Parallel Non-Linear Optimization : Towards The Design Of A Decision Support System For Air Quality Management

Andrew Lewis
Queensland Parallel Supercomputing Facility
Division of Information Services
Griffith University
Brisbane, QLD, 4111
Australia
A.Lewis@its.gu.edu.au

David Abramson
Department of Digital Systems
Monash University
Clayton, VIC, 3168
Australia
davida@dgs.monash.edu.au

Rod Simpson
Faculty of Environmental Sciences
Griffith University
Brisbane, QLD, 4111
Australia
R.Simpson@ens.gu.edu.au

Abstract

Large numerical simulation codes have been applied to a wide range of scientific and engineering problems. In the environmental arena the ability to predict the results of certain scenarios by computational science has allowed the choice of strategies which maximize desired outcomes (e.g. financial return) whilst minimizing environmental damage. Access to high performance computing resources has focussed attention on the development of environmental decision support systems which can be used by regulatory agencies and industry planners in evaluating different policy options. A common objective is to find a solution which optimizes some pre-defined criteria. In environmental modelling, the type of optimization problems which need to be considered involve non-linear cost functions over both discrete and continuous parameter values.

In this paper we address the optimization component of a decision support system, and perform some initial benchmark studies to assess the effectiveness of the overall

Proceedings of the ACM/IEEE SC97 Conference (SC'97)
0-89791-985-8/97 $ 17.00 © 1997 IEEE

approach**.** The algorithm selected for initial study is based on the quasi-Newton BFGS method. Whilst the BFGS algorithm is generally implemented sequentially, because of the focus of the decision support systems described in the paper we are interested in parallelizing the basic algorithm. This is achieved by concurrent evaluation of functions in finite difference approximations to the derivative and a method of interval subdivision in simple bound constrained line searching.

In a realistic problem of air quality management, use of the parallel optimization algorithm as part of an optimizing decision support system is shown to have significant performance gains over other methods of solution. In initial tests it uses less than half the evaluations of a computationally demanding numerical simulation previously used simple enumeration techniques require and is four times faster than traditional sequential optimization methods. This case study has successfully demonstrated the application of an optimization system to a core environmental model, and the feasibility of its use to solve real world problems using parallel and distributed supercomputers.

Keywords

non-linear optimization, parallel optimization, numerical simulation, decision support systems, environmental management

1. Introduction.

Large numerical simulation codes have been applied to a wide range of scientific and engineering problems [1]. In the environmental arena the ability to predict the results of certain scenarios by computational science has allowed the choice of strategies which maximize desired outcomes (e.g. financial return) whilst minimizing environmental damage. In the area of air pollution, such technology has been used for real world studies predicting air pollution levels over a planning horizon, and then examining control strategies which are realistic [2,3,4].

Access to high performance computing resources has focussed attention on the development of environmental decision support systems which can be used by regulatory agencies and industry planners in evaluating different policy options [5]. A common objective is to find a solution which optimizes some pre-defined criteria. The problems may be posed as questions such as "What set of control parameters will minimize production of smog over the planning horizon?" or "How can we minimize the financial burden of maintaining a desired air quality standard?" These are the straightforward uses for which optimizing decision support systems are intended.

Performing real world optimization using computational models is computationally very demanding. Parallel and distributed supercomputing is generally seen as the only cost effective pathway for the delivery of timely results. It is routinely used in the preparation of meteorological forecasts, an application which whilst under considerable pressure to deliver results in short time frames, is possibly orders of magnitude less demanding in terms of computing cycles required. An optimal solution to an environmental problem may not need to be delivered with less than 24 hour turnaround, but it should be available before the reasons for the question are forgotten! It is for this reason that a major focus of the work described in this paper

Proceedings of the ACM/IEEE SC97 Conference (SC'97)
0-89791-985-8/97 $ 17.00 © 1997 IEEE

is parallelization of the optimization algorithms used for solving these problems.

