

# **A Comparison of Heuristic, Meta-Heuristic and Optimal Approaches to the Selection of Conservation Area Networks**

Christopher Pappas<sup>1</sup>, Radhakrishnan K. Srinivasan<sup>2</sup>, J. Wesley Barnes<sup>2</sup>, and Sahotra Sarkar<sup>1,\*</sup>

<sup>1</sup>Biodiversity and Biocultural Conservation Laboratory, Section of Integrative Biology, University of Texas at Austin, 1 University Station, C3500, Austin, TX 78712 –1180;

and

<sup>2</sup>The Graduate Program in Operations Research and Industrial Engineering, University of Texas at Austin, Austin, TX 78712 –1063.

## **Abstract**

This paper reports the superior performance of a meta-heuristic tabu search approach to the selection of conservation area networks for the economical (efficient) representation of biodiversity, both with and without budgets. Using six empirical and twenty artificially constructed data sets, the tabu search method was compared to four other techniques: (i) a traditional one-pass complementarity heuristic; (ii) simulated annealing; and a classical integer programming procedure used (iii) heuristically, as well as (iv) to find optimal solutions. It appears that an advanced carefully tailored tabu search implementation holds great promise for generating near optimal solutions to problems with large data sets with small amounts of computational effort. In particular, tabu search can be used to incorporate multiple criteria into conservation planning including issues of spatial coherence and socioeconomic factors. Tabu search algorithms can also incorporate rules with clear biological interpretations such as complementarity (which is used in the implementation reported here).

*Keywords:* reserve selection; area selection; heuristics; meta-heuristics; complementarity; simulated annealing; tabu search.

*Running head:* Tabu Search for Conservation Area Network Selection.

\* Corresponding author; comments invited. Please send comments to [sarkar@mail.utexas.edu](mailto:sarkar@mail.utexas.edu), or fax: (512) 471-4806.

## 1. Introduction

The effective design of conservation area networks (CANs), which is a central part of systematic conservation planning (Margules and Pressey 2000; Sarkar 2005), typically requires the solution of one or both of the following two problems (ReVelle *et al.* 2002; Sarkar 2005; Sarkar *et al.* 2004): given a set of cells (land areas), the probabilistic expectations of the presence of biodiversity surrogates (such as species or environmental parameters) in each cell, and a target of representation for each surrogate in the set,

- (i) find the smallest set of cells such that each surrogate meets its target;
- (ii) given a maximum number of cells that can be included in a network (a budget), find the set of cells that maximizes the number of surrogates that meet their targets.

With the simplification that the expectation of surrogate presence is 1 or 0 (binary “presence-absence” data), the first problem was originally formulated in the context of biodiversity conservation by Margules *et al.* (1988). Its formal equivalent in operations research, the Location Set Covering Problem (LSCP), was introduced by ReVelle *et al.* (1971). The probabilistic version in conservation biology is known as the Expected Surrogate Set Covering Problem (ESSCP) (Sarkar *et al.* 2004). In the context of biodiversity conservation, the second problem was first analyzed by Cocks and Baird (1989) and studied in detail by Polasky *et al.* (2000). Its formal equivalent in operations research is called the Maximal Covering Location Problem (MCLP) (Church and ReVelle 1974). The probabilistic version in conservation biology is known as the Maximal Expected Surrogate Covering Problem (MESCP) (Sarkar *et al.* 2004). Both

problems are typically encountered in the practical design of CANs for the adequate representation of the biodiversity features of a region though the MESSCP is of greater practical relevance because of the ubiquitous presence of budgetary constraints.

Standard formulations for both the ESSCP and the MESSCP result in NP-hard integer linear optimization problems with no known polynomial-time algorithms to solve them (Papadimitriou 1994). Therefore, optimal algorithms, such as branch-and-bound (Nemhauser and Wolsey 1988), which find optimal solutions are often practically useful only for small problems (Garey and Johnson 1979). This intractability led biologists, starting in the 1980s, to devise several one-pass heuristic approaches which used biological principles for cell incorporation into a network. The most important outcome of that work was the *principle of complementarity*, the iterative selection of cells to maximize the representation of only those surrogates that have not yet met their targets (Margules *et al.* 1988; Vane-Wright *et al.* 1991; Justus and Sarkar 2002). (In Operations Research this is known as a greedy algorithm.)

Optimal methods should be used if they admit of solutions in reasonable amounts of time demanded by the situation. But when their time requirements become excessive or strictly prohibitive heuristic methodologies must be employed. For the ESSCP, continuing efforts have demonstrated that such heuristics usually require much less computation time than optimal algorithms and are consistently capable of achieving near optimal solutions (Csuti *et al.* 1997; Pressey *et al.* 1997; Sarkar *et al.* 2004). Recent results by Sarkar *et al.* (2004) refute a counter-claim by Rodrigues *et al.* [2000] and Rodrigues and Gaston [2002] about the ESSCP. Sarkar *et al.* achieves faster solutions than Rodrigues and Gaston with optimal methods, but find that heuristic methods are

even faster without sacrificing the quality of solutions. Sarkar *et al.* also show that even some small problems take an inordinate amount of time for resolution by optimal algorithms. But problem size is not the only factor that determines a problem's computational tractability. Besides the specific formulation of the problem, the structural properties of the data involved can also have a significant impact on the problem's complexity (Pressey *et al.* 1999; Fischer and Church 2003). These features coupled with the inherent uncertainties and inaccuracies in the biogeographical input data reveal the limitations of purely optimal approaches.

One purpose of this paper is to extend the results of Sarkar *et al.* (2004) in two ways: (i) by analyzing the computational properties of the MESCP; and (ii), for both problems (ESSCP and MESCP), performing tests on a wider class of data sets. However, the main purpose is to demonstrate that "meta-heuristic" algorithms are available that routinely achieve comparable or superior solutions, in terms of time efficiency and solution quality, to those generated using traditional methods. In the past, simulated annealing (SA) (Kirkpatrick *et al.* 1983) has been the only meta-heuristic algorithm that has been systematically used for the selection of conservation area networks (Possingham *et al.* 2000). However, this meta-heuristic achieves, at best, relatively small improvements over simple one-pass heuristic algorithms such as complementarity while taking much longer computational times (Kelley *et al.* 2002). In contrast, even a relatively unsophisticated implementation of a tabu search (TS) metaheuristic uniformly achieves significantly better results than SA (Hao and Pannier 1998). These initial results indicate that TS is a promising method available to attack the design of CANs for the large and complex data sets currently being generated from the field. What often makes

faster and reliable computation critical is that, in many practical planning contexts, hundreds of alternative solutions must be computed (Moffett and Sarkar 2005; Sarkar 2005). This is particularly true when, beyond biodiversity surrogate representation, multiple criteria such as vulnerability, spatial coherence of CAN design, and socio-economic constraints must be simultaneously incorporated into network design.

Throughout this paper the various methods will be judged with respect to their *optimality* (the extent to which they produce the best possible minimum set of cells for ESSCP or the maximum list of surrogates with satisfied targets for the MESCP) and their computational *efficiency* (the time they take to produce a solution). Section 4 will also discuss their *flexibility* in solving the variety of problems encountered in systematic conservation planning.

## 2. Methods

### 2.1 Formal Problem Specifications

Both the ESSCP and the MESCP share the following formal features: a list of cells  $\Sigma(\sigma_j \in \Sigma, j = 1, 2, \dots, n)$ , a list of surrogates  $\Lambda(\lambda_i \in \Lambda, i = 1, 2, \dots, m)$ , a target  $\tau_i$  ( $i = 1, 2, \dots, m$ ) for each biodiversity surrogate, and an  $m \times n$  matrix where each  $p_{ij}$  entry is the probabilistic expectation of surrogate  $\lambda_i$  in cell  $\sigma_j$ . (Since the  $p_{ij}$ 's are expected values, they may be summed over cells or over a list of surrogates without assuming independence. The contrast, here, is with probabilities—see Sarkar *et al.* 2004 for details.)

Solutions to both problems have the form of a list of selected cells,  $I$ , and a list of surrogates that have met their targets,  $K$ . Let  $I'$  and  $K'$  be the complements of these sets,

respectively. Formally, a *solution* is an ordered pair  $(\mathbf{x}, \mathbf{y})$ , where  $\mathbf{x}$  is a binary  $|\mathcal{A}|$ -vector with  $x_i = 1$  if  $\lambda_i \in K$  and  $x_i = 0$ , otherwise; and  $\mathbf{y}$  is a binary  $|\Sigma|$ -vector with  $y_j = 1$  if  $\sigma_j \in \Gamma$  and  $y_j = 0$ , otherwise.

