A Parallel Stochastic Method for Solving Linearly Constrained Concave Global Minimization Problems

A. T. PHILLIPS[1], J. B. ROSEN[2], and M. VAN VLIET[3]
[1]*Computer Science Department, United States Naval Academy, Annapolis, MD 21402, U.S.A.*;
[2]*Computer Science Department, University of Minnesota, Minneapolis, MN 55455, U.S.A.*;
[3]*Tinbergen Institute, Erasmus University Rotterdam, The Netherlands.*

(Received: 11 November 1991; accepted: 21 November 1991)

Abstract. A parallel stochastic algorithm is presented for solving the linearly constrained concave global minimization problem. The algorithm is a multistart method and makes use of a Bayesian stopping rule to identify the global minimum with high probability. Computational results are presented for more than 200 problems on a Cray X-MP EA/464 supercomputer.

Key words. Constrained global minimization, stochastic method, multistart technique, Bayesian stopping rule, parallel computation.

1. Introduction

This paper presents a computational method for finding a solution to the problem

$$\operatorname*{global\,min}_{x\in\Omega} \varphi(x) \qquad \text{(GP)}$$

where $\varphi(x)$ is an arbitrary differentiable strictly concave function, $\Omega = \{x: Ax \leqslant b,\ x \geqslant 0\}$ is assumed to be nonempty and bounded, and $x \in \mathbf{R}^n$, $A \in \mathbf{R}^{m\times n}$, and $b \in \mathbf{R}^m$.

Problem (GP) is a constrained combinatorial optimization problem for which many well known problems are special cases. For example, the concave quadratic global minimization problem is a special case of problem (GP) for which $\varphi(x) = (1/2)x^tQx + c^tx$ where $Q \in \mathbf{R}^{n\times n}$ is symmetric and negative definite. This concave quadratic global minimization problem is known to be NP-hard (Phillips 1988), and hence it follows that problem (GP) is NP-hard. From a computational viewpoint, this means that, in the worst case, the computing time required to obtain a solution will grow exponentially with the size of the problem (in terms of the number of variables and constraints). An important property of problem (GP), which is basic to many solution methods (Phillips 1988), is that the global minimum point is always found at a vertex of the convex polytope Ω. In fact, as a consequence of the *strict* concavity of $\varphi(x)$ over Ω, every local minimum point must also be a vertex of Ω (Phillips 1988). For this reason, linear programming is an essential part of any computational algorithm to solve problem (GP).

A substantial literature exists describing applications of this global minimization problem. The recent monograph by Pardalos and Rosen (1987) contains references to many of these applications. There has also been an active research effort on computational methods for solving problem (GP), and many of these methods are summarized in Pardalos and Rosen (1986) and Mockus (1989). Most methods are restricted to a certain class of concave functions, including quadratic functions (Phillips and Rosen 1988), nonconvex separable functions (Falk and Soland 1969, Phillips and Rosen 1990), and factorable functions (McCormick 1976). Only a few methods have been developed for solving the general constrained concave minimization problem (Tuy 1964, Falk and Hoffman 1976, Horst 1984). None of these general methods, however, seem to be suitable for the efficient computational solution of problems with more than 25 variables. More recently, Hansen, Jaumard, and Lu (1991) have proposed a method for more general classes of nonconvex minimization problems which include concave problems as special cases.

This paper presents a stochastic approach for solving the linearly constrained global minimization problem. This paper continues the work begun by Rosen and van Vliet (1987) in which results are presented for a preliminary version of the method on the Cray 2 and the NCUBE-7 hypercube. The approach presented is based on a multistart technique (Rinnooy Kan and Timmer 1984, 1987). This technique repeatedly employs two phases during the solution process: a global and a local phase. In the global phase, a random search direction is selected and used to obtain a starting point from which the local phase may begin. The local phase then attempts to find a local minimum by starting from this point. Since the global minimum is generally unknown, the objective of a multistart method is to find all of the existing local minima for the problem. But, since the total number of local minima is also unknown, an optimal Bayesian estimate of the number of local minima must be used to terminate the method. This Bayesian stopping rule (Boender and Rinnooy Kan 1987) indicates that, with very high probability, all of the local minima have been found; hence, the one with the lowest function value will be the global minimum.

2. Theory and Algorithm

Existing multistart methods have considered only the unconstrained global minimization problem (Byrd *et al.* 1990). This paper considers the case where the feasible domain is a nonempty and bounded polyhedron Ω. Since it is known that the global minimum point will occur at a vertex of Ω, a local search procedure using linear programming and based on the ideas in Frank and Wolfe (1956) will be presented. This procedure is based on the following theorem:

THEOREM 1. *Let v be a vertex of Ω. If, starting from vertex v, v' solves the linear program*

$$\min_{x \in \Omega} \nabla\varphi(v)^t(x - v) \qquad \text{(LP)}$$

then either (1) $v = v'$ *and hence* v' *is a Karush–Kuhn–Tucker point for problem* (*GP*), *or* (2) $v \neq v'$ *and hence* $\varphi(v') < \varphi(v)$.

