A CONTINUOUS GRASP TO DETERMINE THE RELATIONSHIP BETWEEN DRUGS AND ADVERSE REACTIONS

M. J. HIRSCH, C. N. MENESES, P. M. PARDALOS, M. A. RAGLE, AND M. G. C. RESENDE

ABSTRACT. Adverse drug reactions (ADRs) are estimated to be one of the leading causes of death. Many national and international agencies have set up databases of ADR reports for the express purpose of determining the relationship between drugs and adverse reactions that they cause. We formulate the drug-reaction relationship problem as a continuous optimization problem and utilize C-GRASP, a new continuous global optimization heuristic, to approximately determine the relationship between drugs and adverse reactions. Our approach is compared against others in the literature and is shown to find better solutions.

1. INTRODUCTION

An adverse drug reaction (ADR) is the onset of an untoward medical condition caused by the administration of a therapeutic drug [6]. In the United States, five percent of hospital admissions, 28 percent of emergency room visits, and five percent of hospital deaths can be attributed to ADRs [20]. ADRs are often not recognized during the testing phase of a new drug, as these testing studies are generally small-scale and of short duration. Thus, most ADRs are detected as a result of post-marketing studies, i.e. when the drug is available and prescribed to the general public. The U.S. Food and Drug Administration (FDA), The World Health Organization (WHO), as well as other countries and organizations, have created databases of ADRs reported by practicing physicians and hospitals [2, 3, 18, 19].

The *adverse drug reaction problem* can be set up as follows. Consider n patients, m drugs, and c possible reaction classes. Each of the n patients has taken a known subset of the m drugs. In addition, each patient exhibits symptoms from a subset of the c possible reactions. The goal of the adverse drug reaction problem is to determine, for each drug-reaction pair $\{d, r\}$, a weight or *probability*, that drug d is the cause of the adverse reaction r. Many solutions based on optimization techniques have been proposed for the ADR problem [1, 6, 10, 11, 12, 13, 14, 15, 17, 20, 21]. The approach of Mammadov et al. [13, 14, 15] (herein referred to as the *Mammadov Algorithm*) sets up the ADR problem as a nonlinear optimization problem subject to bound constraints.

In this paper, we propose a modification to the Mammadov et al. formulation. We solve an instance of this continuous optimization problem using real data from the *Australian Adverse Drug Reaction Dataset* with C-GRASP (continuous GRASP), a new continuous global optimization heuristic [7, 8, 9]. To analyze the performance of our algorithm, we also examine the *average precision* metric [13, 14, 15], as well as propose two new metrics. The paper is organized as follows. We begin, in Section 2, by describing the Australian adverse drug reaction dataset. In Section 3, we introduce both the Mammadov algorithm as well as our modifications of the Mammadov formulation. C-GRASP is described in

Key words and phrases. Adverse drug reactions/events, global optimization, continuous optimization, heuristic, GRASP, C-GRASP, stochastic local search, nonlinear programming.

AT&T Labs Research Technical Report TD-6UPR92. October 18, 2006.

Section 4. We compare our approach with the approach of Mammadov et al. in Section 5. Finally, Section 6 provides concluding remarks and future research directions.

2. Australian ADR dataset

The aim of the Australian Adverse Drug Reaction Advisory Committee (ADRAC) is to detect signals from adverse drug reactions as early as possible [2]. The ADRAC dataset contains 137,297 records collected from 1971 to 2001. There exist 18 system organ reaction classes, one of those being the cardiovascular class. The cardiovascular reaction class is broken down into four subclasses. We limit our study to data that has at least one reaction from the cardiovascular class. Therefore, for each record of the dataset, there are five possible reactions to consider. Four are from the cardiovascular class and a fifth reaction encompasses all reactions that are not part of the cardiovascular class.

For the data we received[1], each record corresponds to a patient. Each record contains 26 fields. The first ten fields represent up to ten drugs that the patient was prescribed. Each drug is represented as a unique identification number. If a patient took less than ten drugs, then the remaining drug fields are set to 0. Fields 11 through 15 contain a binary vector representing which reactions were observed for the patient. Field 16 contains the year that this ADR was written and fields 17 through 26 contain a binary vector representing which of the drugs in the first ten fields are suspected of causing the observed reactions, in the opinion of the doctor writing the ADR. As an example, consider the following record:

$$\{1984, 4871, 5544, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1972, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0\}.$$

This adverse drug report was written in 1972. The patient was prescribed drugs 1984, 4871, and 5544. Reactions 1, 4, and 5 occurred. The doctor writing the report was of the opinion that drug 4871 caused the observed reactions.

3. ADR problem formulation

We begin this section by presenting the Mammadov algorithm. We describe the problem formulation as well as the solution technique. We then present our modification to this formulation.

