Split-and-Merge Method for Accelerating Convergence of Stochastic Linear Programs

Akhil Langer[1] and Udatta Palekar[2]
[1]*Department of Computer Science, University of Illinois at Urbana-Champaign, Illinois, USA*
[2]*Department of Business Administration, University of Illinois at Urbana-Champaign, Illinois, USA*
{*alanger, palekar*}*@illinois.edu*

Keywords: Stochastic Optimization, Decomposition, Scenario-based Decomposition, Multicut L-shaped Method, Resource Allocation, US Military Aircraft Allocation

Abstract: Stochastic program optimizations are computationally very expensive, especially when the number of scenarios are large. Complexity of the focal application, and the slow convergence rate add to its computational complexity. We propose a split-and-merge (SAM) method for accelerating the convergence of stochastic linear programs. SAM splits the original problem into subproblems, and utilizes the dual constraints from the subproblems to accelerate the convergence of the original problem. Our initial results are very encouraging, giving up to 58% reduction in the optimization time. In this paper we discuss the initial results, the ongoing and the future work.

1 INTRODUCTION

Stochastic programs are used for decision making under uncertainty. For example, product production decisions under resource constraints are to be taken at a time when future demand for the products is not known with certainty. Stochastic programs assume that a probabilistic distribution of these uncertain future outcomes is known. An instantiation of the uncertain parameters is called as a scenario. Equation 1 gives a standard representation of a stochastic program.

$$\begin{aligned} \min \quad & cx + \sum_s p_s(q_s y_s) \\ s.t. \quad & Ax \leq b \\ \forall s, \quad & W_s y_s + T_s x \leq h_s \end{aligned} \tag{1}$$

where, x corresponds to the strategic decisions corresponding to the known parameters that are to be taken now, and y_s corresponds to the operational decisions that will be taken when the scenario s is realized, and p_s is the probability that scenario s will occur. The objective function is sum of the costs of strategic decisions and the weighted average of the cost of operational decisions for all scenarios.

When the unknown parameters become known in multiple stages over a period of time, the corresponding optimization problem is called a multistage stochastic program. In a two-stage stochastic program, all the unknown parameters become known at the same time. We propose our work on two-stage stochastic programs but the strategy is easily generalizable to multi-stage stochastic programs. Moreover, multistage stochastic programs can be solved as a sequence of two-stage stochastic programs.

Equation 2 and 3 shows the first and second stage programs of the two-stage stochastic program, respectively.

Stage 1 Program:

$$\begin{aligned} \min \quad & cx + \sum_s p_s Q_s(x, y_s) \\ s.t. \quad & Ax \leq b \end{aligned} \tag{2}$$

Stage 2 Program:

$$\begin{aligned} \min \quad & Q_s(x, y_s) \\ & W_s y_s \leq h_s - T_s x \end{aligned} \tag{3}$$

In this work, we focus on the stochastic linear programs, that is, problems that have linear variables and constraints both in Stage 1 and Stage 2.

The usual method of solving stochastic linear program uses Bender's decomposition (Benders, 1962). In this method, a candidate Stage 1 solution is obtained by optimizing the Stage 1 program. The candidate Stage 1 solution is evaluated against all the scenarios in Stage 2. Stage 2 optimization gives the scenario costs for the given Stage 1 solution, and optimality/feasibility cuts that are fed back to Stage 1.

Stage 1 is re-optimized with the addition of new set of cuts to obtain another candidate solution. The iterative Stage 1-Stage 2 optimization continues until the optimal solution is found which is determined by a convergence criteria. A Stage 1 optimization followed by a Stage 2 optimization is called an iteration.

There are two variants of the Bender's approach. In one, a single cut using a weighted combination of Stage 2 dual objective function is added to the Stage 1 in each iteration. This method is called the L-shaped method (Van Slyke and Wets, 1969). In the other, in an iteration a cut constraint is added to Stage 1 for each scenario. This multicut method (Birge and Louveaux, 1988) has the advantage that the set of cuts in each iteration dominates a single L-shaped cut. However, the number of cuts can become very large quickly, particularly for problems with large number of scenarios.

In most real world applications, the number of uncertain parameters are large, and therefore the number of scenarios are also very large. In addition to the complexity of the focal application, factors such as the number of Stage 2 evaluations, number of rounds it takes to converge to optimality (within the user-specified convergence criteria), and the size of the Stage 1 linear program which increases with the increase in number of scenarios, add to the computational complexity of the stochastic programs.

Our research focuses on multicut Bender's method. Equation 4 shows the Stage 1 program after r rounds.

$$\begin{aligned} \min \quad & cx + \sum_s p_s \theta_s \\ s.t. \quad & Ax \leq B \\ \forall s \text{ and } l \in [1,r], \quad & E_{sl}x + \theta_s \leq e_{sl} \end{aligned} \tag{4}$$

where, $E_{sl} + \theta_s \leq e_{sl}$ are the cut constraints obtained from Stage 2 optimization and θ_s is the cost of scenario s.

Because of the large number of cuts in the multicut method, it is imperative that the cuts generated in each round are strong cuts in the sense that they allow the Bender's program to converge quickly. This paper considers a scenario split-and-merge approach to accelerate the convergence of multicut Bender's method. In Section 2, we do a literature review on stochastic optimization methods and their convergence properties. In Section 3, we propose a split-and-merge method for accelerating the convergence of multicut L-shaped method. We corroborate our ideas with some initial results in Section 4. Finally, we conclude the paper with the ongoing work and future work in Section 5.

2 RELATED WORK

Magnanti and Wong in their seminal paper (Magnanti and Wong, 1981), proposed a method for accelerating Bender's decomposition by selecting good cuts to add to the master problem. A cut $\theta \leq \pi_1^* h + \pi_1^* Tx$ dominates or is stronger than the cut, $\theta \leq \pi_2^* h + \pi_2^* Tx$, if $\pi_1^* h + \pi_1^* Tx \leq \pi_2^* h + \pi_2^* Tx$ for all $x \in X$ with a strict inequality for at least one $x \in X$, where π_1^* and π_2^* are any two dual optimal solutions of the degenerate Stage 2 problem. They define a cut as pareto optimal if it has no dominating cut. The corresponding Stage 2 dual optimal solution is called the pareto optimal solution. Given the set of Stage 2 dual optimal solution set $S(x^*)$, the pareto optimal solution (π^p) solves the problem:

$$\min_{\pi \in S(x^*)} \pi h + \pi T x^c$$

where, x^c is a core point of X i.e. $x^c \in$ relative interior of X and $S(x^*) = \{\pi | \pi \text{ maximizes } Q(x^*)\}$. The downside of this approach is that it requires solving additional optimization problem to identify pareto optimal cuts in every iteration which can trade-off the benefit of reduction in total number of iterations.

Linderoth et al (Linderoth and Wright, 2003) developed asynchronous algorithms for stochastic optimization on computational grids. They use a multicut method and add a cut of a particular scenario to the master program only if it changes the objective value of the proposed model function corresponding to that scenario. This requires solving several additional optimization problems at each iteration to determine the usability of each cut, which can be prohibitive.

Initial iterations in the multicut method are often inefficient because the solution tends to oscillate between different feasible regions of the solution space. Ruszczyński (Ruszczyński, 1986) proposed a regularized decomposition method that adds a quadratic penalty term to the objective function to minimize the movement of the candidate solution. Linderoth and Wright (Linderoth and Wright, 2003) use a linearized approach to this idea by binding the solution in a box called the trust region. Trust region method is used to decide the major iterates that significantly change the value of the objective function in each iteration. This requires doing several minor iterations at each major iteration to come-up with a good candidate solution x^k. Trust-region method at minor iterations limits the step-size by adding constraints of the form $||x - x^k||_\infty \leq \Delta$. Heuristics are used to decide and update Δ. The cuts generated during the minor iterations can be discarded without affecting the convergence of the problem.

The Progressive Hedging algorithm proposed by Rockafellar and Wets (Rockafellar and Wets, 1991) solves each scenario independently by introducing lagrangean multipliers for the Stage 1 variables in the objective function of the individual problems. This approach requires search for the optimal lagrangean multipliers which can be computationally prohibitive.

Langer et al (Langer et al., 2012) propose clustering schemes for solving similar scenarios in succession that significantly reduces the Stage 2 scenario optimization times by use of advanced/warm start. However, this does not address the slow convergence rate of the problem. Depth and breadth of applications of stochastic optimization can be found in (Gassmann et al., 2013). Other stochastic program decomposition studies can be found in (Li et al., 2012; Becker, 2011; Guan and Philpott, 2011).

3 PROPOSED APPROACH

In each iteration of the multicut method, as many cut constraints are added to the Stage 1 program as there are scenarios. In the initial iterations of the multicut Bender's method, all the scenario cut constraints are not active in the Stage 1 linear program optimization. This is because few cuts are needed to perturb the previous Stage 1 solution and provide a new candidate solution. Therefore, the cuts from the Stage 2 evaluation of most of the scenarios remain inactive in Stage 1 during the initial iterations of the Bender's method. For such scenarios, similar cuts will be generated in successive iterations, and hence a lot of computation is wasted.

We propose a split-and-merge (SAM) algorithm (Algorithm 1) that divides the scenarios into N clusters ($S_1, S_2, ..., S_n$).

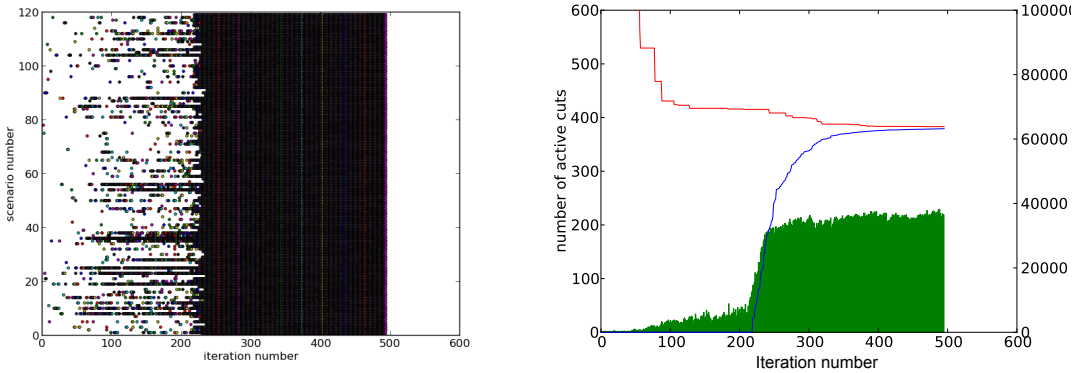
In SAM, n stochastic programs ($P_1, P_2, ..., P_n$) are created and each of these is assigned one cluster of scenarios (lines 3-4). Probabilities of the scenarios in each of these subproblems are scaled up so that they add up to 1 (lines 5-6). We then apply the Bender's multicut method to these n stochastic programs independently of each other (lines 8-16). For multicore machines, these problems can be solved in parallel on multiple cores by adding a simple OpenMP (Dagum and Menon, 1998) parallel construct to parallelize the for loop that iterates over each of these subproblems (line 8). Bender's decomposition is applied to these subproblems for a fixed number of rounds (r) or till the subproblem has converged to optimality, whichever is the earliest (line 12). Once this criteria has been met for all the subproblems, the cut constraints from these problems are collected (lines 21-22). The cuts from subproblems are also valid for the original problem with all the scenarios. These cuts are used as the initial set of cut constraints for applying the multicut Bender's method to the original stochastic linear program.

Algorithm 1 Split-and-Merge (SAM)

```
1  Input: S (set of scenarios), Original Stochastic Program (P)
2  Divide S into n clusters, S_1, S_2, ...., S_n
3  Generate n stochastic programs, P_1, P_2, ...., P_n, with
4   scenarios from S_1, S_2, ...., S_n, respectively
5  Scale scenario probabilities in each of these subproblems
6   such that they sum up to 1
7
8  #pragma omp parallel for
9  for i in range(1,n):
10   scosts_i = [] #scenario costs
11   cuts_i = [] #scenarios cut constraints
12   while r_i < r or hasConverged(i):
13     x_i = solveStage1(P_i, scosts_i, cuts_i)
14     scosts_i, cuts_i = solveStage2(x_i)
15     r_i=r_i + 1
16   end while
17
18 #wait until all the subproblems have returned
19 cuts = []
20 scosts = []
21 for i in range(1,n):
22   cuts.add(getCutConstraints(P_i))
23
24 #now solve the original problem
25 while not hasConverged(P):
26   x = solveStage1(P, scosts, cuts)
27   scosts, cuts = solveStage2(x)
28 end while
```