Proof. Since the vertex v' solves the linear program (LP), then the Karush–Kuhn–Tucker conditions for problem (LP) are satisfied at v'. Specifically, there exists $\lambda \in \mathbf{R}^m$ and $\mu \in \mathbf{R}^n$ such that

(1) $\lambda \geq 0\,, \quad \mu \geq 0\,,$
(2) $Av' \leq b\,, \quad v' \geq 0\,,$
(3) $\mu^t v' = 0\,, \quad \lambda^t(Av' - b) = 0\,,$ and
(4) $\nabla\varphi(v) + A^t\lambda - \mu = 0\,.$

If $v = v'$, then $\nabla\varphi(v') + A^t\lambda - \mu = 0$, which along with conditions 1, 2, and 3 satisfy the Karush–Kuhn–Tucker conditions for problem (GP) at v'. Hence, if $v = v'$ then v' is a Karush–Kuhn–Tucker point for problem (GP).

If $v \neq v'$ then v cannot be optimal for problem (LP) so that

$$\nabla\varphi(v)^t(v' - v) < \nabla\varphi(v)^t(v - v) = 0\,.$$

In addition, $\varphi(x)$ is concave over Ω so that $\varphi(v') - \varphi(v) \leq \nabla\varphi(v)^t(v' - v) < 0$. That is, $\varphi(v') < \varphi(v)$. □

Note that in the previous theorem, if $v = v'$ then v' is a Karush–Kuhn–Tucker point for problem (GP), but not necessarily a local minimum for problem (GP). For example, consider the function $\varphi(x, y) = -x^2 - (y - 2)$ with Ω defined by $0 \leq x \leq 1$, $0 \leq y$, and $x + y \leq 2$. The global minimum of φ over Ω occurs at the vertex $(1, 0)$. The two vertices $(0, 2)$ and $(0, 0)$ of Ω are both Karush–Kuhn–Tucker points of φ, but neither is a local minimum. In fact, the vertex $(0, 2)$ is the global *maximum* of φ. The final vertex $(1, 1)$ of Ω is not even a Karush–Kuhn–Tucker point of φ. It is clear from this example that only in the case when the gradient of φ at v is identically 0 or is orthogonal to the active constraints at v will the vertex v satisfy the Karush–Kuhn–Tucker conditions but not necessarily be a local minimum. The following corollary is related to theorem 1 and will be useful in the proof of Theorem 2 to be presented below.

COROLLARY 1. *If the vertex* $v \in \Omega$ *is a Karush–Kuhn–Tucker point for problem* (GP), *then* v *is optimal for the linear program*

$$\min_{x \in \Omega} \nabla\varphi(v)^t(x - v)\,.$$

Proof. Let $x \in \Omega$. Applying the Karush–Kuhn–Tucker conditions listed in the proof of theorem 1, $\nabla\varphi(v)^t(x - v) = (\mu - A^t\lambda)^t(x - v) = \mu^t x - \lambda^t Ax + \lambda^t b = \mu^t x - \lambda^t(Ax - b)$. But $x \geq 0$ and $Ax \leq b$ (since $x \in \Omega$), and $\mu \geq 0$ and $\lambda \geq 0$ so that $\nabla\varphi(v)^t(x - v) \geq 0 = \nabla\varphi(v)^t(v - v)$. Hence, v is optimal for the linear program (LP). □

The stochastic algorithm to be presented in this paper is based on solving a sequence of linear programs of the form of problem (LP) above. Since the solution to problem (LP) need only be a Karush–Kuhn–Tucker point for problem (GP) and not necessarily a local minimum, then the local phase to be applied will attempt to find all of the existing Karush–Kuhn–Tucker vertices (a superset of the set of all local minimum vertices) for problem (GP). Since the polytope Ω has only a finite number of vertices, then the total number of Karush–Kuhn–Tucker vertices of Ω (and hence also the total number of local minimum vertices of Ω) is finite. Let $K = \{v_1, v_2, \ldots, v_k\}$ represent this set of Karush–Kuhn–Tucker vertices of Ω for problem (GP).

DEFINITION. The *region of attraction* of a Karush–Kuhn–Tucker vertex $v \in K$, denoted by $R(v)$, is the set of all search directions $u \in \mathbf{R}^n$ such that the following local search procedure results in obtaining the vertex v:

1. Set $j := 1$ and solve the linear program
$$\min_{x \in \Omega} u^t x$$
to get the vertex z_0.
2. Starting from vertex z_{j-1} solve the linear program
$$\min_{x \in \Omega} \nabla\varphi(z_{j-1})^t(x - z_{j-1})$$
to get the vertex z_j.
3. If $\varphi(z_j) \neq \varphi(z_{j-1})$ then set $j := j + 1$ and go to step (2). Otherwise, stop.

THEOREM 2. *The regions of attraction* $R(v_1)$, $R(v_2), \ldots, R(v_\kappa)$ *are nonempty and* $R(v_1) \cup R(v_2) \cup \cdots \cup R(v_\kappa) = \mathbf{R}^n$.

Proof. Let $u \in \mathbf{R}^n$. Theorem 1 guarantees that the solution of step (2) satisfies either $z_j = z_{z-1}$ (i.e. $\varphi(z_j) = \varphi(z_{j-1})$) in which case $z_j \in K$, or $z_j \neq z_{j-1}$ in which case $\varphi(z_j) < \varphi(z_{j-1})$. But $\varphi(x)$ is bounded below on Ω by φ^*, the global minimum function value, so that the above local search procedure must halt in a finite number of steps at some $z_j \equiv v_i \in K$. Hence, $u \in R(v_i)$, so that $R(v_1) \cup R(v_2) \cup \cdots \cup R(v_\kappa) = \mathbf{R}^n$.

