

Computational Experience with the Parallel Progressive Hedging Algorithm for Stochastic Linear Programs

Amal de Silva ^{*} David Abramson [†]

December 6, 1994

1 Introduction

Recent advances in computer architecture has started to influence the field of numerical optimisation. Several methods to solve large mathematical programming problems which were tried and given up in the last three decades has gained renewed interest. stochastic optimisation is such area which has a huge potential of providing successful results due to its high parallel content.

In most practical mathematical programming applications the parameters are assumed to be deterministic (i.e the value of the parameters are known before hand). Due inherent randomness of many practical problems the model builder has to be satisfied with a good approximation. Stochastic optimisation tries to alleviate this type difficulty. Consideration of uncertainties dramatically increases the size of resulting mathematical program. When these problems are formulated appropriately the resulting block structure could be exploited for parallization.

Stochastic optimisation with recourse is the most commonly used technique to include randomness of parameters to the problem. (see Ermoliev and Wets [4]). Uncertainty is incorporated into the problem by the use of scenarios. Each realization of random quantities is referred to as a scenarios. The two stage stochastic linear programs with recourse is a class of problems which has been studied extensively.

In the first stage of two stage stochastic optimisation problems decisions have to be made which will provide a good solution without the knowledge of what scenario is going to occur in the second stage. At the second stage when the scenario becomes apparent the solution provides a recourse action which will

^{*}School of Computer and Information Technology, Griffith University, Nathan, QLD 4111.

[†]School of Computer and Information Technology, Griffith University, Nathan, QLD 4111.

restore feasibility at some cost. Most of the methods used are variants of the L-shaped method[11]. This method has been extended to multistage methods by Birge [1].

In this paper a technique known as the progressive hedging method [8] is tested for its parallization properties. This algorithm was successfully applied to portfolio management by Mulvey and Vladimirou.

A quadratic program has to be solved in each iteration of progressive hedging algorithm for each scenario. This can be done in parallel as each subproblem is independent. Mulvey and Vladimirou solve the quadratic program by the use of a nonlinear network optimiser known as [7]. This exploits the generalised network structure of the subproblem. In this study a quadratic primal-dual interior point method is used to solve the subproblems.

Eight problems on portfolio management from Mulvey and Vladimirou are solved on a Silicon Graphics on IRIS 4D/330 with 4 CPU's and a Encore Multimax NS32032 with 20 CPU's.

2 Two Stage Stochastic Linear Programming Problem

The two stage stochastic linear program with recourse can be formulated as shown below. This is referred to as the compact formulation.

$$\begin{array}{llllll}
 \min & & & & & \\
 & cx & + & p_1 q_1 y_1 & + & \dots & + & p_K q_K y_K \\
 & Ax & & & & & & = & b \\
 & T_1 x & + & W_1 y_1 & & & & = & h_1 \\
 & \dots & \dots & \dots & & & & \dots & \dots \\
 & \dots & \dots & \dots & & & & \dots & \dots \\
 & T_K x & & & + & W_K y_K & & = & h_K
 \end{array}$$

where

- $K =$ is the number of scenarios
- $S =$ the set of possible scenarios
- $x =$ is the first stage decision variables which have to determined here and now before any scenario is occurs.
- $y_s =$ the second stage decision (or recourse variable) of scenario $j \quad \forall j \in S$
This provides the recourse action required for each scenario to restore feasibility.

- p_s = the probability of j th scenario
- q_s = the cost associated with second stage variables
- h_s = The random right hand side vector
- A = The coefficient matrix of deterministic constraints and first stage variables.
- T_s = The coefficient matrix of stochastic constraints and first stage variables for scenario s .
- W_s = The coefficient matrix of stochastic constraints and second stage variables for scenario s .

In this formulation there is a different W, T, h for each scenario. Decomposition methods based on the L-Shaped method(also called cutting plane methods) are generally for problems with a fixed W matrix for all the scenarios. (This is known as the two stage problem with fixed recourse). The portfolio management problems tested by us have random W matrices. Thus cutting plane methods are not suitable for these problems.

3 Progressive hedging algorithm

The above two stage stochastic linear program re- formulated by the following method.

A new set of x_j ($s = 1, 2, \dots K$) are substituted for the common x variable along the first column. Then additional constraints such as $x_j - z = 0 \forall j \in S$ S is the set of possible scenarios are included.

