

# On Heuristic Mapping of Decision Surfaces for Post-Evaluation Analysis\*

Major Bill Branley  
U.S. Army Artificial Intelligence Center  
DIRECTOR INFO SYS  
107 ARMY PENTAGON  
WASHINGTON DC 20310-0107  
<http://www.pentagon-ai.army.mil/~branley/>

Steven O. Kimbrough  
University of Pennsylvania  
3620 Locust Walk  
Philadelphia, PA 19104-6366  
kimbrough@wharton.upenn.edu  
<http://opim.wharton.upenn.edu/~sok/>

Russell Fradin  
Box 0522  
3910 Irving Street  
Philadelphia, PA 19104-6007  
fradin@wharton.upenn.edu  
<http://opim.wharton.upenn.edu/~fradin/>

Tate Shafer  
101 S. 39th Street Apt. D401  
Philadelphia, PA 19104-3159  
shafer38@wharton.upenn.edu  
<http://futures.wharton.upenn.edu/~shafer38/>

## Abstract

*The value for decision making of high-quality post-solution analysis of decision-supporting models can hardly be overestimated. Candle-lighting analysis (CLA) takes this notion very seriously and brings to the table a series of computational approaches for gaining insight into decision problems by performing heuristic mapping of decision surfaces for management science models. The principal aims of this paper are to present, illustrate, and demonstrate the value, and extensive range, of candle-lighting analysis (CLA) ideas in the context of a new decision support system, CLAP-NT. In addition to illustrating how heuristic decision surface maps may be used for post-evaluation analysis, we present preliminary results comparing the efficacy of a genetic algorithm and a random search for mapping decision surfaces.*

## 1 Introduction

Achieving any single objective typically requires consumption of scarce, or not fully sufficient, resources.

---

\*Send correspondence to Steven O. Kimbrough. Thanks to Jack Hershey for pointing us to the ASSIST papers. This work was supported, in part, by the U.S. Army AI Center. File: clap-nt-hicss97-final. 960930. From: clap-nt-may96-960609. We wish to thank two anonymous referees for a number of very helpful comments.

Further, goals and objectives rarely exist in isolation. Partial conflict among them, and mutual competition for limited resources, is the norm. Thus, the world presents us with nearly unrestricted opportunities for optimization in the presence of constrained resources.

Optimization, as a practical endeavor, is properly concerned with much more than merely a recommendation for the allocation of scarce resources. Any such recommendation must be based upon assumptions, and models, that are at best approximate and that nearly always must leave out important information. Optimization procedures optimize models, not the original problem. Thus, much of the decision maker's work, at least the thoughtful, deliberative part of it, occurs during the interpretation phase of the modeling and optimization effort. Interpreting and seeking to understand what a model has to say is variously called *post-evaluation analysis* or *post-solution analysis* or *post-optimality analysis*,<sup>1</sup> although, as we shall see, it also has a useful rôle in model validation.

There are essentially three kinds of techniques for supporting post-solution analysis:

1. Analytic. Taking partial derivatives of a model for purposes of sensitivity analysis is an example. This may work well for some problems, especially for unconstrained optimization of differ-

---

<sup>1</sup>Among other things, e.g., *sensitivity analysis*, which we prefer to think of as a particular technique for performing post-evaluation analysis.

entiable models, but it is too limited for general practical purposes.

2. Solver by-product. When, e.g., linear programs are solved using the simplex method, a great deal of information for post-evaluation analysis is produced as a by-product of solving the problem. This information includes dual variable values, reduced costs, and basis-maintaining ranges on variables and parameters.
3. Multiple-solution. There are several popular computational techniques for supporting post-solution analysis. These techniques all require repeated solution of the model in question, using perturbed input parameter values. The main multiple solution techniques are: what-if analysis (usually manually directed), goal-seeking, data tables (a feature supported in most spreadsheet programs), and grid searches. Although much used and quite valuable, these multiple solution techniques are limited in important ways. Manually-directed what-if questioning is slow, unsystematic, and at risk of distortion because of human biases and cognitive limitations. Complete grid searches may in principle produce a wealth of information for post-solution analysis, but they are often computationally impracticable.

With no essential loss of generality (for present purposes), we may think of the post-solution analysis task for an optimization problem as presenting the decision maker with a finite *evaluation* or *objective function hypersurface*, which we shall refer to as the *decision surface*. Each point on the decision surface represents the value of the objective function for some permitted setting of the variables for the problem.<sup>2,3</sup> Post-evaluation analysis may be thought of as the task of exploring the decision surface for the sake of gaining insight and coming to a decision.

Clearly, the decision maker will want to know which point or points on the decision surface are optimal.<sup>4</sup>

---

<sup>2</sup>We assume that ranges are given for the decision variables. A permitted setting is one in which each decision variable is set at a value within its range. This setting need not result in a feasible solution to the optimization problem.

<sup>3</sup>The variables used to generate the decision surface will normally include the decision variables for the optimization problem. Also, for purposes we explain in the sequel, the surface-generating variables may include model parameters, e.g., right-hand side constants, which are allowed to vary over a specified range.

