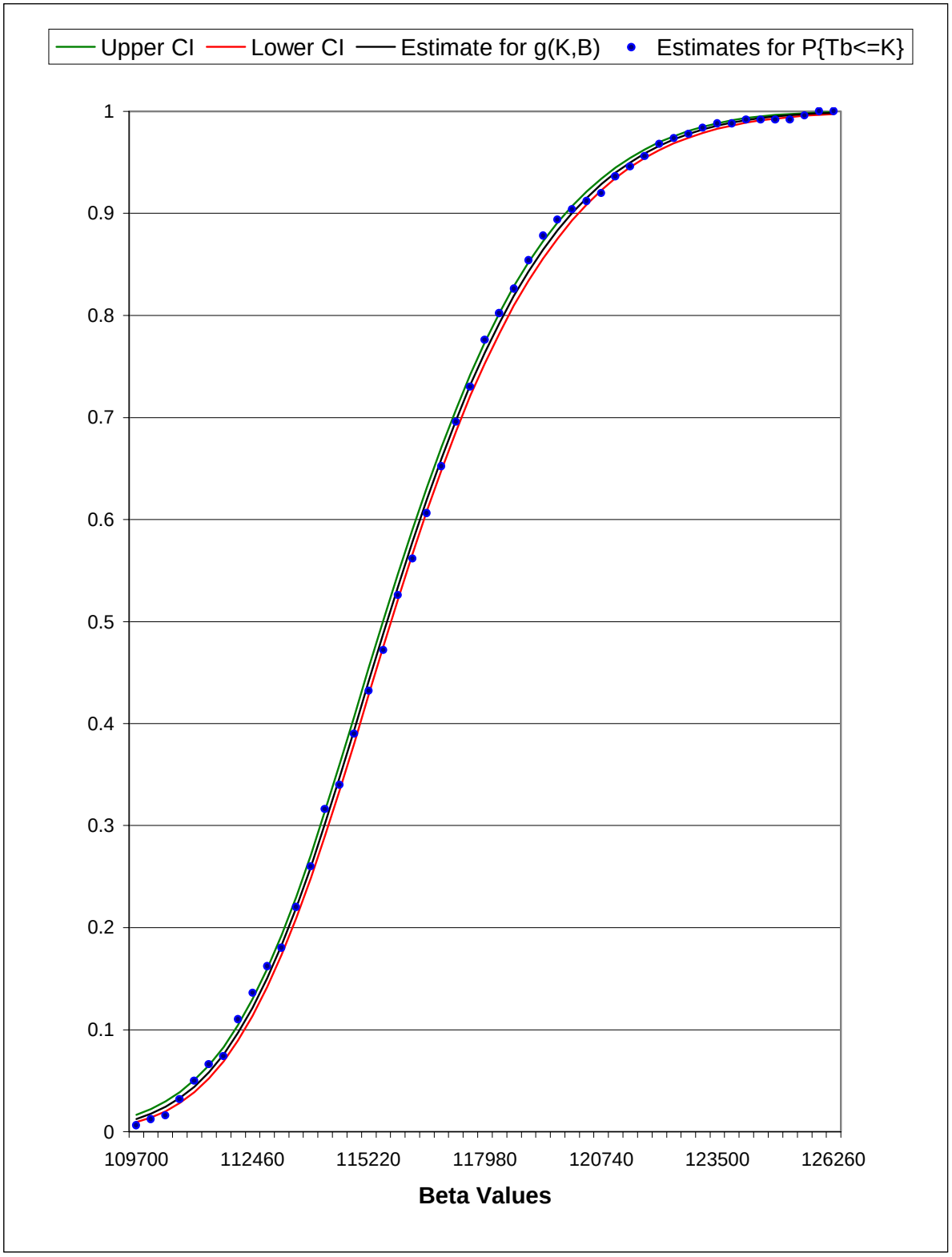
$$\hat{f}(K,\beta) = -2649.6 + 10.3371\beta - 0.013494\beta^2 + 5.8948E\text{-}06\beta^3$$

Figure 15: Estimators for the PR76 Traveling Salesman Problem (LS)



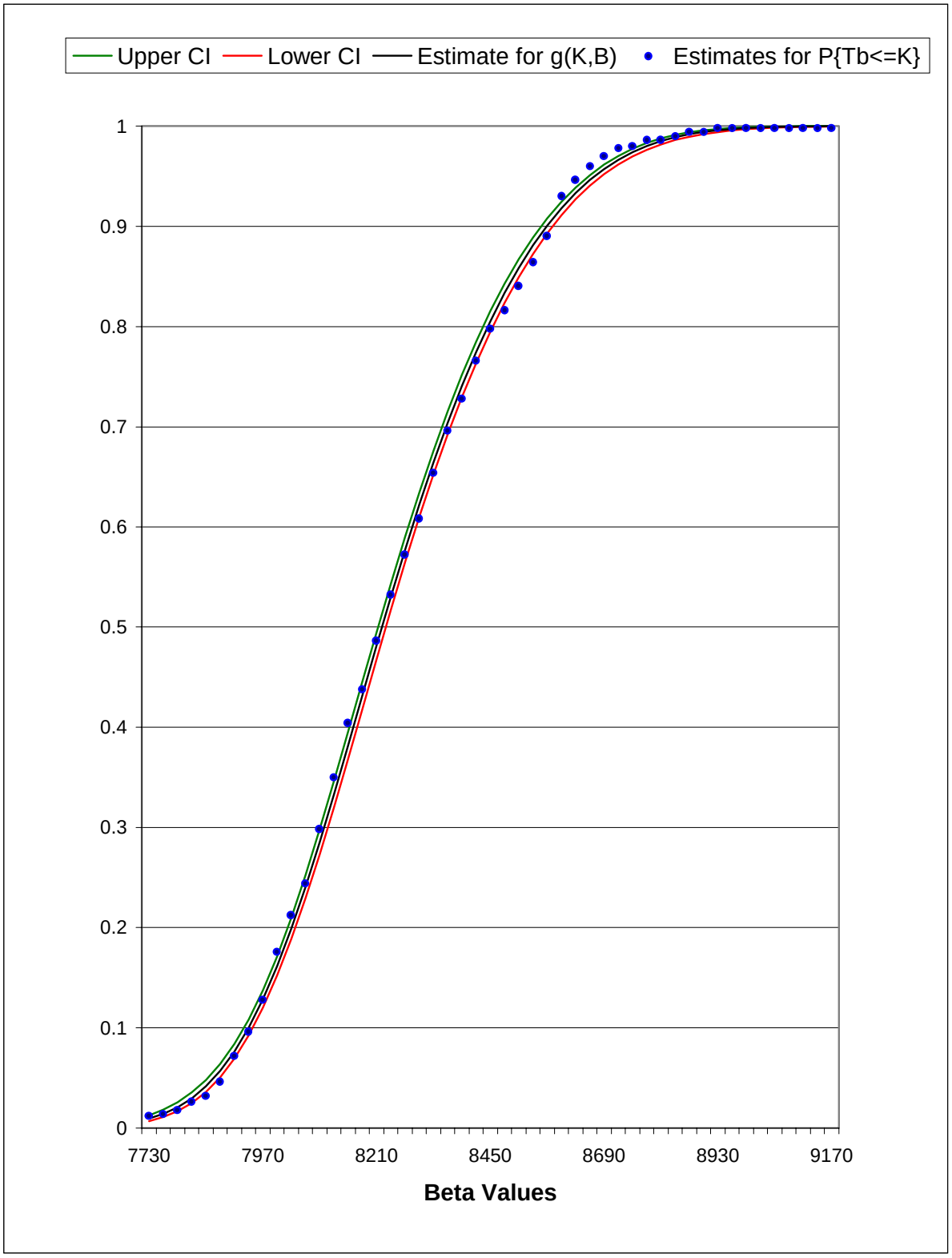
$$\hat{f}(K,\beta) = -5261.9 + 0.13277\beta - 1.120E\text{-}06\beta^2 + 3.1627E\text{-}12\beta^3$$

