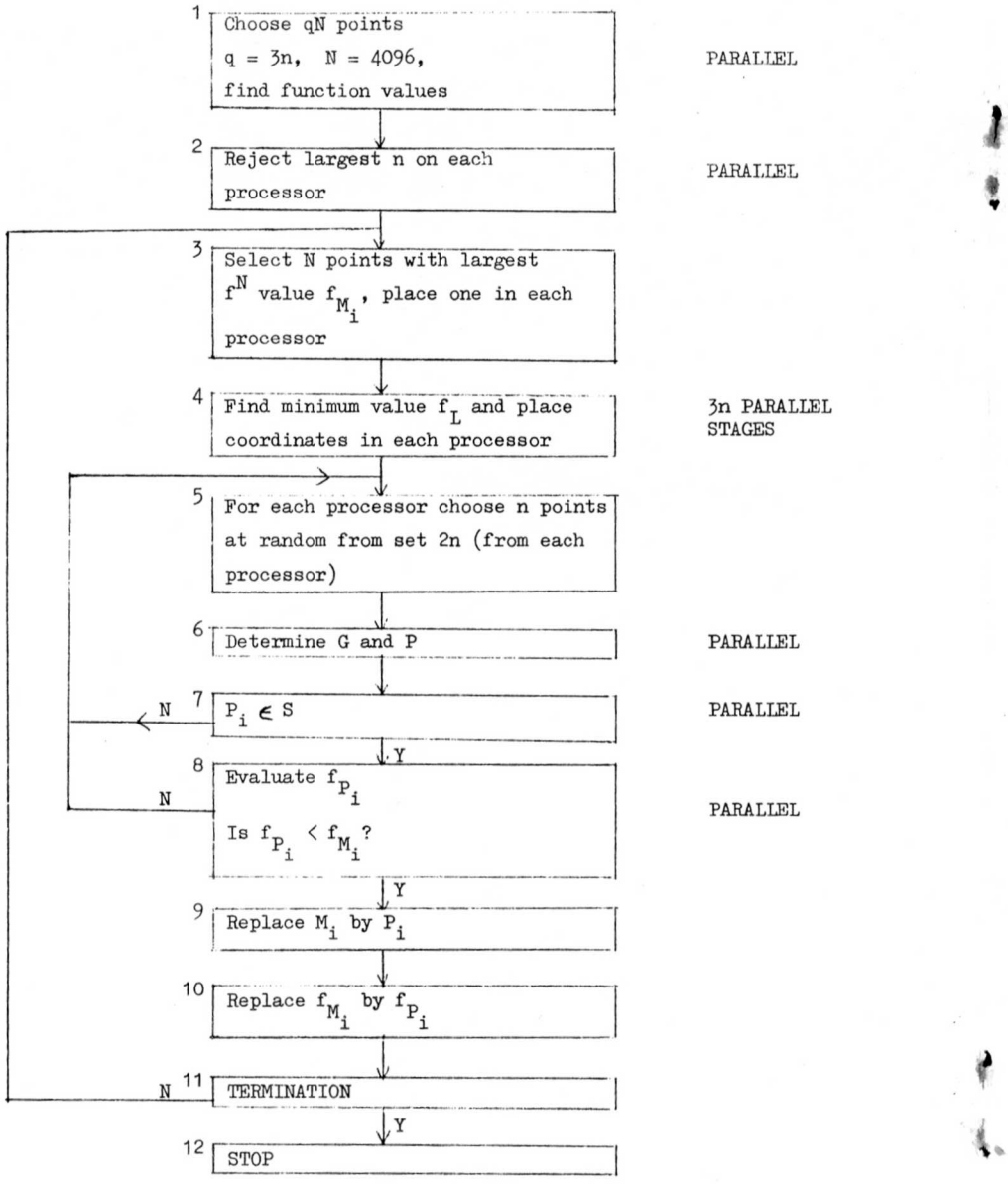

Figure 1

5. Initial results

The initial results obtained from this program were rather disappointing when compared with the sequential algorithm (the sequential one being run on a DEC system 10, 1091 here at Hatfield) as can be seen from Table 1 (sequential vs. 'old' parallel). It was the case that three sections in the program had needed to be coded sequentially and these sections were pinpointed as being the main time consumers. The available diagnostic facilities on the DAP allow for the timings of individual sections of code (or even individual statements) to be recorded. They were as follows:

(i) Step 3 on Figure 1, when choosing the overall 4096 largest function values it was necessary to code it sequentially, namely, find the largest value, mask it out, then find the next largest, etc.