To show that each region $R(v_i)$ is nonempty, consider any search direction $u \in \mathbf{R}^n$ which lies in the nonempty positive cone generated by the set of linearly independent normal vectors of the active constraints at v_i. For any such search direction, the solution of the linear program
$$\min_{x \in \Omega} u^t x$$
is the Karush–Kuhn–Tucker vertex v_i. Noting that $z_0 \equiv v_i$ and applying corollary 1, v_i is optimal for the linear program
$$\min_{x \in \Omega} \nabla\varphi(z_0)^t(x - z_0)$$
Hence $R(v_i) \neq \emptyset$. □

The following theorem states that if a large enough sample of random search directions are chosen from a uniform distribution, then every Karush–Kuhn–Tucker vertex of Ω for problem (GP), and hence every local minimum of φ over Ω (since all local minima *must* occur at vertices of Ω), will be found using the above local search procedure. Hence, for a large enough sample of random search directions, the global minimum vertex v^* will be found

THEOREM 3. *If the search directions $u_1, u_2, \ldots, u_N$ are chosen from a uniform distribution over $\mathbf{R}^n$, then as $N \rightarrow \infty$, every $v_i \in K$, for $i = 1, \ldots, \kappa$, will be found with probability* 1.

Proof. For $i = 1, \ldots, \kappa$, let θ_i denote the probability that the Karush–Kuhn–Tucker vertex v_i is found by the local search procedure starting with some uniformly distributed random search direction. Hence, the probability that v_i is found at least once by the local search procedure using N uniformly distributed random search directions is $1-(1-\theta_i)^N$. Hence as $N \rightarrow \infty$, the Karush–Kuhn–Tucker vertex v_i will be found with probability

$$\lim_{N\rightarrow\infty} (1-(1-\theta_i)^N) = 1$$

since $0 < \theta_i$ (by Theorem 2). □

Clearly, the repeated application of this local search procedure can be terminated as soon as all of the Karush–Kuhn–Tucker vertices $v \in K$ have been discovered. Unfortunately, the number of such vertices κ is unknown. Hence, for a practical implementation of the method, a reliable estimate of the number of distinct Karush–Kuhn–Tucker vertices κ is required. As discussed earlier, only in certain pathological cases are the Karush–Kuhn–Tucker *vertices* of Ω for problem (GP) not also local minima of $\varphi(x)$ over Ω. Hence, a reliable estimate of the number of local minima of $\varphi(x)$ over Ω would also be a reliable estimate of κ. This estimate is called the *optimal Bayesian estimate* and is provided by the following theorem due to Boender and Rinnooy Kan (1987).

THEOREM 4. *Let ω be the number of different observed local minima obtained as a result of performing N uniformly distributed random local searches. Then the optimal Bayesian estimate of the number of local minima is given, for $N \geq \omega + 3$, by*

$$\frac{\omega(N-1)}{N-\omega-2} \, .$$

Proof. See Boender and Rinnooy Kan (1987). □

Byrd *et al.* (1990) suggest that a practical implementation would terminate the algorithm when this estimate exceeds ω by less than 0.5. More generally, the algorithm can be terminated when

$$\omega \leqslant \frac{\omega(N-1)}{N-\omega-2} \leqslant \omega + \delta$$

for some $0 < \delta < 1$. The right hand side of this expression can be rewritten in the form

$$N \geqslant \frac{\omega^2 + \omega}{\delta} + \omega + 2 .$$

Based on these previous theorems, we can present an algorithm for obtaining a stochastic solution to problem (GP). If P represents the number of available processors, then this procedure is as follows:

```
StochasticAlg($\varphi$, $\Omega$, $\delta$, P):
  1. Find a feasible vertex $v \in \Omega$.
     Set $v^{(1)} := v^{(2)} := \cdots j := v^{(P)} := v$.
     Set $\omega := 0$, $N := 0$, and $K := \emptyset$.
  For $i := 1, 2, \ldots, P$ (in parallel) do steps (2) through (10):
  2. Pick a random vector $u^{(i)} \in \mathbf{R}^n$.
     Set $j_i := 1$ and $N := N + 1$.
  3. Starting from vertex $v^{(i)}$ solve the linear program
         $\min_{x \in \Omega} u^{(i)^t} x$
     to get the vertex $z_0^{(i)}$.
  4. Starting from vertex $z_{j_i-1}^{(i)}$ solve the linear program
         $\min_{x \in \Omega} \nabla\varphi(z_{j_i-1}^{(i)})^t (x - z_{j_i-1}^{(i)})$
     to get the vertex $z_{j_i}^{(i)}$.
  5. If $\varphi(z_{j_i}^{(i)}) \neq \varphi(z_{j_1-1}^{(i)})$ then set $j_i := j_i + 1$ and go to step (4).
  6. Set $v^{(i)} := z_{j_i}^{(i)}$.
  7. If $K \neq K \cup \{z_{j_i}^{(i)}\}$ then set $K := K \cup \{z_{j_i}^{(i)}\}$ and $\omega := \omega + 1$.
  8. If $N < \omega + 3$ then go to step (2).
  9. If $N < (\omega^2 + \omega)/\delta + \omega + 2$ then go to step (2).
  10. Stop all processors ($i = 1, 2, \ldots, P$) and set $\varphi^* := \min\{\varphi(z) : z \in K\}$.
```

In the above algorithm, the set K represents the set of all Karush–Kuhn–Tucker vertices found by the local search procedure, $\omega = |K|$, and N is the number of random search directions.