The ESSCP requires finding the smallest set of cells (set of minimum cardinality) such that every surrogate meets its assigned target. The formal specification of the ESSCP is:

$$\text{Minimize } |\Gamma| = \sum_{j=1}^n y_j \quad \text{s. t.} \quad \forall i = 1, \dots, m \left( \sum_{j=1}^n y_j p_{ij} \geq \tau_i \right). \quad (1)$$

The biological interpretation of the ESSCP is that all surrogates are being represented as economically as possible, that is, in as few cells as possible. The MESCP requires finding the maximal cardinality set of satisfied surrogate targets such that the set of cells used to satisfy those targets is less than some budget constraint  $\beta (\leq n)$ . The formal specification of the MESCP is:

$$\text{Maximize } |K| = \sum_{i=1}^m x_i \quad \text{s. t.} \quad \sum_{j=1}^n y_j \leq \beta \quad \text{and} \quad \forall i = 1, \dots, m \left( \sum_{j=1}^n y_j p_{ij} \geq \tau_i \cdot x_i \right). \quad (2)$$

The biological interpretation of the MESCP is that as many surrogates as possible are being adequately represented while being constrained by a budget that limits the number of cells that can be included in a network. For both the ESSCP or MESCP problems, any  $\Gamma$  uniquely determines a corresponding  $K$ . (The converse is not necessarily true.) Thus, stating  $\Gamma$  is sufficient to stipulate the solution to either problem.

## 2.2 Optimal Method (CPLEX 8.1)

An optimal method attempts both to find an optimal solution to a problem and prove that the solution is optimal. Historically, optimal methods for both the ESSCP and the MESCP have used a variation of the classical branch-and-bound method for solving integer-linear programming (IP) problems. In this analysis, the ILOG CPLEX 8.1 software package was used. (ILOG CPLEX is an industry standard, commercial software package that has traditionally been used for these problems [Rodrigues *et al.* 2000; Rodrigues and Gaston 2002; Sarkar *et al.* 2004].)

## **2.3 Heuristics**

### **2.3.1 Complementarity**

The quickest heuristics for these problems select cells for inclusion using rarity and complementarity. For binary presence-absence data, using both rarity and complementarity yields the best results (Csuti *et al.* 1997; Pressey *et al.* 1997). With probabilistic data, using complementarity alone is best (Sarkar *et al.* 2004). For this reason, two greedy heuristic algorithms (ESSCP-C and MESCP-C) based on complementarity, encoded into probabilistic versions of the ResNet software package (Garson *et al.* 2002), were used in this analysis.

Both the ESSCP-C and MESCP-C proceed iteratively. First, each calculates the complementarity value for every unselected cell; second, each chooses the cell with the highest complementarity value, breaking remaining ties on the basis of lexical order. The two procedures differ only with respect to their termination conditions and the handling of “redundant cells” (*i.e.*, cells the removal of which does not cause the representation of any surrogate to fall below its target). The selection routine in the ESSCP-C procedure terminates when its list of selected cells ensures that all surrogates meet their targets (*i.e.*,

when the solution is *feasible*). It then removes redundant cells from the solution. The selection routine in the MESCP-C procedure terminates when  $|\Gamma| = \beta(\leq n)$ . No redundancy removal is used in MESCP-C.

### **2.3.2 CPLEX Heuristics**

Because integer-linear programming is NP-hard, the task of use of a stand-alone branch-and-bound algorithm to solve even moderate size IP problems is often impractical. Thus, most software packages based on branch-and-bound algorithms also provide a variety of heuristic tools to generate feasible starting solutions quickly (Williams 1993). These starting solutions can then be improved upon by standard branch-and-bound algorithms, though the efficiency with which this is done is usually highly sensitive to the particular starting solution.

We used this option in CPLEX 8.1 with the Solver 5.3 libraries. CPLEX 8.1 marks a significant improvement over its predecessor (CPLEX 7.5) in finding feasible solutions to IP problems. This increase in performance is the result of CPLEX 8.1's use of undisclosed search heuristics at the beginning of a search to generate better feasible solutions in a shorter amount of time than would otherwise be required.

## **2.4 Meta-heuristics**

A meta-heuristic algorithm: (i) controls the execution of a simpler heuristic method; and (ii) unlike a one-pass heuristic, does not automatically terminate once a locally optimal solution is found. Thus, unlike the one-pass complementarity heuristic described in Section 2.3.1, explicit termination conditions must be set for meta-heuristic procedures that go beyond feasibility or local optimality. We compared two meta-heuristic methods, simulated annealing (SA) and tabu search (TS). Both are "iterative-

improvement methods”: they begin with a feasible solution and attempt to improve upon it throughout the course of a search.

### **2.4.1 Simulated Annealing**

SA is a general purpose meta-heuristic optimization procedure (Kirkpatrick *et al.* 1983). The performance of simulated annealing (SA) for the ESSCP and MESCP was assessed using the MARXAN software package (Ball and Possingham 2000). For a description of the implementation of simulated annealing in MARXAN, see Ball (2000).

In this analysis, all SA runs for both the ESSCP and MESCP (hereafter referred to as ESSCP-M and MESCP-M, respectively) were conducted using MARXAN’s “adaptive annealing schedule”, which requires the user to specify only the number of iterations,  $k$  (Ball and Possingham 2000). To determine the annealing schedule, MARXAN samples the given problem, setting values for the other internal parameters from the data. Initial solutions were generated using MARXAN’s “greedy” complementarity-based heuristic. MARXAN’s “normal iterative improvement” (Ball and Possingham 2000) was used.

Because MARXAN does not accept probabilistic data, ESSCP-M and MESCP-M could only be used for our binary data sets. The ESSCP runs were conducted with  $k = 250 \times 10^6$  and the MESCP runs had  $k = 125 \times 10^6$  with three exceptions: (i) the ESSCP runs for Namibia had  $k = 100 \times 10^6$ ; (ii) the ESSCP runs for West Virginia had  $k = 20\,000$ ; and the MESCP runs for West Virginia had  $k = 5\,000$ . Lower values of  $k$  were used because computation times became prohibitive (greater than 12 hours) without discernible solution improvement.

Due to SA’s stochastic nature, individual runs with MARXAN on the same data set often result in different solutions. To correct for statistical error, ESSCP-M and

MESCP-M were run on each individual problem 2 -10 times. The best solution for each problem is presented in Section 3. While more runs may improve solution quality (or economy), it will also increase the computational effort involved, requiring either more time on a single machine, or more machines with each running MARXAN on an instance of the problem in parallel. More MARXAN runs were not performed, because it would have required an even more drastic increase in the amount of computational effort allotted when compared with the other methods (see Table 2). The quality of the MARXAN results was typically not an issue compared to the other methods.

Unlike the other procedures used in this analysis, MARXAN does not record the exact time during a search when the best solution was found. Therefore, it is possible that the time to find the best known solution was less than the time required for the procedure to terminate. Runs conducted with MESCP-M were made despite an undocumented “Overflow Error” message outputted to the execution window during the runs; there was no superficial evidence of any adverse effect of this possible error on the final solutions.

#### **2.4.2 Tabu Search**

Since tabu search (TS) has apparently not been previously used for the CAN selection problem in the published literature (see, however, Okin’s [1997] thesis for a simple implementation), we will describe it in some detail. TS is a direct search method that intelligently *escapes* from local optima to find near-optimal solutions quickly (Glover and Laguna 1993; Glover and Laguna 1997). One of its main innovations is to use memory to prevent recycling to a recently-visited solution. Given an *incumbent* solution, a *move* is a transition to another solution in the solution space defined by the *search neighborhood*. The search neighborhood, relative to an incumbent solution and

the state of the search, is the set of solutions reachable from that solution in a single move. A move is made at each iteration in a TS, and the best-known solution is stored in memory. This allows global direction of the improvement strategy. A solution *attribute* captures some salient characteristic of a solution that can be used to prohibit the search from returning to solutions with that attribute for a specified number of moves, the *tabu tenure*. This restriction is implemented by storing the attribute in a tabu memory structure, the *tabu list*, for its tabu tenure number of iterations. Permissible moves are those that, if applied, do not result in a solution with an attribute that is on the current tabu list.