(ii) Step 5 on Figure 1, when choosing n points at random for each processor it proved to be extremely difficult to find a method for doing it in parallel, since it was required to choose n points at random, avoiding the points previously rejected and keeping track of where the chosen points had come from (eventually the coordinates were recorded namely, row, column, plane).

(iii) Steps 9 and 10 on Figure 1, these two sections formed the third major sequential point in the program. When replacing the points M_i by new points P_i and similarly f_{M_i} by f_{P_i}, they had to be replaced in the positions from which they initially came.

These three points in the program were modified by means of changes to the algorithm, resulting in the new version of the parallel algorithm to be described in Section 6.

6. The new Price's algorithm for a DAP

The algorithm described in Section 4 was modified to remove the three main sequential sections of code. The main details of the new algorithm are shown in Figure 2 but briefly they are as listed below:

(i) After rejecting the largest n points on each processor we 'compact down' the matrices MI and FMI (holding the coordinates and function values of points respectively) by removing the rejected values, this means we no longer need to take care to avoid them.

(ii) Instead of choosing the overall 4096 maximum values, we now select 4096, but the maximum from each processor, which can be done quite simply in parallel.

(iii) Instead of choosing n points at random, one from each processor, we now choose n matrices at random. This means that it is now no longer necessary to record the positions of the randomly selected points for later reference, since they have all come from one of n matrices.

(iv) Since we are now choosing whole matrices at random rather than individual points, then the replacement of old points by new ones is a lot simpler, i.e. we can generate a logical mask, containing TRUE values, only at positions where the new point is contained within the domain S AND the function value f_{P_i} is less than f_{M_i}. This mask then being used in a series of simple assignments.

(v) When we compacted down the two matrices there was space left over, n planes in FMI, n^2 in MI, this space is now used for holding the centroids, new trial points, maximum values and the function values of new trial points.

(vi) The random number initialisation routine and generator were also changed to faster but adequate routines.

set of 2n*4096 and together with the minimum poing f_L, determine the centroid followed by the new trial point matrix P.

i.e.

Suppose we have chosen n points R_2, R_3, ..., R_{n+1} at random and R_1 = L then

$$\underline{G} = \left(\sum_{i=1}^{n} \underline{R}_i\right)/n$$

and

$$\underline{P} = 2\underline{G} - \underline{R}_{n+1}$$

If the new points generated are within the domain S then we evaluate F(P) otherwise we must repeat by choosing more points at random, for those which are not within S, we also repeat from this stage if F(P) is worse than the existing point.

At this stage (after one or possibly more iterations) we have a full set of 4096 points which are contained within the domain S and whose function values are less than the corresponding maximum function values. It is these improved values that are used in the replacement process and the test for termination is carried out.

The termination criterion examines the 3 smallest values of the function value L1, L2, L3 such that L1 < L2 < L3 then if $|L1 - L2| <$ and $|L2 - L3| <$ and $|L2 - L3| < \epsilon$ (ϵ = some tolerance) we stop, otherwise repeat by selecting the 4096 new largest values etc.

The algorithm also contains as a safety measure an upper bound on the number of iterations (in this case 40 - purely an arbitrary figure).

2. System for the implementation

The ICL-DAP is a single instruction multiple data machine (SIMD) having a total of 4096 individual processors (arranged in a 64 x 64 matrix) each one having its own 1 bit full adder, 1 bit accumulator, a carry register and what is known as an activity switch used for "switching off" a particular processor if need be.

The total store of the machine is at present 4096 bits per processor, and a typical single length matrix requires 32 bits, one bit per plane, allowing approximately 120 matrices of length 32 bits to be held. The DAP is connected to an ICL 2980 host computer at Queen Mary College.

3. Language for the implementation

The language used on the DAP is the DAP-Fortran language which is an extension to the existing ICL 2900 Fortran, the extension being a set of 51 built-in aggregate functions for manipulation of vector and matrix values. Details of the languages and the supporting system can be found in technical publications [1, 2, and 3].

4. Price's algorithm for a DAP

The sequential algorithm of Price has been previously modified for running on an ICL-DAP for small values of n, as is described in Dixon & Patel [4]. The main features of the modified Price's algorithm are contained on the flow chart (Fig. 1).

Briefly what happens is this, 3n x 4096 points are generated over the domain S and their function values determined. (3n points allocated to each processor) the largest valued n points per processor are then rejected and the overall 4096 remaining largest values are chosen, one being placed in 'each' processor.