We first introduce some essential notation. Let n be the number of patients in the dataset, m be the total number of drugs over all patients, and c be the number of possible reactions. Let X^a be the $n \times m$ binary matrix defined as

$$X^a_{i,j} = \begin{cases} 1 & \text{if drug } j \text{ was prescribed for patient } i \\ 0 & \text{otherwise,} \end{cases}$$

where drug j refers to the j-th drug in the dataset. The matrix X^a corresponds to the patient/drug information if we consider all prescribed drugs for each patient. In a similar way, we can define X^s to represent the patient/drug information if we only consider the suspect drugs for each patient, as

$$X^s_{i,j} = \begin{cases} 1 & \text{if drug } j \text{ is a suspect drug for patient } i \\ 0 & \text{otherwise.} \end{cases}$$

[1]Data was received from Musa Mammadov on April 6, 2006 in private email communication.

We assume that $X^s_{i,j} = 1 \Longrightarrow X^a_{i,j} = 1$. Let Y be the $n \times c$ binary matrix defined as

$$Y_{i,k} = \begin{cases} 1 & \text{if reaction } k \text{ was observed for patient } i \\ 0 & \text{otherwise} \end{cases}$$

that represents which reactions were observed for each patient. We are tasked with finding the $m \times c$ matrix W, where $W_{j,k}$ is the weight, or probability, that drug j causes reaction k.

The goal of the Mammadov algorithm is to minimize

$$F(W;\ p,\ X) = \frac{1}{n}\sum_{i=1}^{n}\Big[\|Y_i\|_1^{-p} \cdot \|\frac{\|Y_i\|_1}{\|X_iW\|_1}(X_iW) - Y_i\|_2^2\Big], \tag{1}$$

where $p \in \mathbb{Z}$ and $X \in \{X^a, X^s\}$, such that $0 \le W \le 1$. However, since the size of this optimization problem can be very large, the Mammadov algorithm does not solve this problem exactly. Rather, it solves an approximation to this problem which we next describe.

For drug j, define $X[j]$ to be the set of all patients that were only prescribed drug j, i.e.

$$X[j] = \{i\ \in\ \{1,\dots,n\} \mid X_{i,j} = 1 \wedge X_{i,j'} = 0,\ \forall\ j' \neq j\}.$$

In a similar way, define $X_{all}[j]$ to be the set of patients that were prescribed drug j. It is clear that $X[j] \subseteq X_{all}[j]$. The solution technique taken in the Mammadov algorithm is to compute

$$W_{j,k} = \begin{cases} W^*_{j,k} & \text{if } a|X[j]| + 100b\frac{|X[j]|}{|X_{all}[j]|} \ge \rho^* \\ W^{**}_{j,k} & \text{otherwise,} \end{cases}$$

where

$$W^*_{j,k} = \Big[\sum_{i \in X[j]} \|Y_i\|^{2-p}\Big]^{-1} \cdot \Big[\sum_{i \in X[j]} \|Y_i\|^{1-p} Y_{i,k}\Big], \tag{2}$$

$$W^{**}_{j,k} = \Big[\sum_{i \in X_{all}[j]} \|Y_i\|^{2-p}\Big]^{-1} \cdot \Big[\sum_{i \in X_{all}[j]} \|Y_i\|^{1-p} \frac{rem(Y_{i,k})}{|\Delta''[i]|}\Big],$$

$\Delta''[i]$ is the set of all drugs prescribed to patient i whose weights are not calculated using Equation (2), and $rem(Y_{i,k})$ can be seen as the remainder of the reaction available after accounting for the drugs j whose weights are computed using Equation (2). In all experiments to follow, a, b, and ρ^* were set to 1, 1, and 80, respectively.

3.1. **Modification to the ADR formulation.**

As seen above, the Mammadov formulation allows X to be either X^a or X^s. We propose to consider X as a convex combination of X^a and X^s, i.e. $X(\beta) = \beta X^a + (1-\beta)X^s$, where $\beta \in [0,1]$ is unknown. Thus, $X(\beta)$ is a convex combination of all drugs and suspected drugs for each patient. Looked at in another way, this definition of X can be seen as providing a discount measure to the non-suspected drugs, as, for each patient, the suspect drugs are found in both X^a_i and X^s_i. Our formulation,

$$\min_{W,\beta} F(W;\ p,\ X(\beta)) \ni 0 \le W, \beta \le 1, \tag{3}$$

is a minimization problem with unknowns W and β. Note that the objective function, defined in Equation (1), is a nonlinear function in both W and β.

4. Continuous GRASP

Feo and Resende [4, 5] describe the metaheuristic GRASP as a multi-start local search procedure, where each GRASP iteration consists of two phases, a construction phase and a local search phase. In the construction phase, interactions between greediness and randomization generate a diverse set of good-quality solutions. The local search phase attempts to improve the solutions found by construction. The best solution over all of the multi-start iterations is retained as the final solution.