This yields the following linear program.

$$\begin{array}{rcllcl}
 p_1\{cx_1 + q_1y_1\} & + & \dots & + & p_K\{cx_K + q_Ky_K\} & & \\
 Ax_1 & & & & & & = b \\
 T_1x_1 & + & W_1y_1 & & & & = h_1 \\
 \dots & & \dots & & \dots & & \dots \\
 Ax_K & & & & & & = b \\
 T_Kx_K & & & & & + W_Ky_K & = h_K \\
 x_1 & - & z & = & 0 & & \\
 x_2 & - & z & = & 0 & & \\
 \dots & & \dots & & \dots & & \dots \\
 x_K & - & z & = & 0 & &
 \end{array}$$

This formulation is known as the split variable formulation. If not for the common constraints $x_j - z = 0$ each scenario could have solved independently

without any trouble. These constraints are referred to as the nonanticipativity constraints.

The progressive hedging algorithm of Rockafellar and Wets [10] iteratively solves a version of the scenario subproblem and progressively enforces the nonanticipativity constraints. The basis of this method comes from duality principles and the proximal point algorithm [9].

The augmented Lagrangian Function after relaxing the nonanticipativity constraints

$$\Phi(X, Y, z, W) \equiv \sum_{s=1}^K p_s \{c_s x_s + q_s y_s + w(s) \cdot (x_s - z) + \rho \|x_s - z\|^2\}$$

- ρ = penalty parameter of the augmented Lagrangian method.
- $W = \{w_1, w_2, \dots, w_K\}$
- w_s = The vector of the dual variables of the nonanticipativity constraints of scenarios s

The steps of the progressive hedging algorithm as given by Mulvey [8]

step 1 set the initial value of the penalty parameter ρ the iteration counter v to 0
set the dual variables $w_s^v = 0 \quad \forall s \in S$

step 2 Solve the following Linear Program for each scenario s
 $\min \{c x_s + q_s y_s \text{ s.t. } A x_s = b, T_s x_s + W_s y_s = h_s, x(s), y(s) \geq 0\}$
This can be done in parallel as each scenario is independent.
Let the solution of each subproblem be x_s^v and y_s^v

step 3 Evaluate an estimate for z by $z^v = \sum_{s \in S} p_s x_s^v$ if $x_s^v = z^v \forall s \in S$ then optimal solution found else goto step 4

step 4 Solve the following Quadratic Program for each scenario s
 $\min \{c x_s + q_s y_s + w_s^v \cdot x_s + \rho \|x_s - z^v\|^2 \text{ s.t. } A x_s = b, T_s x_s + W_s y_s = h_s, x(s), y(s) \geq 0\}$
This also can be done in parallel as each scenario is independent.
Let the solution of each subproblem be x_s^{v+1} and y_s^{v+1}

step 5 Evaluate an estimate for z by
 $z^{v+1} = \sum_{s \in S} p_s x_s^{v+1}$

step 6 Update the dual variable W

$w_s^{v+1} = w_s^v + \rho(x_s^{v+1} - z^{v+1})$ check if convergence criteria $\delta \leq$ convergence tolerance. If not increment iteration counter v and goto step 4.

The convergence criteria is given by the following norm

$$\delta = \{\|z^v - z^{v+1}\|^2 + \sum_{s \in S} p_s \|x_s^v - z^v\|^2\}^{1/2}$$

The Lagrangian function does not separate into independent scenarios due to the term $\|x_s - z_s\|$. The progressive hedging algorithm works by alternatively fixing x and solving z and fixing z and solving x . This decomposes the problem into independent scenarios.

4 Implementation and Parallization of the algorithm

The algorithm was tested on a four processor Silicon Graphics Power series 4D/380 and a twenty processor Encore Multimax NS32032.

Nine problems from Mulvey and Vladimirou [8] were tested. Details of these test problems are given in table 1.

Problem Name	number of Scenarios	number nodes	number of arcs
LP0	18	121	334
LP1	52	121	335
LP2	80	91	248
LP3	72	121	335
LP4	70	61	163
LP5	48	121	335
LP6	40	121	335
LP7	60	121	335
LP8	36	121	335

The penalty parameter for the augmented Lagrangian method ρ is fixed at 0.0015 for all problems. This value is different to the values used by Vladimirou and Mulvey[8]. With this value all the problems converge in a reasonable amount of iterations. It has been noted that the behaviour of the progressive hedging algorithm is extremely sensitive to the value of ρ [3]. As the objective of this study was to evaluate the parallel performance of the progressive hedging algorithm a constant ρ was used. Most problems could be solved faster by properly tuning the value of ρ .