There are also a number of interesting optimization problems during the development of such an environmental decision support system**.** Modelling photochemical pollution has a number of essential components. Meso-scale meteorological flows can have a profound effect on smog production. Successful simulation of photochemical smog production in turn requires the accurate simulation of these air flows. Over the years, a number of prognostic mesoscale wind field models have been developed to compute 3-dimensional wind fields [6,7]. Wind field models are based on the dynamics and thermodynamics of large scale flow, as described by the Navier Stokes equations. Physical processes, such as the effect of solar and longwave radiation and interactions with the surface, must also be included. It is these effects in particular that may require considerable tuning to minimize the variance between computed and measured wind fields. This provides another opportunity for application of optimizing systems to internally verify the environmental model itself.

A decision support system can thus be applied in a number of different ways and, if engineered properly, will serve all these purposes using one architecture.

2. The Architecture of an Optimizing Decision Support System.

The architecture of the envisaged decision support system is shown in Figure 1. It will have the following characteristics:

- It should utilize high performance parallel and distributed platforms, as the most cost effective way of providing sufficient computing cycles to perform real world optimization using computational models.

- It should incorporate a wide range of optimization algorithms where the cost of a solution is computed by a modelling application, to tackle the various different types of optimization problems encountered in environmental modelling.

- It should allow for rapid prototyping of an experiment, since users of such systems often like to experiment with different aspects of the model and explore a wide range of scenarios.

- It should incorporate user interaction in the decision making process. Since it is often difficult to formulate a good cost measure for some environmental problems, it is useful if the user can guide the system using domain specific knowledge.

In this paper we only address the optimization component of the architecture, and perform some initial benchmark studies to assess the effectiveness of the overall approach**.** While numerical non-linear optimization is a large, well established field [8,9,10,11], combining optimization with computational models is quite recent. In general, **e**nvironmental models are computationally expensive, and thus high performance computing platforms are required in order to deliver solutions in a timely manner. The development of parallel algorithms for the tasks to be completed is thus an integral part of system development.

Proceedings of the ACM/IEEE SC97 Conference (SC'97)
0-89791-985-8/97 $ 17.00 © 1997 IEEE

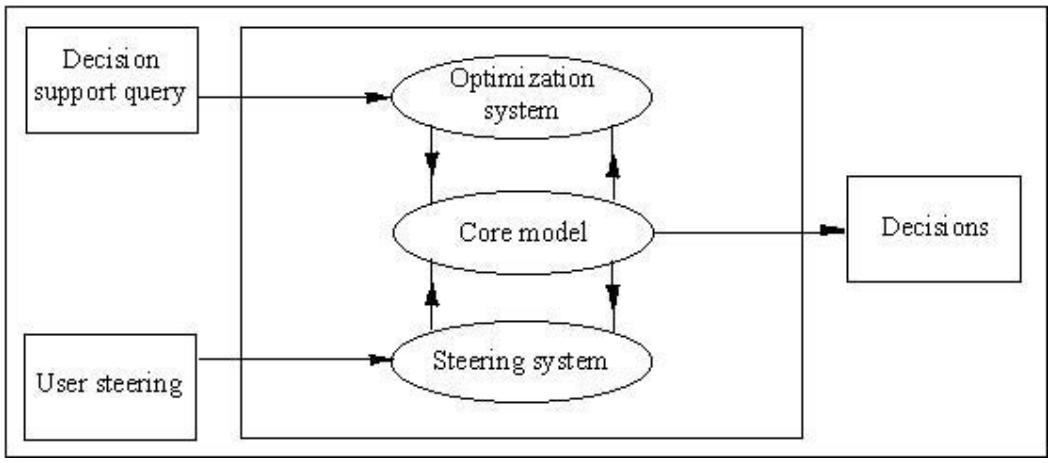

Figure 1. Decision Support System Architecture

In environmental modelling, the type of optimization problems which need to be considered involve non-linear cost functions over both discrete and continuous parameter values. Many robust codes for such purposes have been developed. Finding optimal solutions to discrete problems in which the cost is computed by a numerical simulation is extremely demanding. Meta-heuristics like simulated annealing [12], genetic algorithms [13], and Tabu search [14], are candidates for searching the parameter space. However, these can be computationally expensive in their own right and the difficulties involved in using them with expensive application models and integrating them with continuous function solution requires further investigation.