Hirsch et al. [7] describe Continuous GRASP (C-GRASP), an adaptation of GRASP to handle continuous optimization problems. C-GRASP works by discretizing the domain into a uniform grid. Both the construction and local improvement phases move along points on the grid. As the algorithm progresses, the grid adaptively becomes more dense. C-GRASP resembles GRASP in that it is a multi-start stochastic search metaheuristic that uses a randomized greedy construction procedure to generate starting solutions for a local improvement algorithm. The main difference is that an iteration of C-GRASP does not consist of a single greedy randomized construction followed by local improvement, but rather a series of construction-local improvement cycles with the output of construction serving as the input of the local improvement, as in GRASP. Unlike GRASP, however, the output of the C-GRASP local improvement procedure serves as the input of the C-GRASP construction procedure.

Hirsch et al. [9] proposed modifications to the original C-GRASP algorithm, resulting in a significant decrease in the number of objective function evaluations required to converge to the global optimum. In the remainder of this section, we detail this version of C-GRASP, along with changes specified to handle the formulation (3).

4.1. **The heuristic.**

Pseudo-code for the C-GRASP heuristic is shown in Figure 1. The procedure takes as input the problem dimension n, lower and upper bound vectors ℓ and u, the objective function $f(\cdot)$, as well as the parameters `Maxi`, and ρ_{lo}. Parameter `Maxi` defines the number of iterations to perform in the C-GRASP algorithm while parameter ρ_{lo} defines the portion of the neighborhood of the current solution that is searched during the local improvement phase.

Line 1 of the pseudo-code initializes the objective function value f^* of the best solution found to infinity. While the C-GRASP implementations of [7, 9] are multi-start procedures, because of the size of the ADR problem (see Table 2), we do not implement C-GRASP as a true multi-start algorithm. Rather, we perform a single multi-start iteration. Line 2 initializes the solution x to a random point distributed uniformly over the box in $\mathbb{R}^n$ defined by ℓ and u. The parameter h, that controls the discretization density of the search space, is initialized to 0.1. The construction and local improvement phases are then called sequentially in lines 7 and 8, respectively, `Maxi` times. The solution returned from the local improvement procedure is compared against the current best solution in line 9. If the returned solution has a smaller objective value than the current best solution, then, in lines 10–11, the current best solution is updated with the returned solution. In line 13, if the variables $\texttt{Impr}_\texttt{C}$ and $\texttt{Impr}_\texttt{L}$ are still set to `false`, then the grid density is increased by halving h, in line 14. The variable $\texttt{Impr}_\texttt{C}$ is `false` upon return from the construction procedure if and only if no improvement is made in the construction phase. Section 4.3 shows that the $\texttt{Impr}_\texttt{L}$ variable is `false` on return from the local improvement procedure if and only if the input solution to local improvement is determined to be an *h-local minimum*. We increase the grid density at this stage because repeating the construction procedure with the same grid density will not improve the solution. This approach allows C-GRASP to start with

```
procedure C-GRASP($n, \ell, u, f(\cdot)$, Maxi, $\rho_{lo}$)
1    $f^* \leftarrow \infty$;
2    $x \leftarrow$ UnifRand($\ell, u$);
3    $h \leftarrow 0.1$;
4    for $i = 1, \ldots,$ Maxi do
5        $\texttt{Impr}_\texttt{C} \leftarrow$ false;
6        $\texttt{Impr}_\texttt{L} \leftarrow$ false;
7        $[x, \texttt{Impr}_\texttt{C}] \leftarrow$ ConstructGreedyRandomized($x, f(\cdot), n, h, \ell, u, \texttt{Impr}_\texttt{C}$);
8        $[x, \texttt{Impr}_\texttt{L}] \leftarrow$ LocalImprovement($x, f(\cdot), n, h, \ell, u, \rho_{lo}, \texttt{Impr}_\texttt{L}$);
9        if $f(x) < f^*$ then
10           $x^* \leftarrow x$;
11           $f^* \leftarrow f(x)$;
12       end if
13       if $\texttt{Impr}_\texttt{C}$ = false and $\texttt{Impr}_\texttt{L}$ = false then
14           $h \leftarrow h/2$;    /* make grid more dense */
15       end if
16   end for
17   return($x^*$);
end C-GRASP;
```

FIGURE 1. Pseudo-code for C-GRASP.

a coarse discretization and adaptively increase the density as needed, thereby intensifying the search in a more dense discretization when a good solution has been found. The best solution found, at the time the stopping criteria are satisfied, is returned.

4.2. **Construction procedure.** In this section, we describe in detail the construction procedure. The construction algorithm combines greediness and randomization to produce a diverse set of good-quality solutions from which to start the local improvement phase. The construction algorithm is shown in Figure 2. The input is a solution vector x. To start, line 1 of the algorithm allows MaxUnFixed (user-defined parameter) coordinates of x to change in the current construction call (i.e. these coordinates are unfixed). In turn, in line 10 of the pseudo-code, if ReUse is false, a line search is performed in each unfixed coordinate direction i of x with the other $n-1$ coordinates of x held at their current values. In lines 10 and 11 of the pseudo-code, the value z_i for the i-th coordinate that minimizes the objective function, together with the objective function value g_i, are saved. In line 11, $\check{x}^i$ denotes x with the i-th coordinate set to z_i.