The two reactive TS (RTS) procedures (Battiti and Tecchiolli 1994) used in this analysis (ESSCP-RTS and MESCP-RTS) are characterized by three features:

- (i) an *iterative improvement strategy*: this defines the neighborhoods encountered throughout a search and the criteria for selecting a (*non-escape*) move;
- (ii) a *tabu strategy*: this specifies what attributes make a solution “tabu” and gives them tabu tenure values; and
- (iii) an *explicit escape strategy*: this determines when and how a search should escape from an unpromising or exhaustively searched region of the solution space.

Moves are defined by a transition from an incumbent solution,  $\Gamma_0$ , defined by its associated vector,  $y_0$ , to another solution in the solution space. Moves are of three types:

- (a) *delete*: neighbors (solutions) generated by changing a  $y_{0j} = 1$  to  $y_{0j} = 0$ , *i.e.*, unselecting a cell;
- (b) *insert*: neighbors generated by changing a  $y_{0j} = 0$  to  $y_{0j} = 1$ , *i.e.*, selecting an unselected cell;

- (c) *swap*: neighbors generated from dual “bit toggles” of  $y_0$ , *i.e.*, for some  $\sigma_i \in \Gamma_0$ ,  $y_{0i}$  is set to 0 (a delete move) and for some  $\sigma_j \notin \Gamma_0$ ,  $y_{0j}$  is set to 1 (an insert move).

Although the neighborhood implied by all *possible* moves from  $\Gamma_0$  can be large, the set of permissible, non-tabu, moves tends to be much smaller. The effectiveness of a TS procedure depends on intelligently enforcing its tabu move restrictions. Moreover, TS must efficiently determine a unique ‘best’ move from a given set of permissible moves.

Although the ESSCP-RTS and MESCP-RTS involve quite different strategies in their iterative stages, they are similar insofar as they both maintain a single tabu list,  $z$ , where  $z$  is a binary  $|\Sigma|$ -vector with  $z_j = 1$  if  $\sigma_j$  is “tabu” and  $z_j = 0$ , otherwise. In both procedures, at every iteration, if a delete move or an insert move is made, the cell that is deleted or inserted is made tabu, but if a swap move is made, *only* the deleted cell is made tabu. Moreover, both procedures use its complementarity score to evaluate cells for inclusion: the complementarity heuristic is used because of its straightforward biological interpretation. Thus, our TS procedure can be regarded as a generalization of a complementarity heuristic.

The *assignment* of tabu tenure values—the second feature of a tabu strategy—is identical for ESSCP-RTS and MESCP-RTS and part of the *reactive tabu search* method. Simpler static TS methods freeze the tabu tenure at a constant value. For more complex problems such as the ESSCP and MESCP, the more robust RTS approach is required. The RTS strategy (Battiti and Tecchiolli, 1994) is used in both the ESSCP-RTS and MESCP-RTS to adjust the tabu tenure values dynamically and to determine when each procedure should perform an explicit escape strategy. A detailed narrative description of the RTS strategy is provided in Appendix 1. The two ESSCP-RTS procedures that were

used differ according to: (i) how they search for solutions of minimal cardinality; and (ii) what values they use for the lower bound of the search. ESSCP-RTS-LP uses a lower bound based on the linear programming (LP) relaxation of the problem. (The LP lower bound is easily found using CPLEX.) ESSCP-RTS-H uses a lower bound calculated using an embedded heuristic algorithm. Details and pseudo-code of both the ESSCP-RTS and MESCP-RTS procedures are provided in Appendices 2 and 3, respectively.

## 2.5 Data Sets and Problems

Six empirical data sets and twenty artificial data sets were used to compare the procedures described above. The empirical data sets consist of the type of field data that can be and, in some cases, have been used for practical conservation planning (see, *e.g.*, Sarakinos *et al.* 2001). Those six data sets are characterized by the number of entries in the  $n \times m$  representation matrix where  $n$  is the number of cells and  $m$  is the number of surrogates. Table 1 details these data sets. Appendix 4 describes the construction of the artificial data sets. These artificial datasets, as characterized by their density, fall within the range of those that have been encountered with empirical data sets. However, caution should be used on interpreting results from such data sets because: (i) it is known that data set properties have strong effects on algorithm performance (Pressey *et al.* 1999; Fischer and Church 2003); but (ii) the specific influence of different such properties are yet to be characterized (Sarkar *et al.* 2004).

The tested problems were created by setting representation targets for both the ESSCP and the MESCP, and by assigning a budget for the MESCP. The choice of an appropriate target for conservation depends on biological assumptions which are sometimes controversial (Soulé and Sanjayan 1998); we use a variety of targets for the

ESSCP and the MESCP. For the ESSCP, for the empirical data sets, two different targets were used: a single (1) representation and a representation of 10 % of the total for each surrogate in the data sets; for the artificial data sets, the target was always 10 %. For the MESCP targets were uniformly set to 10 %. Two budgets were used: 7.5 % of the number of cells (representing 7.5 % of the total area) and  $\mu - 1$ , which is one cell less than the LP lower bound on the number guaranteed to allow all surrogates to achieve their targets. (The computations required to obtain  $\mu$  are discussed in Appendix 2.)

Thus, tests were run on six classes of problems:

- (i) ESSCP for integer-valued data sets.
- (ii) ESSCP for probabilistic data sets.
- (iii) MESCP with a budget of  $\mu - 1$  for integer-valued data sets.
- (iv) MESCP with a budget of 7.5 % of the total number of cells for integer-valued data sets.
- (v) MESCP with a budget of  $\mu - 1$  for probabilistic data sets.
- (vi) MESCP with a budget of 7.5% of the total number of cells for probabilistic data sets.

## 2.6 Computations

The CPLEX 8.1 software package with Solver 5.3 libraries was run on a Dell PowerEdge 2650 operating under Windows™ Terminal Server 4.0. The machine specifications were: four 2.4 GHz Xeon processors (each with a 533 MHz front side bus and 512 K cache), 3 GB of primary memory, 4.5 GB of virtual memory, and 13 GB of free hard disk space to store the CPLEX “branch and cut” files. All CPLEX runs were allowed a maximum time of 7 200 seconds (2 hours).

All other tests were run on a Dell Precision Workstation 650 operating under Windows™ XP Professional. The machine specifications were: four 2.4 GHz Xeon processors (each with a 533 MHz front side bus and 512 K cache), 3.5 GB of primary memory and 4 GB of virtual memory.

### **3. Results**

Table 2 presents results for optimality and computational efficiency for the ESSCP for the data sets with the binary presence-absence (binary) data sets (all the empirical data sets except Ecuador); Table 3 does the same for the data sets with probabilistic expectation data (Ecuador and the artificial data sets). Table 2, but not Table 3, includes results from MARXAN because it cannot use probabilistic data. The reported times, except those from the MARXAN runs, are for when the best solutions were found. MARXAN does not track this statistic. For the TS results, the reported times are those taken to find the best solution after being initialized with a solution from ESSCP-C (the complementarity-based heuristic algorithm); the maximum allowed time was 3 600 seconds (1 hour), half of what was allowed to CPLEX. In Table 2 there is only one data set in which a TS procedure cannot find an optimal solution: For Québec (10 %) the optimum is missed by 0.03 %; for Québec (1) the optimum is missed by 1.84 %. In Table 3 there are no cases where the optimal CPLEX algorithm (CPLEX-O) returns a solution superior to TS. TS finds two optimal solutions; CPLEX-O never does. The heuristic CPLEX-H procedure produces marginally better solutions than TS for 6 problems. The maximal difference in performance for these six problems was 0.5 %.

Tables 4 and 5 report results for the MESCP using the presence-absence (binary) data sets with budgets of  $\mu-1$  and 7.5 %, respectively. For the budget of  $\mu-1$ , CPLEX-O outperformed the TS procedure (MESCP-RTS) for 6 out of the 10 problems but only with an average improvement of 1.49 %. For the budget of 7.5 %, CPLEX-O and CPLEX-H perform better than TS for only two data sets (Québec—10 %, by 18 %; and West Virginia—10 %, by 6.2 %); the latter also outperforms TS for another data set (Latin America—10 %, by 1.2 %). MARXAN usually performed much worse than its competitors, finding only three optimal solutions at much higher computational cost.