For continuous functions, gradient descent methods use the derivative of the cost function, as well as its value, to select a search direction, essentially reducing the multivariate optimization problem to univariate minimization along a search vector; a line search. They thus consist of two main operations which are executed repeatedly, gradient calculations and line searching. The algorithm selected for initial study is based on the quasi-Newton BFGS method, widely regarded as one of the most efficient and robust gradient descent methods for use with continuous functions[8,9,15].

When the cost function is the computed result of a numerical simulation, as proposed in this paper, finite difference approximations to the derivative must usually be employed. Whilst the BFGS algorithm is generally implemented sequentially, because of the focus of the decision support systems described above we are interested in parallelizing the basic algorithm. The gradient information gathering phase of the algorithm appeared a promising candidate for parallelization. The degree of parallelism achievable is limited, however, by the dimensionality of the problem domain.

For unconstrained optimization the line search phase is often inherently sequential, but for the class of problems under consideration there will often be restrictions on the expected range of variables determined by physical constraints on their possible values. The restricted expected range of the variables allows the formulation of these problems as nonlinear optimization

Proceedings of the ACM/IEEE SC97 Conference (SC'97)
0-89791-985-8/97 $ 17.00 © 1997 IEEE

problems with simple bounds, and so the line search phase of the algorithm may be parallelized by using a method of interval subdivision, rather than the typically used sequential stepping methods. A brief outline of the parallel algorithm is given in Appendix A. A detailed description is given elsewhere [16].

3. Air Quality Management – An Initial Benchmark

In the target domain of air quality management, one of the major uses of photochemical airshed models is to compute oxidant concentrations. Oxidants, such as ozone, are generated as a result of the chemical interaction between various precursors such as oxides of nitrogen (NOx) and other reactive organic compounds (ROCs) in the presence of ultra-violet radiation. Ozone is of particular importance because of its health related side effects; ozone levels in Australian urban areas recently have been observed to exceed 0.12 ppm, which is a widely adopted health standard level. Results in this area are of considerable and immediate interest to our environmental research colleagues.

The prototype parallel optimization code is being used with the CIT model developed by McRae et al [17], to compute the transport and production of photochemical smog within an urban airshed. As such it forms one half of the decision support architecture described in this paper. The addition of user steering is to be part of future development work, and may draw on other research to support this function [18]. The system is currently being used to determine the sensitivity of oxidant photo-chemistry to various input parameters.

The benchmark described in this paper was based on detailed meteorological and pollution emission data for a particular scenario have been collected for the Australian city of Perth. Further sampling studies are being performed for a number of other Australian cities. The model behaviour is well understood, as it has served as the basis for earlier work including simple enumeration studies on parallel and distributed computing platforms [19]. As an initial benchmark for the work the optimization algorithm is used to determine the NOx and ROC concentrations, as input parameters, that would produce a minimal peak hourly average ozone concentration over a 24 hour period. Future studies are expected to require minimization over up to 10 different input parameters, and objective functions related to overall population exposure to a variety of chemical species.

Figure 2 shows a sample ozone contour for the model region. It clearly shows the non-linear effect of varying ROCs and NOx on the ozone concentration. In some regions of the control space, increasing one of the precursors can increase the ozone, and in others it can decrease the ozone. Thus, a simplistic strategy of decreasing the precursors may not have the desired effect of decreasing the oxidants. It is this effect that makes it essential to correctly model the process when evaluating control strategies.

The parameter ranges are limited to a subset of the whole domain to exclude the trivial solution of reducing all precursor concentrations to zero. These limits were initially arbitrarily set to 50% in each dimension, but the range permissible for NOx was expanded down to 25% to allow the distinct non-linear feature seen in Figure 2 to extend across the whole domain. The desired solution accuracy is arbitrarily set to ±5% in the input parameters, a reasonable estimate for the

Proceedings of the ACM/IEEE SC97 Conference (SC'97)
0-89791-985-8/97 $ 17.00 © 1997 IEEE

accuracy of the simulation.