<sup>4</sup>Here and in the sequel we assume without loss of generality that we have a maximization problem, so that the decision maker

Providing this information is what optimization algorithms are mainly about. Post-evaluation analysis is about asking and answering other important questions regarding the evaluation hypersurface.

This paper is about heuristic, multiple-solution mapping of decision surfaces for purposes of supporting post-evaluation analyses, a process we call *candle-lighting analysis* or CLA (cf., [7, 8, 9, 10, 11, 12]). In general, *heuristic* mapping of decision surfaces is needed because the computational cost of exhaustive mapping is prohibitive. In what follows we implicitly describe our CLA concepts by directly describing a series of examples and results obtained from an existing implementation, CLAP-NT (candle-lighting analysis program in NT).<sup>5,6</sup>

## 2 Sensitivity Analysis

Given a model, a set of data (input parameters) for the model, and a solution for, or evaluation of, the model, there are a number of general questions we need to ask before a decision is made and a course of action taken. These questions include the following:<sup>7</sup>

1. How sensitive (or robust) is the model to minor changes in its recommendations (particularly to minor changes in the values of its decision variables)?
2. How sensitive (or robust) is the model to minor changes in its assumptions (particularly to minor changes in the values of its input parameters)?
3. Are there attractive alternative decisions, and, if so, what are they and what are their sensitivity properties?

For each of these questions, CLAP-NT provides general, heuristic approaches for finding answers. These

---

is seeking the “highest” points on the decision surface.

<sup>5</sup>Space is greatly restricted in this paper. See [12] for additional information on CLAP-NT and other aspects of this work.

<sup>6</sup>One reviewer of the paper commented that the results we achieve involve more than heuristic mapping of the decision surfaces. Yes, but—and there is insufficient space to go into this point—all of the results we report are obtained by extraction of information from heuristically-mapped decision surfaces. The key point to note is that these surfaces may be generated by more than variations in the decision variables. We also examine ranges of input parameters, e.g., right-hand side constants, in generating our surfaces. In short, given the evaluation purposes at hand, we will often blur the distinction between decision variables and input parameters in the model.

<sup>7</sup>For more detailed development of these concepts, see [11].

approaches apply to linear and nonlinear models with or without integer-valued decision variables.

$x_i$	Solution Rank				
	1	2	3	4	5
$x_1$	1	1	1	1	1
$x_2$	1	1	1	1	1
$x_3$	0	0	0	0	0
$x_4$	0	0	0	0	0
$x_5$	1	1	1	1	1
$x_6$	0	0	0	0	0
$x_7$	1	1	1	0	1
$x_8$	0	0	0	0	0
$x_9$	1	1	1	1	1
$x_{10}$	1	1	1	1	1
$x_{11}$	0	0	0	0	0
$x_{12}$	1	1	1	1	1
$x_{13}$	1	1	1	1	1
$x_{14}$	1	1	1	1	1
$x_{15}$	0	0	0	0	1
$x_{16}$	1	1	1	1	0
$x_{17}$	1	0	1	1	1
$x_{18}$	0	0	0	0	0
$x_{19}$	0	0	0	0	0
$x_{20}$	0	0	1	1	1
$x_{21}$	1	1	0	1	0
$x_{22}$	0	0	0	0	0
$x_{23}$	0	0	0	1	1
$x_{24}$	1	1	1	0	1
$x_{25}$	0	0	0	1	0
$x_{26}$	0	0	0	0	0
$x_{27}$	1	1	1	1	0
$x_{28}$	1	1	1	1	1
$x_{29}$	1	1	1	1	1
$x_{30}$	1	1	1	1	1

Table 1: From BestNSaveSet: Top solutions found by CLAP-NT for test knapsack problem with  $m = 80$  and  $n = 30$ . Values for  $w_i$  and  $p_i$  were randomly drawn. Found values of  $z$ :  $z_1 = 274.93$ ,  $z_2 = 266.99$ ,  $z_3 = 266.20$ ,  $z_4 = 261.66$ ,  $z_5 = 261.63$ .

To illustrate, consider the knapsack problem having the form given in Expression 1:

$$\begin{aligned} \max z &= \sum_{i=1}^n p_i \cdot x_i \\ \text{where } \sum_{i=1}^n w_i \cdot x_i &\leq m \\ x_i &\in \{0, 1\} \end{aligned} \quad (1)$$

Knapsack problems offer easily-interpreted, simply-created integer programs that typically are solved by creation of an integer linear program whose relaxed LP version is solved with the simplex algorithm, and whose integer constraints are then enforced with the branch-and-bound method. Because the simplex algorithm is not the final step in solving knapsack problems this way, the extensive set of sensitivity information generated in its solution process is effectively lost.<sup>8</sup>

CLAP-NT presents a way of creating sensitivity analysis information as a residue of the solution mechanism. Using genetic search (in the current version of CLAP-NT), the user directs the system to save the best  $N$  solutions and corresponding fitness values *that it finds*, thereby producing the BestNSaveSet for the run. These data may then be explored by the user, both graphically and in tabular form with filters of various sorts, in order to yield sensitivity information.