Tables 6 and 7 present the MESCP results for the probabilistic problems with budgets of 7.5% and  $\mu-1$ . On the problems of Table 6, CPLEX-O failed to satisfy the target for even a single surrogate on all 22 problems. CPLEX-H failed to satisfy the target for even a single surrogate on 14 problems. MESCP-RTS significantly outperformed CPLEX-H on all 22 problems with a budget of  $\mu-1$ . On the apparently easier problems of Table 7, CPLEX-O failed to satisfy the target for even a single surrogate on 19 of the 22 problems. On the remaining three problems, CPLEX-O performed well achieving 2 optima. CPLEX-H failed on one problem and outperformed MESCP-RTS on a total of 4 problems. MESCP-RTS significantly outperformed CPLEX-H on 14 of the remaining problems.

#### **4. Discussion**

The results presented above demonstrate, in most cases, that even a prototype RTS methodology is capable of consistently generating markedly superior solutions in

less time than competing approaches. A compelling point is that the RTS advantage is greatest for larger problems and more difficult problems.

The ESSCP Tables (Tables 2 and 3) show that CPLEX-O produced comparable results to the RTS procedures for the small data sets (the binary empirical data sets and the first ten artificial data sets). For the remaining larger data sets, CPLEX-O was not competitive with RTS. CPLEX-H produced comparable results to the RTS methods over all 32 problems. CPLEX-H achieved the best solution on all 10 smaller problems in Table 2 and had the best solutions on 6 of the problems from Table 3. Table 2 shows that MARXAN requires inordinately large times to generate results which are inferior on half of the problems addressed.

As for the MESCP problems, the small binary ones presented in Tables 4 and 5, CPLEX-O and CPLEX-H, obtain superior or equivalent results to the RTS methods in all but one case. It is in the larger MESCP problems presented in Tables 6 and 7 that the RTS method shows its power and potential. MESCP-RTS yields superior solutions, compared to those generated by the CPLEX techniques, for 16 of the problems in Table 6 but is inferior in 4 problems. In Table 7, which comprises the most difficult set of problems, the MESCP-RTS results dominate all of the other methods. The CPLEX-based procedures fail to satisfy even a single surrogate target in 14 of the test problems. MESCP-RTS's superior performance is of greater relevance because the final practical decisions in any conservation area network problem are likely to be made in the presence of a harsh budget constraint.

A well-known advantage of SA over complementarity is that it allows the incorporation of criteria other than surrogate representation into the decision-making

process. In particular, MARXAN allows the use of shape and size preferences in selecting cells. Other SA implementations could potentially use other criteria and RTS shares this advantage of flexibility (that is, the ability to incorporate multiple criteria) with SA. Yet, the results show that TS is computationally much more efficient than SA. Finally, multiple criteria can be easily incorporated into RTS both hierarchically, (with each criterion having a definite biological interpretation), or as a part of a single objective function. When multiple criteria are incorporated hierarchically in RTS, the role played by different criteria in selecting individual cells is transparent, as is also true with one-pass complementarity algorithms (Sarkar *et al.* 2004). With SA, multiple criteria can only be incorporated through the objective function being minimized.

The RTS method used here is a relatively unsophisticated prototype. Forthcoming improvements in the implementation incorporating such components as dynamic neighborhood methodology (Harwig *et al.* 2004; Barnes, *et al.* 2004) and superior candidate list strategies (Carlton and Barnes 1996), intermediate and long term memory structures (Laguna and Barnes 1993), and an abstract algebraic perspective (Colletti and Barnes 2004) will significantly enhance the power of the RTS approach.

In the near future, we plan to incorporate more criteria such as spatial considerations (shape, size, and dispersion of selected units), measures of threat and vulnerability, as well as economic and social costs. In a typical planning environment, we envision that simple heuristics such as complementarity will continue to be used to generate initial solutions very fast. These solutions will then be refined using RTS procedures which can be executed for whatever time is available in the given context. (For instance, a one-pass complementarity heuristic could be used to generate fast

solutions during a planning meeting, with TS used to refine these solutions during breaks to achieve near-optimality.) The results reported here indicate that we can expect substantial improvements in economy from the RTS refinement step.

### **Software Availability**

The software used to generate the results presented in this paper is available, by request, from the Biodiversity and Biocultural Conservation Laboratory at the University of Texas at Austin.

### **Acknowledgments**

We thank David Morton for helpful discussions.

## Appendix 1

### Reactive Tabu Search

Reactive tabu search (RTS) dynamically modifies the tabu tenure based on the history and current status of the search. RTS procedures are required primarily to overcome more complex “trapping mechanisms” that limit the search to a subset of the solution space. Battiti and Tecchiolli (1994) describe the existence of *complex attractor basins* which defy detection when only static tabu tenures are used. RTS methods are invoked to detect both the presence of such basins and explicitly to escape from them. RTS achieves this by means of an additional memory structure, a *reactive tabu solution list*, where it stores the total number of times that a distinct solution has been encountered, along with the last iteration in the search that that solution was encountered. As documented by Carlton and Barnes (1996) and Battiti and Tecchiolli (1994), RTS yields a more robust and powerful search.

Because of the sizes of the solution spaces for the two problems,  $\binom{|\Sigma|}{\beta}$  for MESCP and  $\sum_{i=\mu}^{|\Sigma|} \binom{|\Sigma|}{i}$  for ESSCP), RTS requires an efficient process to determine whether a particular solution has been previously encountered during the search. The solution representation in RTS is an  $|\Sigma|$ -length binary string with 1 indicating a selected cell. The HAVAL technique was used to generate a unique hash value for any solution compressed into a 32-character hexadecimal string, the solution “fingerprint” (Zheng *et al.* 1993). After a move is implemented, the fingerprint is calculated and stored in a dynamic hash table (Carlton and Barnes 1996) for subsequent efficient access. The new

solution is checked for global improvement and the appropriate data structures are updated.

After the move is completed, RTS determines if the current solution has been previously visited. If so, the moving average of the solution repeat cycle length is updated. There are two ways to determine the presence of an attractor basin and the associated need to escape the basin: (1) if enough *different* solutions have been repeated more than a specified number of times, the presence of an attractor basin is indicated and an escape procedure is invoked (implemented in both ESSCP-RTS and MESCP-RTS); (2) if a *single* solution is encountered too often an escape procedure is also invoked (implemented only in MESCP-RTS).

When escape is not necessary, the tabu tenure is dynamically increased and decreased according to the historical status of the search. In essence, if a specified number of iterations have passed without a solution being repeated, the tabu tenure is decreased. This allows an intensified search in a region that holds previously unvisited solutions. If too many solutions have recently been revisited, the tabu tenure is increased. This encourages the search to move to other regions of the solution space. The tabu tenure is increased or decreased by a multiplicative factor which further augments its ability to escape unpromising regions of the search space.

```
Reactive_tabu_search() //Returns TRUE - if we should escape; FALSE
otherwise
{ search for solution in the hash_table
  if(solution found)
  {
    compute cycle_length
    update iteration_number for solution
    increment number_of_repetitions for solution

    if(number_of_repetitions > REP + NUMBER_ABOVE_REP)
```

```

    { return TRUE;} // solution repeated too often (not implemented in
    ESSCP-RTS procedures)

    if(number_of_repetitions == REP + 1)
    {
        chaotic++;
        if(chaotic > CHAOS) {reset chaotic to FALSE; return TRUE;}
    }

    if(cycle_length < CYCLE_MAX)
    { update moving_average; increase tabu tenure;
    steps_since_last_tenure_change = 0;}
    }
    else // solution not found.
    {
        add solution to the hash_table
    }
    if(steps_since_last_tenure_change > moving_average)
    { decrease tabu tenure // (tabu tenure lowerbound is 1)
    set steps_since_last_tenure_change = 0;
    }
}

```

## Appendix 2

### ESSCP-RTS

Given an ESSCP problem,  $\rho_{ESSCP}$ , let  $\mu'$  be the value of the *LP relaxation* of  $\rho_{ESSCP}$  and let  $\mu$  be the smallest integer larger than  $\mu'$ , i.e.,  $\mu = \lceil \mu' \rceil$ . Efficient polynomial-time algorithms are available for calculating  $\mu$ ;  $\mu'$  is the first calculation CPLEX makes in its “branch and cut” procedure. ESSCP-RTS-LP uses  $\mu$  as an effective lower bound on any feasible solution for  $\rho_{ESSCP}$  ( $|\Gamma_{\rho_{ESSCP}}| \geq \mu_{\rho_{ESSCP}}$ ).  $\xi$ , the lower bound for ESSCP-RTS-H, is computed by calculating the minimum number of cells to satisfy each surrogate and setting  $\xi$  to the maximum of these values. This method was implemented recursively for all surrogates not present in the cells already selected.