The linear and quadratic programs of step 2 and 4 was solved by the use a primal- dual interior point method. This was based on the Mehrotra's predictor-corrector interior point method. [6, 2] In each iteration of the interior point method a set of linear equation $Ax = b$ is solved. A is a symmetric positive

definite matrix. Thus Cholesky factorisation method is used to solve this set of equations as this is the most efficient method of solving such a system. The bulk of the time in each call to the interior point method is consumed in a Cholesky factorisation. The Cholesky factorisation is based on the algorithm by George and Liu [5]. Sparse data structures are used in the storage of each problem and the A matrix.

The portions of the nonzeros of each subproblem are identical for each scenario as well as for each iteration of the interior point method. Due to this reason a symbolic factorisation is done once only. In the symbolic factorisation an data structure is created to hold the A matrix. This data structure is copied for each subproblem as the nonzero pattern will be identical.

The major computational effort of this algorithm is taken in solving the subproblems (One for each scenario) at step two and step four. In step two a linear program is solved while in step four a quadratic program is solved. Initially the minimum row ordering and the symbolic factorisation is done by the parent process sequentially. The minimum row ordering permutes the A matrix such that creation of nonzeros while factorisation is minimised. A heuristic is used for this purpose [5]. Also the data structures for each subproblem and the working memory is created at this stage. This shared memory is dynamically allocated. Then a number of child processes are spawned depending on the number of CPU's available. Each CPU will solve each subproblem concurrently. Also step 3 , 5, 6 involves independent loops with a size equal to the number of first stage variables. These loops are also parallized. All the sequential code code is carried out by the parent process. The number of scenarios and the number of first stage variables are divided into nearly equal chunks, which are processed concurrently by different CPU's.

5 Results

problem name	optimal objective value	objective value obtained	number of CPU's	time (sec)	speed up
LP0	-2.045920	-2.045758	1	224	1.00
"	"	"	2	117	1.91
"	"	"	3	81	2.76
"	"	"	4	63	3.55
LP1	-2.557564	-2.557351	1	885	1.00
"	"	"	2	493	1.79
"	"	"	3	335	2.64
"	"	"	4	256	3.46
LP2	-1.829869	-1.829877	1	1276	1.00
"	"	"	2	662	1.93
"	"	"	3	436	2.93
"	"	"	4	333	3.83
LP3	-2.107791	-2.107654	1	1540	1.00
"	"	"	2	833	1.89
"	"	"	3	537	2.87
"	"	"	4	405	3.80
LP4	-1.428035	-1.426476	1	301	1.00
"	"	"	2	154	1.94
"	"	"	3	117	2.57
"	"	"	4	121	2.48
LP5	-2.261224	-2.270544	1	926	1.00
"	"	"	2	501	1.84
"	"	"	3	358	2.58
"	"	"	4	238	3.89
LP6	-2.307987	-2.309094	1	1128	1.00
"	"	"	2	582	1.93
"	"	"	3	437	2.58
"	"	"	4	345	3.26
LP7	-2.228568	-2.228186	1	2129	1.00
"	"	"	2	1144	1.86
"	"	"	3	824	2.58
"	"	"	4	589	3.61
LP8	-2.794296	-2.228186	1	902	1.00 *
"	"	"	2	412	2.18 *
"	"	"	3	310	2.98 *
"	"	"	4	227	3.97 *

Table 2. Results on the Silicon Graphics

problem name	optimal objective value	objective value obtained	number of CPU's	time (sec)	speed up
LP0	-2.045920	-2.045780	1	5604	1.00
"	"	"	2	3306	1.69
"	"	"	3	2287	2.45
"	"	"	6	900	6.22 *
"	"	"	18	394	14.2
LP1	-2.557564	-2.557368	1	17134	1.00
"	"	"	2	8727	1.96
"	"	"	4	4452	3.84
"	"	"	13	1635	10.47
LP2	-1.829869	-1.829876	1	24272	1.00
"	"	"	2	12711	1.91
"	"	"	4	6463	3.75
"	"	"	8	3862	6.28
"	"	"	10	2806	8.65
"	"	"	16	1824	13.30
LP3	-2.107791	-2.107663	1	29847	1.00
"	"	"	2	14835	2.01*
"	"	"	3	9609	3.11*
"	"	"	4	7587	3.933*
"	"	"	8	3723	8.01*
"	"	"	12	2809	10.625*
"	"	"	18	1886	15.826*
LP4	-1.428035	-1.427526	1	16150	1.00
"	"	"	2	9057	1.78
"	"	"	5	3414	4.73
"	"	"	7	2611	6.19
"	"	"	10	1882	8.58
"	"	"	14	1320	12.23