In a test implementation of the knapsack problem, we randomly assigned weights,  $w_i$ , in the range  $[1, 10]$  and profits,  $p_i$ , in the range  $[1, 20]$  for  $i$  from 1 to 30, i.e.,  $n = 30$ . Since the 30 decision variables were binary, the problem had a basic search space of  $2^{30}$ .<sup>9</sup>

Using an implementation of the model in the CLAP-NT system, we conducted a search using a genetic algorithm with 5,000 generations and a population size of 50, for a maximum of  $250,000 = 5,000 \cdot 50$  sample points. Thus, CLAP-NT examined at most  $(250,000/2^{30}) \approx 0.023\%$  of the search space. In each of ten runs, CLAP-NT found the optimal solution. Moreover, it returned important sensitivity information. In what follows, we report on the run in which  $m = 80$ .

## 2.1 Shadow Prices on $m$

The maximum value of  $z$  in our model, Expression 1, is constrained by the given value of  $m$ . If  $m$  is changed, at what rate does  $\max z$  change? This information, called the *shadow price* on  $m$ , is often invaluable for decision making. Let us see how CLAP-NT may produce shadow price information, i.e., heuristically-obtained information for estimating shadow prices.

The data in the BestNSaveSet reveal a difference of 7.9427 in fitness between the best solution found by the CLAP-NT system (and optimum for the model) and the second-best (see Table 1). The best found

<sup>8</sup>But see [2].

<sup>9</sup>This is, of course, a small problem for current optimizers. Our focus is on using heuristics to find interesting feasible alternatives to the—or an—optimal solution of a model, and this is, we believe, something worth investigating even for comparatively small models.

(and optimal) solution differs from the second-best solution (found by CLAP-NT) only in the value of  $x_{17}$ , which is 1 in the best solution and 0 in the second-best. The weight required to carry the optimal mix of items whose total weight is less than or equal to 80 is 78.231. Should  $m$ , the weight constraint of the model, drop below 78.231 yet remain above 77.095, which is required to carry the second-best-found mix, then the (estimated) optimal solution will shift to that found in the second-best solution. Thus, the interpretation of this information is that the (estimated) shadow price on  $m$  is 0 up to the allowable decrease of  $1.769 = 80 - 78.231$ ; after that it has a shadow price of 7.9427 and a zero shadow price for the next  $1.136 = 78.231 - 77.095$  units of change.

All of this information is easily extractable, given the original objective function for the model (cf., Expression 1) and the data in the BestNSaveSet. We note that the shadow price information on  $m$  is available here only for *decreases* in  $m$ , i.e., only for *tightening*, rather than relaxing, the constraint (but see §4, below). However, the number of (estimated) shadow prices for tightening of  $m$  is here determined (in large part) by the size of the BestNSaveSet. The data may (and in our test case, do) probabilistically yield useful results for many levels of tightening on  $m$ .

## 2.2 Reduced Costs

The BestNSaveSet data (see Table 1) also yield (heuristic, estimated) insight into the reduced costs of several items in, or considered for, the knapsack. For example, the best and third-best vectors in the BestNSaveSet (ranks 1 and 3 in Table 1) have identical decision variable values, except that one includes  $x_{20}$ , while the other instead includes  $x_{21}$ . It turns out that  $x_{21}$  fills a larger chunk of the remaining space and increases the profit of the knapsack by 8.729 units. Thus, adding  $x_{20}$  to the knapsack creates a net opportunity cost—or *reduced cost*—of 8.729 and results in  $x_{21}$  being left behind.

We note that this reduced cost, or opportunity cost, information is also available for variables that are *included* in the optimal solution. For example,  $x_{27}$  is included in the optimal solution. What is the reduced cost of *excluding*  $x_{27}$ ? From Table 1 we see that CLAP-NT (via the BestNSaveSet) estimates that the cost is  $z_1 - z_5 = 13.30$ .

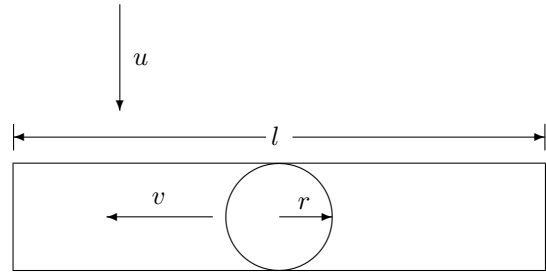


Figure 1: Barrier of length  $l$ , patrolled by a vessel with radius of detection,  $r$ . The Coast Guard vessel, located at the center of the circle, is traveling at a speed of  $v$  to the “west” and a target vessel is approaching with a speed of  $u$ , heading “south.” (The height of the barrier is  $2 \cdot r$ . The patrol area has area  $2 \cdot r \cdot l$ .)

## 3 Model Validation

CLAP-NT may be used to detect flaws, or suspect features, of models during the model validation phase of the modeling life-cycle. This aspect of CLA has received initial exploration for closed-form models [11] as well as for discrete-event simulation models [13]. More important than detecting absolute invalidity in a model, is the ability of CLA to ascertain ranges of validity and invalidity. Very many, otherwise quite valid and useful, models will give untoward results for certain joint ranges of parameter values. It is crucial for using such models that these ranges be detected and avoided—or at least recognized—in actual operations.

By way of illustration, consider the Basic Barrier Patrol model, which was developed for the U.S. Coast Guard in order to estimate the probability of detecting a target vessel attempting to cross a patrol barrier [9, 11].