After looping through all unfixed coordinates (lines 7–16), in lines 17–23 a restricted candidate list (RCL) is formed containing the unfixed coordinates i whose g_i values are less than or equal to $\underline{g} + \alpha \cdot (\bar{g} - \underline{g})$, where $\bar{g}$ and $\underline{g}$ are, respectively, the maximum and minimum g_i values over all unfixed coordinates of x, and $\alpha \in [0,1]$ is randomly determined in line 2. In line 24, a coordinate is chosen at random from the RCL, say coordinate $j \in$ RCL. Line 25 checks whether x_j and z_j are equal. If so, line 26 sets ReUse to the value true. Otherwise, in lines 28–30, ReUse is set to false, $\texttt{Impr}_\texttt{C}$ is set to true, and x_j is set to equal z_j. Finally, in line 30, we fix coordinate j of x, by removing j from the set UnFixed. Choosing a coordinate by selecting at random from the RCL ensures both greediness and randomness in the construction phase. The above procedure is continued until all of the

```
procedure ConstructGreedyRandomized($x, f(\cdot), n, h, \ell, u$, Impr$_C$)
1    UnFixed $\leftarrow$ RandSelect($\{1,2,\ldots,n\}$, MaxUnFixed);
2    $\alpha \leftarrow$ UnifRand(0,1);
3    ReUse $\leftarrow$ false;
4    while UnFixed $\neq \emptyset$ do
5        $\underline{g} \leftarrow +\infty$;
6        $\bar{g} \leftarrow -\infty$;
7        for $i = 1,\ldots,n$ do
8            if $i \in$ UnFixed then
9                if ReUse = false then
10                   $z_i \leftarrow$ LineSearch($x, h, i, n, f(\cdot), \ell, u$);
11                   $g_i \leftarrow f(\check{x}^i)$;
12               end if
13               if $\underline{g} > g_i$ then $\underline{g} \leftarrow g_i$;
14               if $\bar{g} < g_i$ then $\bar{g} \leftarrow g_i$;
15           end if
16       end for
17       RCL $\leftarrow \emptyset$;
18       Threshold $\leftarrow \underline{g} + \alpha \cdot (\bar{g} - \underline{g})$;
19       for $i = 1,\ldots,n$ do
20           if $i \in$ UnFixed and $g_i \leq$ Threshold then
21               RCL $\leftarrow$ RCL $\cup \{i\}$;
22           end if
23       end for
24       $j \leftarrow$ RandomlySelectElement(RCL);
25       if $x_j = z_j$ then
26           ReUse $\leftarrow$ true;
27       else
28           $x_j \leftarrow z_j$;
29           ReUse $\leftarrow$ false;
30           Impr$_C$ $\leftarrow$ true;
31       end if
32       UnFixed $\leftarrow$ UnFixed $\setminus \{j\}$;    /* Fix coordinate j. */
33   end while
34   return($x$, Impr$_C$);
end ConstructGreedyRandomized;
```

FIGURE 2. Pseudo-code for C-GRASP construction phase.

n coordinates of x have been fixed. At that stage, x and Impr$_C$ are returned from the construction procedure.

Note that the ReUse variable is utilized to speed up computations by avoiding unnecessary line searches. More details can be found in [9]. The parameter α controls the size of the RCL and therefore determines the mix of greediness and randomness in the construction procedure. Different values of α throughout the run allow some construction phases to be more greedy while others to be more random.

```
procedure LocalImprovement($x, f(\cdot), n, h, \ell, u, \rho_{lo}$, Impr_L)
1    $x^* \leftarrow x$;
2    $f^* \leftarrow f(x)$;
3    NumGridPoints $\leftarrow \prod_{i=1}^{n} \lceil (u_i - \ell_i)/h \rceil$;
4    MaxPointsToExamine $\leftarrow \lceil \rho_{lo} \cdot$ NumGridPoints$\rceil$;
5    NumPointsExamined $\leftarrow 0$;
6    while NumPointsExamined $\leq$ MaxPointsToExamine do
7        NumPointsExamined $\leftarrow$ NumPointsExamined $+ 1$;
8        $x \leftarrow$ RandomlySelectElement($B_h(x^*)$);
9        if $\ell \leq x \leq u$ and $f(x) < f^*$ then
10           $x^* \leftarrow x$;
11           $f^* \leftarrow f(x)$;
12           Impr_L $\leftarrow$ true;
13           NumPointsExamined $\leftarrow 0$;
14       end if
15   end while
16   return($x^*$, Impr_L);
end LocalImprovement;
```

FIGURE 3. Pseudo-code for C-GRASP local improvement phase.

4.3. **Local improvement procedure.** C-GRASP makes no use of derivatives. Though derivatives can be easily computed for many functions, they are not always available or efficiently computable for all functions. The local improvement phase (with pseudo-code shown in Figure 3) can be seen as *approximating* the role of the gradient of the objective function $f(\cdot)$. From a given input point $x \in \mathbb{R}^n$, the local improvement algorithm generates a neighborhood, and determines at which points in the neighborhood, if any, the objective function improves. If an improving point is found, it is made the current point and the local search continues from the new solution.