To allow the ESSCP-RTS procedures to proceed in an effective and efficient manner, it is necessary to differentiate between two solutions that have the same cardinality. A natural way to do this is to measure either:

(1) the total amount that the  $\tau_i$  are under-satisfied, *i. e.*,

$$D = \sum \left( \tau_i - \sum_{j=1}^n y_j p_{ij} \right) \text{ for all } i \text{ where } \tau_i - \sum_{j=1}^n y_j p_{ij} \geq 0;$$

or

(2) the total amount that the  $\tau_i$  are over-satisfied, *i. e.*,

$$S = \sum \left( \tau_i - \sum_{j=1}^n y_j p_{ij} \right) \text{ for all } i \text{ where } \tau_i - \sum_{j=1}^n y_j p_{ij} \leq 0.$$

These two measures have a natural biological interpretation which is an additional attractive feature of using them during a search.

In comparing two solutions, the move selection criteria for both ESSCP-RTS procedures conformed to the following rules, with ties being broken arbitrarily:

- (a) any feasible solution is superior to an infeasible solution;
- (b) given two solutions which are both either feasible or infeasible, the solution with a smaller  $|\Gamma_0|$  is superior;
- (c) if two infeasible solutions have the same  $|\Gamma_0|$ , the solution with the smaller  $D$  or “total deficit” is superior;
- (d) if two feasible solutions have identical  $|\Gamma_0|$ , the solution with the larger  $S$  or “total surplus” is superior.

The iterative improvement strategy for ESSCP-RTS-LP is based on a novel implementation of a *steepest descent mildest ascent* (SDMA) method. ESSCP-RTS-LP is initialized with a feasible solution,  $\Gamma_{ini}$ , from ESSCP-C and the parameter  $\mu$ . ESSCP-RTS-LP continuously explores solutions within the *active range of cardinalities*,

$\mu$  to  $|\Gamma_{best}| - 1$ , *i.e.*, one less than the cardinality of the best known feasible solution.

Typically, during a search, this range will decrease as feasible solutions are encountered.

The SDMA procedure cycles through three stages: (i) a *descent* stage; (ii) a *swap* stage; and (iii) an *ascent* stage. Whenever a feasible solution is found, the best known solution is updated and ESSCP-RTS-LP enters the descent stage, immediately generating a solution of cardinality  $\mu$ , by applying  $|\Gamma_{best}| - \mu$  single-delete moves in a row. If the resultant solution,  $\Gamma_{new}$ , is not feasible, then ESSCP-RTS-LP enters a swap stage, where it attempts to find a feasible solution of cardinality  $|\Gamma_{new}|$  by repeatedly performing swap moves until that cardinality is deemed “sufficiently explored”. A cardinality *reached from a descent stage* is deemed sufficiently explored on that visit, if either a minimum number of swap moves (20) have been performed, a maximum number of swap moves (100) have been performed, or the total number of swap moves performed at the current cardinality on that visit is  $\geq (50\% * |\Gamma_{new}|)$ .

After sufficient exploration at a cardinality, ESSCP-RTS-LP iteratively *ascends* through the active range of cardinalities by repeatedly making one insert move. After each insert move, the procedure either explores current neighboring solutions or continues ascending. Suppose that, for the entire search, the total number of solutions generated at the current cardinality (including repeats) is less than at the next higher cardinality. If *so*, the procedure will enter the *swap stage* and search at that cardinality until either sufficient exploration is complete or a feasible solution is found. If *not*, ESSCP-RTS-LP continues its ascent. Thus, the procedure distributes its search efforts across all solutions at all active cardinalities.

ESSCP-RTS-LP cycles between the ascent stage and swap stage until it reaches either (i) a cardinality with less visits than the next higher cardinality or (ii) a cardinality one less than the best known feasible solution,  $|\Gamma_{best}|-1$ .

ESSCP-RTS-H uses the same basic cycling methodology as ESSCP-RTS-LP, but with a less aggressive *binary search* (BS) methodology bounded between the current best solution and  $\xi$ . In both procedures, an *aspiration criterion* allowed a move to a “tabu” solution if that solution was better than the global best solution previously encountered.

As described in Appendix 1, both ESSCP-RTS-LP and ESSCP-RTS-H use the *reactive tabu search* strategy to dynamically assign tabu tenure values to cells involved in every move. Both also use the RTS strategy to determine when to perform an escape. Since the *last escape*, if at least 3 solutions have been repeated at least 3 times each, then the procedures invoke an escape at the current cardinality by performing a series of *worst swap moves* (i.e. a worst delete move, followed by a worst insert move). The number of swap moves is either 1 or  $1.5\% * |\Gamma_{new}|$ , whichever is greater.

ESSCP-RTS-LP terminates under one of two conditions: either the search time has exceeded a specified maximum amount of time, or a feasible solution,  $\Gamma^*$ , has been found such that  $|\Gamma^*| = \mu$ . ESSCP-RTS-H terminates only under the former condition.

```

ESSCP_RTS_LP_MAIN(  $\mu$  )
{
     $\Gamma_{init}$  = ESSCP-C(); //generate initial solution
     $\Gamma_{best}^*$  =  $\Gamma_{init}$  ;
    while(maximum time not exceeded)
    {
        if (  $|\Gamma_{best}^*| == \mu$  ) {terminate search} //an optimal solution has been
found
        else if (( $\Gamma_{prev}^*$  is feasible) or (steepest_descent == TRUE))
        {

```

```

    k =  $\left\lfloor \Gamma_{best}^* \right\rfloor - \mu$ ;
    perform a steepest descent by doing k best delete moves in a
row
    steepest_descent = FALSE;
}
else if(Reactive_tabu_search() returns TRUE)
{
    perform an escape with  $1.5\% * \left\lfloor \Gamma_{best}^* \right\rfloor$  bad swap moves
}
else
{
    if(mildest_ascent == TRUE)
    {
        apply the best insert move
        mildest_ascent = Is_new_card_target_ascent_card();
    }
    else
    {
        apply the best swap move at  $\left\lfloor \Gamma_{new}^* \right\rfloor$ 
        curr_cardinality_suff_explored = Is_card_suff_explored();

        if ((curr_cardinality_suff_explored == TRUE) and
( $\left\lfloor \Gamma_{new}^* \right\rfloor == (\left\lfloor \Gamma_{best}^* \right\rfloor - 1)$ ))
        { //here we have ascended up as far as we want to go
            steepest_descent = TRUE;
            curr_cardinality_suff_explored = FALSE;
            mildest_ascent = FALSE;
        }
        else if((curr_cardinality_suff_explored == TRUE) and ( $\left\lfloor \Gamma_{new}^* \right\rfloor$ 
<  $\left\lfloor \Gamma_{best}^* \right\rfloor$ ))
        { //here we can still ascend higher
            mildest_ascent = TRUE;
            steepest_descent = FALSE;
        }
    }
}
}
}

ESSCP_RTS_H_MAIN(  $\xi$  )
{
     $\Gamma_{init}$  = ESSCP-C(); //generate initial solution
     $\Gamma_{best}^* = \Gamma_{init}$ ;
    while(maximum time not exceeded)
    {
        if ( $\left\lfloor \Gamma_{best}^* \right\rfloor == \xi$ ) {terminate search} //an optimal solution has been
found
        else if (( $\Gamma_{prev}^*$  is feasible) or (binary_descent == TRUE))
        {
            k =  $\left\lfloor (\left\lfloor \Gamma_{best}^* \right\rfloor + \xi) / 2 \right\rfloor$ ;
            perform a binary descent by doing k best delete moves in a row