Figure 16: Estimators for the PR76 Traveling Salesman Problem (SA)



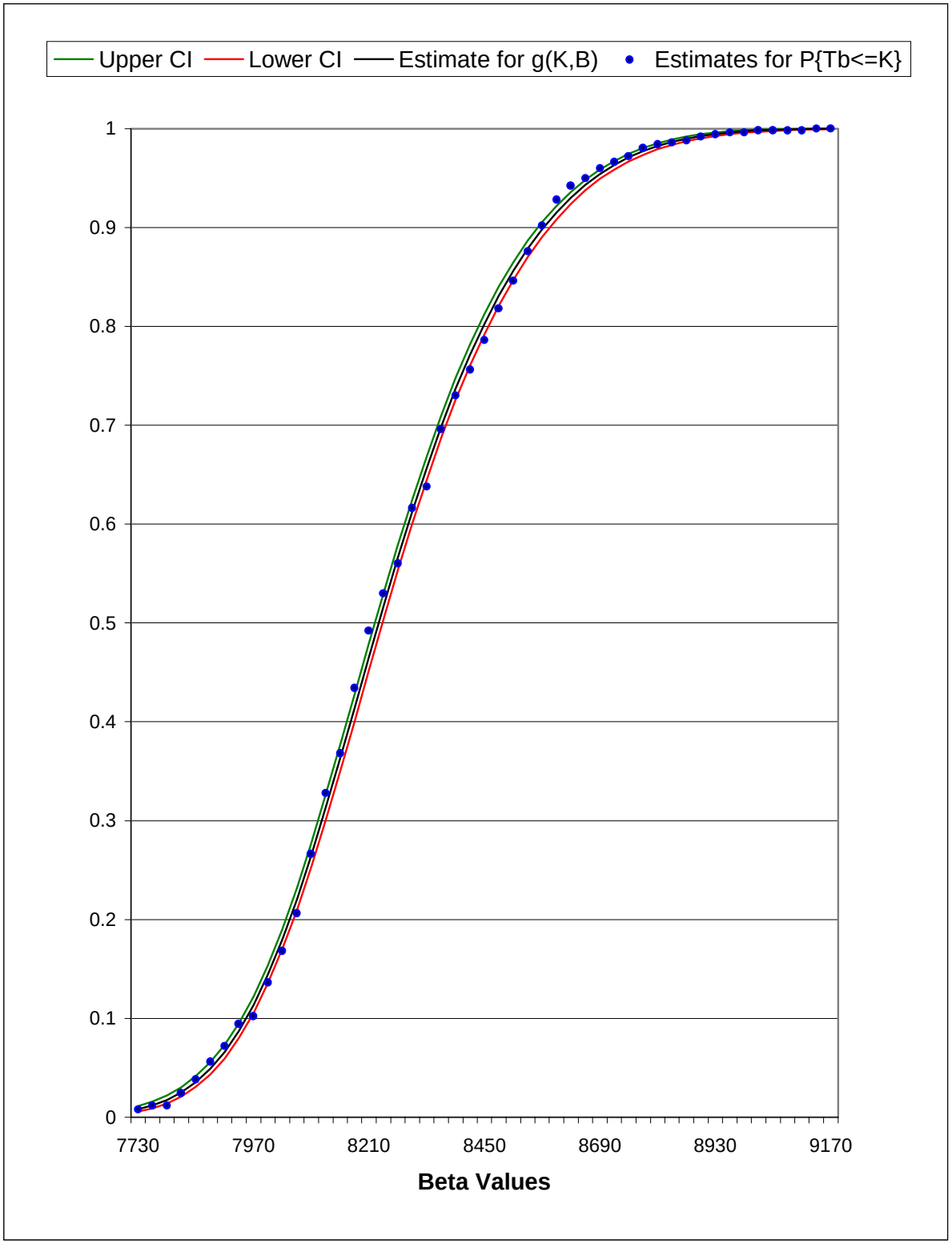
$$\hat{f}(K,\beta) = -4120.9 + 0.10368\beta - 8.734E\text{-}07\beta^2 + 2.4638E\text{-}12\beta^3$$

Figure 17: Estimators for the Random 100 Traveling Salesman Problem (LS)