In order to generate new trial points P_i we choose 4096*n points at random (n points coming from each of the processors) from the remaining

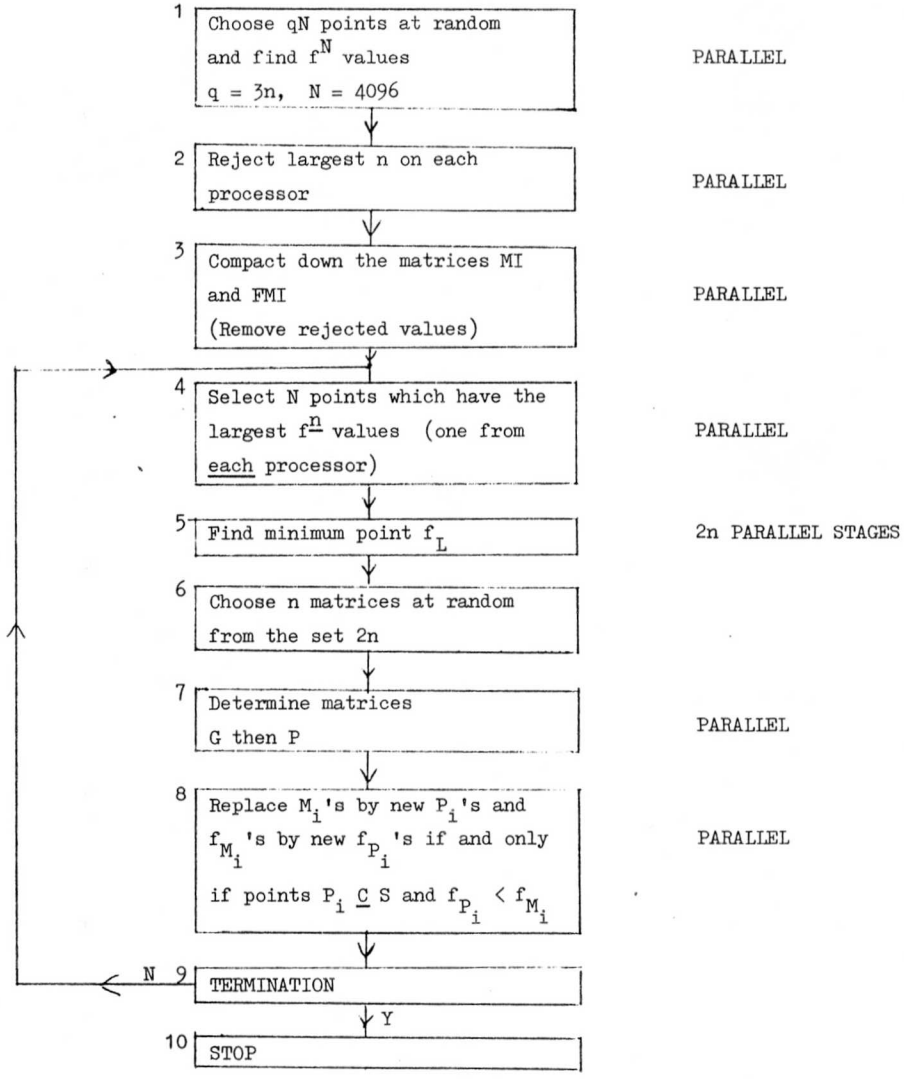

Figure 2

7. Final results

The results of this new version of the algorithm are also displayed in Table 1 (sequential vs. 'new' parallel). This time a positive improvement can be seen compared with the sequential algorithm and an even greater improvement on the 'old' parallel algorithm.

A direct comparison of the times for the sequential and the new parallel yields the improvement factor which can be seen to range from 3.2 up to 68. This considerable range can be explained by the fact that the lower improvement factors came from functions which had just one global minimum, whereas the larger factors came from the functions possessing multiple global minima. The best example is probably the SHUBERT function which has 18 global minima; the 'new' parallel algorithm picked up 11 of these minima in one run, whereas the sequential algorithm required 10 runs to pick up these same 11 minima.

It can also be seen that the number of parallel function evaluations needed has been drastically reduced in most cases to single figures but at the most to just 14.

Table 2 contains the numeric results of the programs and Table 3 the actual values (obtained from [5] and elsewhere). It can be seen that there is a reasonable comparison in all of the test cases except for SHEKEL's function where the algorithm only managed to locate the approximate position of the global minima, this was then allowed to run for 40 iterations and also 8 different random sequences but still failed to improve upon the quoted figure.

Taking the percentage differences of the values we find that the results for the sequential algorithm were within an average of 0.65% of the actual values and for the new parallel algorithm within an average of 1.54% but if we include the results from the SHEKEL function then this figure worsens to 4.57%.

1. The Problem

The problem of global optimisation is as follows:

$$\min F(x) \qquad x \in S \subset R^n$$

where $S = \{x: a_i \leq x_i \leq b_i\}$

Assuming that the function $F(x)$ is non-convex and has more than one local minimum, then the problem is to isolate the point x^* with the least function value (which is by definition, the global minimum).