```

```

    binary_descent = FALSE;
}
else if(Reactive_tabu_search() returns TRUE)
{
    perform an escape with  $1.5\% * |\Gamma_{best}^*|$  worst swap moves
}
else
{
    if(binary_ascent == TRUE)
    {
        perform  $k = \left\lfloor \frac{(|\Gamma_{best}^*| + k_{prev})}{2} \right\rfloor$  insert moves;
    }
    else
    {
        apply the best swap move at  $|\Gamma_{new}^*|$ 
        curr_cardinality_suff_explored = Is_card_suff_explored();

        if ((curr_cardinality_suff_explored == TRUE) and
            ( $|\Gamma_{new}^*| == (|\Gamma_{best}^*| - 1)$ ))
            { //here we have ascended up as far as we want to go
                binary_descent = TRUE;
                curr_cardinality_suff_explored = FALSE;
                binary_ascent = FALSE;
            }
            else if((curr_cardinality_suff_explored == TRUE) and ( $|\Gamma_{new}^*|$ 
                <  $|\Gamma_{best}^*|$ ))
                { //here we can still ascend higher
                    binary_ascent = TRUE;
                    binary_descent = FALSE;
                }
            }
        }
    }
}

```

```

Is_new_card_target_ascent_card(cardinality stats for  $|\Gamma_{new}^*|$  and  $|\Gamma_{new}^*|_{+1}$ )
{
    if ( $|\Gamma_{new}^*| == (|\Gamma_{best}^*| - 1)$ ) {stop_at_curr_card = TRUE;}

    else if(( $|\Gamma_{new}^*|_{+1} \neq |\Sigma|$ ) && (num_visits at  $|\Gamma_{new}^*| \leq$  num_visits at
 $|\Gamma_{new}^*|_{+1}$ ))
    {
        stop_at_curr_card = TRUE;
    }
    else {stop_at_curr_card = FALSE;}

    return stop_at_curr_card;
}

```

```
}
```

```
Is_card_suff_explored(cardinality stats for  $\left| \Gamma_{new}^* \right|$ )  
  
{ suff_explored = FALSE;  
  if (arrived_at_from_delete)  
  { if (num_swaps_sofar >= 20)  
    { if (num_swaps_sofar >= 100){suff_explored = TRUE;}  
      else if (num_swaps_sofar >= (.50 *  $\left| \Gamma_{new}^* \right|$ )){suff_explored =  
TRUE;}  
    }  
  }  
  else if (arrived_at_from_insert)  
  
  { if (num_swaps_sofar >= 30)  
    { if (num_swaps_sofar >= 100){suff_explored = TRUE;}  
      else if (num_swaps_sofar >= (.50 *  $\left| \Gamma_{new}^* \right|$ )){suff_explored =  
TRUE;}  
    }  
  }  
  
  return suff_explored;  
}
```

### Appendix 3

#### MESCP-RTS

The MESCP-RTS starting solution with  $\beta$  cells selected is generated by MESCP-C. MESCP-RTS maintains the cardinality at  $\beta$  by using a variant of the classical *swap* neighborhood (Laguna and Barnes, 1993), *deleting* a selected cell and *inserting* an unselected cell. Simply maximizing the number of satisfied surrogate targets does not provide a sufficient level of discrimination between solutions to allow an efficient and effective search process.

This is overcome by using a *fine-grained* objective function (Carlton and Barnes 1996). MESCP-RTS uses a strict hierarchical structure to compare two solutions:

- (a) a solution with a greater number of satisfied surrogate targets is superior to any solution with fewer;
- (b) if two solutions satisfy the same number of surrogate targets, the solution with the *smallest single-species deficit* is superior;
- (c) if two solutions share the same number of surrogate targets satisfied and the same *smallest single-species deficit*, the solution with the greater total surplus is superior.

Given an incumbent solution, MESCP-RTS iteratively invokes the best non-tabu swap move at each iteration. After each move, the deleted cell may not be selected for tabu tenure iterations, unless inserting that cell would satisfy the *aspiration criterion*.

An additional diversification strategy is implemented: if the number of iterations since the last escape exceeds a stated escape threshold, the escape procedure is invoked. The MESCP-RTS also uses the escape strategy embodied in the ESSCP-RTS procedures of making a series of “worst swaps”, as described in Appendix 2.

#### **MESCP\_RTS\_MAIN**

```

{    $\Gamma_{init}$  = MESCP-C(); //generate initial solution
    while(maximum time not exceeded)
    {
        need_to_escape = Is_Escape_necessary();
        if(need_to_escape == TRUE) {Apply_escape_procedure();}
        else {Apply_swap_move();}
        Reactive_tabu_search(); // update search parameters and status
    }
}

```

#### **Is\_escape\_necessary()**

```

{ if ((time since best solution improved > IMPROVEMENT_THRESHOLD)
    or (Reactive_tabu_search() returns TRUE))
    {return TRUE;}
  else
    {return FALSE;}
}

```

## **Appendix 4**

## Artificial Data Set Creation

Each *artificial* data set was created by selecting  $|\Sigma|$  and  $|\mathcal{A}|$  and randomly assigning uniformly distributed 6 decimal place values (between 0 and 1) to the elements of the surrogate representation  $|\mathcal{A}| \times |\Sigma|$  matrix until a user-specified density was reached. Next, targets are assigned to each surrogate based on a percentage of its total representation in the matrix. Figure 1 describes the process by which the twenty artificial data sets were created using a logarithmic procedure to sample the space systematically.

## References

- Ball, I. R. 2000. "Mathematical Applications for Conservation Ecology: The Dynamics of Tree Hollows and The Design of Nature Reserves." PhD Thesis, The University of Adelaide.
- Ball, I. R. and Possingham, H. P. 2000. "MARXAN (V 1.8.2) User's Manual". <http://www.ecology.uq.edu.au/marxan.htm>.
- Barnes, J. W., Wiley, V., Moore, J. and Ryer, D. 2004. "Solving the Aerial Fleet Refueling Problem using Group Theoretic Tabu Search." *Mathematical and Computer Modeling*, in press.
- Battiti, R. and G. Tecchiolli. 1994. "The Reactive Tabu Search." *Operations Research Journal on Computing*, **6**:2, 126 -140.
- Carlton, W. and Barnes, J. W. 1996. "Solving the Traveling Salesman Problem with Time Windows Using Tabu Search." *Institute of Industrial Engineering Transactions* **28**: 617 -629.
- Church, R. L. and ReVelle, C. 1974. "The Maximal Covering Location Problem." *Papers in Regional Science: The Journal of the Regional Science Association International* **32**: 101 -118.
- Cocks, K. D. and Baird, I. A. 1989. "Using Mathematical Programming to Address the Multiple Reserve Selection Problem: An Example from the Eyre Peninsula, S. Australia." *Biological Conservation*. 49:113-130.
- Colletti, B., and Barnes, J. W. 2004. "Using Group Theory to Construct and Characterize Metaheuristic Search Neighborhoods." In Rego, C. and Alidaee, B. Eds. *Adaptive Memory and Evolution: Tabu Search and Scatter Search*. Dordrecht: Kluwer.
- Csuti, B., Polasky, S., Williams, P. H., Pressey, R. L., Camm, J.D., Kershaw, M., Kiester, A. R., Downs, B., Hamilton, R., Huso, M., and Sahr., K. 1997. "A Comparison of Reserve Selection Algorithms Using Data on Terrestrial Vertebrates in Oregon." *Biological Conservation* **80**: 83 -97.
- Fischer, D. T. and Church, R. L. 2003. "Clustering and Compactness in Reserve Site Selection: An Extension of the Biodiversity Management Area Selection Model" *Forest Science*. 49 (4): 555-565.
- Garey, M. R. and Johnson, D. S. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco: W. H. Freeman.
- Garson, J., Aggarwal, A. and Sarkar, S. 2002. "ResNet Ver 1.2 Manual." Report. Austin: Biodiversity and Biocultural Conservation Laboratory, University of Texas at Austin. <http://uts.cc.utexas.edu/~consbio/Cons/reports.html>.
- Glover, F. and Laguna, M. 1997. *Tabu Search*. Dordrecht: Kluwer.
- Glover, F. and Laguna, M. 1993. "Tabu Search." In Reeves, C. R. Ed. *Modern Heuristic Techniques for Combinatorial Problems*. Oxford: Blackwell, pp. 70 -150.