3. Computational Testing

Because steps (2) through (10) can all be done in parallel with only a minimal amount of communication between processors (the sharing of the set of local minima K, the number of local searches N, and the number of local minima ω), we would expect the speedup obtained by this parallel algorithm over its sequential version (i.e., $P=1$) to be nearly linear for a particular problem instance. In fact, because this problem and its implementation is so similar to the class of problems discussed in Phillips and Rosen (1989), we would expect certain problem instances to produce superlinear speedups and that on the "average" (taken over a large number of problems) the speedup obtained would be linear. For more details on this sort of behavior, see Phillips and Rosen (1989).

All computational results in the following sections were obtained on a Cray X-MP EA/464 with the number of processors P set to 4 (hence, unless otherwise specified, all times reported are "wall clock seconds"), and in all of the tests cited, a tolerance of $\delta = 0.5$ was used in step (9), the termination criteria. The implementation of this method required that the set K satisfy $|K| \leq 400$. If a problem with more than 400 local minima was attempted, then that problem was halted when ω reached 400 and the "best" local minimum found so far was reported. Clearly, since the stopping criteria was not satisfied when the algorithm was stopped with $\omega = 400$, this local minimum is not necessarily the global minimum.

Note that for $\omega = 400$, the number of local searches N required for termination, according to Theorem 4, is 321,202. Each of these local searches involves the solution of a series of related linear programs. Based on the computational results presented in the following sections, the number of linear programs solved for each local search has been observed to be about four or five, and the number of pivot steps required for each linear program is also around four or five. Finally, it has been observed that the average solution time (in CPU seconds) required *per local search* can be approximated by the function (valid at least for $10 \leq n\ m \leq 40$)

$$f(m,n) = \frac{1}{25000}(m \cdot n - 50)\,.$$

The algorithm was tested on a small number of example problems found in the literature (Pardalos and Floudas 1990; Winston 1987), and also on over 200 randomly generated problems. In either case, three different classes of functions were used to test the method: quadratic, exponential, and "fixed charge" type functions. An example of the output provided by the algorithm is provided in Appendix A for the concave quadratic test problem 2.1 from Pardalos and Floudas (1990).

3.1. QUADRATIC FUNCTIONS

The first class of functions consisted of concave quadratic functions in separable form. The example problems chosen were problems 2.1, 2.2, 2.3, and 2.6 from Pardalos and Floudas (1990). The algorithm solved all four problems, obtaining the same global minima that they report. The most difficult test problem (in terms of the number of local minima and, hence, total solution time required) in their collection was problem number 2.6 with $\omega = 73$. The solution to this problem required 7.1 seconds (28.4 CPU seconds), $N = 10879$ local searches, and a total of 32057 linear programs.

The concave quadratic functions which were randomly generated had the following form:

$$\phi(x) = \sum_{i=1}^{n} \lambda_i (x_i - \nu_i)^2$$

where ν is the unconstrained global maximum of $\phi(x)$, and $\lambda_i < 0$ for $i = 1, \ldots, n$. Two sets of data were generated for this type of problem: the first set required that $\nu \in \Omega$, while the second made no restriction on ν. For the set of problems with $\nu \in \Omega$, five problems of each size with values of m and n ranging between 10 and 30 were tested. A summary of the results is presented in Table I where the average solution time (in seconds) is displayed as a function of the problem dimensions m and n. For each set of problems of the same size, the number of problems that were successfully solved (and hence, the number of problems used to compute the average solution time) is given in parentheses. If no problems of a given size were successfully solved, i.e., each of the problems exceeded the limit of 400 enforced on $|K|$, then the solution time is reported as ∞ (since the Bayesian stopping rule was not satisfied at the time the algorithm was stopped). If no problems of a given size were attempted, then the corresponding entry in the table contains "NA" (none attempted). According to Table I, for randomly generated concave quadratic functions with $\nu \in \Omega$, a practical limit on m and n would require that $m + n$ be no larger than about 30.

According to the Bayesian stopping rule of Theorem 4, the number of local searches N should vary as the square of the number of local minima ω. Figure 1 confirms this prediction by plotting the "wall clock solution time" against number of local minima ω for the concave quadratic problems (with $\nu \in \Omega$) of size $m = 10$ and $n = 10$. The quadratic function which best fits the computational data in the least squares sense is also shown. In fact, similar results are obtained for the remaining problems of the other sizes tested.

For the set of problems with no restriction on the location of the global maximum ν, the results indicate that the practical limit on m and n is somewhat larger, and the computation time required to obtain the global optimum, for problems of the same size, is significantly less when ν is unrestricted than when $\nu \in \Omega$. As before, five problems of each size with values of m and n ranging between 10 and 40 were tested. A summary of the results is presented in Table II.