Let $\bar{x} \in \mathbb{R}^n$ be the current solution and h be the current grid discretization parameter. Define

$$S_h(\bar{x}) = \{x \in S \mid \ell \leq x \leq u,\ x = \bar{x} + \tau \cdot h,\ \tau \in \mathbb{Z}^n\}$$

to be the set of points in S that are integer steps (of size h) away from $\bar{x}$. Let

$$B_h(\bar{x}) = \{x \in S \mid x = \bar{x} + h \cdot (x' - \bar{x})/\|x' - \bar{x}\|,\ x' \in S_h(\bar{x}) \setminus \{\bar{x}\}\}$$

be the projection of the points in $S_h(\bar{x}) \setminus \{\bar{x}\}$ onto the hyper-sphere centered at $\bar{x}$ of radius h. The *h-neighborhood* of the point $\bar{x}$ is defined as the set of points in $B_h(\bar{x})$.

The local improvement procedure is given a starting solution $x \in S \subseteq \mathbb{R}^n$. The current best local improvement solution x^* is initialized to x in line 1. Lines 3 and 4 determine the number of grid points, based on the current value of the discretization parameter h, and the maximum number of points in $B_h(x^*)$ that are to be examined. This number of grid points is defined by the parameter ρ_{lo} which is the portion of the neighborhood which is to be examined. If all of these points are examined and no improving point is found, the current solution x^* is considered an *h-local minimum*.

Starting at the point x^*, in the loop in lines 6–15, the algorithm randomly selects MaxPointsToExamine points in $B_h(x^*)$, one at a time. In line 9, if the current point x selected from $B_h(x^*)$ is feasible and is better than x^*, then x^* is set to x, Impr$_\mathrm{L}$ is set to `true`, and the process restarts with x^* as the starting solution. $\texttt{Impr}_{\texttt{L}}$ is used to determine whether the local improvement procedure improved the best solution. Local improvement is terminated when an *h-local minimum* solution x^* is found. At that point, x^* and $\texttt{Impr}_{\texttt{L}}$ are returned from the local improvement procedure.

5. Metrics, scenarios, and results

In this section, we present metrics used to evaluate the solutions found for the ADR problem. We then discuss some characteristics of the particular data scenarios. Finally, we describe the testing environment and discuss the computational results.

5.1. **ADR metrics.** Mammadov et al. [13, 14, 15] define only one metric to measure the performance of their algorithm, the average precision metric. In this section, we describe the average precision metric, present an example detailing its computation, and discuss a potential pitfall of this metric. We also suggest two more conventional metrics that have been used extensively in the literature to measure distance between vectors.

Suppose we have a solution W to the ADR problem. We would like some way of measuring the quality of this solution. For each patient i, define the computed reaction of this patient to be $\hat{Y}_i = X_i W$. Recall from Section 3 that the true, or observed, reaction for this patient is given by Y_i. Let $\mathbb{T}_i$ be the set of all permutations of the indices $1,\ldots,c$ such that the permutation applied to $\hat{Y}_i$ produces a non-increasing sequence. Then, for each $\tau \in \mathbb{T}_i$, the precision for τ, $P_\tau(X_i)$, is defined as

$$P_\tau(X_i) = \frac{1}{\|Y_i\|_1} \sum_{k:Y_{i,k}=1} \frac{|R'_{i,k}(\tau)|}{|R_{i,k}(\tau)|},$$

where

$$R_{i,k}(\tau) = \{\ell \in \{1,\ldots,c\} \mid \ell \text{ is listed before } k \text{ in } \tau\} \cup \{k\}$$
$$R'_{i,k}(\tau) = \{\ell \in \{1,\ldots,c\} \mid Y_{i,\ell} = 1 \ \wedge\ \ell \text{ is listed before } k \text{ in } \tau\} \cup \{k\}.$$

The average precision $P(X_i)$ for patient i is defined to be the average of the best and worst precisions over all $\tau \in \mathbb{T}_i$. The average precision is given as the average over the average precision of each patient, i.e.

$$P_{AP} = \frac{1}{n}\sum_{i=1}^{n} P(X_i).$$

As an example of computing the precision for a patient, suppose that for patient i, the observed reaction is $Y_i = \{1,0,1,1,0\}$ and the computed reaction is found to be $\hat{Y}_i = \{0.9,0.1,0.1,0.9,0.1\}$. The set of all index permutations $\mathbb{T}_i$ that produce a non-increasing sequence for the elements of $\hat{Y}_i$ is listed in Table 1. Looking at the permutation $\tau = \{1,4,5,3,2\}$, we see that $R_{i,1} = R'_{i,1} = \{1\}$, $R_{i,4} = R'_{i,4} = \{1,4\}$, $R_{i,3} = \{1,4,5,3\}$, and $R'_{i,3} = \{1,4,3\}$. Therefore, $P_\tau(X_i) \approx 0.9167$. The best precision for this patient (from permutations that place 3 before 2 and 5) is 1, while the worst precision for this patient (from permutations that place 3 after 2 and 5) is approximately 0.8667. Hence, this patient would have an average precision of $P(X_i) \approx 0.9333$.