An international study comparing the relative efficiency of a number of programs for the solution of the global optimisation problem is reported in Dixon & Szegö [5]. That study indicated that most of the efficient codes are based on probabilistic principles and that the heuristic CRS algorithm proposed by Price [6] was one of the most successful. He has recently proposed a simplified version [7] and this was chosen for implementation on the parallel system.

The probabilistic method of solution relies on the following assumption:

if we have a finite number of points chosen at random and uniformly distributed in the region S, then if A is a subset of S with a measure m such that

$$\frac{m(A)}{m(S)} \geq \alpha > 0$$

and p(A,N) being the probability that at least one point of a sequence of N random points lies in A,

then $\lim_{N \to \infty} p(A,N) = 1$

The heuristic method then aims at improving the best points located in the random search.

THE IMPLEMENTATION OF A PARALLEL VERSION OF PRICE'S (CRS) ALGORITHM ON AN ICL DAP

by

Paul G. Ducksbury

Abstract

This report describes the way in which a parallel version of Price's (controlled random search) algorithm was implemented on an ICL Distributed Array Processor (the DAP situated at Queen Mary College, London) for the solution of multi-extremal global optimisation problems; together with the corresponding results that were obtained from it.

The author would like to thank the SERC for their support and the Queen Mary College DAP Support Unit for permission to use their system.

8. Conclusions

Despite the fact that the numeric results for the new algorithm were very slightly worse than those of the sequential algorithm ($4.57 - 0.65 \approx 3.9\%$) there must be an advantage in a considerable reduction in the number of parallel function evaluations carried out (particularly on functions for which a large proportion of the time is spent in the function evaluation routine) thus leading to a reduced time.

The CRS optimisation method described in [5] on which this method is based is designed for thoroughness of the search rather than speed of convergence and where possible the global minima discovered could be refined using faster methods.

The algorithm is only suitable for small values of n, namely those in the range

$$2 \leq n \leq 5$$

The restriction on larger values is one of the available storage space. A possible alternative for larger dimensional problems would perhaps be to reduce the number of points generated to say 2n per processor but this would later limit the choice of the matrices in a random way.

The program is now successfully running on an ICL DAP. The two versions of the implementation illustrate an interesting property, namely, the algorithm (or any algorithm) must be designed with the available facilities in mind, not necessarily the detailed hardware facilities but rather the software ones, i.e. the built-in DAP functions and the best way in which to use them.

Hence no matter how efficient the machine is, a number of sequential code sections (as is the case with the initial implementation) in the program can totally destroy any possible advantage which comes from using the parallel machine.

FUNCTION	SEQUENTIAL		NEW PARALLEL		OLD PARALLEL		IMPROVEMENT
	Time (s)	f^N eval.	Time (s)	f^N eval.	Time (s)	f^N eval.	Note (3)
SIX HUMP	0.36	288					
CAMEL BACK	0.81	798	0.093	7	95.6	21	x 12.5
(2 global minima)	1.17	1086		Note (1)			
BRANIN	0.47	360					
(3 global minima)	0.97	654	0.118	8	25.7	9	x 26.8
	1.73	1024		Note (1)	25.8	9	
	3.17	2038			Note (2)		
GOLDSTEIN & PRICE (1 global minimum)	0.45	481	0.13	8	37.78	12	x 3.5
HARTMANS N = 3 (1 global minimum)	0.88	503	0.276	11	67.33	13	x 3.2
N = 6			INSUFFICIENT STORE				
SHEKEL (1 global minimum)							
m = 5 Note (5)	3.89	2732	0.453	14	NOT RUN		x 8.6
m = 7 Note (5)	3.82	2423	0.55	14	NOT RUN		x 6.9
m = 10 Note (5)	4.31	2567	0.693	14	NOT RUN		x 6.2
SHUBERT 2-D (18 global minima)	average 1.58/min total 15.8 Note (4)	average 995	0.233 Note (4)	8	NOT RUN		x 68

TABLE 1

NOTES

(1) Multiple global minima were identified in one run
(2) Two runs were needed to locate 3 global minima
(3) Improvement of new parallel as compared with the sequential
(4) The parallel algorithm picked up 11 minima in one run hence the same 11 minima from the sequential version were used for the comparison (these 11 being picked up in 10 runs)
(5) For this particular function the approximate location only was located for the global minimum.

REFERENCES

[1] B.L. Fox, "Counterparts of variance reduction techniques for quasi-Monte Carlo integration", Manuscript, University of Montreal (1983).