Table 3. Results on Encore Multimax**

problem name	optimal objective value	objective value obtained	number of CPU's	time (sec)	speed up
LP5	-2.261224	-2.270516	1	17742	1.00
"	"	"	2	9770	1.81
"	"	"	3	6834	2.59
"	"	"	4	4628	3.83
"	"	"	6	3292	5.38
"	"	"	12	1658	10.7
"	"	"	18	1406	12.61
LP6	-2.307987	-2.309094	1	21904	1.00*
"	"	"	2	9417	2.32*
"	"	"	3	6316	3.47*
"	"	"	4	5613	3.92*
"	"	"	8	2654	8.25*
"	"	"	9	2323	9.49*
"	"	"	12	2137	10.25*
"	"	"	18	1515	14.45*
LP7	-2.228568	-2.228116	1	38366	1.00
"	"	"	2	19618	1.95
"	"	"	3	13432	2.85
"	"	"	4	11403	3.36
"	"	"	9	5050	7.59
"	"	"	12	3924	9.77
"	"	"	15	3391	11.31

Table 3. cont. Results of Encore Multimax contd.**

* Some problems exhibit a super-linear speed up. This is due addition of some shared variables in a non deterministic manner. This leads to small differences of the convergence tolerance with different number of processors. And thus the number of iterations with the number of processors. Thus this makes it difficult to make comparisons in the speedup.

** The number of CPU's was always taken such that number of scenarios was a multiple of the number of CPU's. This balanced the load on the CPU's.

6 Conclusion

The progressive hedging algorithm was tested for its parallization properties on two shared memory parallel machines (i.e. Silicon Graphics IRIS 4D/330 and Encore Multimax NS32032). This machines exhibits very good speed up for the portfolio management problems tested. The Encore Multimax is extremely slow

compared with the Silicon Graphics. But this allowed us to test the algorithm on up to 18 processors.

The biggest weakness of the progressive hedging algorithm is that it is very sensitive to the penalty parameter. This requires the user to determine a good value for this by experimentation.

The authors are hoping to develop a parallel algorithm to solve large stochastic linear programs by the use of interior point methods. It is hoped to parallelize the interior point method. As these problems exhibit a block structure it is hoped to exploit this property.

References

- [1] J.R. Birge. Decomposition and partitioning methods for multistage stochastic linear programs. *Oper. Res.*, 33:989, 1985.
- [2] T.J. Carpenter, I.J. Lustig, R.E. Marsten, and D.F. Shanno. A primal-dual interior point method for convex separable nonlinear programs. Research Report SOR 90-2, Dept. of civil Engineering and operations research, Princeton University, 1990.
- [3] B. J. Chun and S.M. Robinson. Scenario analysis via bundle decomposition. Technical report, Dept of Industrial Engineering, University of Wisconsin, Madison, 1513 University av., Madison, WI 53706-1572, 1992.
- [4] Y. Ermoliev and R.G.B. Wets. *Numerical Techniques for Stochastic Optimization*. Springer, Berlin, 1988.
- [5] A. George and J.W. Liu. *Computer Solutions to Large Sparse Positive Definite systems*. Prentice-Hall Inc, 1981.
- [6] I.J. Lustig, R.E. Marsten, and D.F. Shanno. On implementing mehrotra's predictor-corrector interior point method for linear programming. *SIAM J. Optimization*, 2:3:435–449, 1992.
- [7] J. M. Mulvey and S.A. Zenios. A generalised network optimization system. Working paper 87-12-03, Wharton School University of Pennsylvania, 1987.
- [8] J.M. Mulvey and H. Vladimirov. Applying the progressive hedging algorithm to stochastic generalised networks. *Ann. Oper. Res.*, 31:399–424, 1991.
- [9] R. T. Rockafellar. Augmented lagrangians and application of proximal point algorithms. *Math. Oper. Res.*, 1:2:97–116, 1976.
- [10] R. T. Rockafellar and R.J.B. Wets. Scenarios and policy aggregation in optimization under uncertainty. *Math. Oper. Res.*, 16:1:119–147, 1989.

- [11] R. Van Slyke and R.J.B. Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM J. Appl. Math.*, 17:638–663, 1968.