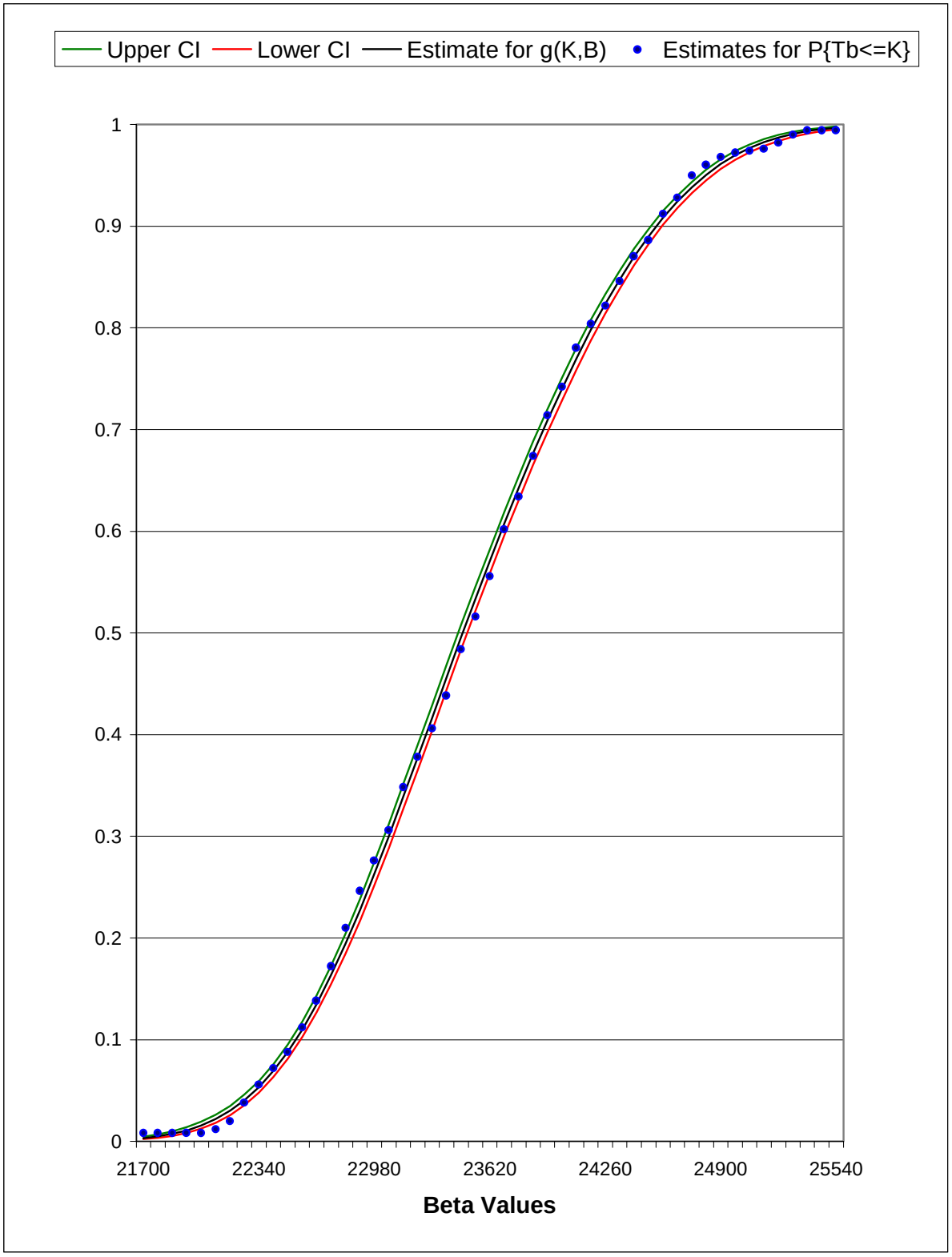
$$\hat{f}(K,\beta) = -3860.41 + 1.372667\beta - 1.6338E\text{-}04\beta^2 + 6.5106E\text{-}09\beta^3$$

Figure 18: Estimators for the Random 100 Traveling Salesman Problem (SA)



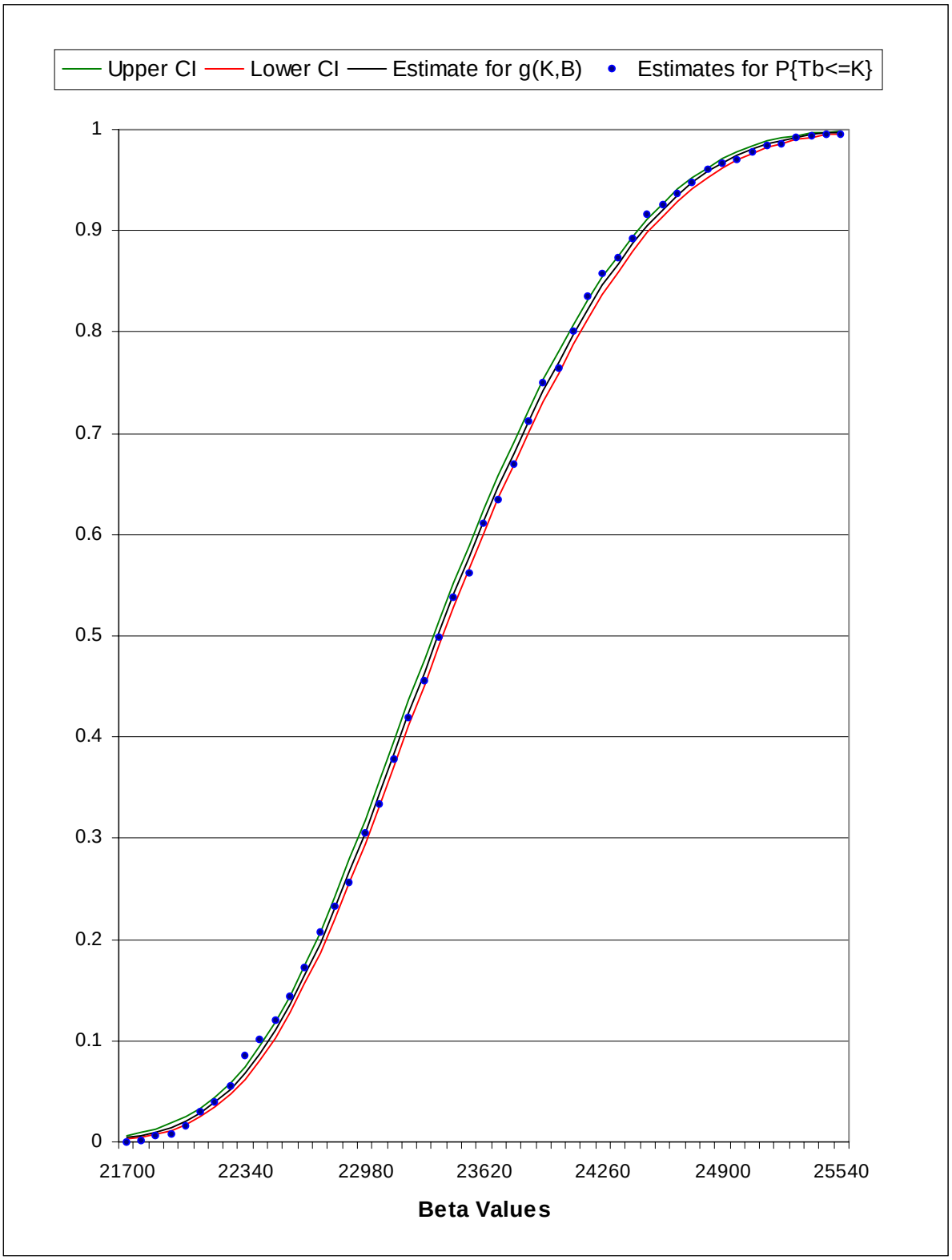
$$\hat{f}(K,\beta) = -3385.7 + 1.1965\beta - 1.416\text{E-}04\beta^2 + 5.6174\text{E-}09\beta^3$$

Figure 19: Estimators for the KROA100 Traveling Salesman Problem (LS)



$$\hat{f}(K,\beta) = -4012.3 + 0.50414\beta - 2.119E\text{-}05\beta^2 + 2.9803E\text{-}10\beta^3$$