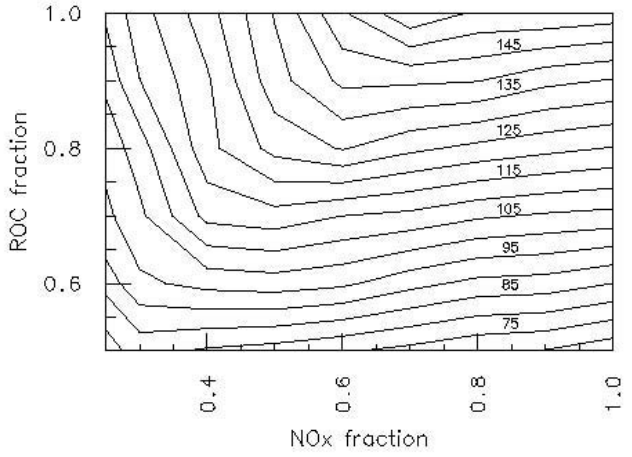

Figure 2. Typical surface: Ozone concentration (ppb) as a function of NOx and ROC

The initial benchmark was run on an IBM SP2 supercomputer. Since the problem was only 2-dimensional, the gradient determination used a maximum of 4 CPUs. This is because central difference approximations are used for gradient calculation, performing two function evaluations for each of two dimensions, giving a total of four independent operations. The configuration of the supercomputer places a current practical maximum of 10 CPUs on the line search phase of optimization. Since the machine is a shared facility, this maximum number of CPUs may not be allocated to every line search request.

During the period of this trial an average 4.6 CPUs were utilized. The parallel distribution of function evaluations was prototyped in Linda and the separate gradient calculation and line search tasks were submitted via system calls to the IBM LoadLeveler batch queueing system with appropriate requests for parallel computing resources for each task.

A solution set of input concentrations of NOx and ROC that minimized generated ozone was found at 100% NOx and 50% ROC. Inspection of Figure 2 confirms this to be the global minimum for the search domain. This result was achieved in under 14 hours of wall-clock compute time (ignoring queue wait time). This represents 46 evaluations of the cost function, using a simplified chemistry [20] in the airshed model to reduce the computational load in these exploratory tests. Each run of the airshed model currently takes about 1 hour 20 minutes of CPU time. A comparable solution by simple enumeration would require an estimated 140 hours of computing.

The same problem was solved using an inherently sequential optimization algorithm. This algorithm uses a combination of bracketing the minimum by parabolic extrapolation followed by isolation of the minimum using Brent's method[21]. The parallel algorithm, as mentioned at the end of section 2, was modified to use an iterative interval subdivision method for line searching. Using a single CPU of the IBM SP the sequential algorithm required 51 hours of wall-clock

Proceedings of the ACM/IEEE SC97 Conference (SC'97)
0-89791-985-8/97 $ 17.00 © 1997 IEEE

compute time, performing 39 evaluations of the cost function. The prototype parallel code thus achieved a speedup of 3.8 relative to the sequnetial algorithm. Using 4.6 CPUs this represents a parallel efficiency of 83%. The effect on speedup of the number of processors allocated to a line search, and thus the number of interval subdivisions performed concurrently, will be investigated in future work.

Unlike the parallel algorithm, the sequential algorithm terminated in a local minimum (79% NOx, 50% ROC) which was close to the global minimum actually discovered by the parallel algorithm. The method of line search adopted in the parallel algorithm has been found to more reliably find global minima in a number of test cases.

4. Discussion

In the example just discussed only 2 parameters are used in the optimization. As previously noted up to 10 parameters may need to be optimized in future studies. While the models in practice cannot be expected to be more than ±5% accurate, even at this accuracy for increasing n simple enumeration requires 10^n cost function evaluations. This rapidly becomes an infeasible method of solving problems of this type. The behaviour of the parallel optimization algorithm for increasing n is a subject of future work. The evidence of these early trials in which it uses far fewer function evaluations suggests it to be considerably more efficient than enumeration. Further tests are also required to compare its convergence rate with that of sequential optimization methods.