- Harwig, J., Barnes, J. W., Moore, J., 2004, "An Adaptive Tabu Search Approach for 2-Dimensional Orthogonal Packing Problems," Technical Report 04-01, The Graduate Program in Operations Research and Industrial Engineering, The University of Texas at Austin, Austin, Texas.
- Hao, J. and Pannier, J. 1998. "Simulated Annealing and Tabu Search for Constraint Solving." *Fifth Intl. Symposium on Artificial Intelligence and Mathematics*. Electronic Proceedings: <http://rutcor.rutgers.edu/~amai/Proceedings.html>.
- Justus, J. and Sarkar, S. 2002. "The Principle of Complementarity in the Design of Reserve Networks to Conserve Biodiversity: A Preliminary History." *Journal of Biosciences* **27**(S2): 421 -435.
- Kelley, C., Garson, J., Aggarwal, A., and Sarkar, S. 2002. "Place Prioritization for Biodiversity Reserve Network Design: A Comparison of the SITES and ResNet Software Packages for Coverage and Efficiency." *Diversity and Distributions* **8**: 297 -306.
- Kirkpatrick, S., Gelatt, Jr., C. D., and Vecchi, M. P. 1983. "Optimization by Simulated Annealing." *Science* **220**: 671 -220.
- Laguna, M., and Barnes, J. W. 1993. "A Tabu Search Experience in Production Scheduling." *Annals of Operations Research* **41**: 41 -61.
- Margules, C. R. and Pressey, R. L. 2000. "Systematic Conservation Planning." *Nature* **405**: 242 -253.
- Margules, C. R., Nicholls, A. O. and Pressey, R. L. 1988. "Selecting Networks of Reserves to Maximize Biological Diversity." *Biological Conservation* **43**: 63-76.
- Moffett, A. and Sarkar, S. 2005. "Incorporating Multiple Criteria into the Design of Conservation Area Networks: A Minireview with Recommendations," *Diversity and Distributions*. (In press).
- Nemhauser, G. and Wolsey, L. 1988. *Integer and Combinatorial Optimization*. New York: John Wiley & Sons.
- Okin, W. J. 1997. "The Biodiversity Management Area Selection Model: Constructing a Solution Approach." M. A. Thesis, Department of Geography, University of California--Santa Barbara.
- Papadimitriou, C. 1994. *Computational Complexity*. Reading: Addison-Wesley.
- Polasky, S., Camm, J. D., Solow, A. R., Csuti, B., White, D. and Ding, R. 2000. "Choosing Reserve Networks with Incomplete Species Information." *Biological Conservation* **94**: 1 -10.
- Possingham, H. P., Ball, I. R. and Andelman, S. 2000. "Mathematical Methods for Identifying Representative Reserve Networks." In Ferson, S. and Burgman, M. Eds. *Quantitative Methods for Conservation Biology*. New York: Springer, pp. 291-305.

- Pressey, R. L., Possingham, H. P., Logan, V. S., Day, J. R. and Williams, P. H. 1999. "Effects of Data Characteristics on the Results of Reserve Selection Algorithms." *Journal of Biogeography*. 26 (1): 179-191.
- Pressey, R. L., Possingham, H. P. and Day, J. R. 1997. "Effectiveness of Alternative Heuristic Algorithms for Approximating Minimum Requirements for Conservation Reserves." *Biological Conservation* 80: 207-219.
- ReVelle, C. S., Toregas, C., Swain, R. and Bergman, L. 1971 "The location of emergency service facilities." *Operations Research* 19:1363 -73.
- ReVelle, C. S., Williams, J. C. and Boland, J. J. 2002. "Counterpart Models in Facility Location Science and Reserve Selection Science." *Environmental Modeling and Assessment* 7: 71 -80.
- Rodrigues, A. S., Cerdeira, J. O., and Gaston, K. J. 2000. "Flexibility, Efficiency, and Accountability: Adapting Reserve Selection Algorithms to More Complex Conservation Problems." *Ecography* 23: 565 -574.
- Rodrigues, A. S. and Gaston, K. J. 2002. "Optimisation in Reserve Selection Procedures-Why Not?" *Biological Conservation* 107: 123 -129.
- Sarakinos, H., Nicholls, A. O., Tubert, A., Aggarwal, A., Margules, C. R. and Sarkar, S. 2001. "Area Prioritization for Biodiversity Conservation in Québec on the Basis of Species Distributions: A Preliminary Analysis." *Biodiversity and Conservation* 10: 1419 -1472.
- Sarkar, S. 2005. *Biodiversity and Environmental Philosophy: An Introduction to the Issues*. New York: Cambridge University Press.
- Sarkar, S., Aggarwal, A., Garson, J., Margules, C. R., and Zeidler, J. 2002. "Place Prioritization for Biodiversity Content." *Journal of Biosciences* 27 (S2): 339 -346.
- Sarkar, S., Pappas, C., Garson, J., Aggarwal, A., and Cameron, S. 2004. "Place Prioritization for Biodiversity Conservation Using Probabilistic Surrogate Distribution Data." *Diversity and Distributions* 10: 125 -133.
- Sarkar, S., Parker N. C., Garson, J., Aggarwal, A. and Haskell, S. 2000. "Place Prioritization for Texas Using GAP Data." *Gap Analysis Program Bulletin* 9: 48 -50.
- Soulé, M. E. and Sanjayan, M. A. 1998. "Conservation Targets: Do They Help?" *Science* 279: 2060 -2061.
- Tognelli, M. 2004. "Assessing the Utility of Surrogate Groups for the Conservation of South American Terrestrial Mammals." *Biological Conservation*, under review.
- Vane-Wright, R. I., Humphries, C. J., and Williams, P. H. 1991. "What to Protect?—Systematics and the Agony of Choice." *Biological Conservation* 55: 235 -254.
- Williams, H. P. 1993. *Model Solving in Mathematical Programming*. Chichester: John Wiley & Sons.

Zheng, Y., Pieprzyk, J., and Seberry, J. 1993, "HAVAL—A One-Way Hashing Algorithm with Variable Length of Output", *Lecture Notes in Computer Science* **718**: 83 - 104.



**Table 1**  
**Empirical Data Sets**

This table presents three properties of the six empirical data sets used in this analysis. The rows specify the number of cells ( $n$ ) in that data set; the number of surrogates ( $m$ ) present in that data set; and the density value of the representation  $m \times n$  representation matrix calculated as *number of non-zero entries* /  $m \times n$ ; ‘Representation Type’ denotes whether the surrogate representation values are probabilistic expectations or presence/absence (*i.e.* ‘binary’) data. The data for Namibia were from termite genera (Sarkar *et al.* 2002), for Texas and West Virginia were from animal species from the GAP Analysis Project (see Sarkar *et al.* [2000]), for Québec mainly from species at risk (Sarakinos *et al.* 2001), for Latin America for animal species (Tognelli 2004), and for Ecuador for vegetation types (Sarkar *et al.* 2004).

Properties	Data Sets					
	Namibia	Texas	Québec	Latin America	West Virginia	Ecuador
Cells ( $m$ )	1250	1183	39537	1774	94771	37727
Surrogates ( $n$ )	33	655	719	662	323	46
Density	0.087	0.469	0.002	0.083	0.548	0.03
Representation Type	binary	binary	binary	binary	binary	probabilistic

**Table 2**  
**ESSCP for Presence-Absence Data**

The table presents results from six procedures used to solve the ESSCP on five empirical data sets with surrogate targets of 1 and 10%. For each procedure and data set pair, the results reported are the size of the best solution found (*i.e.* the number of cells selected) followed below by the time in seconds that that solution was found by the procedure. CPLEX 8.1 was used in two ways: CPLEX 8.1-O denotes the CPLEX 8.1 procedure as an optimal algorithm, whereas CPLEX 8.1-H denotes the procedure with its heuristics activated. (See Sections 2.2 and 2.3.2 for more details.) ESSCP-C denotes the complementarity procedure described in Section 2.3.1. The number of *redundant cells* (see Section 2.3.1) removed to obtain each ESSCP-C solution is also listed. ESSCP-RTS-LP refers to the TS procedure that uses the LP relaxation value as a lower bound; ESSCP-RTS-H is the TS procedure that uses the heuristic calculation of the lower bound. Both procedures are described in Appendix 2. The MARXAN procedure is described in Section 2.4.1. (\*: optimal solution; n/a: not applicable.)