Table I. Average solution time in seconds (and number of problems solved) for randomly generated concave quadratic problems with $\nu \in \Omega$

$m \backslash n$	10	20	30
10	1.4 (5)	113.0 (5)	∞(0)
20	2.7 (5)	∞(0)	NA
30	1.0 (5)	NA	NA

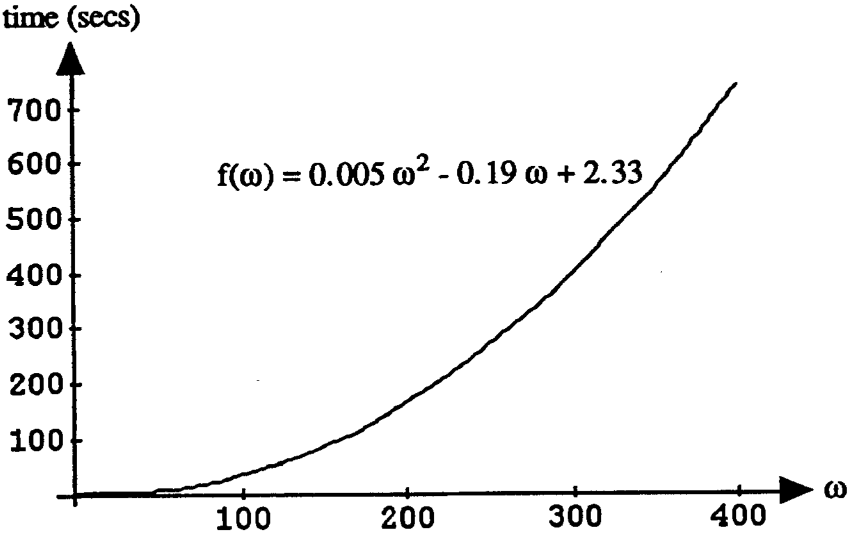

Fig. 1. Time vs number of local minima (ω) for randomly generated concave quadratic problems with $m = 10$, $n = 10$, and $\nu \in \Omega$.

Table II. Average solution time in seconds (and number of problems solved) for randomly generated concave quadratic problems with no restriction on ν

$m\backslash n$	10	20	30	40
10	1.2 (5)	51.5 (5)	488.2 (2)	∞(0)
20	0.6 (5)	462.1 (5)	604.3 (2)	NA
30	0.1 (5)	141.6 (5)	NA	NA
40	0.1 (5)	NA	NA	NA

According to Table II, for randomly generated concave quadratic functions with no restriction on ν, a practical limit on m and n would require that $m + n$ be no larger than about 40 or 50.

As in the case of Figure 1, Figure 2 displays the relationship between the "wall clock solution time" and the number of local minima ω for the concave quadratic problems (with no restriction on the global maximum ν) of size $m = 10$ and $n = 10$. The quadratic function which best fits the computational data in the least squares sense is also shown, and again, similar results are obtained for the remaining problems of the other sizes tested.

Finally, Figure 3 displays the relationship between the number of local ω and the problem dimension n, for randomly generated concave quadratic problems with $m = 10$. The function f_{int} is the quadratic function which best fits the computational data in the least squares sense for the case when $\nu \in \Omega$. Similarly, f_{unr} is the quadratic function which best fits the computational data in the least squares sense for the case when ν is unrestricted. According to Figure 3, for a given m and n, problems with $\nu \in \Omega$ have more local minima than do problems

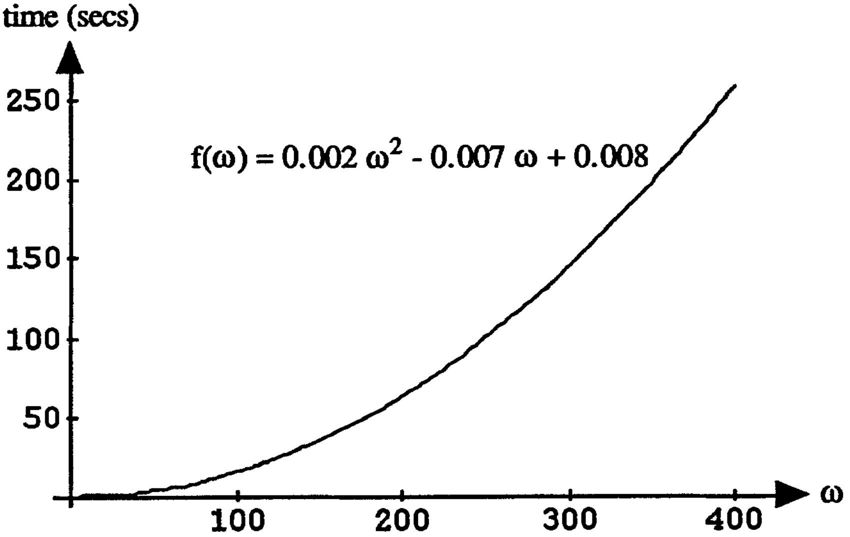

Fig. 2. Time vs number of local minima (ω) for randomly generated concave quadratic problems with $m = 10$, $n = 10$, and no restriction on ν.

with ν unrestricted. This explains why, for a given m and n, the concave quadratic problems with $\nu \in \Omega$ are more difficult than the corresponding concave quadratic problems with ν unrestricted. In both cases, based on the data collected for problems of this size and type, the number of local minima ω tends to grow as the square of the number of nonlinear variables n.

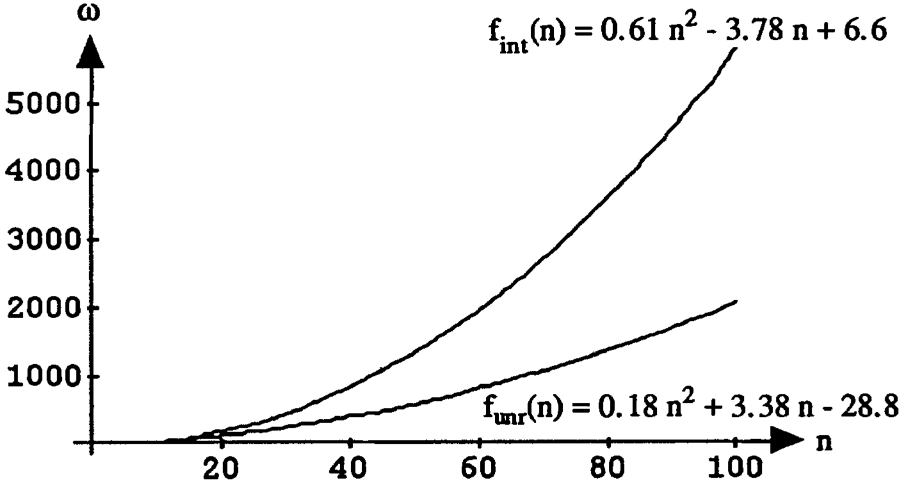

Fig. 3. Number of local minima (ω) vs problem dimension (n) for randomly generated concave quadratic problems with $m = 10$.