As indicated, the benchmark problem used an average 4.6 CPUs during the trial. There are practical limits to the number of CPUs this method can profitably utilize. During the gradient determination phase, two CPUs are required for each dimension, to perform the finite difference approximation. So any more than $2n$ processors are wasted for gradient determination. For the benchmark problem $n = 2$, so the maximum number of processors useful for gradient determination is 4.

During the line minimization phase, interval sub-division is repeated until the sub-intervals are less than the desired tolerance. So for a given fractional tolerance, t, the maximum number of CPUs that can usefully be employed is $1/t$. Since most line searches will be less than the entire span of the search domain, to a first approximation the average number of CPUs usefully employed will be $1/(2t)$. For the benchmark problem $t = 0.1$ so the number of processors useful for line minimization is 5.

For the benchmark problem, the number of processors available actually coincided with the theoretical optimum most of the time, as reflected in the high parallel efficiency. With some code refinements the parallel efficiency could be made very high because of the negligible extent of serial code, most computation being performed in parallel in the objective function evaluation. The algorithm scaling, however, is strictly limited by problem parameters.

Unlike the sequential algorithm, the parallel algorithm is not trapped in some local minima. This is illustrated in Figure 3 which shows the sequential algorithm using a combination of stepping and parabolic interpolation, and the parallel algorithm using interval sub-division. The sequential

Proceedings of the ACM/IEEE SC97 Conference (SC'97)
0-89791-985-8/97 $ 17.00 © 1997 IEEE

algorithm terminates in the first minimum found, while the parallel algorithm, extending its search to the boundary, finds the lower minimum.

To date only simple bounds on the optimization domain have been considered. The end uses of the decision support system may involve more general constraints. Of particular note is the challenging extension to the case study described in this paper, where the objective becomes the minimization of the financial cost of achieving some desired standard of air quality. In this case the problem may be expressed as the minimization of a (possibly discontinuous) multivariate linear function, the financial cost of reductions in emissions, subject to non-linear constraints derived from numerically intensive simulations of smog generation.

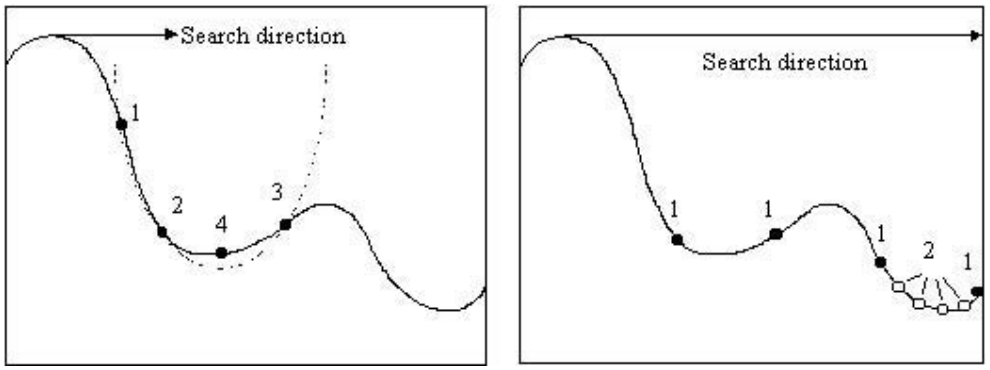

Figure 3: Comparison of sequential and parallel line minimization

The system architecture described in this paper is quite similar to that of the DAKOTA project at Sandia Laboratories [23] in the U.S.A. It additionally allows for interactive steering of the application which is important for the target application domains. DAKOTA has mainly been applied to problems in structural mechanics. The focus of the work reported here is on environmental applications so in this regard too, it is complementary to the work at Sandia Laboratories.

As the DAKOTA project has already shown with successful applications in container shape optimization, coating flow die design and vibration isolation system design, the techniques described in this paper have potential for much wider application than the initial environmental problems for which they are being developed. Interest has already been expressed in their use in vehicle aerodynamic drag minimization studies and prosthetics design in medical engineering.