Figure 20: Estimators for the KROA100 Traveling Salesman Problem (SA)



$$\hat{f}(K,\beta) = -4025.2 + 0.50651\beta - 2.132E\text{-}05\beta^2 + 3.0021E\text{-}10\beta^3$$

The curves depicted in Figures 9-20 demonstrate that the estimates for the coefficients δ_0, δ_1, δ_2, and δ_3 in (24) obtained by applying logistic regression to the data generated in Section 6.3 result in a model that provides a reasonable estimate for $P\{T_\beta \leq K\}$. The logistic regression model in (23) can be used to investigate the behavior of local search algorithms applied to instances of discrete optimization problems. Define ρ as a surrogate for $E[X_{K,\beta}] = P\{T_\beta \leq K\} = g(K,\beta)$ in (21). The logistic regression model can be used to investigate the impact on ρ for different values of $\beta = (1+\lambda)\, f(\omega^*)$ where $0 \leq \lambda \leq 1$ and ω^* is the globally optimal solution or the best-known-solution. A summary of ρ for values of β greater than the objective function value of the globally optimal solution or the best-known-solution for each instance of the TSP is presented in Table 11.

Table 11: ρ Values for Different Values of β = (1+λ)f(ω*)

Problem (Algorithm,K)	ρ				
	λ = 0	**λ = .05**	**λ = .10**	**λ = .15**	**λ = .20**
Random 50 (LS,10000)	0.024679	0.392028	0.817929	0.964243	0.997697
Random 50 (SA,10000)	0.009754	0.392226	0.803343	0.959626	0.999155
Random 50 (TA,10000)	0.021218	0.372665	0.804186	0.947861	0.992333
Berlin 52 (LS,10000)	0.000384	0.054684	0.487830	0.894267	0.995023
Berlin 52 (SA,10000)	0.002038	0.089193	0.545250	0.898152	0.989210
Berline 52 (TA,10000)	0.000548	0.062669	0.520082	0.910213	0.995812
ST 70 (LS,20000)	0.000606	0.041385	0.333477	0.701972	0.926446
ST 70 (SA,20000)	0.000851	0.070952	0.429201	0.757503	0.958627
ST 70 (TA,20000)	0.001717	0.079572	0.456407	0.799104	0.963097
PR 76 (LS,20000)	0.001605	0.258148	0.860223	0.994439	0.999994
PR 76 (SA,20000)	0.002262	0.227744	0.840507	0.992594	0.999980
PR 76 (TA,20000)	0.001395	0.235604	0.835975	0.985922	0.999887
Random 100 (LS,30000)	0.000690	0.104421	0.656211	0.961066	0.999532
Random 100 (SA,30000)	0.000607	0.091547	0.648649	0.957932	0.999164
Random 100 (TA,30000)	0.000931	0.103778	0.630335	0.948454	0.998928
KROA 100 (LS,30000)	0.000240	0.054546	0.470936	0.883883	0.996794
KROA 100 (SA,30000)	0.000352	0.069754	0.516957	0.899615	0.997413
KROA 100 (TA,30000)	0.000542	0.066621	0.487530	0.880750	0.994825

The results in Table 11 suggest that the logistic regression model in (23) can be used to assist practitioners in determining a set of β acceptable solutions that can be reached in a fixed number of iterations. For a fixed number of iterations K, the logistic regression model in (23) can be used to evaluate a range of β values, $\beta_i \in S_\beta$, i = 1,2,…,n, to see if the algorithm has a positive probability of visiting a β-acceptable solution in this range. For example, when pure local search was applied the randomly generated 50 city TSP (Random 50) the values of β ranging from $f(\omega^*) \le \beta \le 1.2f(\omega^*)$ resulted in $0.024679 \le \rho \le 0.997697$. Note that in Tables 4-9, all but one value for γ/H is equal to 1/500 and that thirteen of the eighteen

estimated values of ρ corresponding to $\beta = f(\omega^*)$ (i.e., $\lambda = 0$, hence β is the objective function value of the best-known-solution) in Table 11 are less than 1/500. Therefore, setting ρ=1/500 may provide a reasonable upper bound estimate for $f(\omega^*)$.

One goal of this research is to determine reasonable termination conditions for non-convergent local search algorithms. If a particular algorithm can reach a solution that is "*good enough*" (i.e., reach an acceptable value of β) in a "*reasonable*" amount of time (i.e., number of iterations K) then practitioners can terminate the algorithm at iteration K and be confident that an acceptable solution was achieved. Figures 9-20 illustrate that the logistic regression model presented in (23) can be used to estimate $P\{T_\beta \le K\}$ using the estimated values for $P\{T_{\beta i} \le K\}$, $\beta_i \in S_\beta$, i = 1,2,…, 50 generated from the computational experiments presented in Section 6.3. The estimated logistic regression model in (23) with the corresponding parameter estimates presented in Table 10 can be used to estimate the probability of reaching smaller values of β over K iterations for the six instances of the TSP described in Section 6.3 that would otherwise be too difficult to determine computationally. The logistic regression model in (23) allows practitioners to evaluate a range of values for β (i.e., $\beta_i \in S_\beta$, i = 1, 2,…, n) by providing an estimate for the probability of visiting solutions within K iterations with objective function values in the set S_β. This provides practitioners with a tool for determining if a local search algorithm has a reasonable probability of visiting a solution with an objective function value that is "*good enough*" (i.e., value of β) in a "*reasonable*" amount of time (i.e., K iterations).

6.5 Estimating the Globally Optimal Solution Objective Function Value

Another goal of this research is to use logistic regression to estimate the objective function value of the globally optimal solution for a discrete optimization problem. To assist practitioners in determining realistic expectations for an algorithm applied to a particular instance of a discrete optimization problem, the estimated logit function (33) corresponding to the estimated logistic regression model in (34) can be used to estimate values of β for a range of values for ρ. Recall that ρ is a surrogate for $P\{T_\beta \le K\}$ and that the function $g(K,\beta)$ is used to model $P\{T_\beta \le K\}$ (see (21)). From (24) and (33),

$$\hat{f}(K,\beta) = \ln\left[\frac{\rho}{1-\rho}\right] = \hat{\delta}_0 + \hat{\delta}_1\beta + \hat{\delta}_2\beta^2 + \hat{\delta}_3\beta^3 \tag{38}$$

can be used to estimate values of β for a range of ρ values, where $\hat{\delta}_0$, $\hat{\delta}_1$, $\hat{\delta}_2$, and $\hat{\delta}_3$ are the estimators for the coefficients δ_0, δ_1, δ_2, and δ_3, respectively, in (24). Define $\hat{\beta}$ as an estimator for β computed at a given value of ρ using (38). The coefficient estimators $\hat{\delta}_0$, $\hat{\delta}_1$, $\hat{\delta}_2$, and $\hat{\delta}_3$ were computed using logistic regression applied to the sample data generated in the experiments described in Section 6.3. Expression (38) is used to compute $\hat{\beta}$ for a range of ρ values. The solution to (38) results in one or three real values for $\hat{\beta}$, since solving (38) is equivalent to finding the roots of a third order polynomial (this follows from the fundamental theorem of algebra; see Edwards and Penny 1986, p.975).