[2] J.H. Halton and G.B. Smith, "Algorithm 247: radical inverse quasirandom point sequences", Comm. ACM 7 (1964) 701-702.

[3] H. Niederreiter, "Quasi-Monte Carlo methods and pseudo-random numbers", Bull. Amer. Math. Soc. 84 (1978) 957-1041.

GENERATING TEST PROBLEMS QUASI RANDOMLY

Bennet L. Fox
University of Montreal

Test problems have data $(X_1,\ldots,X_m)$ with m fixed. Here we generate $(X_1,\ldots,X_m)$ from some sample space and evaluate the efficiency of the algorithm as $f(X_1,\ldots,X_m)$. For example, f may be the time to solve the problem with the given algorithm. We want to estimate its expectation $E[f(X_1,\ldots X_m)]$.

One averages $f(X_{1k},\ldots,X_{mk})$ over the samples generated, where the subscript k indicates sample k. One way to do this, apparently universally used by the mathematical programming community, generates these samples via pseudo-random numbers. To the extent that these pseudorandom numbers mimic genuinely random numbers, the expected absolute error is roughly $O(1/\sqrt{N})$ where N is the number of samples used.

The estimation error is an order-of-magnitude less, however, using quasirandom points $(U_{1k},\ldots,U_{mk})$ and setting $X_{jk} = G_j^{-1}(U_{jk})$ where X_{jk} has cumulative distribution function G_j. Using the method for generating quasirandom points in Halton and Smith [2], for example, an upper bound on the error is $O((\log N)^m/N)$. The exponent m suggests making all the data depend on at most a dozen independent (random) parameters. Intuitively, pseudorandom points can cluster or spread out badly because they are "independent". Quasirandom points, by design, are spaced to give a low error bound for numerical integration. Niederreiter [3] surveys quasi-Monte Carlo methods. Fox [1] discusses analogs of variance reduction techniques for quasi-Monte Carlo integration.

FUNCTION	SEQUENTIAL	NEW PARALLEL	OLD PARALLEL
SIX HUMP CAMEL BACK (2 global minima)	f = -1.0312 @ 0.09047, -0.70548 & -0.08851, 0.71195	f = -1 030088 @ 0.09524, -0.7262 & -0.07299, 0.71892	f = -1.03035 @ -0.07846, 0.72161 Note (1)
BRANIN (3 global minima)	f = 0.39797 @ 3.14294, 2.2821 & 9.43796, 2.4653 & -3.1364, 12.2577	f = 0.401413 @ 3.16611, 2.23063 & 9.41012, 2.5422 & -3.12068, 12.12934	f = 0.40254 @ 3.1431, 2.272 & 9.3941, 2.477 & -3.16734,12.375
GOLDSTEIN & PRICE (1 global minima)	f = 3.00008 @ 0.00061, -0.99986	f = 3.041037 @ 0.01146, -1.00212	f = 3.037369 @ 0.010315, -0.99176
HARTMANS N = 3 (1 global minima)	f = -3.86223 @ 0.12838, 0.55814, 0.85407	f = -3.85239 @ 0.12173, 0.57061, 0.85807	f = -3.85532 @ 0.1454, 0.5416, 0.85097
SHEKEL N = 4 (1 global min) m = 5	f = -10.15269 @ 3.99941, 3.99815, 4.00085, 3.99975	f = -4.85933 @ 4.01411, 3.96868, 3.74164, 4.21136	NOT RUN
m = 7	f = -10.4028 @ 4.00118, 3.99993, 3.99974, 4.00025	f = -5.106936 @ 4.01411, 3.96868, 3.74164, 4.21136	NOT RUN
m = 10	f = -10.5362 @ 4.00182, 3.99978, 3.99978, 3.99994	f = -5.2288 @ 4.01411, 3.96868 3.74164, 4.21136	NOT RUN
SHUBERT 2-D	f = -186.38818 @ -7.0817, -7.7046 -1.4555, -0.80274 4.8788, -7.06644 -0 785, 4.8523 -7.7083, -7.0835 4.8663, -0.7934 -1.42505, 5.4828 -7.7083, -0.8001 -7.7077, 5.483 -0.802, -7.7052 5.4828, 4.8579	f = -186.5548 -7.09226, -7.7066 -1.40794, -0.80522 4.88029, -7 0847 -0.79988, 4.89147 -7.66859, -7.0571 4.84246, -0.8475 -1.4149, 5.42207 -7.76933, -0.79236 -7.66386, 5.43725 -0.80595, -7.64545 5.41464, 4.81433	NOT RUN

TABLE 2

NOTE

(1) Only one of the global minima was located, the random sequence was changed several times