We now provide an example highlighting a potential shortcoming of the average precision metric. Consider the same patient observed reaction, namely $Y_i = \{1,0,1,1,0\}$, and suppose one approach to solving the ADR problem produces a computed reaction of $\hat{Y}'_i = \{0.1,0.01,0.07,0.04,0.039\}$, while a second approach produces the computed reaction of $\hat{Y}''_i = \{0.9,0.01,0.7,0.8,0.039\}$. While it is easy to see that both of these computed

TABLE 1. Permutations of $\{1,2,3,4,5\}$ that are elements of $\mathbb{T}_i$.

$\{1,4,2,3,5\}$	$\{4,1,2,3,5\}$
$\{1,4,2,5,3\}$	$\{4,1,2,5,3\}$
$\{1,4,3,2,5\}$	$\{4,1,3,2,5\}$
$\{1,4,3,5,2\}$	$\{4,1,3,5,2\}$
$\{1,4,5,2,3\}$	$\{4,1,5,2,3\}$
$\{1,4,5,3,2\}$	$\{4,1,5,3,2\}$

reactions will produce an average precision of 1.0 for this patient, it is clear that the second computed reaction $\hat{Y}_i''$ predicts much more closely the true observed reaction than does the first. Hence, for this particular example, it is unclear what the average precision metric is really measuring.

To aid in measuring how well the different solution approaches are performing, in addition to using the average precision metric we also look at the standard least squares residual

$$LS(W) = \frac{1}{n}\sum_{i=1}^{n} \|\hat{Y}_i - Y_i\|_2^2$$

and the average maximum deviation

$$AMD(W) = \frac{1}{n}\sum_{i=1}^{n} \max_{k\in\{1,\dots,c\}} [|\hat{Y}_i - Y_i|].$$

Using these conventional metrics will lend stronger support to one algorithm performing better than another.

5.2. **Testing scenarios.** The ADRAC dataset contains data for the years 1972 to 2001 inclusive. The approach taken by Mammadov et al. [13, 14, 15], which we adhere to, was to consider the years 1996 through 2001. For each of these years, a training dataset was formed using the data from 1972 to the previous year. This is the data that will be used to determine the weight matrix W. Once the weight matrix is determined, it is applied to the current year data to measure how well W predicts the true reactions. As an example, for the year 1998, data from years 1972 to 1997 (the training data) is used in the solution approaches to determine W. Then, W is applied to the year 1998 data (the test data) to measure the performance of W. In addition, it is also useful to measure the performance of W on the training data. Table 2 lists characteristics for each of the training and test data sets.

5.3. **Testing environment.** The computational experiments with C-GRASP were carried out on a Dell PowerEdge 2600 computer with dual 3.2 GHz 1 Mb cache XEON III processors and 6 Gb of memory running Red Hat Linux 3.2.3-53. C-GRASP was implemented in C++ and compiled with GNU `g++` version 3.2.3. The compiler options used were `-O6 -funroll-all-loops -fomit-frame-pointer -march=pentium4`. The algorithm used

TABLE 2. ADR data characteristics.

	Training Data			Testing Data
Year	# Patients	# Drugs	# Variables	# Patients
1996	12,600	2,253	11,265	1,049
1997	13,747	2,375	11,875	1,091
1998	15,001	2,497	12,485	1,418
1999	16,684	2,661	13,305	1,746
2000	18,599	2,786	13,930	1,988
2001	20,745	2,916	14,580	1,054

TABLE 3. C-GRASP parameter values used in the experiments.

Parameter	Value
`Maxi`	2,000
`MaxUnFixed`	10
ρ_{lo}	0.7

for random-number generation is an implementation of the Mersenne Twister algorithm described in [16]. The Mammadov algorithm was implemented in Matlab to verify the average precision results, as well as determine values for the least squares and average maximum deviation metrics.

We set the C-GRASP parameters to the values listed in Table 3. While the ideal situation would be to allow all variables to start as *unfixed* in the construction procedure, we limit the number of starting unfixed variables to 10, chosen randomly each time the construction procedure is called. It was estimated that each call to the construction procedure would have taken more than one year of computing time had we allowed all variables to start unfixed, hence the need implement this limit.

5.4. **Computational results.** Tables 4–6 display the performance of both the Mammadov algorithm and the enhanced C-GRASP heuristic for the three metrics. It is clear from these tables that, overall, the C-GRASP heuristic performs better for both the training and testing data, for each year, with the three metrics. The Mammadov algorithm, by not considering all of the data when computing W and not forming a convex combination of all drugs and suspect drugs, appears to be limited in its ability to achieve a solution significantly close to the global minimum. C-GRASP does not have this limitation. Of note is that, for each year, each iteration of C-GRASP improved upon the previous iteration. Therefore, letting C-GRASP run for more than 2,000 iterations would probably produce an even better solution. Table 7 displays the β values (discounting values) found by the C-GRASP for each year.