3.2. EXPONENTIAL FUNCTIONS

The second class of functions were randomly generated concave exponential functions of the following form:

$$\varphi(x) = \sum_{i=1}^{s} \lambda_i e^{c_i^t x}$$

where $c \in \mathbf{R}^n$ satisfies $c_i \geq 0$ for $i = 1, \ldots, n$, and $\lambda_i < 0$ for $i = 1, \ldots, n$. No example problems from the literature could be found which fit this form, although previous work by Rosen and van Vliet (1987) tested a limited number of easier problems (also randomly generated) of this same type. Results from a preliminary version of the algorithm presented in that paper are provided for both an implementation on the Cray 2 and the NCUBE-7 hypercube. The largest of the exponential problems solved with that implementation involved $m = 10$, $n = 50$, and $s = 8$. That problem had $\omega = 14$ and required 18.5 seconds wall clocktime on the Cray 2 (using $P = 4$ processors). For problems of this same size using the current implementation, the average number of local minima (for five problems) was $\omega = 16.2$ and the average solution time was 4.8 seconds on the Cray X-MP EA/464.

This category of functions tended to be the easiest class tested, due to the relatively small number of local minima present for each problem. In all, sixty problems of various sizes were generated, and all were solved. The smallest of the problems tested had dimensions $m = 20$, $n = 20$, and $s = 4$, while the largest had $m = 40$, $n = 80$, and $s = 8$. A summary of the results is presented in Tables III and IV where the average solution time (in seconds) is displayed as a function of the problem dimensions m and n.

3.3. FIXED CHARGE FUNCTIONS

Finally, the third class of functions were randomly generated concave exponential functions in separable form. These functions were chosen because they can be

Table III. Average solution time in seconds (and number of problems solved) for randomly generated exponential problems with $s = 4$

$m\backslash n$	20	40	80
20	0.4 (5)	1.0 (5)	1.9 (5)
40	0.3 (5)	1.3 (5)	4.2 (5)

Table IV. Average solution time in seconds (and number of problems solved) for randomly generated exponential problems with $s = 8$

$m\backslash n$	20	40	80
20	0.3 (5)	5.8 (5)	9.9 (5)
40	0.4 (5)	9.9 (5)	13.4 (5)

used to closely approximate functions of the so-called “fixed charge” type. The fixed charge problem is

$$\min_{x\in\Omega} c'x + \sum_{i\in J(x)} f_i$$

where $\Omega=\{x\in \mathbf{R}^n : Ax\leqslant b,\ x\geqslant 0\}$ is a nonempty polytope, $f_i\geqslant 0$ for $i=1,\ldots,n$, and $J(x)=\{j: x_j>0\}$. This is, a fixed charge is incurred for any problem variable that has a positive value at the point solution point x, but no charge is incurred for problem variables held at 0 value. Problems of this sort are very common in practical applications and are usually solved as 0–1 mixed integer linear programs. Hence, it is expected that the number of local minima may grow exponentially with the number of problem variables n. We choose to model these problems as concave exponential functions in separable form as follows:

$$\varphi(x)=\sum_{i=1}^{n}(c_i x_i + f_i(1-e^{-Mx_i}))$$

where M is a very large positive constant.

Two example problems from Winston (1987) were tested. Problem 8.3 (the “Gandhi Cloth Company Problem”) required $m=2$ constraints, $n=3$ variables, had $\omega=5$ local minima, and was solved in 0.02 seconds. Problem 8.4 (the “Lockbox Problem”) required $m=40$ constraints, $n=20$ variables, had $\omega=10$ local minima, and was solved in 1.23 seconds. In both cases, the solutions obtained are the same as those reported by Winston.

As with the class of quadratic functions, two sets of data were constructed for the randomly generated fixed charge problems: the first set required that the linear costs satisfy $c\leqslant 0$, while the second set required $c\geqslant 0$ and $0\not\in\Omega$.

A summary of results for the case when $c\leqslant 0$ is presented in Table V. According to Table V, for randomly generated fixed charge functions with $c\leqslant 0$, a practical limit on m and n would require that $m+n$ be no larger than about 20 or 25. Figure 4 shows that, like the quadratic problems presented earlier, the solution time tends to grow as the square of the number of local minima ω.