FUNCTION	ACTUAL VALUES
SIX HUMP CAMEL BACK	f = -1.0316285 @ 0.08983, -0.7126 and -0.08983, 0.7126
BRANIN	f = 0.397887 @ 3.14159, 2.275 and 9.42478, 2.475 and -3.14159, 12.275
GOLDSTEIN & PRICE	f = 3.0 @ 0.0, -1.0
HARTMANS N = 3	f = -3.86278 @ 0.11484, 0.555515, 0.852551
SHEKEL N = 4 m = 3	f = -10.1532 @ 4.0003, 4.00016, 4.00002, 4.00011
m = 7	f = -10.4029 @ 4.00052, 4.00078, 3.99943, 3.99957
m = 10	f = -10.5364 @ 4.00066, 4.00054, 3.99955, 3.99948
SHUBERT 2-D	f = -186.73091 @ -7.0835, -7.70831 -1.42513, -0.80032 4.85805, -7.0835 -0.80032, 4.85805 -7.70831, -7.0835 4.85805, -0.80032 -1.42513, 5.48286 -7.70831, -0.80032 -7.70831, 5.48286 -0.80032, -7.70831 5.48286, 4.85805

TABLE 3

now a standard tool for solving constrained nonlinear programming problems, but they were not treated during the Summer School in Urbino. The main topics of the new ASI are not design and implementation of software, but the test and application of optimization algorithms, and should therefore be considered as the continuation of the work, the Committee on Algorithms started in 1977.

In the terminology used by NATO, the director of the Advanced Study Institute is Klaus Schittkowski (University of Stuttgart, Germany, F.R.). Co-directors are Karla Hoffman (National Bureau of Standards, Washington D.C., USA) and Jan Telgen (Rabobank Netherland, Zeist, Netherlands). Limited funds are available for participants from NATO-countries to cover a part of their travel and accommodation expenses. Since the number of participants is limited to approximately 70 persons, participation is possible only by personal invitation. For more information and an application form, write to

Klaus Schittkowski
Institut für Informatik
Universität Stuttgart
Azenbergstraße 12
D-7000 Stuttgart 1
Germany, F.R.

Global optimization (A.H.G. Rinnooy Kan, Erasmus University, Rotterdam, Netherlands)

Generation of test problems, experimental design, comparative performance evaluation (R. Jackson*), National Bureau of Standards, Washington, D.C. USA)

Optimal control (D. Kraft, DFVLR, Oberpfaffenhofen, Germany, F.R.)

Algorithmic procedures for stochastic optimization (R. Wets, University of Kentucky, Lexington, USA)

Parallel computing in optimization (L.C.W. Dixon*), Hatfield Polytechnic, United Kingdom)

During the ASI some time will be reserved for participants to lecture about their own research activities, application models, and solution methods. All participants will thereby learn from the experiences of those who use optimization algorithms to solve real life problems.

An important intention of the ASI will be to combine people working in universities, public research institutions, and industry. Intensive contacts between mathematicians interested in theoretical developments of algorithms and scientists and engineers whose job it is to use these tools in their modelling efforts, will lead to a better understanding of typical difficulties arising in each of these areas and to additional motivations for further co-operative research.

The first ASI organized by the Committee on Algorithms was titled 'The Design and Implementation of Optimization Software' and was held in Urbino, Italy, during 1977. However, so many new mathematical algorithms, numerical implementations, testing methologies and application models have been developed since then, that we are faced now with a new situation. Linear programming models are still as important as in earlier years, but we observe an increasing number of applications which use the tools of nonlinear programming, optimal control, integer programming and stochastic optimization. To give an example, sequential quadratic programming methods are

*)not yet confirmed

9. References

1. "DAP introduction to Fortran programming," Technical Publication TP 6755, ICL Putney, London SW15.

2. "DAP-Fortran Language," Technical Publication TP 6918, ICL Putney, London SW15.

3. "DAP - Developing DAP programs," Technical Publication TP 6920, ICL Putney, London SW15.

4. Dixon, L.C.W. and Patel, K.D., "The place of parallel computation in numerical optimisation II, the multi-extremal global optimisation problem," Technical Report 119, Numerical Optimisation Centre, The Hatfield Polytechnic.

5. "Towards global optimisation 2," edited by L.C.W. Dixon and G.P. Szegö, North Holland Press, 1978.

6. Price, W.L., "A heuristic CRS algorithm," in Towards global optimisation 2, edited by L.C.W. Dixon and G.P. Szegö, North Holland Press, 1978.

7. Price, W.L., "A new version of the controlled random search procedure for global optimisation," Technical Report, Engineering Dept., University of Leicester, 1981.