Expression (38) can also be used to estimate the objective function value of the globally optimal solution for an instance of a discrete optimization problem. If ρ is chosen sufficiently small such that it corresponds to a rare event (e.g., $\beta = f(\omega^*)$), then (38) can be used to solve for the corresponding value for β which results in an estimate for the objective function value of the globally optimal solution. Note that the logistic regression models may not be accurate when extrapolated outside the range of predictor variables used to compute the estimators for the model (Montgomery and Peck 1982). Table 12 reports estimates for β associated with values for ρ ranging from 0.0001 to 0.1 using (38). Note that the column of values in Table 12 related to $\rho = .002$ correspond to 1/H, where H = 500 replications were implemented for each local search algorithm applied to each instance of the TSP described in Section 6.3. Results are reported for pure local search, simulated annealing and threshold accepting applied to the six instances of the TSP described in Section 6.3.

Recall that in Section 6.3, estimates for $P\{T_{\beta_i} \le K\}$, $\beta_i \in S_\beta$, were computed by executing H = 500 independent replications of each local search algorithm applied to the six instances of the TSP. Values of $\hat{\beta}$ were computed using (38) for the range of values for ρ found in Table

13 and compared to the value of the best-known-solution f(ω*) for each instance of the TSP described in Section 6.3. The *percent increase/decrease* (i.e., error) for each $\hat{\beta}$ compared to f(ω*) (i.e., $100(\hat{\beta} - f(\omega^*))/f(\omega^*)$ was computed for four values of ρ and is reported in Table 13, where f(ω*) denotes the objective function value of the best-known-solution for each instance of the TSP (i.e., the known optimal solution for instances of the TSP from the TSP Library (Reinelt 1991) and the best-known-solution for randomly generated instances of the TSP). The mean $\bar{\rho}$ and standard deviation s_ρ of the percent increase/decrease for every $\hat{\beta}$ compared to f(ω*) corresponding to each value of ρ are reported at the bottom of Table 13, as are the mean $\bar{\rho}^a$ and standard deviation s_ρ^a of the absolute values of the percent increase/decrease for every $\hat{\beta}$ compared to f(ω*) corresponding to each value of ρ. Results are reported for pure local search, simulated annealing and threshold accepting applied to the six instances of the TSP described in Section 6.3.

The bold values in Table 13 represent percent increase/decrease for values of $\hat{\beta}$ compared to f(ω*) with absolute value less than one percent (i.e., $|(\hat{\beta} - f(\omega^*)) / f(\omega^*)| \leq .01$). These bold values, as well as the values for $\bar{\rho}$ and $\bar{\rho}^a$, suggest that a value of ρ = 1/2H, H=500 provides reasonable estimates for the objective function value of the globally optimal solution for the three local search algorithms applied to the instances of the TSP described in Section 6.3. Note that these results are for experiments using pure local search, simulated annealing and threshold accepting only and may be different for other local search algorithms applied to the instances of the TSP used in the experiments described in Section 6.4, or for other discrete optimization problems.

Table 12: Estimated Objective Function Values Given Different Values for ρ

Problem (Algorithm,K)		$\hat{\beta}$			
	f(ω*)	**ρ = .0001**	**ρ = .001**	**ρ = .002**	**ρ = .01**
Random 50 (LS,10000)	5657.90	5380.60	5479.18	5512.74	5600.44
Random 50 (SA,10000)	5657.90	5487.38	5564.13	5590.32	5659.07
Random 50 (TA,10000)	5657.90	5384.45	5486.07	5520.49	5609.96
Berlin 52 (LS,10000)	7542.00	7469.86	7598.69	7643.10	7760.62
Berlin 52 (SA,10000)	7542.00	7334.92	7488.25	7540.55	7677.19
Berline 52 (TA,10000)	7542.00	7447.10	7579.03	7624.38	7743.99
ST 70 (LS,20000)	675.00	664.66	678.17	682.84	695.25
ST 70 (SA,20000)	675.00	664.07	675.91	680.01	690.95
ST 70 (TA,20000)	675.00	658.18	671.46	676.04	688.14
PR 76 (LS,20000)	108159.00	106407.58	107830.71	108317.06	109592.20
PR 76 (SA,20000)	108159.00	105936.67	107518.57	108059.28	109475.79
PR 76 (TA,20000)	108159.00	106446.20	107921.95	108423.67	109732.08
Random 100 (LS,30000)	7567.10	7470.92	7587.49	7627.61	7733.58
Random 100 (SA,30000)	7567.10	7474.14	7595.36	7636.89	7746.02
Random 100 (TA,30000)	7567.10	7449.18	7571.28	7613.29	7724.29
KROA 100 (LS,30000)	21282.00	21163.06	21495.95	21611.52	21920.34
KROA 100 (SA,30000)	21282.00	21111.13	21438.95	21552.54	21855.46
KROA 100 (TA,30000)	21282.00	21028.97	21383.70	21506.57	21833.87

Table 13: Estimated Optimal Objective Function Value Error for a Range of ρ Values

Problem (Algorithm,K)	**f(ω*)**	**$(\hat{\beta} - f(\omega^*))/f(\omega^*)$**			
		ρ = .0001	**ρ = .001**	**ρ = .002**	**ρ = .01**
Random 50 (LS,10000)	5657.90	-4.90%	-3.16%	-2.57%	-1.02%
Random 50 (SA,10000)	5657.90	-3.01%	-1.66%	-1.19%	**0.02%**
Random 50 (TA,10000)	5657.90	-4.83%	-3.04%	-2.43%	**-0.85%**
Berlin 52 (LS,10000)	7542.00	**-0.96%**	**0.75%**	1.34%	2.90%
Berlin 52 (SA,10000)	7542.00	-2.75%	**-0.71%**	**-0.02**%	1.79%
Berline 52 (TA,10000)	7542.00	-1.26%	**0.49%**	1.09%	2.68%
ST 70 (LS,20000)	675.00	-1.53%	**0.47%**	1.16%	3.00%
ST 70 (SA,20000)	675.00	-1.62%	**0.13%**	**0.74%**	2.36%
ST 70 (TA,20000)	675.00	-2.49%	**-0.52%**	**0.15%**	1.95%
PR 76 (LS,20000)	108159.00	-1.62%	**-0.30%**	**0.15%**	1.33%
PR 76 (SA,20000)	108159.00	-2.05%	**-0.59%**	**-0.09%**	1.22%
PR 76 (TA,20000)	108159.00	-1.58%	**-0.22%**	**0.24%**	1.45%
Random 100 (LS,30000)	7657.10	-1.27%	**0.27%**	**0.80%**	2.20%
Random 100 (SA,30000)	7657.10	-1.23%	**0.37%**	**0.92%**	2.36%
Random 100 (TA,30000)	7657.10	-1.56%	**0.06%**	**0.61%**	2.08%
KROA 100 (LS,30000)	21282.00	**-0.56%**	1.01%	1.55%	3.00%
KROA 100 (SA,30000)	21282.00	**-0.80%**	**0.74%**	1.27%	2.69%
KROA 100 (TA,30000)	21282.00	-1.19%	**0.48%**	1.06%	2.59%
$\bar{\rho}$, s_ρ		-1.96,1.24	-0.30,1.20	0.27,1.20	1.76,1.24
$\bar{\rho}^a$, s_ρ^a		1.96,1.24	0.83,0.90	0.97,0.73	1.97,0.84

The results presented in Table 13 suggest that (38) can be used to estimate the objective function value of the globally optimal solution with reasonable accuracy. These results suggest that the cubic logistic regression model presented in (23) combined with (38) provide practitioners with a valuable tool for estimating the objective function value of the globally optimal solution for discrete optimization problems without actually visiting such a solution.

The results presented in this section demonstrate that the logistics regression model with corresponding confidence intervals provides insights into the behavior of local search algorithms for a fixed number of iterations K. These results also demonstrate that logistic regression can be used to find the best fitting model to describe the relationship between a predictor variable β and the response variable $P\{T_\beta \leq K)$ as well as estimate the objective function value of solution close to the globally optimal solution.

Table 14: Mean and Standard Deviation of Error for Each Local Search Algorithm

Local Search Algorithm	**Mean μ, Std Dev σ**			
	ρ = .0001	**ρ = .001**	**ρ = .002**	**ρ = .01**
Pure Local Search $\bar{\rho}$, s_ρ	-1.81, 1.57	-0.16, 1.54	0.41, 1.54	1.90, 1.57
Pure Local Search $\bar{\rho}^a$, s_ρ^a	1.81, 1.57	0.99, 1.10	1.26, 0.81	2.24, 0.89
Simulated Annealing $\bar{\rho}$, s_ρ	-1.91, 0.86	-0.29, 0.87	0.27, 0.90	1.74, 0.99
Simulated Annealing $\bar{\rho}^a$, s_ρ^a	1.91, 0.86	0.70, 0.52	0.71, 0.54	1.74, 0.99
Threshold Accepting $\bar{\rho}$, s_ρ	-2.15, 1.39	-0.46, 1.32	0.12, 1.31	1.65, 1.30
Threshold Accepting $\bar{\rho}^a$, s_ρ^a	2.15, 1.39	0.80, 1.11	0.93, 0.83	1.93, 0.70

Chapter 7:

Conclusions and Future Directions of Research

This dissertation introduces the β-acceptable solution probability as a means to evaluate the finite-time performance of a local search algorithm. This dissertation also presents expressions and bounds for the β-acceptable solution probability within the local search algorithm framework, and illustrates how these expressions and bounds can be estimated for particular local search algorithms. Techniques for applying this finite-time performance information to determine how effectively a local search algorithm can be expected to perform in the future are presented.

This dissertation presents an expression for the expected number of iterations that a local search algorithm must execute to encounter a solution with an objective function value less than or equal to the value of a β-acceptable solution, as well as conditions for convergence (with probability one) to a β-acceptable solution. Moreover, an expresssion for the probability of having visited a solution with a sufficiently small objective function value for a fixed number of iterations K is presented. Logistic regression is used to estimate the likelihood of reaching values of β for a fixed number of iterations K that would otherwise be too difficult to determine computationally, as well as for estimating the objective function value of the globally optimal solution.

The research results presented in this dissertation suggest several new directions of study. Another way to measure the performance of local search algorithms is to compare the rate at which the algorithms converge to a global optimum. Practitioners can use these rates of convergence when choosing a local search algorithm for a given instance of an intractable discrete optimization problem. Investigating the impact of the components of the local search algorithm (e.g., the neighborhood function, the hill climbing random variables, the cooling schedule, and the number of inner and outer loops) on the rate at which $P\{D(k,\beta)\}$ approaches one is another area for future research.

Determining how the hill climbing random variable $R_k(\omega(k),\omega)$ relates to convergence of a local search algorithm presents an additional area for future research. Research into the properties of $R_k(\omega(k),\omega)$ are needed to investigate various forms for $R_k(\omega(k),\omega)$ that maximize $P\{D(k,\beta)\}$. Another area of future research is to determine what properties of $R_k(\omega(k),\omega)$ allow an algorithm to reach a given β-acceptable solution rapidly, but still allow the algorithm to continue searching for the globally optimal solution without getting trapped in local minima. Another approach to this research problem would be to determine how many iterations of a given algorithm are required to allow $R_k(\omega(k),\omega)$ to reach a value that allows an the algorithm to reach a given β-acceptable solution.

The methods of collecting data presented in this dissertation can be time consuming and redundant. On-line adaptive estimation techniques will be explored to collect data as the algorithm executes, rather than collecting data off line and applying the results to a local search algorithm. Another potentially useful approach is to track the rate of change for the β-acceptable solution probability between iterations of the local search algorithm and use this feedback to determine the value of continuing the algorithm. Incorporating an experimental design that uses a single long run and batching method for collecting data similar to the batch means method applied in discrete event simulation may prove to be more efficient that multiple replications of a local search algorithm. These on-line adaptive techniques may also be particularly useful in estimating lower bounds for the expected number of iterations required to reach a β-acceptable solution $E[T_\beta]$.

The β-acceptable solution probability can be used to develop guidelines to design effective run strategies for local search algorithms using the GHC algorithm framework that do not converge with probability one to D_β. These guidelines can be used to determine termination conditions once the marginal value of additional iterations is deemed negligible. Research will be conducted to use the performance measures presented in this dissertation to estimate the number of iterations to reach a β-acceptable solution as well as the globally optimal solution for a given instance of an intractable discrete optimization problem. Another area for future research is to observe the behavior of the β-acceptable solution probability as values of β approach the objective function value of the globally optimal solution.

The traveling salesman problem was chosen for the experiments described in this dissertation because it is well known intractable discrete optimization problem. Moreover, several challenging instances of the TSP with known optimal solutions are available (see Reinelt 1991). Future research investigating the finite-time performance of local search algorithms applied to discrete optimization problems other than the TSP is needed to demonstrate the generality of the theory and analysis presented in this dissertation.