The basic geometry of the patrol regime is given in Figure 1. Under reasonable assumptions,  $P(D|A)$ , the probability of detecting a target vessel given that the Coast Guard vessel is available and on patrol is:

$$P(D|A) = \frac{2 \cdot r}{l} \cdot \left(1 + \frac{v}{u}\right) \quad (2)$$

Given the intended interpretations of this model’s parameters— $r, l, v, u$ —it would not be surprising if an invalid input value, for example a negative magnitude for one of the parameters, produced an anomalous value for  $P(D|A)$ . What would be surprising, and certainly interesting, is if individually valid input parameter values produced anomalous values for  $P(D|A)$ . A general procedure for attempting to find such interest-

ing anomalies is as follows.

1. Define a measure of anomalyhood (or weirdness) for the model at hand.
2. Specify a (usually fairly broad) search space for all the parameters of interest.
3. Direct the CLA search process to find anomalous results in this search space.

John Miller, who independently developed some very similar ideas [13], has felicitously named this technique “model busting.” If anomalous results are found, i.e., if the technique breaks the model, then the analyst has work to do in order to understand and possibly revise the model or restrict its use. If no anomalies are found, after extensive search, then the analyst is given a reason to increase the stock of confidence in the model.

The CLAP-NT software supports model busting. The technique, and CLAP-NT’s support of it, is quite general and hardly limited to the Basic Barrier Patrol model. We use that model here only for illustration. Typical ranges for the model’s parameters in practice are:  $r \in [5, 20]$ ,  $l \in [75, 300]$ ,  $u \in [15, 35]$ , and  $v \in [10, 30]$ . We directed the genetic algorithm (GA) in CLAP-NT to search for high values of  $P(D|A)$  in the joint search space corresponding to these parameter ranges. In one run, we ran the GA for 100 generations with a population size of 50. We saved each individual in each generation, for a total of 5,000 solutions (although some 958 are duplicates). A large number of solutions was found—1,820—for which  $P(D|A) > 1$ . For example, CLAP-NT found that at (roughly)  $r = 19.9$ ,  $l = 75.9$ ,  $u = 15.3$ , and  $v = 30.0$ , the value of  $P(D|A)$  is 1.6. A very high probability indeed!

What has gone wrong? Further examination of the model’s equational form, Expression 2, reveals that these computational findings are not *mathematically* anomalous. There is nothing in the results to indicate that the implementation is incorrect. (Had it been the case, however, that the model was improperly implemented, then the model busting technique might well have helped diagnosed this condition.) Examination in CLAP-NT of the 5,000 saved solutions reveals quite reassuring behavior of the model: the expected directional changes occur (e.g., increasing  $v$  tends to increase  $P(D|A)$ ), response of  $P(D|A)$  is generally smooth, and so on. In short, there is nothing in the data, other than  $P(D|A) > 1$ , that indicates a problem. This might suggest the hypothesis that the model is roughly accurate for parameter values resulting in  $P(D|A) < 1$ , and that higher values indicate a degree of overkill, or margin of

safety. Because the Basic Barrier Patrol model is so simple algebraically, it is easy to confirm this hypothesis by adverting to the mathematics (Expression 2) and the geometry (Figure 1) of the model. In more complex cases, we can expect the benefits—of examining large data sets of model busting efforts—to be correspondingly greater.

## 4 Option Discovery

Models can support decision making, or the taking of a course of action, in two quite different but complementary ways. First, and standardly, a model may be used to find optimal or near-optimal courses of action under a given set of assumptions. This might be called the *decision-oriented* use of models. Here, key issues are model validity and sensitivity analysis. If we are to make a decision based on the recommendations of the model, we need to be confident that the model is valid. Also, we need to explore for alternative recommendations that on balance might be better (hence, sensitivity analysis).

Under the second way of using a model, a model may be used to search for courses of action that may be taken in order best to alter the given set of assumptions. This might be called the *option-discovery-oriented* use of models. Here, the key issues are the costs and benefits of altering assumptions, particularly parameter values and constraints, that normally are assumed fixed for the purpose of solving the model. Given a set of assumptions, and a resulting set of decision options, often the best thing to do is to act so as to alter the assumptions. “It is,” as the saying goes, “better to light one candle than to curse the darkness.”<sup>10</sup> As in the case of sensitivity analysis (§2, above), CLAP-NT provides general, heuristic approaches for option-discovery-oriented use of models. Note: Option-discovery-oriented use of models (our term) is standard practice as reported in the OR/MS literature, although this mode of using models is usually not distinguished terminologically from the technique usually employed to pursue it, sensitivity analysis. For further discussion of option-discovery use of models, see [9, 11].

We now take up an example—the ASSIST model and associated decision making—which has appeared in the general literature. We will rely extensively on the published record in order to describe the essentials of the decision problem [5, 15].

<sup>10</sup>Motto of the Christopher Society.

In response to the mounting human and economic toll from tobacco smoking, the National Cancer Institute (NCI) has, since 1982, funded a series of studies and experiments designed to find effective mechanisms for reducing smoking prevalence . . . These have culminated in the planning of the American Stop Smoking Intervention Study (ASSIST), the largest public health initiative ever undertaken by the National Institutes of Health. [5, page 1040]

ASSIST was given a budget of \$114 million and states were invited to submit funding proposals that addressed the goals of the ASSIST program.