MIXED INTEGER PROGRAMMING IN MATHEMATICAL PROGRAMMING SYSTEMS

The special issue has arrived. Anyone interested in a copy of this report should send a self addressed, stamped envelope (weight: 4 oz) to:

Dr. Karla Hoffman
Center for Applied Mathematics
National Bureau of Standards
Washington, D.C. 20234

and will receive this exciting publication post haste!

SPECIAL ISSUE
MIXED INTEGER PROGRAMMING IN
MATHEMATICAL PROGRAMMING SYSTEMS
RICHARD H.F. JACKSON AND RICHARD P. O'NEILL, EDITORS

Contents:

A JOINT PUBLICATION OF:
THE COMPUTER SCIENCE TECHNICAL SECTION OF ORSA
THE COMMITTEE ON ALGORITHMS OF THE MATHEMATICAL PROGRAMMING SOCIETY

NATO ADVANCED STUDY INSTITUTE ON

COMPUTATIONAL MATHEMATICAL PROGRAMMING

Klaus Schittkowski

The Committee on Algorithms of the Mathematical Programming Society announces a summer school on Computational Mathematical Programming to be held in Bad Windsheim, Germany F.R., from July 23 to August 2, 1984, under the sponsorship of NATO. The Advanced Study Institute intends to bring together optimization specialists developing algorithms or software, and scientists and engineers who use these tools in their modelling efforts. The ASI consists of tutorials emphasizing new mathematical programming algorithms, software products, computational experiments, numerical test results, and practical optimization models. The following list summarizes the topics and the invited lecturers:

Integer programming (M. Beale, SCICON, Milton Keynes, United Kingdom)
Model building in linear and large integer programming (H.P. Williams, University of Edinburgh, United Kingdom)
Networks (D. Klingman, University of Texas, Austin, USA)
Nonlinear programming (R. Fletcher*), University of Dundee, United Kingdom)
Model building and practical implementation aspects in nonlinear programming (P.E. Gill, Stanford University, USA)
Large linear systems (J. Stoer, University of Würzburg, Germany, F.R.)
Large scale nonlinear programming (P. Toint*), University of Namur, Belgium)
Geometric programming (M. Rijckaert, University of Leuven, Belgium)
Nondifferentiable optimization (J. Zowe, University of Bayreuth, Germany, F.R.)

MESSAGE FROM THE CHAIRMAN

I open this column with a strong and heartfelt expression of Thanks to Klaus Schittkowski and Jan Telgen for all their time and effort in organizing the NATO Advanced Study Institute on Computational Mathematical Programming. This meeting will be held July 23 to August 2, 1984 in Bad Windsheim, Germany. The program has been designed to allow both for state-of-the-art lectures and for extensive discussions among participants who we hope will be both developers and users of MP software.

At this conference, COAL will have a special session to discuss how COAL can better serve the MP community. I have received many suggestions as to what future directions the committee should take. Most people who have contacted me believe that COAL should expand its domain to all issues related to Computational Mathematical Programming. We have subconsciously been moving in that direction, but now I think it is time for COAL to state explicitly the expansion of its objections. A draft list of objectives will soon be distributed to all COAL members for discussion.

The proposal for a prize for excellence in Computational Mathematical Programming has been submitted to the Mathematical Programming Society Council and I hope to hear (positively) from Council by early fall so that the first prize could be awarded at the XII MPS Symposium in 1985.

Finally, a COAL-sponsored session on "Implementation Aspects of MP Software" will be held at the ORSA/TIMS San Francisco Meeting.

Clearly, the Committee has been active this summer and I extend my thanks to each of you for making all of these activities possible.

****

SIAM Conference on Numerical Optimization
(communicated by Paul T. Boggs, NBS)

****

SIAM will sponsor a conference on numerical optimization on June 12-14, 1984 in Boulder, Colorado. The meeting will focus on three timely and important topics of interest to both researchers and practitioners. These topics are mathematical software for nonlinear optimization, techniques for solving nonlinearly constrained problems, and methods for global optimization. In addition, there will be a special session on applications of optimization as well as contributed papers sessions.

The invited speakers include D. M. Gay (Bell Laboratories), W. Murray (Stanford University), and M. J. D. Powell (Cambridge University) speaking on software; A. Conn (University of Waterloo), R. Fletcher (University of Dundee), and L. Lasdon (University of Texas) speaking on nonlinearly constrained problems; and A. V. Levy (University of Mexico) and A. Rinnooy-Kan (Erasmus University) on global optimization.

The conference will be preceded by a one-day short course which will provide an overview of the state of the art in numerical optimization. The course is intended for scientists, mathematicians and engineers interested in learning about numerical optimization and will be oriented towards the topics of the conference. The course will be taught by P. T. Boggs (National Bureau of Standards) and R. B. Schnabel (University of Colorado).