Selection of the set of β-acceptable solutions for a given discrete optimization problem is an interesting area for future research. Without full knowledge of the solution space, the minimum and maximum β-acceptable solution values for the discrete optimization problem must be determined through experimentation. This process of choosing β-acceptable solution values through experimentation is cumbersome with larger problems. Techniques for determining bounds on the objective function value of the globally optimal solution can be used to assist the practitioner in choosing a realistic set of β-acceptable solutions.

The number of iterations K that a local search algorithm executes, combined with the set of β-acceptable solutions, has a significant impact on the estimates for $P\{T_\beta \leq K\}$. Future research investigating the influence of the number of iterations K that a local search algorithm executes when applied to larger discrete optimization problems is needed to demonstrate the generality of the theory and analysis presented in this dissertation.

On a broader scale, the framework developed in this dissertation shows how different local search algorithms applied to intractable discrete optimization problems can be compared and evaluated using the GHC algorithm framework as a single, common performance measure. The results in this research represent a first step towards the development of a general performance theory for local search algorithms using the GHC algorithm framework, hence foster new avenues for future research on the evaluation of local search algorithm performance, independent of the specific problems being addressed.

Bibliography:

Aarts, E.H.L. and Korst, J., 1989, *Simulated Annealing and Boltzmann Machines : A Stochastic Approach to Combinatorial Optimization and Neural Computing*, Wiley: Chichester [England] ; New York.

Aarts, E.H.L. and Lenstra, J.K., 1997, *Local Search in Combinatorial Optimization*, Wiley: Chichester [England] ; New York.

Althofer, I. and Koschnick, K.U., 1991, On the Convergence of Threshold Accepting, *Applied Mathematics and Optimization*, 24, 183-195.

Anily, S. and Federgruen, A., 1987, Simulated Annealing Methods with General Acceptance Probabilities, *Journal of Applied Probability*, 24: 3, 657-667.

Balaguer, C., Giménez, A., Pastor, J.M., Padrón, V.M., and Abderrahim, M., 2000, A Climbing Autonomous Robot for Inspection Applications in 3D Complex Environments, *Robotica*, 18, 287-297.

Cantoni, O. and Cerf, R., 1997, The Exit Path of a Markov Chain with Rare Transitions, *ESAIM: Probability and Statistics*, 1: February 1997, 95-144.

Cerf, R., 1998, Asymptotic Convergence of Genetic Algorithms, *Advances in Applied Probability*, 30: 2, 521-550.

Charon, I. and Hudry, O., 1993, The Noising Method - a New Method for Combinatorial Optimization, *Operations Research Letters*, 14: 3, 133-137.

Chiang, T.S. and Chow, Y.Y., 1989, A Limit-Theorem for a Class of Inhomogeneous Markov-Processes, *Annals of Probability*, 17: 4, 1483-1502.

Croes, G.A., 1958, A Method for Solving Traveling-Salesman Problems, *Operations Research*, 6, 791-812.

Davis, T.E., 1991, *Toward an Extrapolation of the Simulated Annealing Convergence Theory onto the Simple Genetic Algorithm*, (Doctoral dissertation), University of Florida: Gainesville.

Delport, V., 1998, Parallel Simulated Annealing and Evolutionary Selection for Combinatorial Optimisation, *Electronics Letters*, 34: 8, 758-759.

Desai, M.P., 1999, Some Results Characterizing the Finite Time Behaviour of the Simulated Annealing Algorithm, *Sadhana-Academy Proceedings in Engineering Sciences*, 24, 317-337.

Draper, N.R. and Smith, H., 1998, *Applied Regression Analysis*, John Wiley & Sons, INC.: New York, NY.

Dueck, G. and Scheuer, T., 1990, Threshold Accepting - a General-Purpose Optimization Algorithm Appearing Superior to Simulated Annealing, *Journal of Computational Physics*, 90: 1, 161-175.

Edwards, C.H., Penny, D.E., 1986, *Calculus and Analytic Geometry*, Prentice-Hall: Englewood Cliffs, New Jersey.

Eglese, R.W., 1990, Simulated Annealing: A Tool for Operational Research, *European Journal of Operational Research*, 46, 271-281.

Faigle, U. and Kern, W., 1991, Note on the Convergence of Simulated Annealing Algorithms, *SIAM Journal On Control and Optimization*, 29: 1, 153-159.

Faigle, U. and Kern, W., 1992, Some Convergence Results for Probabilistic Tabu Search, *ORSA Journal on Computing*, 4, 32-37.

Faigle, U. and Schrader, R., 1988, On the Convergence of Stationary Distributions in Simulated Annealing Algorithms, *Information Processing Letters*, 27: 4, 189-194.

Fleischer, M.A., 1995, Simulated Annealing: Past, Present, and Future. In *1995 Winter Simulation Conference* (Alexopouolos, C., Kang, K., Lilegdon, W.R., Goldsman, D. ed.). pp. 155-161, IEEE Press.

Fleischer, M.A. and Jacobson, S.H., 1999, Information Theory and the Finite-Time Behavior of the Simulated Annealing Algorithm: Experimental Results, *INFORMS Journal On* Computing, 11: 1, 35-43.

Foerster, H. and Wäscher, G., 1998, Simulated Annealing for Order Spread Minimization in Sequencing Cutting Patterns, *European Journal of Operational Research*, 126, 106-130.

Garey, M.R. and Johnson, D.S., 1979, *Computers and Intractability : A Guide to the Theory of NP-Completeness*, W. H. Freeman: San Francisco.

Glover, F., 1989, Tabu Search-Part I, *ORSA Journal on Computing*, 1, 190-206.

Hajek, B., 1988, Cooling Schedules for Optimal Annealing, *Mathematics of Operations Research*, 13: 2, 311-329.

Helsgaun, K., 2000, An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic, *European Journal of Operational Research*, 126, 106-130.

Hoel, P.G., Port, S.C. and Stone, C.J., 1971, *Introduction to Probability Theory*, Houghton Mifflin Company: Hopewell, New Jersey.

Hosmer, D.W. and Lemeshow, S., 1989, *Applied Logistic Regression*, John Whiley & Sons: New York, New York.

Hu, T.C., Kahing, A.B. and Tsao, C.W.A., 1995, Old Bachelor Acceptance: A New Class of Non-Monotone Threshold Accepting Methods, *ORSA Journal on Computing*, 7, 417-425.

Jacobson, S.H., Sullivan, K.A. and Johnson, A.W., 1998, Discrete Manufacturing Process Design Optimization Using Computer Simulation and Generalized Hill Climbing Algorithms, *Engineering Optimization*, 31: 2, 247-260.

Jacobson, S.H. and Yucesan, E., 2001a, On the Effectiveness of Generalized Hill Climbing Algorithms, Technical Report, University of Illinois, Urbana, Illinois.

Jacobson, S.H. and Yucesan, E., 2001b, A Generalized Hill Climbing Algorithm Framework for Studying the Performance and Convergence of Simulated Annealing and Local Search Algorithms, Technical Report, University of Illinois at Urbana-Champaign, Urbana, Illinois.