The process of issuing a complex request for proposal (RFP) and reviewing the resulting proposals defined an interesting decision problem: how to make awards among many competing contract proposals while balancing a number of considerations critical to the long-term viability and effectiveness of ASSIST. [5, page 1041]

Thus, NCI's decision problem was framed as an optimization problem, and a 0–1 integer programming model was built (see [5, 15] for the—quite nontrivial—details). The model had 23 decision variables (1 fund, 0 do not fund), corresponding to the 23 proposals received (each from a distinct state). Objective function coefficients represented proposal ranks for technical merit, and “Other important criteria such as budget, diversity in smoking prevalence, and [*sic*] diversity in decline in smoking rate, and diversity in geographical area,” were represented in the model as constraints [5, page 1046].

The published reports [5, 15] emphasize the extensive sensitivity analyses that were performed with the model.

The overall purpose in formulating and solving a model for the project funding problem is to provide a short list of solutions. Those solutions should be within, or very close to, the estimated budget figure, and also among the best available solutions with respect to total rank function and other important criteria. Decision makers at NCI will then have the option to select from among the recommended solutions [note: decision-oriented use], or seek additional information from the model [note: option-discovery-oriented use]. [5, page 1045]

A number of sensitivity analyses were performed, mainly by grid search: the model was repeatedly reoptimized/resolved across different ranges of parameter values. Perhaps the most interesting result from this process concerned the budget constraint.

Our initial goal had been to stay within a \$114 million budget, and we found a very competitive solution at \$114. However, when we examined [the results of the grid search] we found a superior solution for less than 0.5% over budget (\$114.485). . . . This solution was quite attractive even when compared to other options with costs approaching \$115.5 million.

Our model produced many solution sets. Solutions with the \$114 million and \$115.5 million constraints were shown to the decision-making team. The superiority of the \$114.485 million solution, in terms of smokers reached and geographical balance, was obvious to the decision makers and thus contributed to this combination of states being awarded contracts. The selected solution set, however, was not an obvious one initially, and almost certainly would not have been discovered without the use of the model. [15, pages 119-120]

So we have here a testimonial to the value of post-solution analysis. How much of the achievement reported in the ASSIST papers could have been obtained via heuristic mapping of decision surfaces and the sort of software exemplified by CLAP-NT? Unfortunately, while the functional form of the ASSIST model is published [5, page 1051], confidentiality requirements prevent us from having access to the parameterization of the model. The papers do present a specific, scaled-down version of the model, using a notional parameterization, Expression 3. We have implemented this model in CLAP-NT and will now present our discussion in terms of it.<sup>11</sup> Note: In Expression 3 the right-hand-side value of the budget constraint is 37.

We implemented the model to incorporate a soft budget constraint ( $\leq 37$ ).<sup>12</sup> The penalty we chose for violating the budget constraint was simply equal to a weight,  $w_1$ , times the absolute amount of constraint

<sup>11</sup>The parameterization in Expression 3 is not significant and was meant by the authors to be taken only for illustrative purposes. We note, in particular, that the constraints are variously redundant.

<sup>12</sup>The preference function constraint ( $\geq 17$ ) turns out to be quite slack, so we will not discuss it further here.

$$\begin{array}{rcccccccccc}
\max z & = & 1.9x_1 & +1.8x_2 & +1.6x_3 & +1.5x_4 & +1.3x_5 & +1.2x_6 & +1.1x_7 & +1.0x_8 & \\
\text{where} & & & & & & & & & & \\
6.3x_1 & +4.5x_2 & +8.0x_3 & +5.2x_4 & +4.7x_5 & +7.0x_6 & +9.1x_7 & +7.7x_8 & & & \geq 17.0 \\
9.8x_1 & +7.4x_2 & +4.9x_3 & +3.9x_4 & +8.1x_5 & +6.1x_6 & +7.3x_7 & +5.6x_8 & & & \leq 37.0 \\
x_1 & & +x_3 & & & & & & & & \leq 1 \\
& & & x_4 & & & & +x_8 & & & \leq 1 \\
& x_2 & & & & & +x_7 & & & & \leq 1 \\
& & & & x_5 & +x_6 & & & & & \leq 1 \\
& x_2 & +x_3 & & & & & & & & \leq 1 \\
& & & x_4 & & & & +x_8 & & & \leq 1 \\
x_1 & & & & +x_5 & +x_6 & & & x_7 & & \leq 1 \\
& x_2 & & & & & & & & & \leq 1 \\
& & x_3 & +x_4 & & & +x_7 & & & & \leq 1 \\
& & & & x_5 & & & & & & \leq 1 \\
x_1 & & & & & +x_6 & & & & & \leq 1 \\
& & & & & & & x_i & \in & & \{0, 1\}
\end{array} \tag{3}$$

violation. Thus, for example, if a solution yielded an objective function value of 8.5 but had a budget of 37.7, then the net fitness would be  $8.5 - w_1 \cdot (37.7 - 37.0)$ .