Further information is available from the conference organizers, P. T. Boggs, R. H. Byrd (University of Colorado) and R. B. Schnabel, or from SIAM.

MATHEMATICAL PROGRAMMING

a publication of the mathematical programming society

AIMS & SCOPE

MATHEMATICAL PROGRAMMING publishes original work dealing with every theoretical, computational, and applicational aspect of mathematical programming; that is, everything of direct or indirect use for questions surrounding the problem of finding the extreme values of functions of many variables.

Included, along with the conventional topics of linear, nonlinear, integer and stochastic programming, are computer experimentation, techniques for formulating and applying mathematical programming models, computer programming devices of special interest to the subject; unconstrained optimization, convexity, polytopes, and control and game theory done in the spirit of mathematical programming.

Expositions and surveys, original research, reports on computer routines and computer experimentation, and new or ingenious practical applications are published.

MATHEMATICAL PROGRAMMING contains a 'Short Communication' section consisting of brief and timely announcements of new results, applications, codes or experiments. These submissions benefit from accelerated refereeing. Papers omitting proofs and details are encouraged if accompanied by supporting material establishing the validity of the assertions made.

THE MATHEMATICAL PROGRAMMING SOCIETY

ENROLLMENT

I hereby enroll as a member of the Society for the calendar year 1983.

PLEASE PRINT Name _____
Mailing address _____
_____
_____

My subscription to *Mathematical Programming* is for my personal use and not for the benefit of any library or other institution.

Signed _____

The dues for 1983 are:
35 Dollars (U.S.A.)
20 Pounds (U.K.)
72 Francs (Switzerland)
230 Francs (France)
85 Marks (Fed. Rep. Germany)
95 Guilders (Netherlands)

Please send this application with your dues to:
The Mathematical Programming Society
%The International Statistical Institute
428 Prinses Beatrixlaan
2270 AZ Voorburg, Netherlands

Student Applications: Dues are one-half the above rates. Have a faculty member verify your student status below and send application with dues to: Professor John Mulvey, School of Engineering and Applied Science, Princeton University, Princeton, N.J. 08544.
Faculty verifying status _____ Institution _____

EDITORIAL COLUMN

This is the first COAL Newsletter published under the co-editorship of Jan Telgen and Bob Meyer. Final responsibility for production and publication of the Newsletter still resides with Jan Telgen, but both co-editors will solicit and accept papers for publication. Therefore authors may contact either of the co-editors regarding potential contributions for the Newsletter.

On our part we will try to improve upon the readability of the Newsletter in two ways. First, the Newsletter will be produced from camera-ready material that is as uniform in quality and style as possible within our budget. Second, we will do some screening on the papers to be published. Both these points were put into practice starting with this issue.

A discussion is in progress on the position and function of the COAL Newsletter as indicated in our previous issue. In particular, it may not be possible to continue free distribution of the Newsletter to non-members of MPS. The next issue of the Newsletter will contain the viewpoints of the Committee on this matter and a request for reactions from the readership. The final decision on these issues is planned for the next COAL-business meeting to take place at the NATO ASI in Bad Windsheim (see pages 3-5).

Robert R. Meyer

Jan Telgen

COAL OBJECTIVES

The Committee on Algorithms is involved in computational developments in mathematical programming. There are three major goals: (1) ensuring a suitable basis for comparing algorithms, (2) acting as a focal point for computer programs that are available for general calculations and for test problems, and (3) encouraging those who distribute programs to meet certain standards of portability, testing, ease of use and documentation.

NEWSLETTER OBJECTIVES

The newsletter's primary objective is to serve as a forum for the Friends of COAL. Through an informal exchange of opinions, members have an opportunity to share their experiences. To date, our profession has not developed a clear understanding on the issues of how computational tests should be carried out, how the results of these tests should be presented in the literature, or how mathematical programming algorithms should be properly evaluated and compared. These issues will be addressed in the newsletter.

Dr. JAN TELGEN
RABOBANK NEDERLAND
LAAN VAN EIKENSTEIN 9
3705 AR ZEIST
THE NETHERLANDS

RES

Mr. T. Steihaug,
~~Dept. of Mathematical Sciences,~~ STATOIL, FORUS
~~Rice University,~~ P.O Box 300
~~P.O. Box 1892, HOUSTON, TX 77001,~~ 4001 STAVANGER
~~USA.~~
NORWAY

Mathematical Programming Society

Committee on Algorithms Newsletter

No. 9
September 1983

JAN TELGEN
BOB MEYER
EDITORS

Contents: