Title
The SITES reserve selection system: A critical review

Journal
Environmental Modeling and Assessment, 10(3)

ISSN
1420-2026

Authors
Fischer, Douglas T.
Church, R L

Publication Date
2005-09-01

DOI
10.1007/s10666-005-9005-7

Peer reviewed

JrnlID 10666_ ArtID 9005_Proof# 1 - 15/07/2005
Environmental Modeling and Assessment (2005)
DOI: 10.1007/s10666-005-9005-7

The SITES reserve selection system: A critical review

Douglas T. Fischer[a,]* and Richard Church[b]
[a] Department of Geography, University of California, Santa Barbara, CA 93106-4060, USA
E-mail: fischer@geog.ucsb.edu
[b] Department of Geography/National Center for Geographic Information and Analysis, University of California, Santa Barbara, CA 93106-4060, USA
E-mail: church@geog.ucsb.edu

Numerous models have been put forth to help with the growing demand for the establishment of biodiversity reserves. One site selection model that has been used in several recent studies is SITES [S.J. Andelman, I. Ball, F.W. Davis and D.M. Stoms, SITES V 1.0: an analytical toolbox for designing ecoregional conservation portfolios, Unpublished manual prepared for the nature conservancy, 1999, 1–43. (available at http://www.biogeog.ucsb.edu /projects/tnc/toolbox.html)]. SITES includes two heuristic solvers: based on Greedy and Simulated Annealing. We discuss the formulation of the SITES model, present a new formulation for that problem, and solve a number of test problems optimally using off-the-shelf software. We compared our optimal results with the SITES Simulated Annealing heuristic and found that SITES frequently returns significantly suboptimal solutions. Our results add further support to the argument, started by Underhill [L.G. Underhill, Optimal and suboptimal reserve selection algorithms, Biol. Conserv. 70 (1994) 85–87], continuing through Rodrigues and Gaston [A.S.L. Rodrigues and K.J. Gaston, Optimization in reserve selection procedures – why not?, Biol. Conserv. 107 (2002) 123–129], for greater integration of optimal methods in the reserve design/selection literature.

Keywords: reserve site selection, optimization, integer programming, heuristics, model formulation, Simulated Annealing

1. Introduction

As awareness of conservation issues has grown over the last several decades, a growing number of planners have focused on ways to conserve individual species, whole ecosystems, and other natural resources. A key strategy for conservation has been the establishment of reserves that can be managed for the benefit of the targeted conservation elements, be they endangered species, threatened vegetation communities, unique habitat types, or some other element of conservation concern. Until recently, most efforts to establish reserves have focused on areas with scenic and recreational value, resulting in ad hoc reserve networks with substantial redundancy and many gaps. As more areas experience environmental degradation and more species are threatened with extinction, greater attention has focused on designing comprehensive sets of reserves, where all conservation elements in a region are adequately represented in the reserve system [4–6].

Social and economic considerations often preclude simply conserving all land in a region; so the problem of reserve design has focused on selecting small portions of a region for conservation. Because there may be considerable flexibility in which portions are selected, the problem is far from simple. This design dilemma has fueled the development of a wide variety of computer-based reserve site selection models (e.g., [4–17]).

Reserve site selection models are different from many other applications of optimization techniques. In most cases the ecological data available to conservation planners contain much larger uncertainties than in more traditional optimization applications in business or the military. In addition, unmodeled objectives (e.g., aesthetics, public opinion, politics, etc.) often play a much more influential role in the implementation of a reserve system than in other applications (e.g., [18]). The primary utility of reserve site selection algorithms, then, is not to produce single, prescriptive solutions. Rather, the principle utility of reserve site selection algorithms is to explore the ranges of performance possible for various modeled objectives, and the potential tradeoff curves that may exist between them. The optimal solutions thus produced then provide benchmarks against which specific on-the-ground plans can be compared.

Prendergast et al. [19] argue that there is a gap between theory and application and that current site-selection algorithms do not address many of the pressing and practical issues in reserve design, including ease of use for managers and decision makers. Although Pressey and Cowling [20] answer many of the issues raised by Prendergast et al. [19], there remains a real need to make better selection models, solution procedures, and decision support systems for reserve planning and design. Within this same spirit, there is a need to test and compare existing models and solution methods.

One model that has received significant attention and has been used in recent studies of reserve design [21–24] is the SITES model [1,25] that was developed for The Nature Conservancy. The SITES program includes various mapping and analysis functions, all built around a conceptual model for reserve selection, and a pair of heuristics for solving reserve selection problems based on this model. The model is an area-representation model similar to the

* Corresponding author.

BMAS model [6] with a goal-programming approach and added terms to encourage clustering. Our objective in this paper is to present a reformulation of this model, test off-the-shelf software in solving this model, and compare these results to the solvers provided with the SITES program. We will show that the performance of the existing heuristics for the SITES model can be improved.

In the next section, we describe the SITES model, based upon both the SITES user manual as well as several supporting papers. Working from this conceptual model, the SITES developers opted for the development of two heuristics. These have been tested on several problems [13] and have been compared against each other [26]. Up to now, the SITES model has been solved only heuristically; thus, the quality of the solutions determined from the heuristic solvers has never been fully evaluated. In a subsequent section, we offer an alternate formulation for the SITES model. We demonstrate that this new formulation can be used to solve problem instances optimally. Thus, we are able to provide an assessment of the efficacy of the heuristics already developed for solving SITES. We provide a comparison of heuristic and optimal approaches and then conclude with a summary and final assessment.

2. The SITES model

The SITES program is described as picking from among a number of feasible sites, a set that comprises a portfolio [1]. The objective is to pick sites for the portfolio in such a manner that all conservation goals are met and that the cost is minimized. That is, the stated objective is to find the minimal cost set of sites such that each conservation goal is satisfied. The conservation goals can include representation goals (coverage of species or area of habitat) and spatial configuration goals. The SITES program attempts to select a minimal cost portfolio where the portfolio cost is defined as:

$$\begin{aligned}\text{Total Portfolio Cost} &= (\text{cost of selected sites}) \\ &\quad + \big(\text{penalty cost for not meeting the stated} \\ &\qquad \text{conservation goals for each element}\big) \\ &\quad + \big(\text{cost of spatial dispersion of the selected sites} \\ &\qquad \text{as measured by the total boundary length of} \\ &\qquad \text{the sites in portfolio}\big). \end{aligned} \tag{1}$$

This is further described as:

$$\begin{aligned}\text{Total Cost} = &\sum_i \text{Cost site } i \\ &+ \sum_j \text{Penalty cost for element } j \\ &+ w_b \sum \text{Boundary length.}\end{aligned} \tag{2}$$

This “total cost” function represents the sum of the costs of selected sites (e.g., site area, acquisition cost, opportunity cost, habitat quality), plus the sum of the penalties for not meeting specific conservation targets, plus the weighted perimeter of all selected sites. The third term of the cost objective is weighted by the term, w_b (and is actually a measure of clustering and compactness rather than of spatial dispersion [17]). The higher the value of w_b, the more important it is to select a set of sites that are clustered with a small perimeter, even if doing so increases the other costs somewhat. Thus, the total cost function allows for tradeoffs between boundary length (i.e., an encouragement to cluster elected sites) and the costs of sites and penalty costs for not meeting specific conservation targets. In addition to the terms described above, the SITES documentation [1] includes references to the selection of spatially separated clusters of reserves. Our understanding is that this functionality was never fully implemented in SITES (D. Stoms, personal communication, 2001), and so we have omitted it.

The second term of the total cost function, as described by Andelman et al. [1], involves the penalty costs associated with falling short of any conservation targets. In minimizing total portfolio costs, the penalty function encourages sites to be chosen in such a manner that all conservation targets are met. When all targets are met or exceeded, then the penalty costs are zero for all conservation elements. A conservation target for an element is stated in terms of a minimum desired value. An element may represent a species, habitat type, or other factor of interest. It is assumed that each site contains a specific quantity of each element. The total of that element over all selected sites represents the amount protected among the sites in the selected portfolio. If the total is lower than the target for that element, then a penalty cost is incurred that is proportional to the shortfall. For example, consider the hypothetical problem and solution portfolio comprised four sites presented in table 1.

For this example, assume that these sites do not share any boundary in common. There are five different elements with conservation targets. The first two elements involve specific types of habitat (called habitat types 1 and 2). The remaining three elements involve the representation of three different species. For example, site 65 contains 200 ha of habitat type 1, 2,500 ha of habitat type 2, contains species C, but not species A or B. (Note, for species presence data, a 1 means the species is present at the site and a 0 means that the species is absent). Together, this portfolio of four sites contains 3,500 ha of habitat type 1 and 4,500 ha of habitat type 2. Because the target values for each habitat type is 4,000 ha, the portfolio falls short of habitat type 1 by 500 ha and meets the target for habitat type 2. The element target shortfall amounts are given in the penultimate column of the table. For species presence targets, it is desired to pick sites so that each species is present in at least two sites in the portfolio. Note that species B is found at sites 21 and 109. Thus, species B is found at two sites and meets the conservation target. Unfortunately, species A is present at only one site so a shortfall of representation occurs for this species. In calculating the total cost, we have multiplied each cost by a

Table 1
Sample SITES problem with two habitat protection targets, and three species representation targets.

Problem definition			Selected portfolio				Portfolio cost	
	Weight	Target value	Site 21	Site 65	Site 13	Site 109	Element shortfall	Weighted objective
Boundary	0		11,267	14,321	22,456	16,728		0
Cost	1		1,000	1,000	1,000	1,000		4,000
Habitat type 1	3	4,000	2,000	200	0	1,300	500	1,500
Habitat type 2	3	4,000	100	2,500	1,900	0	0	0
Species A	500	2	1	0	0	0	1	500
Species B	500	2	1	0	0	1	0	0
Species C	500	2	0	1	1	0	0	0

The selected portfolio of four sites meets the habitat protection target for type 2, but not type 1, and represent species B and C adequately, but not species A. The portfolio cost is a weighted sum of the costs of the selected sites, and the penalties for not meeting protection targets.

weight listed on the left side of the table. While not included in formula (2), SITES provides for separate weights for each element shortfall, as described below. (While SITES does not provide for weights for site costs, such weights are easily applied before loading cost data into the program.) The total cost for this portfolio, as shown in the rightmost column of table 1 is:

$$\begin{aligned}\text{Total cost} &= 1 \times 4,000(\text{site cost}) \\ &\quad + 3 \times 500(\text{penalty cost for habitat 1}) \\ &\quad + 3 \times 0(\text{penalty cost for habitat 2}) \\ &\quad + 500 \times 1(\text{penalty cost for Species A}) \\ &\quad + 500 \times 0(\text{penalty cost for Species B}) \\ &\quad + 500 \times 0(\text{penalty cost for Species C}) \\ &\quad + 0 \times 64,772(\text{penalty for boundary length}) \\ &= 4,000 + 1,500 + 0 + 500 + 0 + 0 + 0 \\ &= 6,000\end{aligned} \tag{3}$$

It is important to note that the overall penalty cost for a given shortfall is proportional to the amount of shortfall. That is, the penalty cost is a linear function with respect to shortfall. In McDonnell et al. [26], the penalty cost for a given target is normalized by the amount of the target (and multiplied by an additional, heuristically determined weight), so that a balance can be struck between targets involving small acreage and those involving larger acreage. Each possible portfolio has a calculated total cost. The objective is to identify the portfolio with the smallest “total cost.” If the units of site cost and element penalty are very different in magnitude, lowest-cost solutions may involve selecting all the area to eliminate any penalties, or accepting all penalties to avoid the cost of selecting any sites, or somewhere in between.

McDonnell et al. [26] provide additional description of the underpinning model of SITES. The following notation is necessary to describe their formalism:

c_i – Total area or cost of site i;
a_{ik} – Area or other measure of conservation value k on site i;
b_i – Total boundary length of site i;
b_k – Required area or amount of conservation value k needed in portfolio;
b_{ij} – Length of shared boundary between sites i and j;
K – Set of all conservation elements k;
I – Set of all sites i;
m – Total number of sites available for selection

The decision to select a site for the portfolio can be represented by the following 0–1 decision variable:

$$x_i = \begin{cases} 1, & \text{if site } i \text{ is selected for the portfolio} \\ 0, & \text{otherwise} \end{cases}$$

Using this notation, McDonnell et al. [26] describe the following “crisp” optimization problem:

Minimize $C(x)$

$$= \sum_i c_i x_i + w_b \left(\sum_i b_i x_i - 2 \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} b_{ij} x_i x_j \right) \tag{4}$$

But ensure that sites selected for portfolio contain a minimum quantity of element k

$$\sum_i a_{ik} x_i \geq L_k \text{ for each element } k \in K \tag{5}$$

Subject to:
Enforce integer restrictions on site decision variables

$$x_i = 0, 1 \text{ for each site } i \in I \tag{6}$$

This formulation is an integer nonlinear programming model. The objective (4) involves the minimization of site costs and weighted total boundary length. The boundary length is calculated as the sum of all boundaries of each of the selected units minus twice the distances of the shared edges (since each shared edge is counted twice in the sum). If a pair of sites i and j are both selected for the portfolio, then the term $x_i x_j$ will equal 1, and two times the shared boundary of b_{ij} will be subtracted from the total sum of the individual site boundary lengths. If the term $x_i x_j$ is zero, then at least one of the two sites i and j has not been selected and no shared boundary is subtracted. If $b_{ij} = 0$, then the two sites i and j units are not adjacent and selecting them will not alter the total boundary length. The first type of constraint (5) ensures at least a prescribed minimum area (or some other measure) of each conservation element is achieved by the selected sites. The second type of constraint (6) refers to the integer restrictions on the decision variables. We refer to this as a "crisp" model as each conservation element must be protected by selecting a set of sites that contain at least a minimum amount of area (or representation) for the element.

Recognizing that it is easier to develop a heuristic for an unconstrained optimization problem, McDonnell et al. [26] present a reformulated version of the above model where shortfalls in each conservation target are allowed. Consider the following additional notation:

- u_k – Amount of protection shortfall, if any, for conservation element k;
- w_k – Penalty weight per unit of shortfall for conservation element k;
- SPF_k – User-specified weight for each element.

The value of w_k is determined heuristically as described in McDonnell et al. [26].

With these additional terms they formulated the following expanded model, which is the mathematically explicit version of equations (1) and (2):

$$\text{Minimize } C(x) = \sum_i c_i x_i + \sum_k \left(\frac{w_k SPF_k}{L_k}\right) u_k + w_b\left(\sum_i b_i x_i - 2\sum_{i=1}^{m-1}\sum_{j=i+1}^{m} b_{ij}x_i x_j\right) \quad (7)$$

Subject to:
Define the amount of shortfall in conservation element k associated with sites selected for portfolio

$$\sum_i a_{ik}\, x_i + u_k \ \geq\ L_k \text{ for each element } k \in K \quad (8)$$

Enforce integer restrictions on site decision variables

$$x_i = 0, 1 \text{ for each site } i \in I \quad (9)$$

Enforce nonnegativity on Shortfall variables

$$u_k \geq\ 0 \text{ for each element } k \in K \quad (10)$$

This second formulation contains the penalty terms described by Andelman et al. [1] in the SITES manual. That is, this model solves for the optimal portfolio set of sites that together minimize the total cost function described above in equations (1) and (2).

Although both integer nonlinear programming models convey exactly what is modeled in the SITES program, the formulations are not amenable to direct, optimal solution by commercially available software (except for relatively small problem instances), because they are nonlinear.

3. Description of the two SITES heuristics

There are two possible approaches to dealing with the difficulties of solving the above integer quadratic programming problem: (1) rely on heuristics, or (2) attempt a reformulation, to create a similar, linear problem that would be solvable. Taking the first route (i.e., rely on a heuristic approach) is pragmatic where the second approach proves unsatisfactory. To rely solely on the development of a heuristic approach, however, means that it may be impossible to truly assess the quality of the solutions generated.

Where the second approach is feasible, Rodrigues and Gaston [3] effectively show that it is often advantageous. Moreover, the skill and effort required to program an efficient heuristic from scratch is significantly greater than that required to format a problem for solving by an off-the-shelf optimization code.

The SITES program provides two solvers designed to select a portfolio [1]. Details of these approaches can also be found in [1, 13, 25, 26]. The first solver is a Greedy heuristic. Starting with no sites in the portfolio, Greedy selects the site that yields the lowest value of the total cost. For the second site Greedy picks the site that reduces the total cost the most and adds it to the portfolio. At each step, Greedy adds one more site to the portfolio [1]. The heuristic stops when no site that can be added to the portfolio would lower total cost. Thus, Greedy may stop short of meeting all conservation targets because the reduction of penalty costs by selecting additional sites may be overwhelmed by additional site costs or weighted boundary length. At this point, according to the objective function, it is not "cost effective" to add any more planning units to the portfolio, even though it is possible that not all goals have been satisfied for all elements [1].

It should be understood that the greedy heuristic has been of interest in the Operations Research and Computer Science literature because it is relatively easy to prove worst-case bounds for complex problems. It is rarely used in practice because other techniques have proved to be considerably better. The Greedy heuristic in the SITES program should be used with considerable caution, especially if the Simulated Annealing (SA) process described below is not used. Church and Revelle [27] described how the greedy process can perform poorly in location and siting problems. Essentially, as sites are added to the portfolio, newly added sites tend to marginalize sites that are already members of the portfolio. Without the ability to remove sites from the portfolio as it is being constructed, Greedy suffers in performance because some sites added early on may not, in the end, be needed, or be justified in terms of net cost minus target penalties. This means that Greedy tends to construct solutions that are "bloated," with more sites than are necessary.

One important note is that SITES Greedy is not deterministic. Most greedy heuristics have defined tie-breaking rules, so that multiple runs will always produce the same solution. SITES Greedy may produce different solutions (with different objective values) in different runs, because any ties during the solution process are broken by random selection [26].

We conducted a small test of SITES Greedy using the five zero-perimeter Santa Barbara datasets (described below). The heuristic produced different solutions every time for each of 50 restarts on all five problems. In contrast, when using the Small Sierra dataset (SPF of 10, also described below), SITES Greedy produced only one solution for each of the five problems with 50 restarts each. Whatever the specifics of SITES Greedy, it appears more likely to be consistent on sparser datasets because of the lower likelihood of ties. In all cases, solution quality was inferior to solutions generated by the second solver included with SITES.

The second solver that is provided with the SITES program is based upon a solution technique called Simulated Annealing (SA). SA is based upon a statistical analogy between solution quality and energy states of particles in the process of tempering glass and metals by systematic heating and cooling [28]. The SA procedure in SITES starts with a random set of sites. At each iteration, the procedure identifies a single site at random, and then examines the possibility of either adding that site to the portfolio, or, if currently selected, of discarding it from the portfolio. If the change (dropping a site or adding a site) produces an improved solution, the change is automatically accepted. If it does not produce an improvement, the change *may still be* accepted (based on comparing a random number to a probability distribution). The probability of accepting a change that degrades solution quality is taken from the Boltzmann distribution (which describes the number of particles that will have a higher energy state than a specified state, at a given temperature). Statistically, when the simulated temperature is high, and the proposed change is not substantially worse, the probability of accepting a change is relatively high. But, as the simulated temperature is lowered (systematically as the process runs), the probability of accepting changes that worsen a portfolio (by even a small amount) decreases. This process has the capability of converging to a local optimum and backing out of the local optimum (making a portfolio worse) and then finding even better local optima. SA has been applied to other site selection problems with varying degrees of success [29,30]. As with any SA heuristic, the success in application is somewhat dependent on the problem being solved and the parameter settings used (e.g., cooling rate and the number of iterations). The only parameters in SITES SA that can be set by the user are the number of restarts and the number of iterations per restart.

Most applications of SA require multiple restarts of the process, where only the best solution or solutions found among the different restarts is/are considered. SITES includes an option for examining the sum of results from multiple restarts, showing how many times each site was selected. Andelman et al. [1] suggest that this analysis provides a measure of the "robustness" of a solution, as though the number of times a site was chosen by SA was an indicator of its importance to an optimal solution. This conclusion is not supportable. Fischer [31] described SITES problems where the median solution quality was more than 50% worse than the best solution. With most solutions being very inferior, any site that was selected a majority of the time was necessarily a part of many very inferior solutions. In each case examined, Fischer [31] also found numerous "popular" sites (selected in more than 50% of the solutions) that were not part of an optimal solution, and numerous "unpopular" sites (selected in fewer than 20% of the solutions) that were. The "summed solution" approach described in SITES is a haphazard approach to modeling robustness (a field reviewed by Owen and Daskin [32]) that appears to be uninformative at best.

McDonnell et al. [26] present a comparison of the Greedy approach and the SA approach in solving the SITES model applied to a vegetation dataset of Northern Territory, Australia. Their comparison demonstrates that at times the Greedy approach outperforms SA, although they conclude that the SA process is probably better suited to solving the SITES problem. As with all heuristics, there is no guarantee that the Greedy or SA processes will find optimal solutions. It should also be understood that the quality of the solutions cannot be ascertained without actual optimal solutions with which to make a comparison. That is, the results of a heuristic, used by itself, should be interpreted with caution.

To evaluate the SITES solution process further, an assessment is needed in terms of how close to optimal either technique solves the SITES problem. In the next section we present a reformulated model for SITES and then describe how this model can be solved in practice. With this model we will provide an assessment of the SITES heuristics later in this paper.

4. A reformulation of the SITES model

The major obstacle to using an optimal solver for the formulation of the SITES model (as described in McDonnell et al. [26]) is the set of quadratic terms that are used to define the boundary length of the sites selected for the portfolio:

$$\sum_{i=1}^{m-1} \sum_{j=i+1}^{m} b_{ij} x_i x_j \qquad (11)$$

With the exception of these terms, the model is an integer-linear programming problem. These terms make the previous two formulations difficult or impossible to solve optimally. It is, however, possible to reformulate the model and circumvent the need for the quadratic terms. We can do this by introducing the following new variable:

$$z_{ij} = \begin{cases} 1, & \text{if sites } i \text{ and } j \text{ have been selected for the portfolio} \\ 0, & \text{otherwise} \end{cases}$$

We need to define such a variable for each pair of sites that share an edge. Therefore, consider:

$$Z = \text{set of site pairs}(i,j) \text{ which share boundaries, where } i < j$$

Each pair of adjacent sites will be addressed where the smaller of the two site indices is given first in the site pair. This distinction allows us to represent each possible edge with one decision variable. Using the two discrete decision variables x_i and z_{ij}, it is now possible to construct a model that represents the SITES problem and eliminates the quadratic terms:

$$\text{Minimize } \; Obj = w_c \sum_i c_i x_i + \sum_{k \in K} \left(\frac{w_k \, spf_k}{L_k} \right) u_k + w_b \left(\sum_i b_i x_i - 2 \sum_{(i,j) \in Z} b_{ij} z_{ij} \right) \tag{12}$$

Subject to:
Define amount of shortfall for target involving conservation element k

$$\sum_i a_{ik} x_i + u_k \geq L_k \quad \text{for each element } k \in K \tag{13}$$

Ensure z_{ij} is only allowed to be 1 if adjacent sites i and j are both selected

$$(a)\; x_i - z_{ij} \geq 0 \quad \text{for each shared edge where } (i,j) \in Z$$
$$(b)\; x_j - z_{ij} \geq 0 \quad \text{for each shared edge where } (i,j) \in Z \tag{14}$$

Enforce integer requirements on site decision variables

$$x_i = 0, 1 \quad \text{for each site } i \in I \tag{15}$$

Enforce nonnegativity on Shortfall variables

$$u_k \geq 0 \quad \text{for each element } k \in K \tag{16}$$

The first term in the objective function sums the costs of all of the selected sites. The second term represents the weighted penalty costs of incurring any shortfall in meeting conservation targets and the third term calculates the total boundary of the sites selected, accounting for any shared edges. The first type of constraint (13), the same as (8) used in the previous model, defines any shortfall in conservation targets that may exist in the set of sites chosen. The second type of constraint (14) is used to define the values of the z_{ij} variables. Each z_{ij} variable must equal zero unless both sites i and j are selected. If both i and j are selected, z_{ij} is allowed to have any value between zero and one. Since the objective function encourages z_{ij} to be as large as possible (to reduce total boundary length), z_{ij} is effectively 0 or 1. Defining z_{ij} this way accurately captures the boundary length of the selected sites, without the computational burden of defining each z_{ij} as an integer variable. This type of model construct was recently introduced for a related reserve design model by Fischer and Church [17].

The above model represents a reformulation of the SITES problem. This formulation is an integer-linear programming problem. Since commercially available software exist for solving this type of programming problem, it makes sense to test such software first, instead of investing in development of a special solver for this specific type of problem. As the heuristic solvers now exist, it makes sense to test their performance, and in the next sections, we present information for three different datasets and compare results from SITES heuristics with results from a commercial code for solving the reformulated SITES model.

5. Comparing SITES solvers with an IP/LP approach

In the following sections we provide a comparison of the SA heuristic of the SITES model with solutions generated for the reformulated SITES model. With this new model formulation, it is straightforward to solve the SITES model using the techniques of linear and integer programming. We used an off-the-shelf optimization package called CPLEX (ILOG Corporation), which is a widely used, general-purpose, linear-integer programming solver. We do not give further details of SITES Greedy since, overall, we found that the SA heuristic performed considerably better than Greedy on the problems that we analyzed.

As with any SA heuristic, the SITES solver has a number of parameters (other than weights for the different objectives) that affect its performance [30]. In the interests of operational simplicity, the designers of SITES have hard-coded a number of parameters, such as initial temperature, cooling rate, etc. Remaining variables that must be set to solve a SITES problem are the number of iterations, the number of restarts, and the species penalty factor.

The SITES manual recommends using at least 10^6 iterations. In the interest of speed, we tried a few exploratory runs using 10^5 iterations and found solutions significantly inferior. All of the results presented here used 10^6 iterations. For each problem that we solved, we tested using 50 restarts and 500 restarts of the SA heuristic. We also tested the sensitivity of model results to varying levels of the species penalty factor.

As mentioned earlier, the value of one of the weights applied to protection shortfalls, w_k (7), is determined heuristically in SITES during each run [26]. The method of calculation of this term is not adequately defined to allow independent replication of its values. As a result, we were unable to explain or replicate the penalty values derived by SITES for solutions that did not meet all protection targets. We conducted a number of tests with CPLEX and found that unconstrained problems where shortfalls were allowed, with proportional penalties, were solved quite a bit faster than problems where shortfalls were not allowed. Only solutions that met all protection targets are compared below.

SITES is compiled for a Windows/Intel platform. Our testing was conducted on a PC with an 800-MHz AMD Athlon processor and 256 MB of RAM, running Windows 98SE. For each problem instance, we ran the SA heuristic twice, first with 50 restarts, then with 500 restarts. We wrote a short code to then reformat the SITES input files into an MPS file (a standard text file format for Linear and Integer Programming codes), to allow side-by-side testing. Each MPS file was loaded into CPLEX version 6.6 on a Sun Ultra SPARC 10 Station with a 467-MHz processor. To confirm that SITES was not running on a slower machine, we benchmarked the computers with a billion iterations of the main elements of the SA solver (generate a random number, compare to an array, and store results to an array). The SITES machine performed the benchmark 1.7 times faster than the CPLEX machine.

CPLEX utilizes a branch-and-bound process to solve mixed-integer problems like these [33]. At each step in the process, CPLEX looks for an improved solution, and establishes a lower bound below which it has proven that no feasible solutions exist. As the process continues, the lower bound rises and (hopefully) the objective value of the best-known solution decreases. We used three stopping rules. The first was a gap of 0.01% (0.0001). That is, when the lower bound is within 0.01% of the best known solution, we declare the current solution optimal (while a better solution is possible, it could be no more than 0.01% better than the current solution). We also terminated the CPLEX run after branching to 500,000 nodes or if the branch-and-bound tree exceeded 500 Mb.

Data from three reserve selection problems were used, each providing a different test environment. The datasets and the results from each are described in the next two sections.

6. Sierra datasets

The first two datasets were prepared as part of the Sierra Nevada Ecosystem Project [34]. Both datasets are from primarily forested areas of the northern Sierra Nevada that roughly correspond to the Jepson Northern Sierra Ecoregion [35]. They consist of distribution and conservation information for different plant communities. Distribution is given as an areal extent of the vegetation formation within each of the several hundred watersheds making up the study area.

The first dataset is Okin's [36] *north.M.35*. This dataset uses 663 watersheds as the sites. These sites have 1,834 edges that are shared between adjacent sites. Thirty-six plant community types are used as conservation elements. The minimum protection target for each element is 10% of the existing range. This dataset is hereafter referred to as the Small Sierra dataset.

The second dataset is distributed as sample data by Andelman et al. [1]. This dataset uses an expanded area of the northern Sierra Nevada, encompassing 776 watersheds. These sites have 2,148 edges that are shared between adjacent sites. It includes 55 plant community types as conservation elements, and requires 25% of their existing ranges as minimum protection targets. As a result, the problem requires a substantially greater number of sites to be selected as reserves, and serves as a significantly different test environment for the model. This dataset is hereafter referred to as the Large Sierra dataset. Maps of three solutions from this dataset are included as figure 1.

In both datasets, area is measured in hectares. Site costs are represented in part by a site suitability index defined by Davis et al. [34], where small numbers indicate planning units that are very suitable for conservation, and high numbers indicate areas that are unsuitable due to road density, fragmented ownership, degraded habitat, etc. Because

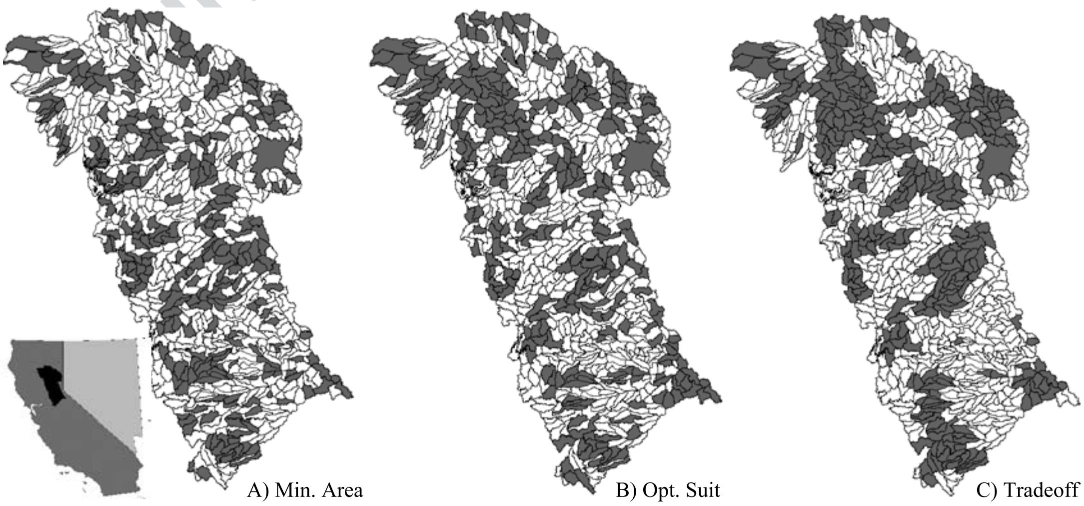

Figure 1. Maps of the three optimal reserve networks for the Large Sierra dataset. Map A shows the network requiring minimum land area. The network that optimizes land suitability scores is shown in map B. Map C shows an optimal solution trading off area and suitability with perimeter to encourage clustering of selected land. The Small Sierra dataset covers all but the northwest corner of this area.

SITES utilizes only a single cost term for each site, we summed weighted area and weighted suitability index values for each site to generate a single cost term before preparing each set of data files for SITES. Perimeter is measured in meters.

For the purposes of this study, we chose five sets of weights based on methods originally outlined by Cohon et al. [37]. Figure 2 gives an example of their method for efficiently estimating two-dimensional tradeoff curves using Suitability and Perimeter for the Large Sierra dataset. Solanki et al. [38] extended this method for systematically varying objective weights to estimate *n*-dimensional tradeoff surfaces, and we used their method for calculating the three-dimensional weights. Table 2 shows the weights used and provides a quick comparison between the three datasets used in this study.

Previous experience with SITES showed that solution quality can vary a great deal depending on the magnitude of SPF values compared to cost and perimeter values [31]. With that in mind, we solved each problem for the Small and Large Sierra datasets with three different SPF values.

The remainder of this section will compare the solvers in terms of solution speed and solution quality. For each problem instance, table 3 lists the lower bound derived by CPLEX, the solution time required by CPLEX (to reach one of its stopping rules), and solution quality for each solver. Solution quality is expressed as "gap" defined as the difference between the best known objective and the lower bound, expressed as a percentage of the bound. The first line of table 3 lists the result from the Small Sierra dataset minimum area problem. CPLEX solved the problem in 6 min or 0.10 h. The best solution found by CPLEX was 0.01% above the proven lower bound of 78308. The first row under SA shows results using an SPF of 10. The best solution found by SA in 50 restarts was 10.82% above the bound. After 500 restarts, the best solution found by SA was better, at 6.61% above the bound. Finally, 49.4% of the SA500 solutions met all protection targets.

For all five problems using the Small Sierra dataset, CPLEX solved to within 0.01% of optimality with a median solve time of 39 h (i.e., each final solution was within 0.01% of its respective final lower bound, and was thus declared optimal). The two problems with perimeter weights of zero was solved in less than 7 min, whereas those with perimeter considerations took much longer to solve optimally. After running CPLEX for 1 h, both multiobjective problems had solutions that eventually proved to be within 3% of optimality (at the time, the gap was only known to be less than 16%). The minimum perimeter solution proved difficult to solve, with the best solution after 1 h being 18% above optimal, and still 8% above optimal after 12 h.

For the large Sierra dataset, solution times were generally shorter, and CPLEX solved all but one problem to

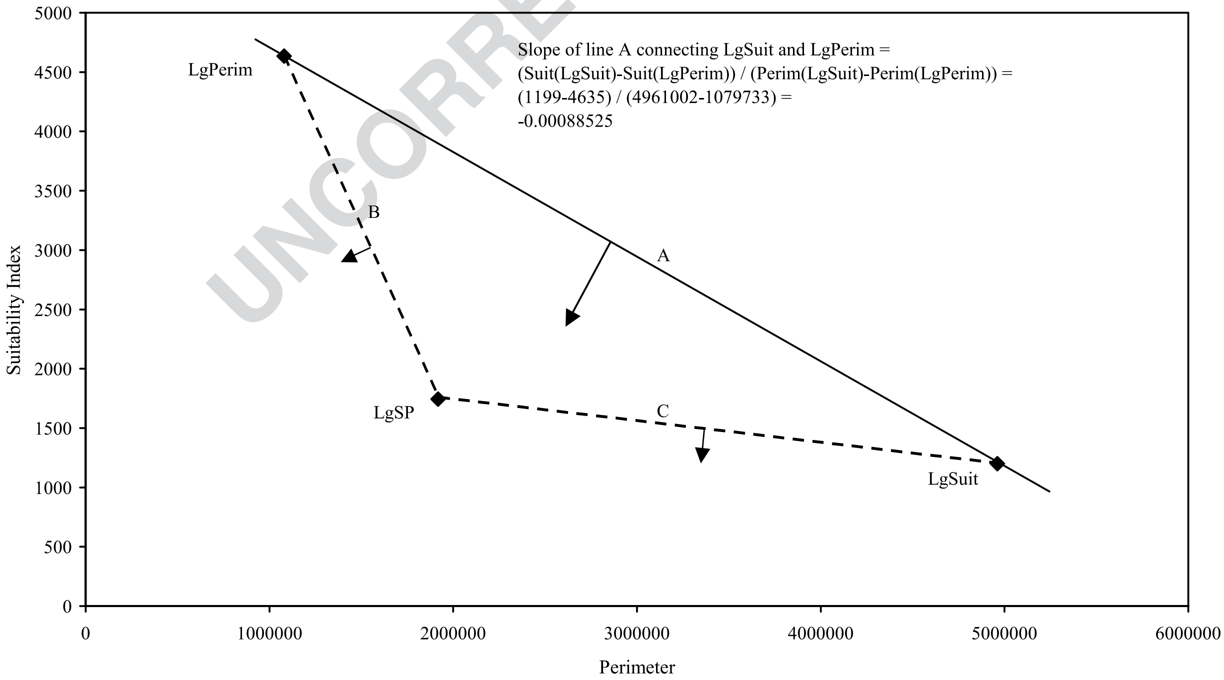

Figure 2. Estimation of a two-dimensional trade-off curve using the method of Cohon et al. [38] using the Large Sierra dataset. Steps are as follows. Optimize each single objective. Calculate the slope of line A connecting those two solutions in objective space. This is the estimated trade-off curve with two points. Apply the absolute value of the slope as the weight for the objective on the *x*-axis and a weight of 1 to the *y*-axis objective (the total weighted objectives for both solutions will be equal under the new weights). Weighting Suitability with 1, the weight for Perimeter would be 0.00088525. For scaling, we multiplied both weights by 1,000 to find solution LgSP. With three solutions, the trade-off curve is estimated as dashed lines B and C. To find additional solutions on the trade-off curve, the process is repeated for line B and C.

Table 2
Comparison of the three datasets, showing objective weights used for each problem.

Dataset		Problem name	Weights		
Name	Description		Area	Suitability	Perimeter
Small Sierra dataset		SmArea	1	0	0
	Sites = 663	SmSuit	0	1	0
	Spp. = 36	SmPerim	0	0	1
	Edges = 1834	SmSP	0	1,000	0.5052
		SmASP	1	733	0.4732
Large Sierra dataset		LgArea	1	0	0
	Sites = 776	LgSuit	0	1	0
	Spp. = 55	LgPerim	0	0	1
	Edges = 2148	LgSP	0	1,000	0.8853
		LgASP	1	2,004	0.4194
Santa Barbara dataset		SB25/P	1	0	0/0.01
	Sites = 1906	SB40/P	1	0	0/0.01
	Spp. = 57	SB50/P	1	0	0/0.01
	Edges = 5709	SB60/P	1	0	0/0.01
		SB75/P	1	0	0/0.01

The description of each dataset includes the number of sites, the number of conservation elements (species), and the number of shared edges. Five problems were solved for each Sierra dataset. The weights used for minimizing each of the three objectives (the sums of area, suitability index, and perimeter) are given in the right three columns. Zero weights indicate the objective was not considered in that problem. For the Santa Barbara dataset, we solved ten problems, five just minimizing area (area weight = 1, suitability and perimeter weights = 0), and then five more with a small additional weight on minimizing perimeter (0.01).

within 0.01% of optimality before stopping. The remaining problem was proven to be within 0.62% of optimal. Median solution time was slightly over 10 h. The Optimal Suitability and Minimum Perimeter problems solved in 3 and 1.7 h, respectively, while the multiobjective problems took longest. It is not immediately apparent why the larger dataset should have had shorter solve times. Solution speed of the branch-and-bound process (the process employed by CPLEX) is dependent on problem characteristics and just because a problem might be smaller or a subset of a larger problem does not guarantee that solution times will be less than that needed for the larger problem.

The quality of the best SITES solution for each of the Sierra problems ranged from optimal (determined using CPLEX) to 11% worse than optimal, depending on the problem and SPF value used. The average gap for the best solutions after 50 restarts was 4.5% for those problems where SITES produced feasible solutions (solutions that met all protection targets). For perspective, however, just as we examine only the best solution in reporting SA performance for a given problem, we must look at the worst performance within a set of problems to give some idea of how well SITES can be expected to perform on a similar set of problems. Thus, when faced with problems similar to those in the Sierra datasets, the best solutions generated by the SITES solver after 50 restarts may vary considerably from optimal.

One important question is whether it makes sense to run a great many iterations of the heuristic. For the Sierra datasets, running a problem for 500 restarts instead of 50 resulted in a mean decrease in the gap of 1.5 percentage points. That reflects improvements in 21 of the 30 problems tested, and no feasible solutions returned in six cases. In the remaining three problems, the gap after 500 restarts was greater than the gap after a separate run of 50 restarts, illustrating the important point that SA works by probability, and that increasing the number of restarts only increases the probability of finding a near-optimal solution. Even using a large number of restarts provides no guarantee of "getting lucky." In the minimum perimeter problem, using the Small Sierra dataset, SITES found the optimal solution once in 50 restarts. Subsequently, in a run of 500 restarts, SA found the optimal solution again, only once. However, in a subsequent run of 1,000 restarts, the best solution that SA found was 1.7% worse than optimal.

For both Sierra datasets, SITES performed very differently depending on the SPF values used. With the recommended SPF value of 1, SITES failed to return any feasible solutions (i.e., solutions meeting all protection targets) in six of the ten problems. In all cases, the percentage of feasible solutions increased with increases in the SPF value used. The setting of SPF values is important to the solution process in SITES, with lower values allowing the heuristic more flexibility to explore infeasible solutions on the way to finding a good solution. Higher SPF values limit the heuristic's flexibility, but ensure a greater number of feasible solutions from which to choose. Because the number of iterations is so important to SA, it is expected that the increased number of solutions from which to choose may be more important to solution quality than the reduced flexibility. We looked for a consistent pattern in solution quality as a function of SPF, but found none.

An obvious point of comparison between the SITES heuristic and an optimal solver such as CPLEX is solution speed. Results presented above indicate that CPLEX takes longer to solve these problems (optimally) than SITES

Table 3
Comparison of results from CPLEX and SITES Simulated Annealing (SA) heuristic for solutions actually meeting all element protection goals.

Problem name	CPLEX			SA50		SA500	
	Bound	Time	Gap (%)	SPF	Gap (%)	Gap (%)	%Feas
A	B	C	D	E	F	G	H
SmArea	78308	0.10	0.01	10	10.82	6.61	49.4
				5	7.80	7.60	19.6
				1	–	–	0.0
SmSuit	95.40	0.00	0.00	10	7.97	5.03	45.8
				5	8.18	2.31	24.0
				1	0.00	0.00	1.0
SmPerim	444255	94.79	0.01	10	0.01	0.01	82.0
				5	8.79	3.72	67.8
				1	1.67	4.24	4.2
SmSP	428477	98.87	0.01	10	4.07	2.83	67.4
				5	2.69	2.00	48.4
				1	–	0.01	1.0
SmASP	467085	38.69	0.01	10	7.57	5.65	62.4
				5	10.41	2.92	39.2
				1	–	–	0.0
LgArea	1040988	10.44	0.62	10	3.29	3.55	14.0
				5	3.81	2.92	2.8
				1	–	–	0.0
LgSuit	1199.20	3.02	0.01	10	7.18	5.51	11.6
				5	6.10	7.36	0.4
				1	–	–	0.0
LgPerim	1079734	1.69	0.01	10	0.56	0.25	96.0
				5	1.28	0.25	86.8
				1	1.05	0.05	53.4
LgSP	3442077	26.52	0.01	10	1.52	0.73	52.6
				5	1.44	0.83	26.8
				1	–	–	0.0
LgASP	5066718	12.31	0.01	10	3.58	2.88	34.6
				5	4.39	2.71	8.4
				1	–	–	0.0
SB25	118552	13.81	0.02	*	9.07	8.19	54.0
SB40	203165	2.40	0.01	*	5.39	5.22	40.0
SB50	261672	1.42	0.01	*	4.12	3.90	49.4
SB60	321340	0.29	0.01	*	3.12	3.16	40.8
SB75	412543	0.09	0.01	*	2.08	1.86	32.6
SB25P	149321	12.00	1.75	*	25.34	20.06	18.2
SB40P	240383	12.00	1.55	*	15.17	14.06	25.6
SB50P	301493	12.00	1.39	*	11.91	11.19	39.6
SB60P	363019	12.00	1.12	*	8.56	7.85	44.4
SB75P	456677	12.00	0.85	*	5.86	5.34	17.4

Problem name (A) is the same as in table 2. Column B is the lower bound below which CPLEX has proven that no solutions exist. Time (C) is the number of hours before CPLEX reached a stopping rule. Gap (D) is the difference between the best-known solution and the bound (B), expressed as a percentage of the bound. SPF (E) is a weight used in SITES SA. Columns F and G show the gap of the best known SITES solution after 50 and 500 restarts, respectively (expressed as a percentage of the bound in Column B). Column H is the percentage of 500 restarts that met all element protection goals. For a time comparison, the SA heuristic required 0.6–0.8 h to run 500 restarts on the Small Sierra dataset, 1.1–1.2 h for the Large Sierra dataset, and 1.8–2.0 h for the Santa Barbara dataset.

takes to find a solution of unknown quality. In many cases, even the longest of these run times would be acceptable given the scope of large planning efforts. However, in cases where rapidity of results is paramount (e.g., an interactive planning exercise), a key question is the quality of solutions that can be produced without taking much computer time. Table 4 shows the quality of CPLEX solutions after the first 2 min. The first column shows the gap at 2 min; this is the percentage difference between the best solution (after 2 min) and the bound known at that time (i.e., how good was the solution known to be at 2 min?). The second column shows the difference between that solution and the final bound proven by CPLEX at termination (i.e., how good did the 2-min solution turn out to be?). The third column lists the final gap CPLEX produced at termination. The fourth column lists the gap for the best SITES solution after running 50 restarts at each of the three SPF values (compared to the final bound proven by CPLEX). The SITES solutions represent 5–18 times more processing time on the faster SITES computer

Table 4
Solution quality comparison between CPLEX after 2 min, final CPLEX solution, and best SA solution from a run of 50 restarts.

Problem name	2 min Gap		Final CPLEX Gap	Best SA50
	Known	Actual		
A	B (%)	C (%)	D (%)	E (%)
SmArea	1.05	0.08	0.01	7.80
SmSuit	0.00	0.00	0.00	0.00
SmPerim	138.87	46.53	0.01	0.01
SmSP	35.63	10.46	0.01	2.69
SmASP	21.77	6.77	0.01	7.57
LgArea	1.23	1.18	0.62	3.29
LgSuit	1.20	0.45	0.01	6.10
LgPerim	16.76	4.65	0.00	0.56
LgSP	9.36	2.68	0.01	1.44
LgASP	3.22	1.20	0.01	3.58
SB25	0.21	0.20	0.02	9.07
SB40	0.07	0.06	0.01	5.39
SB50	0.06	0.05	0.01	4.12
SB60	0.10	0.10	0.01	3.12
SB75	0.03	0.03	0.01	2.08
SB25/P	13.17	12.36	1.75	25.34
SB40/P	12.75	12.23	1.55	15.17
SB50/P	13.91	13.49	1.39	11.91
SB60/P	8.28	7.98	1.12	8.56
SB75/P	14.02	13.83	0.85	5.86

Problem name (A) is the same as in table 2. Column B lists the difference between the best solution known after 2 min and the bound known at 2 min (expressed as a percentage of the 2-min bound). Column C shows the gap between the best 2-min solution and the final bound (expressed as a percentage of the final bound). Column D is the difference between the best CPLEX solution and the final bound, and replicates table 3 Column D. Column E shows the gap between the best SITES SA solution (of the three SPF levels used) after 50 restarts, and the final CPLEX bound. Note that for SmSP the 2-min solution was known to be within 36% of optimal. As CPLEX proceeded, the lower bound rose 25 percentage points, so this solution eventually proved to be within 10% of optimal. Meanwhile the best known solution improved by 10 percentage points. For the SB problems, most of the improvement after 2 min was due to finding improved solutions, with relatively small changes in the lower bound.

(or 8–31 times more computational effort). After only 2 min of CPLEX processing time, the known gaps were still reasonably large for many of the problems, but, importantly, the actual solutions were generally quite good.

For nine of the ten Sierra problems, CPLEX identified at least one solution in the first 2 min that eventually proved to be within 11% (median 1.9%) of optimal. [Only the minimum perimeter problem for the Small Sierra dataset had a worse solution after the first 2 min (47% above optimal), but, with CPLEX reporting a gap of 138%, there was no question that more solution effort was indicated.] These 2-min solutions compare favorably to the best solutions returned by SITES after significantly longer times (with 50 restarts for each SPF value), which had median solution quality of 3.0% above optimal.

7. Santa Barbara dataset

The third dataset was originally prepared for Santa Barbara County in connection with a land-use planning project [39]. County-administered land was divided into 1,906 sites, based largely on watershed boundaries. These sites have 5,709 edges that are shared between adjacent sites. Habitat ranges for 57 species of regulatory concern were mapped based on known ranges and wildlife– habitat relationship models. Each species was assigned a species penalty factor between zero and one based on regulatory status, degree of endemicity, and threats to local habitat.

In addition to having significantly more sites than either Sierra dataset (see table 2), the Santa Barbara dataset has elements that are more widely distributed, with each element occupying an average of 639 sites, compared to 135 for the Large Sierra dataset and 57 for the Small Sierra dataset. For this dataset, we explored a trade-off curve of how much total land would be required to protect increasingly large percentages of each species range. We established five different conservation targets (25, 40, 50, 60, and 75% of each species range), and attempted to find the minimum amount of land that might be required to meet these protection targets for all 57 species. For the current study, we used the existing species-specific SPF values; we opted against performing additional SPF sensitivity testing. After solving these five problems, we solved a second problem for each of the five protection levels where we included a small weight for minimizing perimeter (to increase clustering and compactness of the reserve system).

For the perimeter problems, we also decided in advance to limit the amount of computing time to 12 h per run. That is, we used the solver to find some solutions of known quality (whether they qualify as “good” depends on standards) in a reasonable amount of time. With the relatively large areas being selected, a certain amount of clustering was inevitable. Analyzing a tradeoff curve for perimeter weights was therefore deemed unnecessary; we simply chose an arbitrarily small, nonzero weight for perimeter to somewhat further encourage compactness in our test data.

Solution times for the zero-perimeter problems were generally shorter for the Santa Barbara dataset than for the smaller, but relatively sparser Sierra datasets (see table 3). In all but one case, CPLEX proved the optimality of the solution – that problem terminated on reaching the tree size limit, with a remaining gap of 0.02%. While some of the solution times may still appear long, most of that time was spent improving solutions that were already very close to optimal, or in proving their optimality.

For each of the first five Santa Barbara problems, CPLEX found at least two solutions, with a proven gap of less than 0.5%, within the first 8 s of solution time. The remaining hours often revealed a number of slightly better solutions, but had we used a stopping rule of 0.25% instead of 0.01%, only one problem would have run longer than 9 s. This illustrates the utility of having a lower bound, and the importance of looking at solution quality as well as total solution time.

For the first five Santa Barbara problems, CPLEX returned solutions within 0.2% of optimal (median 0.06%) in the first 2 min (table 4), comparing favorably to solutions ranging 2–9% (median 4.1%) for SITES SA after 50 restarts. The perimeter problems had CPLEX solutions ranging 8–14% above the final bound (median 12.4%) compared to SITES SA, which ranged 6–25% (median 11.9%). For the perimeter problems, CPLEX returned solutions within 1.75% of optimality after 12 h, with decreasing gaps as the targets increased. By contrast, gaps of the SITES solutions varied from 5% to 25%.

8. Conclusion

Optimal solvers offer the significant advantage of revealing the quality of the present solution. At each step in a branch-and-bound process, the solver can display the lower bound, below which it has proven no integer solutions exist, as well as displaying the value of the current solution, and the gap between the two. With this information, a planner can watch the rate of closure between lower bound and current solution to forecast the longest the solver is likely to run (this is a worst-case estimate, assuming no new, better solutions are found). The planner can also choose to abandon the analysis at any point with a clear understanding that no new solutions exist that can beat the current solution by more than the reported gap. By contrast, most heuristics provide little or no information on the quality of the solutions they generate. Due to the random seed used in SA, it is very difficult to answer the question of how many restarts are needed to offer a high probability of finding a near-optimal solution. As seen with the SmPerim problem, 50 restarts may be sufficient to find an optimal solution, while the next 1,000 restarts may fail to find as good a solution again. Based on our analysis, the user's best approach with SITES is to use the highest number of restarts possible, explore a range of SPF values, apply weights efficiently (see figure 2), and carefully screen out solutions based on objective values and success at meeting protection targets. The "summed solution" feature, erroneously reported to reflect a measure of robustness, is not informative.

Performance of SA heuristics has been discussed by numerous authors, as reviewed by Murray and Church [30]. The particular SA process used by SITES has a very narrow search neighborhood, and, echoing Underhill's [2] call for greater interaction between the mathematical and biological research communities, might be readily improved [e.g., by allowing the heuristic to randomly swap two (or more) sites at each iteration as well as randomly adding or dropping sites [30,36]]. SA processes in general do statistically converge on optimal solutions if the cooling rate is very slow and the number of restarts is very high. Statistically speaking, computer-based SA processes are more like "simulated quenching" since the computation required for true "annealing" is well beyond realistic design [30], and far greater than for other, superior solvers. Because of this limitation, SA heuristics may not find optimal solutions with any regularity for complex problems like SITES, in any reasonable amount of computer time. In essence, a heuristic such as SITES SA provides predictable solution times, but inconsistent and unpredictable solution quality, whereas an optimal solver such as CPLEX provides solution times that can only be predicted once started [17], but precise information about solution quality.

Rodrigues and Gaston [3] offer compelling evidence that the hardware and software limitations that in the past have prevented optimal solvers from being used on realistically large reserve selection problems have now been largely overcome. Our results support that argument, with CPLEX outperforming the SITES SA heuristic by larger margins on larger problems. More research could profitably be directed at increasing the use of optimal solvers for existing and future reserve models.

Andelman et al. [1] claim as an advantage the multiple solutions returned by SA. Rather than relying on chance to produce a diversity of solutions of variable quality, another area ripe for further research is to incorporate into reserve selection models the existing literature in "modeling to generate alternatives" (e.g., [40]). Having used existing models to develop trade-off curves for a given application, planners might then use related reserve selection models to generate a number of optimally different alternate solutions that all lie on or near the optimal trade-off curve. These deliberately different alternatives are likely to offer radically different performance toward the sorts of unmodeled objectives that are often important in reserve selection problems. This is a systematic approach toward addressing those concerns raised by Pressey et al. [18] that have not already been answered by Rodrigues and Gaston [3].

9. Data policy

The Large Sierra datasets are available with the SITES software distribution at http://www.biogeog.ucsb.edu/projects/tnc/toolbox.html. The Small Sierra dataset is available at http://www.geog.ucsb.edu/~fischer. Use of the small dataset should reference Davis et al. [34]. The Santa Barbara dataset may be available in the future, depending on county Policy (contact fischer@geog.ucsb.edu).

Acknowledgements

During this work, Douglas Fischer was supported in part by grants from the U.S. Forest Service and the U.S. Geological Survey. We would like to acknowledge the support of the Western Ecological Research Center of the U.S. Geological Survey and thank Judd Howell and Deborah Maxwell there. The U.S. Fish and Wildlife Service has also provided support. We would also like to thank Frank Davis, David Stoms, and Chris Pyke for providing data and comments. Finally, thanks are due to both Pete Stine and Klaus Barber of the U.S. Forest Service for their help and support. The comments of Justin Williams and three anonymous reviewers were very helpful in improving this manuscript.

References

[1] S.J. Andelman, I. Ball, F.W. Davis and D.M. Stoms, SITES V 1.0: an analytical toolbox for designing ecoregional conservation portfolios, Unpublished manual prepared for the nature conservancy, 1999, 1–43. (available at http://www.biogeog.ucsb.edu/projects/tnc/toolbox.html).

[2] L.G. Underhill, Optimal and suboptimal reserve selection algorithms, Biol. Conserv. 70 (1994) 85–87.

[3] A.S.L. Rodrigues and K.J. Gaston, Optimization in reserve selection procedures – why not?, Biol. Conserv. 107 (2002) 123 – 129.

[4] B. Csuti, S. Polasky, P. Williams, R. Pressey, J. Camm, M. Kershaw, A. Kiester, B. Downs, R. Hamilton, M. Huso and K. Sahr, A comparison of reserve selection algorithms using data on terrestrial vertebrates in Oregon, Biol. Conserv. 80 (1997) 83–97.

[5] R. Gerrard, R.L. Church, D.M. Stoms and F.W. Davis, Selecting conservation reserves using species-covering models: adapting the Arc/Info GIS, Trans. Geol. Inf. Sci. 2 (1997) 45–60.

[6] R. Church, D. Stoms, F. Davis and B.J. Okin, Planning management activities to protect biodiversity with a GIS and an integrated optimization model, in: *Proceedings, Third International Conference/Workshop on Integrating GIS and Environmental Modeling, Santa Fe, NM, January 21–26, 1996* (National Center for Geographic Information and Analysis, Santa Barbara, CA, 1996) http://www.ncgia.ucsb.edu/conf/SANTA_FE_CD-ROM/main.html.

[7] R.L. Pressey and A.O. Nicholls, Efficiency in conservation evaluation: scoring versus iterative approaches, Biol. Conserv. 50 (1989) 199–218.

[8] K.D. Cocks and I.A. Baird, Using mathematical programming to address the multiple reserve selection problem: an example from the Eyre Peninsula, South Australia, Biol. Conserv. 49 (1989) 113 – 130.

[9] J.G. Hof and L.A. Joyce, A mixed integer linear programming approach for spatially optimizing wildlife and timber in managed forest ecosystems, For. Sci. 39 (1993) 816–834.

[10] D. Faith, P. Walker, J. Ive and L. Belbin, Integrating conservation and forestry production: exploring trade-offs between biodiversity and production in regional land-use assessment, For. Ecol. Manage. 85 (1996) 251–260.

[11] M. Bevers, J. Hof, D.W. Uresk and G.L. Schenbeck, Spatial optimization of prairie dog colonies for black-footed ferret recovery, Oper. Res. 45 (1997) 495–507.

[12] C. Loehle, Optimizing wildlife habitat mitigation with a habitat defragmentation algorithm, For. Ecol. Manage. 120 (1999) 245–251.

[13] H. Possingham, I. Ball and S. Andelman, Mathematical methods for identifying representative reserve networks, in: *Quantitative Methods for Conservation Biology*, eds. S. Ferson and M. Burgman (Springer-Verlag, Berlin Heidelberg New York, 2000) pp. 291–305.

[14] R.G. Haight, C.S. ReVelle and S.A. Snyder, An integer optimization approach to a probabilistic reserve site selection problem, Oper. Res. 48 (2000) 697–708.

[15] R.L. Church, R. Gerrard, A. Hollander and D.M. Stoms, Understanding the tradeoffs between site quality and species presence in reserve site selection, For. Sci. 46 (2000) 157–167.

[16] D.J. Nalle, J.L. Arthur and J. Sessions, Designing compact and contiguous reserve networks with a hybrid heuristic algorithm, For. Sci. 48 (2002) 59–68.

[17] D.T. Fischer and R.L. Church, Clustering and compactness in reserve site selection: an extension of the biodiversity management area selection model, For. Sci. 49 (2003) 555–565.

[18] R.L. Pressey, H.P. Possingham and C.R. Margules, Optimality in reserve selection algorithms: when does it matter and how much?, Biol. Conserv. 76 (1996) 259–267.

[19] J.R. Prendergast, R.M. Quinn and J.H. Lawton, The gaps between theory and practice in selecting nature reserves, Conserv. Biol. 13 (1999) 484–492.

[20] R.L. Pressey and R.M. Cowling, Reserve selection algorithms and the real world, Conserv. Biol. 15 (2000) 275–277.

[21] C. Groves, L. Valutis, D. Vosick, B. Neely, K. Wheaton, J. Touval and B. Runnels, *Designing a Geography of Hope: A Practitioners' Handbook for Ecoregional Conservation Planning* (The Nature Conservancy, Arlington, VA, 2000).

[22] M.W. Beck and M. Odaya, Ecoregional planning in marine environments: identifying priority sites for conservation in the northern Gulf of Mexico, Aquat. Conserv. 11 (2001) 235–242.

[23] S.J. Andelman and M.R. Willig, Alternative configurations of conservation reserves for Paraguayan bats: considerations of spatial scale, J. Soc. Conserv. Biol. 16 (2002) 1352–1363.

[24] H. Leslie, M. Ruckelshaus, I.R. Ball, S. Andelman and H.P. Possingham, Using siting algorithms in the design of marine reserve networks, Ecol. Appl. 13 (2003) S185–S198 Suppl.

[25] I.R. Ball, Mathematical applications for conservation ecology: the dynamics of tree hollows and the design of nature reserves, Ph.D. thesis, University of Adelaide, Adelaide, Australia, 2000.

[26] M.D. McDonnell, H.P. Possingham, I.R. Ball and E.A. Cousins, Mathematical methods for spatially cohesive reserve design, Environ. Model. Assess. 7 (2002) 107–114.

[27] R.L. Church and C.S. Revelle, The maximal covering location problem, Pap. Reg. Sci. Assoc. 32 (1974) 101–118.

[28] N.A. Metropolis, M. Rosenbluth, A. Rosenbluth and E. Teller, Equation of state calculations by fast computing machines, J. Chem. Phys. 21 (1953) 1087–1092.

[29] B.L. Golden and C.S. Skiscim, Using simulated annealing to solve routing and location problems, Naval Res. Logist. Q. 33 (1986) 261–279.

[30] A. Murray and R. Church, Applying simulated annealing to location-planning models, J. Heuristics 2 (1996) 31–53.

[31] D.T. Fischer, Clustering and compactness in reserve site

selection: an extension of the Biodiversity Management Area Selection Model, MA thesis, Univ. of Calif. Santa Barbara, 2001, p. 48.

[32] S.H. Owen, M.S. Daskin, Strategic facility location: a review, Eur. J. Oper. Res. 111 (1998) 423–447.

[33] ILOG. ILOG CPLEX 6.5 User's Manual, Ilog Corp. Incline Village, NV, 1999.

[34] F.W. Davis, D.M. Stoms, R.L. Church, W.J. Okin and K.N. Johnson, Selecting biodiversity management areas, in: *Sierra Nevada Ecosystem Project: Final Report to Congress, vol. II, Assessments and Scientific Basis for Management Options* (Centers for Water and Wildlands Resources, University of California, Davis, 1996) pp. 1503–1527.

[35] Willis Linn Jepson, *The Jepson Manual: Higher Plants of California*, ed. James C. Hickman (University of California Press, Berkeley, 1993) p. 1400.

[36] W. Okin, The biodiversity management area selection model: constructing a solution approach, MA Thesis, Univ. of Calif. Santa Barbara, 1997, p. 67.

[37] J. Cohon, R.L. Church and D. Sheer, Generating multiobjective trade-offs: an algorithm for bicriterion problems, Water Resour. Res. 15 (1979) 1001–1010.

[38] R. Solanki, P.A. Appino and J.L. Cohon, Approximating the noninferior set in multiobjective linear programming problems. Eur. J. Oper. Res. 68 (1993) 356–373.

[39] Watershed Environmental, Rural resource protection project sensitive biological resources study, prepared for County of Santa Barbara Planning and Development Department, 2003.

[40] E.D. Brill, S.Y. Chang and L.D. Hopkins, Modelling to generate alternatives – the HSJ approach and an illustration using a problem in land-use planning, Manage. Sci. 28 (1982) 221–235.

AUTHOR QUERY

AUTHOR PLEASE ANSWER QUERY.

Q1. Please provide a legend for the asterisks found in Table 3.