Setting  $w_1 = 1$ , we ran the model with a genetic algorithm in CLAP-NT, using 50 generations and a population size of 50.<sup>13</sup> The optimal (hard constraint) solution to the model (found both by an integer programming solver and by CLAP-NT) is to fund all proposals except  $x_1$  and  $x_2$ , which happen to be the most-preferred projects according to the rank function (i.e.,  $x_1$  and  $x_2$  have the highest objective function coefficients). Decision makers will naturally ask, “What happens as the budget constraint is relaxed?” CLAP-NT (with  $w_1 = 1$ ) also finds a good solution that slightly violates the budget constraint: fund  $x_2$  through  $x_7$ . This yields an objective function value of 8.5 with a budget of 37.7. Thus, this solution produces an increase of  $8.5 - 7.7 = 0.8$  in the objective function value for a budget increase of  $37.7 - 35.9 = 1.8$ . So, just as in the original ASSIST study, there is a happy solution just beyond the budget constraint.

One naturally also wonders what the consequences would be if the budget constraint could be tightened. This information is not reported in the ASSIST papers, and would have required different grid searches than those described in the papers, but it is easily available from CLAP-NT (see §2). As it happens, the optimal solution to the original problem (Expression 3) has an

<sup>13</sup>We also performed a number of runs in which the GA also searched on  $w_1$ , e.g., in the range [1, 2]. The results corroborate the general findings we report here, but space limitations prevent us from giving details.

objective function value of 7.7 and a budget expenditure of 35.9. The next best found solution yields funding for  $x_1, x_2, x_4, x_5$ , and  $x_7$ , with an objective function value of 7.6 and a budget of 36.5. All of this (and other) information is easily and quickly obtained by using CLAP-NT to view and filter the BestNSaveSet for the model runs.

## 5 Reducing Infeasibility

An extreme case in which the need for an option-discovery-oriented use of models is appropriate is when an optimization model is infeasible due to conflicting requirements (rather than errors of implementation). Such events are far from rare. Standard optimization and model solution techniques are then pretty much at a loss as to what to do, unless, of course, the models are reformulated and solved as goal programs, a option not without difficulties of its own.<sup>14</sup> CLA offers a general, computational, and heuristic approach to finding changes in assumptions that are attractive (or minimally unattractive) and that can (probabilistically) result in feasible solutions.

We take our example for this section from a goal programming case in a popular textbook by Ragsdale (Davis McKeown example, [14, pages 254-263]). Briefly, Davis McKeown is considering expanding his hotel and convention center and has purchased a consulting study.

<sup>14</sup>We note, however, important work on certain aspects of this problem done by Harvey Greenberg [4] as well as others [3].

$$\begin{array}{rcll}
\min z & = & 18000x_1 & +33000x_2 & +45150x_3 \\
\text{where} & & & & \\
x_1 & & & & \geq 5 \\
& & x_2 & & \geq 10 \\
& & & & x_3 \geq 15 \\
400x_1 & +750x_2 & +1050x_3 & & \geq 25000 \\
1800x_1 & +3300x_2 & +45150x_3 & & \leq 1000000 \\
& & x_i & \in & \{0, 1, \dots\} \\
& & & & (4)
\end{array}$$

The results of this study indicated that Davis’s facilities should include at least 5 small (400 square foot) conference rooms, 10 medium (750 square foot) conference rooms, and 15 large (1,050 square foot) conference rooms. Additionally, the marketing research firm indicated that if the expansion consisted of a total of 25,000 square feet, Davis would have the largest convention center among his competitors—which would be desirable for advertising purposes. [14, pages 254]

Small, medium, and large conference rooms cost \$18,000, \$33,000, and \$45,150 respectively, and Davis has a budget of \$1,000,000. Given this, we can formulate the optimization problem as in Expression 4.

The model, as formulated, is infeasible. A standard response, and that undertaken in Ragsdale [14], is to reformulate the model as a goal program. This results in a new objective function which here is linear, but involves weights that are chosen on largely subjective grounds. Ragsdale’s suggested objective function for the goal programming formulation is:

$$\begin{aligned}
\min z &= \frac{w_1^-}{5}d_1^- + \frac{w_2^-}{10}d_2^- + \frac{w_3^-}{15}d_3^- + \\
&\frac{w_4^-}{25000}d_4^- + \frac{w_4^+}{25000}d_4^+ + \frac{w_5^+}{1000000}d_5^+
\end{aligned} \tag{5}$$

where the  $d_i^\pm$  are the new decision variables and represent the amounts under or over ( $\pm$ ) goal  $i$ . The  $w_i$  weights are constants, but would normally be subject to the kind of sensitivity analysis by grid search we saw in §4. Solving this goal program produces a solution that is about 10% over budget and 1% over the square feet goal of 25,000, but that otherwise meets the goals/constraints.

By now it should be clear how we would approach this problem with CLA concepts and CLAP-NT. We have a choice of either relaxing the right-hand-side goals in the original problem and performing search

over appropriate ranges, or of solving the goal programming reformulation, but with ranges on the  $d_i^\pm$ . In fact, we implemented the latter approach, and much as described in previous sections, we not only obtained the goal programming optimal solution, but a rich data set of sensitivity information.