Similarly, Table VI presents the results for the case when $c\geqslant 0$ and $0\not\in\Omega$. Clearly, for problems of the same size, this form of fixed charge problem ($c\geqslant 0$) tends to be much easier (even though $0\not\in\Omega$) than the case $c\leqslant 0$ because there are far fewer local minima to consider. Unfortunately, due to the highly exponential nature of the fixed charge problem, the practical limits on m and n do not change

Table V. Average solution time in seconds (and number of problems solved) for randomly generated fixed charge problems with $c\leqslant 0$

$m\backslash n$	10	15	20
10	43.6 (5)	206.7 (1)	∞(0)
15	280.9 (4)	∞(0)	NA
20	630.8 (4)	NA	NA

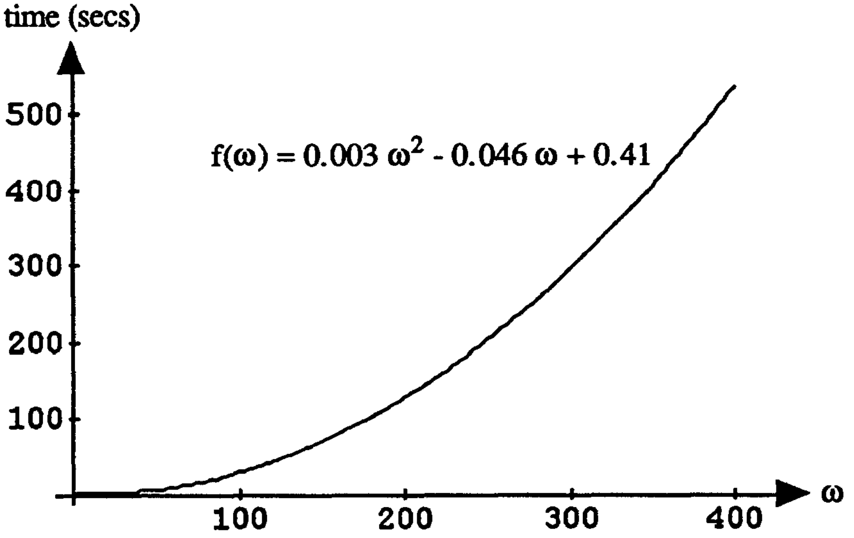

Fig. 4. Time vs number of local minima (ω) for the randomly generated fixed charge with $m = 15$, $n = 10$, $c \geqslant 0$, and $0 \not\in \Omega$.

Table VI. Average solution time in seconds (and number of problems solved) for randomly generated fixed charge problems with $c \geqslant 0$ and $0 \not\in \Omega$

$m \backslash n$	10	15	20
10	0.5 (5)	112.5 (5)	∞(0)
15	2.7 (5)	352.7 (4)	NA
20	0.7 (5)	317.3 (4)	NA

significantly. In this case, the limit would require that $m + n$ be no larger than about 25 or 30. For each problem tested with $n = 15$ and $m = 15$ or 20, the number of local minima exceeded 300. Hence, no problems with $n = 20$ and $m = 15$ or 20 were attempted since the number of local minima would very likely exceed 400.

4. Conclusions

Table VII presents a sample of the results obtained for 20 randomly generated concave quadratic problems with $m = 10$, $n = 10$, and $\nu \in \Omega$. In this table, N^* represents the number local searches (i.e. trials) needed to obtain x^*, and hence N^*/N is the fraction of local searches required before x^* was discovered. In addition, if x^* was the (ω^*)th local minimum found, then ω^*/ω is the fraction of local minima found prior to x^*. It is clear from this table that the global minimum vertex x^* is found by the stochastic algorithm very early in the sequence of local searches. In fact, on the average, only 0.55% of the total number of local searches are required to obtain the eventual global minimum x^*. The remaining 99.45% of

Table VII. Selected solution statistics for 20 randomly generated concave quadratic problems with $m = 10$, $n = 10$, and $\nu \in \Omega$

Problem#	N^*	N	ω	ω^*	$(N^*/N)\%$	$(\omega^*/\omega)\%$
1	3	2851	37	3	0.11	8.11
2	39	4006	44	13	0.97	29.54
3	1	497	15	1	0.20	6.67
4	8	2416	34	7	0.33	20.59
5	3	436	14	2	0.69	14.28
6	1	562	16	1	0.18	6.25
7	1	1771	29	1	0.06	3.45
8	23	12091	77	14	0.19	18.18
9	1	2017	31	1	0.05	3.22
10	2	1541	27	2	0.13	7.41
11	2	2851	37	2	0.07	5.40
12	8	947	21	5	0.84	23.81
13	4	947	21	4	0.42	19.05
14	2	781	19	2	0.26	10.53
15	4	1892	30	3	0.21	10.00
16	3	3487	41	2	0.09	4.88
17	37	2279	33	15	1.62	45.45
18	18	497	15	10	3.62	66.67
19	13	6671	57	9	0.19	15.79
20	24	3004	38	16	0.80	42.10
Means	9.8	2577.2	31.8	5.6	0.55	18.07

the local searches are required to find the other local minima and satisfy the Bayesian stopping rule. Furthermore, the global minimum x^* is, again on the average, found among the first 18% of the total number of local minima found.

In conclusion, the computational results obtained seem to indicate that the randomly generated fixed charge problems with $c \leqslant 0$, and the randomly generated concave quadratic problems with $\nu \in \Omega$ are among the most difficult test problems available. In addition, because the stochastic method does not take advantage of any special problem structure, this method is likely to be useful in only two types of problem instances: if it desired that all (or at least a large number) of the local minima for a given problem be found, then this is the fastest method available to date; alternatively, if there is no known structure to the problem and it is suspected that only a moderate number (e.g. $\leqslant 100$) of local minima exist, then this method can be applied with great success. On the other hand, for problems in which there is some special structure (e.g. separability of the nonlinear function $\varphi(x)$), or for problems in which the number of local minima is suspected to be large, other deterministic methods (see Phillips and Rosen 1990) would be preferable.