TABLE 4. ADR average precision results.

	Mammadov				C-GRASP	
	All		Suspect			
Year	Train	Test	Train	Test	Train	Test
1996	0.8145	0.7997	0.8291	0.8025	0.8780	0.8517
1997	0.8153	0.7905	0.8280	0.7979	0.8779	0.8528
1998	0.8158	0.7786	0.8282	0.7802	0.8758	0.8569
1999	0.8152	0.8041	0.8272	0.8091	0.8777	0.8727
2000	0.8169	0.7757	0.8272	0.7788	0.8755	0.8380
2001	0.8135	0.7687	0.8232	0.7685	0.8722	0.8283

TABLE 5. ADR least squares residual results.

	Mammadov				C-GRASP	
	All		Suspect			
Year	Train	Test	Train	Test	Train	Test
1996	0.0091	0.0330	0.0089	0.0355	0.0090	0.0345
1997	0.0088	0.0321	0.0085	0.0352	0.0087	0.0344
1998	0.0084	0.0264	0.0082	0.0297	0.0085	0.0310
1999	0.0079	0.0245	0.0077	0.0245	0.0080	0.0245
2000	0.0075	0.0239	0.0073	0.0232	0.0076	0.0230
2001	0.0071	0.0331	0.0069	0.0372	0.0074	0.0348

TABLE 6. ADR average maximum deviation results.

	Mammadov				C-GRASP	
	All		Suspect			
Year	Train	Test	Train	Test	Train	Test
1996	0.7551	0.7869	0.7306	0.8138	0.6977	0.7836
1997	0.7572	0.7653	0.7321	0.8196	0.7058	0.7719
1998	0.7525	0.7486	0.7321	0.7994	0.7140	0.7802
1999	0.7497	0.7569	0.7296	0.7484	0.7124	0.7305
2000	0.7464	0.7877	0.7267	0.7588	0.7081	0.6950
2001	0.7469	0.7902	0.7284	0.8521	0.7255	0.7544

TABLE 7. Discount factor (β) found by the C-GRASP heuristic.

Year	β
1996	0.27
1997	0.25
1998	0.28
1999	0.25
2000	0.28
2001	0.34

6. CONCLUSIONS

From the statistics given in Section 1, determining the relationship between drugs and adverse drug reactions is an extremely important problem in health-care. This paper has generalized the formulation of Mammadov et al. [13, 14, 15] to include a convex combination of suspected drugs and non-suspected drugs. Depending on the data, the number of variables in the optimization formulation can be quite large. We have contrasted the approach taken by Mammadov et al. [13, 14, 15] with our new approach. Using three different metrics, we have shown that the C-GRASP approach produced better solutions than did the Mammadov algorithm.

One area of future research concerns discounting the non-suspected drugs of each patient by a different discount factor. This would have the effect of giving less weight to those patients having the same observed reactions and conflicting suspect drug sets. At the same time, however, this would significantly increase the number of variables in the problem. Another area of future research concerns applying our C-GRASP approach to other adverse drug reaction data. The United States Food and Drug Administration [18], the World Health Organization Uppsala Monitoring Center [19], and the Canadian Adverse Drug Reaction Information System [3] all collect adverse drug reaction reports. We are currently beginning the task of collecting data from these sites.

ACKNOWLEDGMENT

This work has been partially supported by NSF, NIH, and CRDF grants.

REFERENCES

[1] J. S. Almenoff, W. DuMouchel, L. A. Kindman, X. Yang, and D. Fram. Disproportionality analysis using empirical Bayes data mining: a tool for the evaluation of drug interactions in the post-marketing setting. *Pharmacoepidemiology and Drug Safety*, 12:517–521, 2003.

[2] Australian Department of Health and Ageing Therapeutic Goods Administration. Australian Adverse Drug Reactions Bulliten. Last time accessed: August 2006. `http://www.tga.gov.au/adr/aadrb.htm`.

[3] Canadian Adverse Drug Reaction Information System. Canadian Adverse Drug Reaction Database. Last time accessed: August 2006. `http://www.cbc.ca/news/adr/database/`.

[4] T. A. Feo and M. G. C. Resende. A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters*, 8:67–71, 1989.

[5] T. A. Feo and M. G. C. Resende. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133, 1995.

[6] J. T. Harvey, C. Turville, and S. M. Barty. Data mining of the Australian adverse drug reactions database: a comparison of Bayesian and other statistical indicators. *International Transactions in Operations Research*, 11:419–433, 2004.

[7] M. J. Hirsch, C. N. Meneses, P. M. Pardalos, and M. G. C. Resende. Global optimization by Continuous GRASP. *To appear in Optimization Letters*, 2006.

[8] M. J. Hirsch, P. M. Pardalos, and M. G. C. Resende. Solving systems of nonlinear equations using Continuous GRASP. *Submitted to Journal of Heuristics*, 2006.

[9] M. J. Hirsch, P. M. Pardalos, and M. G. C. Resende. Speeding up Continuous GRASP. *Submitted to European Journal of Operational Research*, 2006.

[10] Y. Ji, H. Ying, J. Yen, S. Zhu, R. M. Massanari, and D. C. Barth-Jones. Team-based multi-agent system for early detection of adverse drug reactions in postmarketing surveilance. In *Annual Meeting of the North American Fuzzy Information Processing Society*, pages 644–649, 2005.

[11] M. Mammadov and A. Banerjee. An optimization approach identifying drugs responsible for adverse drug reactions. In J. Ryan, P. Manyem, K. Sugeng, and M. Miller, editors, *Proc. of the Sixteenth Australasian Workshop on Combinatorial Algorithms*, pages 185–200, 2005.

[12] M. Mammadov and A. Banerjee. An optimization approach to the study of drug-drug interactions. In J. Ryan, P. Manyem, K. Sugeng, and M. Miller, editors, *Proc. of the Sixteenth Australasian Workshop on Combinatorial Algorithms*, pages 201–216, 2005.

[13] M. Mammadov, E. Dekker, and G. Saunders. An optimization approach to the study of drug-reaction relationships on the basis of adrac dataset: Neurological class of reactions. In A. Rubinov and M. Sniedovich, editors, *Proc. of The Sixth International Conference on Optimization: Techniques and Applications (ICOTA6)*, 2004.

[14] M. Mammadov, A. Rubinov, and J. Yearwood. An optimization approach to identify the relationship between features and output of a multi-label classifier. In A. Rubinov and M. Sniedovich, editors, *Proc. of The Sixth International Conference on Optimization: Techniques and Applications (ICOTA6)*, 2004.

[15] M. Mammadov and G. Saunders. A comparison of two methods to establish drug-reaction relationships in the adrac database. In S. S. Raza Abidi, editor, *Proc. of The Fourth International ICSC Symposium on ENGINEERING OF INTELLEGENT SYSTEMS (EIS 2004)*, 2004.

[16] M. Matsumoto and T. Nishimura. Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation*, 8(1):3–30, 1998.

[17] E. Roux, F. Thiessard, A. Fourrier, B. Begaud, and P. Tubert-Bitter. Evaluation of statistical association measures for the automatic signal generation in pharmacovigilance. *IEEE Transactions on Information Technology in Biomedicine*, 9(4):419–433, 2005.

[18] U. S. Food and Drug Administration. Adverse Event Reporting System. Last time accessed: August 2006. `http://www.fda.gov/cder/aers/default.htm`.

[19] Uppsala Monitoring Center. Uppsala Monitoring Center - Vigilbase. Last time accessed: August 2006. `http://www.umc-products.com/DynPage.aspx?id=4910`.

[20] A. M. Wilson, L. Thabane, and A. Holbrook. Application of data mining in pharmacovigilance. *British Journal of Clinical Pharmacology*, 57(2):127–134, 2003.

[21] H. Yu, J. Yang, W. Wang, and J. Han. Discovering compact and highly discriminative features or feature combinations of drug activities using support vector machines. In *Proc. of the IEEE Bioinformatics Conference*, 2003.

(Michael J. Hirsch) RAYTHEON, INC., NETWORK CENTRIC SYSTEMS, P.O. BOX 12248, ST. PETERSBURG, FL, 33733-2248, AND DEPARTMENT OF INDUSTRIAL AND SYSTEMS ENGINEERING, UNIVERSITY OF FLORIDA, 303 WEIL HALL, GAINESVILLE, FL, 32611, USA.
E-mail address: `mjh8787@ufl.edu`

(Claudio N. Meneses) DEPARTMENT OF INDUSTRIAL AND SYSTEMS ENGINEERING, UNIVERSITY OF FLORIDA, 303 WEIL HALL, GAINESVILLE, FL, 32611, USA.
E-mail address: `claudio@ufl.edu`

(Panos M. Pardalos) DEPARTMENT OF INDUSTRIAL AND SYSTEMS ENGINEERING, UNIVERSITY OF FLORIDA, 303 WEIL HALL, GAINESVILLE, FL, 32611, USA.
E-mail address: `pardalos@ufl.edu`

(Michelle A. Ragle) DEPARTMENT OF INDUSTRIAL AND SYSTEMS ENGINEERING, UNIVERSITY OF FLORIDA, 303 WEIL HALL, GAINESVILLE, FL, 32611, USA.
E-mail address: `raglem@ufl.edu`

(Mauricio G. C. Resende) ALGORITHMS AND OPTIMIZATION RESEARCH DEPARTMENT, AT&T LABS RESEARCH, 180 PARK AVENUE, ROOM C241, FLORHAM PARK, NJ 07932 USA.
E-mail address: `mgcr@research.att.com`