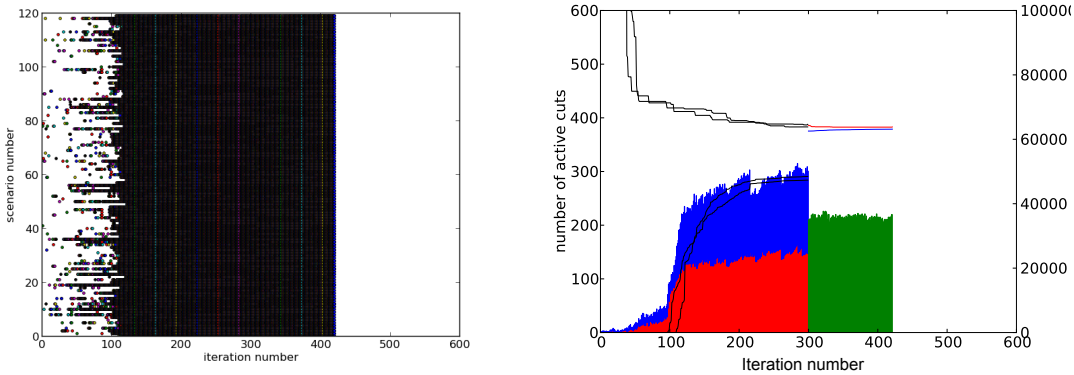
There are several benefits of this approach. The chances of a scenario having active cuts is higher in the subproblems because of the smaller number of scenarios present in the subproblems. Scenario cut activity helps in generating newer and different cuts for those scenarios, and thus doing more useful work, as compared to the original problem in which most of the scenarios remain inactive in the initial iterations.

Stage 1 optimization is often a serial bottleneck in Bender's decomposition, especially when the number of scenarios is large. In the decomposition approach, the number of scenarios per subproblem are much smaller than the original problem, which speeds up the Stage 1 optimization and thus the candidate solutions for Stage 2 evaluation become available much earlier. Additionally, this also gives an opportunity to have parallelism in Stage 1, in addition to the obvious Stage 2 parallelization available in stochastic linear programs. These subproblems being independent of each other, can be optimized in parallel in Stage 1.



(a) Scenario Activity (b) Cut Activity

Figure 1: Multicut Benders Method. Total Iterations = 495, Time to Solution = 1190s



(a) Scenario Activity (b) Cut Activity

Figure 2: SAM with decomposition into 2 subproblems for 300 iterations. Total Iterations = 415, Time to Solution = 784s

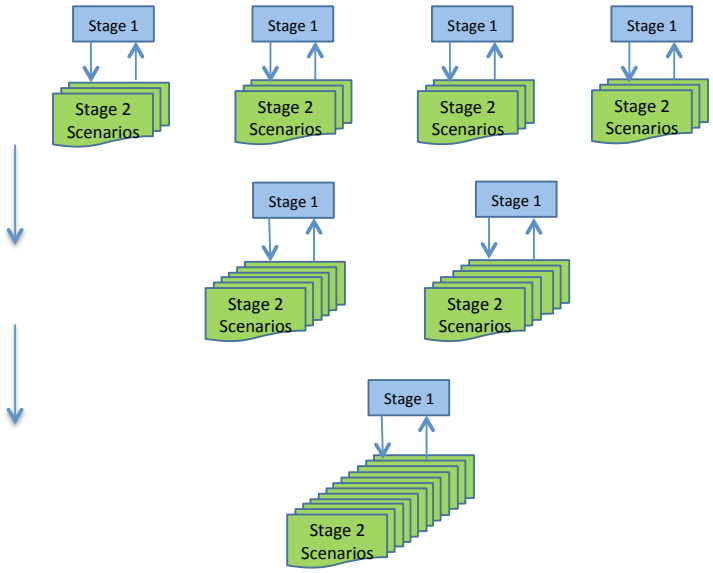


Figure 3: Hierarchical Scenario Decomposition Scheme

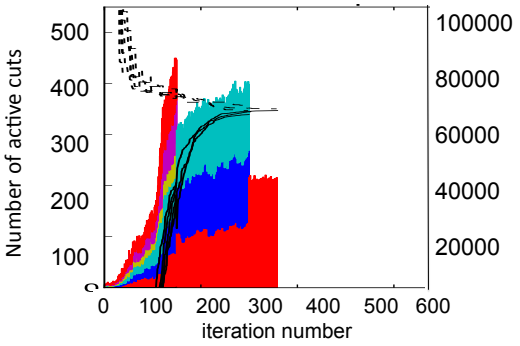


Figure 4: SAHM with decomposition into 6 subproblems for 150 iterations followed by 3 subproblems for 150 iterations. Total Iterations = 360, Time to Solution = 507s

4 INITIAL RESULTS

Our initial experiments of the proposed SAM approach are based on a military aircraft allocation problem. The U.S. Tanker Airlift Control Center (TACC) has to allocate aircraft to various missions and wings one month in advance to the actual scheduling of those aircraft. The demands of these missions and wings are not known in advance and are subject to enormous uncertainty even during peacetime. We formulated the problem as a two-stage stochastic program in which aircraft are allocated to different wings and missions in Stage 1. In Stage 2 these allocations are evaluated by scheduling these aircraft for various missions once the demands are realized in a scenario. More details of the problem and its stochastic formulation can be found in (Langer, 2011; Baker et al., 2012).

For our experiments, we took a problem that allocates aircraft for a period of 10 days. We consider 120 scenarios. For this problem, Stage 1 has 270 variables and 180 constraints, while Stage 2 has 25573 variables and 16572 constraints per scenario. All the experiments were done on the same number of cores, on a machine with Intel HP X5650 2.66Ghz 6C Processor.

In Figure 1(a), we show the scenarios that have active cuts in Stage 1 in each iteration of the Bender's multicut method. The x-axis is the iteration number, and y-axis is the scenario number. In the vertical line corresponding to any iteration number, a dot in the horizontal line corresponding to a scenario number means that a cut obtained from the Stage 2 optimization of that scenario was active in that iteration. As can be seen in the figure (Figure 1(a)), very few scenarios have active cuts in the initial few rounds. As the optimization progresses, the number of scenarios with active cuts increases with the increase in the iteration number. And eventually, after approximately 220 iterations, all the scenarios have active cuts in Stage 1. The total number of active cuts in each iteration are shown in Figure 1(b). The upper line shows the upper bound, and the lower line shows the lower bound as the number of iterations increase.

For testing the proposed SAM algorithm, we divided the original problem with 120 scenarios into two subproblems each with 60 scenarios. The subproblems are solved for a maximum of 300 rounds, after which the cut constraints are collected from both of them and these cut constraints are used as the initial set of constraints for solving the original problem with 120 scenarios. Figure 2(a) shows the scenario activity for this method. As can be seen in the figure, the overall scenario activity is much higher in the initial iterations of the SAM approach than in the original Bender's method. Figure 2(b) shows the number of cuts that were active in each of the subproblems. The two bar shades correspond to the two subproblems. The bars are stacked on top of each other to show the total number of active cuts in both the subproblems. Bars after iteration 300 show the number of active cuts for the original problem (P), which begins optimization at iteration 301. As in Figure 1(b), the lower and upper lines correspond to the lower and upper bounds, respectively - initially of the subproblems, and then of the original problem. Total time to optimization is 784 seconds with the SAM approach as compared to 1190 seconds with the original Bender's method.

We have extended our algorithm to split-and-hierarchical-merge (SAHM) algorithm, in which the merging of the subproblems into the original problem is done in stages instead of at once as in the SAM algorithm. Figure 3 shows a schematic diagram of the SAHM approach. We tested the SAHM approach by dividing the original problem into 6 subproblems each with 20 scenarios. In the first stage, a set of 2 subproblems combine to form one subproblem, giving a total of 3 subproblems. In the second stage, these three subproblems are combined into the original problem. Each of these stages is executed for 150 rounds, after which optimization of the original problem begins. Figure 4 shows the cut activity for this setup. Various shades correspond to different subproblems. Total time to solution using SAHM was 507 seconds, giving us an improvement of 58% over the Bender's method.

5 FUTURE WORK

We plan to evaluate and extend the proposed scenario decomposition schemes in the following ways:

- Try decomposing the problem into different number of subproblems, and determine the optimal subproblem size.
- Exhaustively study the SAHM scheme.
- Currently, the number of rounds for which the subproblems are executed before they are merged is hard-coded by the user/programmer. An important milestone is to dynamically determine during the execution of the program, the optimal time to merge the subproblems into the original problem. This could be based on the cut activity of the subproblems.
- Use distributed computing to solve the stochastic linear programs. As discussed in Section 3 scenario decomposition gives us Stage 1 parallelism in addition to the Stage 2 parallelism already available in stochastic linear programs. This can be used to increase the parallel efficiency of stochastic linear programs.

REFERENCES

Baker, S., Palekar, U., Gupta, G., Kale, L., Langer, A., Surina, M., and Venkataraman, R. (2012). Parallel Computing for DoD Airlift Allocation. MITRE Technical Report, 2012. `www.mitre.org/work/tech_papers/2012/11_5412/`.

Becker, A. B. D. (2011). *Decomposition Methods for Large scale Stochastic and Robust Optimization Problems*. PhD thesis, Massachusetts Institute of Technology.

Benders, J. (1962). Partitioning Procedures for Solving Mixed Variables Programming Problems. *Numerische Mathematik 4*, pages 238–252.

Birge, J. and Louveaux, F. (1988). A Multicut Algorithm for Two-stage Stochastic Linear Programs. *European Journal of Operational Research*, 34(3):384–392.

Dagum, L. and Menon, R. (1998). OpenMP: An Industry-Standard API for Shared-Memory Programming. *IEEE Computational Science & Engineering*, 5(1).

Gassmann, H., Wallace, S. W., and Ziemba, W. T. (2013). *Stochastic Programming: Applications in Finance, Energy, Planning and Logistics*, volume 4. World Scientific.

Guan, Z. and Philpott, A. (2011). A Multistage Stochastic Programming Model for the New Zealand Dairy Industry. *International Journal of Production Economics*, 134(2):289 – 299. Robust Supply Chain Management.

Langer, A. (2011). Enabling Massive Parallelism for Two-Stage Stochastic Integer Optimizations: A Branch and Bound Based Approach. Master's thesis, Department of Computer Science, University of Illinois. http://hdl.handle.net/2142/29700.

Langer, A., Venkataraman, R., Palekar, U., Kale, L. V., and Baker, S. (2012). Performance Optimization of a Parallel, Two Stage Stochastic Linear Program: The Military Aircraft Allocation Problem. In *Proceedings of the 18th International Conference on Parallel and Distributed Systems (ICPADS 2012)*, Singapore.

Li, X., Tomasgard, A., and Barton, P. (2012). Decomposition Strategy for the Stochastic Pooling Problem. *Journal of Global Optimization*, 54(4):765–790.

Linderoth, J. and Wright, S. (2003). Decomposition algorithms for stochastic programming on a computational grid. *Computational Optimization and Applications*, 24(2):207–250.

Magnanti, T. L. and Wong, R. T. (1981). Accelerating Benders Decomposition: Algorithmic Enhancement and Model Selection Criteria. *Operations Research*, 29(3):464–484.

Rockafellar, R. T. and Wets, R. J.-B. (1991). Scenarios and Policy Aggregation in Optimization Under Uncertainty. *Mathematics of operations research*, 16(1):119–147.

Ruszczyński, A. (1986). A regularized decomposition method for minimizing a sum of polyhedral functions. *Mathematical programming*, 35(3):309–333.

Van Slyke, R. M. and Wets, R. (1969). L-shaped Linear Programs with Applications to Optimal Control and Stochastic Programming. *SIAM Journal on Applied Mathematics*, 17(4):638–663.