Acknowledgements

This research was supported in part by the National Science Foundation grants ASC-8902036 and DCR-8420935, the Air Force Office of Scientific Research grants AFOSR 87-0127 and AFOSR 89-0031, and by the Minnesota Supercomputer Institute and the Army High Performance Computing Research Center.

Appendix A

The following is typical of the output provided by the algorithm. The particular results shown here were obtained on the Cray X-MP EA/464 (with $P = 4$) for the concave quadratic test problem 2.1 from Pardalos and Floudas (1990).

```
Problem Number 2.1 - Problem Dimensions: m = 6 n = 5

- Global Minimum vertex -
  x[1] = 1.000
  x[2] = 1.000
  x[3] = 0.000
  x[4] = 1.000
  x[5] = 0.000
- Global Minimum Function Value = -17.000

Number of Local Minima = 34
Number of Trials = 2416
Number of Trials Since Final Local Min Found = 1794
Global Minimum was Local Minimum #5
Most Frequently Found was Local Minimum #20 (148 times, value = -13.500)
Global Minimum was Found 89 times
Global Minimum was Found on Trial Number 9

Average Local Minimum Function Value = -9.620
Range on Local Minimum Function Values = 17.000

CPU Time = 1.755 secs
Wall Clock Time = 0.442 secs
Random Number Seed Value = -422436315054

Total Number of Pivots = 7190
Total Number of LP Problems = 4971
Average Pivots/LP Problem = 1.45
```

References

Boender, C. G. E. and A. H. G. Rinnooy Kan (1987), Bayesian Stopping Rules for Global Optimization Methods, *Mathematical Programming* **37** (1), 59–80.

Byrd, R. H., C. L. Dert, A. H. G. Rinnooy Kan, and R. B. Schnabel (1990), Concurrent Stochastic Methods for Global Optimization, *Mathematical Programming* **46** (1), 1–29.

Falk, J. E. and K. R. Hoffman (1976), A Successive Underestimation Method for Concave Minimization Problems, *Mathematics of Operations Research* **1** (3), 251–259.

Falk, J. E. and R. M. Soland (1969), An Algorithm for Separable Nonconvex Programming Problems, *Management Science* **15** (9), 550–569.

Frank, M. and P. Wolfe (1956), An Algorithm for Quadratic Programming, *Naval Research Logistics Quarterly* **3**, 95–110.

Hansen, P., B. Jaumard, and S. H. Lu (1991), An Analytical Approach to Global Optimization, *Mathematical Programming* **52** (2), 227–254.

Horst, R. (1984), On the Global Minimization of Concave Functions: Introduction and Survey, *Operations Research Spektrum* **6**, 195–205.

McCormick, G. P. (1976), Computability of Global Solutions to Factorable Nonconvex Programs: Part I – Convex Underestimating Problems, *Mathematical Programming* **10** (2), 147–175.

Mockus, J. (1989), *Bayesian Approach to Global Optimization*, Dordrecht: Kluwer Academic Publishers.

Pardalos, P. M. and C. A. Floudas (1990), A Collection of Test Problems for Constrained Global Optimization Algorithms, in *Lecture Notes in Computer Science 455*, ed. G. Goos and J. Hartmanis, Berlin: Springer–Verlag.

Pardalos, P. M. and J. B. Rosen (1986), Methods for Global Concave Minimization: A Bibliographic Survey, *SIAM Review* **28** (3), 367–379.

Pardalos, P. M. and J. B. Rosen (1987), Constrained Global Optimization: Algorithms and Applications, in *Lecture Notes in Computer Science 268*, ed. G. Goos and J. Hartmanis, Berlin: Springer–Verlag.

Phillips, A. T. (1988), Parallel Algorithms for Constrained Optimization, Ph.D. dissertation, University of Minnesota, Minneapolis, MN.

Phillips, A. T. and J. B. Rosen (1988), A Parallel Algorithm for Constrained Concave Quadratic Global Minimization, *Mathematical Programming* **42** (2), 421–448.

Phillips, A. T. and J. B. Rosen (1989), Anomalous Acceleration in Parallel Multiple-Cost-Row Linear Programming, *ORSA Journal on Computing* **1** (4), 247–251.

Phillips, A. T. and J. B. Rosen (1990), A Parallel Algorithm for Partially Separable Non-Convex Global Minimization: Linear Constraints, *Annals of Operations Research* **25**, 101–118.

Rinnooy Kan, A. H. G. and G. T. Timmer (1984), Stochastic Methods for Global Optimization, *American Journal of Mathematical and Management Sciences* **4**, 7–40.

Rinnooy Kan, A. H. G. and G. T. Timmer (1987), Stochastic Global Optimization Methods. Part I: Clustering Methods, *Mathematical Programming* **39** (1), 27–56.

Rosen, J. B. and M. van Vliet (1987), A Parallel Stochastic Method for the Constrained Concave Global Minimization Problem. Econometric Institute Report 8734/A, Erasmus University Rotterdam, The Netherlands.

Tuy, H. (1964), Concave Programming under Linear Constraints, *Doklady Akademii Nauk SSSR* **159**, 32–35. English translation in *Soviet Mathematics Doklady* **5**, 1437–1440.

Winston, Wayne L. (1987), *Operations Research: Applications and Algorithms*, Boston: Duxbury Press, pp. 369–375.