5. Concluding Remarks.

Other authors [22] have demonstrated domination of computing time by user-supplied function-gradient evaluations. In the class of problems under study in this work, this trend reaches an extreme where the computational cost of the optimization algorithm is essentially negligible. In this light, the concurrent performance of function evaluations becomes of paramount importance.

Proceedings of the ACM/IEEE SC97 Conference (SC'97)
0-89791-985-8/97 $ 17.00 © 1997 IEEE

In a realistic problem of air quality management, use of a parallel optimization algorithm as part of an optimizing decision support system has shown significant performance gains over other methods of solution. In initial tests it uses less than half the number of evaluations of a computationally demanding numerical simulation than previously used simple enumeration techniques require and is more than four times faster than traditional sequential optimization methods. This case study has successfully demonstrated the application of an optimization system to a core environmental model, and the feasibility of its use to solve real world problems using parallel and distributed supercomputers.

Future work is anticipated to improve the robustness of the methods in application on a wider range of problems. Applications other than those studied to date are anticipated to include discrete parameters as well as continuous variables, and the probable inclusion of a "floor function" termination criterion to facilitate go/no-go design decisions. Other issues involved in building a real decision support system, such as the user steering system, integration of visualization sub-systems and user interface will be part of future development work.

Acknowledgements.

The authors would like to thank their colleague Rok Sosic for his helpful suggestions tendered in a number of discussions and Martin Cope and Steven Spencer for their advice regarding airshed chemistry modeling. This work was performed as part of a collaborative project between Griffith University, Monash University, Queensland Department of Environment and Victorian Environment Protection Agency, funded in part by an ARC Collaborative Grant. Thanks go to the Queensland Parallel Supercomputing Facility for the use of their IBM SP.

Proceedings of the ACM/IEEE SC97 Conference (SC'97)
0-89791-985-8/97 $ 17.00 © 1997 IEEE

References.

[1] G. Bell, **The Future of High Performance Computers in Science and Engineering**, Communications of the ACM, **32**(9), pp 1091-1101, 1989.

[2} M. Cope, F. Carnovale, and B. Cook, **An Air quality Modelling System of Oxidant Transport and Production in Port Phillip Control Region**, Proceedings of the 10th International Conference of the Clean Air Society of Australia and New Zealand, 1990.

[3] R.A. Harley, A.G. Russell, G.J. McRae et al**. Continued Development of a Photochemical Model and Application to the Southern California Air Quality Study (SCAQS) Intensive Monitoring Periods**. Carnegie Mellon University, Pittsburgh, PA and California Institute of Technology, Pasadena, CA. Report to the Coordinating Research Council under project SCAQS-8, 1992.

[4] G.J. McRae and A.G. Russell, **Effectiveness of the Air Quality Management Plan to Improve Air Quality**, report to the South Coast Air Quality Management District, El Monte, CA, 1989.

[5] B. Breckling and F. Müller, **Current trends in ecological modelling and the 8th ISEM conference on the state-of-the-art**, Ecological Modelling, **75/76**, pp 667-675, 1994.

[6] M.E. Cope and G.D. Hess, **Meso-scale meteorological modelling for input into a 3-dimensional photochemical oxidant model**, Proceedings International Congress on Modelling and Simulation, Perth, Australia, 1993.

[7] W.L. Physic, **Review: Mesoscale modelling in complex terrain**, Earth-Sci Rev., 25, 199-235, 1988.

[8] P.E. Gill, W. Murray, & M.H Wright, **Practical Optimization**, Academic Press, London, 1981.

[9] R. Fletcher, **Practical Methods of Optimization**, 2nd ed., Wiley, New York, 1987.

[10] P.E. Gill & W. Murray, eds., **Numerical Methods for Constrained Optimization**, Academic Press, London.

[11] D.G. Luenberger, **Introduction to Linear and NonLinear Programming**, Addison-Wesley, Reading, MA, 1973.