Algorithm	Data Sets									
	Namibia		Texas		Québec		Latin America		West Virginia	
	1	10 %	1	10 %	1	10%	1	10%	1	10%
CPLEX 8.1-O	6*	41*	21*	119*	163*	3063*	108*	193*	6*	9476*
	1	1	1	29	1	2	1	18	179	203
ESSCP-C	6*	41*	23	133	167	3079	113	210	7	9918
	42	1	1	6	84	1247	3	7	39	14321
(redundant cells)	1	1	2	23	6	10	42	4	1	411
CPLEX 8.1-H	6*	41*	21*	119*	163*	3063*	108*	193*	6*	9476*
	1	1	1	50	1	1	1	17	219	224
ESSCP-RTS-LP	6*	41*	22	119*	167	3077	108*	193*	6*	9476*
	n/a	n/a	29	314	n/a	497	322	204	1	2001
ESSCP-RTS-H	6*	41*	21*	119*	166	3073	110	200	6*	9476*
	n/a	n/a	2	36	473	674	243	865	58	382
MARXAN	6*	41*	21*	124	183	3425	110	202	9	9636
	144	793	11580	11880	6232	6255	10828	5920	43416	46551

**Table 3**  
**ESSCP for Probabilistic Data**

This table presents results from five procedures used to solve the ESSCP on 21 probabilistic data sets: one empirical probabilistic data set (Ecuador) with surrogate targets of 1 and 10% and 20 artificial data sets. For notation details, see Table 2.

Algorithm	Data Sets																					
	Ecuador		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
	1	10%																				
CPLEX 8.1-O	51 2482	3784 1501	401 906	227 77	129 86	77 1246	449 189	251 276	141 236	492 498	271 491	526 1432	0 n/a	1102 6140	595 4819	0 n/a	0 n/a	0 n/a	0 n/a	0 n/a	0 n/a	0 n/a
ESSCP-C	57	3780	404	231	131	76	455	256	146	503	278	542	2105	1127	607	331	2228	1185	636	2343	1235	2441
(redundant cells)	5	223	7	3	2	1	9	5	3	19	11	37	372	192	102	57	732	389	211	1501	793	2426
	4	6	5	1	1	1	4	1	0	2	2	4	12	6	5	2	7	6	4	12	16	4
CPLEX 8.1-H	50 605	3780 369	398 2	224 835	127 2	73 7	447 9	248 9	138 27	489 9	268 15	523 1872	2062 62	1100 74	584 145	314 426	2184 179	1149 301	607 858	2283 615	1190 1483	2376 4129
ESSCP- RTS-LP	48 1989	3777 27	397* 72	222* 129	124 29	68 80	446 96	246 91	134 1066	486 222	270 1734	519 826	2069 153	1096 3375	575 1257	301 1109	2189 292	1155 379	597 3531	2290 8840	1196 1261	2383 1944
ESSCP- RTS-H	51 1866	3776 1323	398 150	225 1	127 9	72 295	448 1352	248 1296	138 68	491 538	267 2686	525 524	2069 154	1103 2486	586 74	314 157	2190 483	1153 210	609 148	2294 950	1197 673	2389 3505

**Table 4**  
**MESCP for Presence-Absence Data (Budget =  $\mu - 1$ )**

This table presents results from five procedures used to solve the MESCP with a budget level of  $\mu - 1$  on five empirical data sets each with surrogate targets of 1 and 10%. MESCP-C refers to the MESCP complementarity heuristic described in Section 2.3.1. MESCP-RTS refers to the MESCP tabu procedure described in Section 2.4.2.2. For notation details, see Table 2.

Algorithm	Data Sets									
	Namibia		Texas		Québec		Latin America		West Virginia	
	1	10 %	1	10 %	1	10%	1	10%	1	10%
CPLEX 8.1-O	32*	32*	654*	0	718*	718	661*	656	322*	322*
	1	1	2	n/a	1	275	1	1526	215	219
MESCP-C	32*	32*	651	384	708	691	654	576	321	321
	1	1	2	4	65	1024	3	5	26	111114
CPLEX 8.1-H	32*	32*	654*	355	718*	718	661*	660	322*	322*
	1	1	1	59	1	782	1	4830	211	229
MESCP-RTS	32*	32*	651	461	708	691	654	640	321	321
	n/a	n/a	n/a	2165	n/a	n/a	n/a	228	n/a	n/a
MARXAN	31	31	645	334	439	588	659	653		0
	6300	8404	5367	15783	11691	8734	11567	2084		197640

**Table 5**  
**MESCP for Presence-Absence Data (Budget = 7.5 %)**

This table presents results from five procedures used to solve the MESCP with a budget level of 7.5% on five empirical data sets each with surrogate targets of 1 and 10%. (See Appendix 4 for details on data sets 1-20.) For notation details, see Tables 2 and 4.

Algorithm	Data Sets									
	Namibia		Texas		Québec		Latin America		West Virginia	
	1	10 %	1	10 %	1	10%	1	10%	1	10%
CPLEX 8.1-O	33*	33*	655*	254	719*	718*	662*	453	323*	195
	1	1	2	1386	1	168	1	6188	194	5026
MESCP-C	33*	33*	655*	228	719*	590	662*	347	323*	178
	1	1	2	4	66	948	2	3	30	8260
CPLEX 8.1-H	33*	33*	655*	290	719*	718*	662*	478	323*	195
	1	1	1	3203	1	6	1	6460	202	4937
MESCP-RTS	33*	33*	655*	291	719*	590	662*	472	323*	183
	n/a	n/a	n/a	57	n/a	n/a	n/a	2700	n/a	541
MARXAN	33*	33*	645	121	484	593	662*	253	323*	
	195	4587	13397	21226	9125	1020	10504	5760	15420	

**Table 6**  
**MESCP for Probabilistic Data (Budget =  $\mu-1$ )**

This table presents results from four procedures used to solve the MESCP with a budget level of  $\mu-1$  on 21 probabilistic data sets: one empirical probabilistic data set (Ecuador) with surrogate targets of 1 and 10% and 20 artificial data sets. (See Appendix 4 for details on data sets 1 to 20.) For notation details, see Tables 2 and 4.

Algorithm	Data Sets																					
	Ecuador		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
	1	10%																				
CPLEX	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8.1-O	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
MESCP-C	25	13	19	32	61	131	30	61	128	63	121	124	81	147	318	683	146	296	660	305	635	602
	3	171	3	1	1	0	7	10	0	1	1	4	292	147	77	41	578	297	156	1169	602	2342
CPLEX	36	37	18	4	1	81	9	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
8.1-H	626	2178	2031	3479	4248	5442	5736	n/a	6956	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
MESCP-	41	45	22	39	73	153	39	70	150	74	141	144	95	168	353	736	173	347	698	330	700	658
RTS	82	2	1	38	1	11	15	26	23	8	1	6	213	167	20	10	695	70	5	182	272	218

**Table 7**  
**MESCP for Probabilistic Data (Budget = 7.5%)**

The table presents results from four procedures used to solve the MESCP with a budget level of 7.5% on 21 probabilistic data sets: one empirical probabilistic data set (Ecuador) with surrogate targets of 1 and 10% and 20 artificial data sets. (See Appendix 4 for details on the data sets.) For notation details, see Tables 2 and 4.

Algorithm	Data Sets																					
	Ecuador		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
	1	10%																				
CPLEX 8.1-O	46*	40	25*	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	277	161	205	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
MESCP-C	46	0	25	36	21	35	42	7	15	7	7	0	0	0	0	2	0	0	0	0	0	0
	3	128	3	1	0	0	7	4	1	13	6	24	266	126	63	33	496	245	125	961	482	1866
CPLEX 8.1-H	46*	44	25*	26	2	30	9	9	4	12	11	12	11	9	10	11	10	6	9	7	10	0
	238	1394	348	3	5667	937	4142	5	1111	1828	1149	767	4993	6041	2609	6973	5478	2432	3046	4826	502	n/a
MESCP-RTS	46*	24	25*	42	40	63	47	37	49	37	36	31	14	21	31	41	2	19	29	1	1	47
	n/a	42	n/a	1	636	2055	16	133	485	332	2702	1027	1182	982	1536	1417	84	802	561	107	49	3540

## Figure Captions

**Figure 1:** The class of data sets from 1 to 10 and the class from 11 to 20 were created using a logarithmic method to vary the number of cells and surrogates evenly across the class. The relationship between the data sets in a class is captured in the tree structure, where a data set's specifications are given below its name in terms of '# cells/ # surrogates'. **Data Sets 1 –10:** Data set 10, at the root of the tree, has 6 000 cells and 200 surrogates. Halving the number of cells and maintaining the same number of surrogates create the left child in the tree. Halving the number of species and maintaining the same number of cells create the right child in the tree. **Data Sets 11 –20:** Data set 20 is the root of the tree with 25 000 cells and 1000 surrogates. Left and right children are created exactly as in Data Set 1-10. The density is 50 % for all data sets.

**Figure 1**