## 6 On the Rôle of the GA

As we have seen, CLAP-NT uses a genetic algorithm (GA) to generate heuristic maps of the decision surfaces we investigate for post-evaluation analysis. Why use a GA? Are GAs fundamental to the CLA (heuristic mapping of decision surfaces) enterprise? On the latter question, the answer is clear and immediate: no. If a decision surface—or some relevant part of it—could be exhaustively searched, that would be better. The consequence of using a heuristic is that our results—shadow prices, etc., described above—can only be estimates. But for problems sufficiently large, such exhaustive enumeration is not practicable, and *some* heuristic will be required. Further, if heuristics are found for CLA that are better than genetic algorithms, then certainly one would want to use them. Then why use a GA?

GAs, for present purposes, have two important merits: (1) they are general; and (2) they work by finding good solutions and improving on them. Regarding (1), given a GA search engine, encoding a particular model for it is a fairly straightforward thing, regardless of whether the model is linear or not, and whether or not the decision variables are integer. (This is not to say that the GA approach will be generally successful.) Regarding (2), GAs may be thought of as working via an implicitly parallel hill-climbing effort, which typically produces many good solutions. Our approach has been to save the best of these solutions for examination during post-evaluation analysis. The assumption has been that for decision making the most interesting alternatives to the optimal solution are those that are also reasonably close to being optimal. For many purposes, we think this is a reasonable assumption, but to the extent that it is not, the GA approach would seem to be unpromising.

These two reasons, however, are only plausibility reasons. They are only reasons for exploring use of GAs for post-evaluation analysis. It is hard to see why this should *not* be done. But what is more important is (a) to define appropriate goodness, or success, criteria for CLA (heuristic mapping of decision surfaces), and

(b) to determine, either analytically or computationally, which heuristic approaches are most likely to best satisfy these criteria in a given situation.<sup>15</sup> We have begun such investigations, and here offer some preliminary thoughts and initial results.

Here are two criteria (really criteria schema) that answer to (a). Suppose we have two heuristics for generating decision surface maps,  $h_1$  and  $h_2$ . Then, for a given level of computational effort,  $h_1$  is better than  $h_2$  if

*Best N criterion:* The  $N$  best distinct solutions found by  $h_1$  have associated objective function values that are generally closer to the optimum than are the  $N$  best distinct solutions found by  $h_2$ .

*Best within P criterion:* There are more and better distinct solutions found by  $h_1$  than by  $h_2$  that are within  $P$  percent of the optimum objective function value.

Using the Best N criterion, we performed computational experiments to test the CLAP-NT GA against a random search for 12 randomly-generated knapsack problems. Our measure of computational effort was the number of function (or fitness) evaluations performed. Thus, for example, a GA with population size 100, run for 50 generations, requires 5,000 fitness evaluations. This was, we assumed, computationally equivalent to randomly generating 5000 solutions (which may or may not be feasible) and determining their values. (Our findings are not very sensitive to this assumption.) The randomly generated solutions used the same variable ranges that the GA used, so the spaces searched were identical.

In the experiments we are reporting here,  $N$  was 30. Of our 12 knapsack problems, 3 had 20 decision variables, 3 had 25, 3 had 30 and 3 had 35. Fixing the decision variable count, we randomly generated 3 problems for each of the 4 cases (numbers of decision variables). Finally, we had two settings for computational effort: 5,000 function evaluations and 10,000 evaluations. In all, then, we had 48 runs: 12 models  $\times$  2 heuristics  $\times$  2 levels of computational effort.

As should be expected, the GA generally performed better, at least in a gross sense. Table 2 summarizes the results. In all 12 cases, the GA averages better than the random search in the sense that the best 30 distinct solutions it finds have objective function values on average higher than their counterparts from the random search. In fact, the average maximum value from the random searches is just slightly higher than

the average minimum (i.e., 30<sup>th</sup> best) value found by the GA searches.

These computational results are hardly definitive. The performance of this GA versus this random search is hardly stunning, yet there does seem to be some advantage, provided we find the Best N criterion a relevant one. Clearly, much remains to be done if we are to find the most effective heuristic for mapping decision surfaces.

Model Optimum	GA		Random	
	Max	Min	Max	Min
127.33	126.09	112.86	110.39	78.33
182.70	179.60	168.04	175.51	158.12
126.20	122.43	114.00	122.30	104.10
163.38	157.08	139.78	145.39	121.19
174.77	167.29	154.06	155.24	134.93
213.73	202.58	185.13	193.67	172.93
175.31	162.71	149.52	138.58	106.14
313.81	300.74	288.72	276.23	247.93
244.14	226.87	211.39	221.22	198.49
342.40	325.83	311.08	293.34	267.03
310.94	283.83	266.92	272.13	247.31
312.78	281.43	265.51	273.79	254.10

Table 2: Summary statistics for the 30 best distinct results on runs of 12 random knapsack models, for the CLAP-NT GA (population size = 100) versus a random search in the same space. Max and Min values are averaged across the 30 best distinct solutions found in each of two runs, at 5,000 and 10,000 function evaluations.