[12] N. Collins, R. Eglese and B. Golden, **Simulated annealing: An annotated bibliography**, American Journal of Mathematical and Management Sciences, **8**, Nos 3 & 4, pp 209-307, 1988.

[13] D. Goldberg, **Genetic Algorithms: In Search, Optimization and Machine Learning**, Addison-Wesley, 1989.

[14] F. Glover, **Tabu Search - part 2**, ORSA Journal on Computing, **2**(1), pp 4-32, 1990.

[15] W.H. Press, **Numerical Recipes: the Art of Scientific Computing**, Cambridge University Press, 1986.

[16] A.Lewis and D. Abramson, **A Parallel Algorithm for Bound Constrained Optimization of Numerical Simulations**, in preparation.

[17] G.J. McRae, A.G. Russell and R.A. Harley, **CIT photochemical airshed model -users manuals**, Carnegie Mellon University, Pittsburgh, PA and California Institute of Technology, Pasadena, CA, 1992.

[18] S.G. Parker and C.R. Johnson**, SCIRun: A Scientific Programming Environment for Computational Steering**, Proceedings of the 1995 ACM/IEEE Supercomputing Conference, San Diego, U.S.A., 1995.

Proceedings of the ACM/IEEE SC97 Conference (SC'97)
0-89791-985-8/97 $ 17.00 © 1997 IEEE

[19] D.Abramson, M. Cope, and R. McKenzie, **Modelling Photochemical Pollution using Parallel and Distributed Computing Platforms**, Proc. PARLE-94, pp 478-489, Athens Greece, July 1994.

[20] M. Azzi, G. Johnson, and M. Cope, **An introduction to the Generic Reaction Set photochemical mechanism**, Proceedings of the 11th International Conference of the Clean Air Society of Australia and New Zealand, Brisbane, Australia, 1992.

[21] R.P. Brent, **Algorithms for Minimization without Derivatives**, Prentice-Hall, Englewood Cliffs, N.J., 1973.

[22] J.J. Moré, **On the Performance of Algorithms for Large-Scale Bound Constrained Problems**, Large-scale Numerical Optimization, Ch. 4, SIAM, Philadelphia, 1990.

[23] M.Eldred, D.Outka, C.Fulcher, and W. Bohnhoff. **Optimization of complex mechanics simulations with object-oriented software design**, Proc. 36th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conf., pp 2406-2415, New Orleans, LA, April 1995.

Proceedings of the ACM/IEEE SC97 Conference (SC'97)
0-89791-985-8/97 $ 17.00 © 1997 IEEE

Appendix A - Parallel Algorithm

Initialization

- Evaluate the objective function at the starting point
- Calculate the gradient at the starting point by finite difference approximation (parallel function evaluation)
- Set the inverse Hessian to the unit matrix
- Set the initial line direction to the inverse of the gradient

Perform line minimization

- Truncate the line search vector for active constraints
- Determine the nearest boundary in the search direction and set maximum excursion accordingly
- Sub-divide the interval as desired (usually an integer multiple of processors available)
- Evaluate the objective function at the sub-intervals (parallel function evaluation)
- Select a bracket of three points containing the minimum value
    - Yes - Line minimization complete
    - No - Sub-divide the bracket and repeat the evaluation

Test for convergence

If (step change in function value and largest step change in position in any dimension are less than the desired tolerance)
Then {Calculate the gradient at the "minimum"
  If (gradient is less than an empirically determined constant)
  Then optimization is complete
  Else {if (the "minimum" is at a boundary normal to the gradient)
    Then optimization is complete
    Else { if (change in function value over previous two steps is less than desired tolerance)
      Then optimization is complete
      Else {Reset the inverse Hessian to the unit matrix, reset the line search direction to the inverse of the gradient and **repeat line minimization**}
      }
    }
  }

Perform BFGS update

- Calculate a new gradient at the line minimum
- Calculate the step change in the gradient
- Apply the BFGS update to the inverse Hessian
- Calculate the new line search direction and **repeat line minimization**

Proceedings of the ACM/IEEE SC97 Conference (SC'97)
0-89791-985-8/97 $ 17.00 © 1997 IEEE