Johnson, A.W., 1996, *Generalized Hill Climbing Algorithms for Discrete Optimization Problems*, (Ph.D. Dissertation) Industrial and Systems Engineering, Virginia Tech: Blacksburg, VA.

Johnson, A.W. and Jacobson, S.H., 2001a, A Class of Convergent Generalized Hill Climbing Algorithms, *Applied Mathematics and Computation, (To Appear).*

Johnson, A.W. and Jacobson, S.H., 2001b, On the Convergence of Generalized Hill Climbing Algorithms, *Discrete Applied Mathematics, (To Appear).*

Kobayashi, S., Edahiro, M., and Kubo, M., 1999, A VLSI Scan-Chain Optimization Algorithm for Multiple Scan-Paths, *IEICE Transactions on Fundamentals of Electronics, Communications, and Computer Science*, 11, 2499-2504.

Koulamas, C., Antony, S.R. and Jaen, R., 1994, A Survey of Simulated Annealing Applications to Operations- Research Problems, *OMEGA-International Journal of Management Science*, 22: 1, 41-56.

Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G. and Shmoys, D.B., 1985, *The Traveling Salesman Problem*, John Wiley and Sons, Chichester, United Kingdom.

Liepins, G.E. and Hilliard, M.R., 1989, Genetic Algorithms: Foundations and Applications, *Annals of Operations Research*, 21, 31-58.

Lin, C.K.Y., Haley, K.B. and Sparks, C., 1995, A Comparative Study of Both Standard and Adaptive Versions of Threshold Accepting and Simulated Annealing Algorithms in Three Scheduling Problems, *European Journal of Operational Research*, 83, 330-346.

Lin, S., 1965, Computer Solutions to the Traveling Salesman Problem, *Bell Technical Journal*, 44, 2245-2269.

Lin, S. and Kernihan, B.W., 1973, An Effective Heuristic for the Traveling Salesman Problem, *Operations Research*, 21, 498-516.

Mazza, C., 1992, Parallel Simulated Annealing, *Random Structures & Algorithms*, 3: 2, 139-148.

McCullagh, P. and Nelder, J.A., 1989, *Generalized Linear Models*, Chapman and Hall: New York, New York.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A. and Teller, E., 1953, Equation of State Calculations by Fast Computing Machines, *Journal of Chmical Physics*, 21, 1087-1092.

Mitra, D., Romeo, F. and Sangiovanni-Vincentelli, A.L., 1986, Convergence and Finite Time Behavior of Simulated Annealing, *Advances in Applied Probability*, 18, 747-771.

Montgomery, D.C. and Peck, E.A., 1982, *Introduction to Linear Regression Alalysis*, John Wiley & Sons: New York, New York.

Moscato, P. and Fontanari, J.F., 1990, Convergence and Finite-Time Behavior of Simulated Annealing, *Advances in Applied Probability*, 18, 747-771.

Muhlenbein, H., 1997, Genetic Algorithms, In *Local Search in Combinatorial Optimization* (Aarts, E. & Lenstra, J.K., eds.). pp. 137-172, John Wiley & Sons: New York, New York.

Nissen, V. and Paul, H., 1995, A Modification of Threshold Accepting and Its Application to the Quadratic Assignment Problem, *OR Spektrum*, 17: 2-3, 205-210.

Reinelt, G., 1991, TSPlib - a Traveling Salesman Problem Library, *ORSA Journal on Computing*, 3: 4, 376-385.

Rice, J.A., 1988, *Mathematical Statistics and Data Analysis*, Wadsworth & Brooks/Cole Advanced Books & Software: Pacific Grove, CA.

Romeo, F. and Sangiovanni-Vincentelli, A., 1991, A Theoretical Framework for Simulated Annealing, *Algorithmica*, 6: 3, 302-345.

Ross, S.M., 1988, *A First Course in Probability*, Macmillan; Collier Macmillan: New York, London.

Rudolph, G., 1994, Convergence Analysis of Cononical Genetic Algorithms, *IEEE Transactions on Neural Networks, Special issue on Evolutional Computing*, 5, 96-101.

Scheermesser, T. and Bryngdahl, O., 1995, Threshold Accepting for Constrained Half-Toning, *Optics Communications*, 115: 1-2, 13-18.

Schuur, P.C., 1997, Classification of Acceptance Criteria for the Simulated Annealing Algorithm, *Mathematics of Operations Research*, 22: 2, 266-275.

Srichander, R., 1995, Efficient Schedules for Simulated Annealing, *Engineering Optimization*, 24: 3, 161-176.

Storer, R.H., Wu, S.D. and Vaccari, R., 1992, New Search Spaces for Sequencing Problmes with Application to Job Shop Scheduling, *Management Science*, 38: 10, 1495-1509.

Sullivan, K.A. and Jacobson, S.H., 2000, Ordinal Hill Climbing Algorithms for Discrete Manufacturing Process Design Optimization Problems, *Discrete Event Dynamical Systems*, 10: 4, 307-324.

Sullivan, K.A. and Jacobson, S.H., 2001, A Convergence Analysis of Generalized Hill Climbing Algorithms, *IEEE Transactions on Automatic Control, (To Appear)*.

van Laarhoven, P.J.M. and Aarts, E.H.L., 1987, *Simulated Annealing: Theory and Applications*, D. Reidel: Boston, MA, U.S.A. Sold and distributed in the U.S.A. and Canada by Kluwer Academic Publishers: Norwell MA.

Yao, X., 1995, A New Simulated Annealing Algorithm, *International Journal of Computer Mathematics*, 56: 3-4, 161-168.

Yao, X. and Li, G., 1991, General Simulated Annealing, *Journal of Computer Science and Technology*, 6, 329-338.

Yucesan, E. and Jacobson, S.H., 1996, Computational Issues for Accessibility in Discrete Event Simulation, *ACM Transactions on Modeling and Computer Simulation*, 6: 1, 53-75.

Vita:

Darrall Henderson
Lieutenant Colonel, United States Army
Department of Mathematical Sciences
United States Military Academy
West Point, New York, 10996
darrall@stanfordalumni.org

CIVILIAN EDUCATION

2001 Doctor of Philosophy Industrial and Systems Engineering), Virginia Tech,
2000 Master of Science (Operations Research), Stanford University,
1981 Bachelor of Building Construction, University of Florida.

MILITARY EDUCATION

1996 Joint and Combined Staff Officer School, Armed Forces Staff College, National Defense University,
1994 Command and General Staff Officer's Course, U.S. Army Command and General Staff College,
1986 Combined Armed Services Staff School, U.S. Army Command and General Staff College,
1985 Infantry Officer Advanced Course,
1981 Engineer Officer Basic Course.

EMPLOYMENT

Lieutenant Colonel, United States Army Corps of Engineers (1981 - Present)