## 7 Summary

There is more to the story of candle-lighting analysis,<sup>16</sup> but we may summarize the main points of this paper as follows:

- (1) Optimization problems, ubiquitously encountered, present to the decision maker objective function hypersurfaces—decision surfaces—that are typically rugged and complex, and that need to be explored in order to support high-quality decisions.
- (2) If these objective function hypersurfaces could be fully mapped and the resulting information made effectively available to the decision maker, then a rich, if not complete, body of information for post-solution analysis would be available to the decision maker.

<sup>15</sup>For very thoughtful, and useful, general surveys on assessment and measurement of contributions to decision making made by a system such as CLAP-NT see [1] and [6].

<sup>16</sup>See [11] and other CLA references for additional details.

- (3) It is generally not possible, for realistic problems, to produce anything approaching a complete mapping of an objective function hypersurface. Computational complexity forbids this.
- (4) Short of a full mapping of a decision surface, it may be hoped that a partial mapping, based upon intelligently-directed heuristics, can reliably produce much valuable information, and be generally superior to alternative methods for post-solution analysis (e.g., manually-directed what-if questioning, goal-seeking, grid search, and so forth).
- (5) Genetic algorithms are a promising form of heuristic for mapping decision surfaces for purposes of post-evaluation analysis. Other heuristics need to be investigated.
- (6) Much work—both conceptual and computational—remains to be done in order to understand adequately effective means of generating and exploring decision surfaces.

## References

- [1] Barr, R.S., B.L. Golden, J.P. Kelly, M.G.C. Resende, and W.R. Stewart, "Designing and Reporting on Computational Experiments with Heuristic Methods," *Journal of Heuristics*, **1**, no. 1, Fall 1995, pp. 9–32.
- [2] Geoffrion, A.M., and R. Nauss, "Parametric and Postoptimality Analysis in Integer Linear Programming," *Management Science*, **23**, no. 5 (January 1977), pp. 453–466.
- [3] Greenberg, Harvey J., "A Bibliography for the Development of an Intelligent Mathematical Programming System," University of Colorado at Denver, Center for Computational Mathematics, report number 59, July 1995.
- [4] Greenberg, Harvey J., "The ANALYZE Rulebase for Supporting LP Analysis," University of Colorado at Denver, Center for Computational Mathematics, report number 60, July 1995.
- [5] Hall, Nicholas G., John C. Hershey, Larry G. Kessler, and R. Craig Stotts, "A Model for Making Project Funding Decisions at the National Cancer Institute," *Operations Research*, **40**, no. 6 (November-December 1992), pp. 1040–1051.
- [6] Hooker, J.N., "Testing Heuristics: We Have It All Wrong," *Journal of Heuristics*, **1**, no. 1, Fall 1995, pp. 33–42.
- [7] Kimbrough, Steven O., Scott A. Moore, Clark W. Pritchett, and Cynthia A. Sherman, "On DSS Support for Candle Lighting Analysis," *Transactions of DSS '92*, June 8-10, 1992, pp. 118–135.
- [8] Kimbrough, Steven O. and Jim R. Oliver, "Candle Lighting Analysis: Concepts, Examples, and Implementation," in Veda C. Storey and Andrew B. Whinston, eds., *Proceedings of the Second Annual Workshop on Information Technologies and Systems*, Dallas, Texas, December 12-13, 1992, pp. 55–63. File: Candle lighting GA 5.22.93.
- [9] Kimbrough, Steven O., Jim R. Oliver, and Clark W. Pritchett, "On Post-Evaluation Analysis: Candle-Lighting and Surrogate Models," *Interfaces*, **23**, 7, May-June 1993, pp. 17–28.
- [10] Kimbrough, Steven O. and Jim R. Oliver, "On Automating Candle Lighting Analysis: Insight from Search with Genetic Algorithms and Approximate Models," in Jay F. Nunamaker, Jr. and Ralph H. Sprague, Jr., eds., *Proceedings of the Twenty-Sixth Annual Hawaii International Conference on System Sciences, Volume III: Information Systems: Decision Support and Knowledge-Based Systems*, IEEE Computer Society Press, Los Alamitos, CA, January 1994, pp. 536–544. File: Candle Lighting HICSS 7/20/93.
- [11] Kimbrough, Steven O. and Jim R. Oliver, *Candle-Lighting Analysis: Automating "What-if" Questioning with Genetic Search*, October 26, 1995. File: \*\*armyai-m1. Available in PostScript at [12].
- [12] Kimbrough, Steven O., "Home Page for Candle-Lighting Analysis," <http://opim.wharton.upenn.edu/~sok/cla/>.
- [13] Miller, John H., "Active Nonlinear Tests (ANTs) of Complex Simulation Models," CMU working paper, 5 February 1996. <http://zia.hss.cmu.edu/econ/misc/wp.html>.
- [14] Ragsdale, Cliff T., *Spreadsheet Modeling and Decision Analysis*, Course Technology, Inc., Cambridge, MA, 1995. ISBN: 1-56527-277-3.
- [15] Stotts, R. Craig, Larry G. Kessler, John C. Hershey, Nicholas G. Hall, and Jessie G. Gruman, "Awarding Contracts at the National Institutes of Health: a Sensitivity Analysis of the Critical Parameters," *Int. Trans. Opl. Res.*, **1**, no. 2 (1994), pp. 117–124.