EMPIRICAL ANALYSIS OF SCENARIO CLUSTERING IN BENDERS DECOMPOSITION

A PREPRINT

Nakul Upadhya
Department of Mechanical and Industrial Engineering
University of Toronto
Toronto, ON, M5S 2E4
nakul.upadhya@mail.utoronto.ca

Zhijian Ling
Department of Mechanical and Industrial Engineering
University of Toronto
Toronto, ON, M5S 2E4
zhijian.ling@mail.utoronto.ca

ABSTRACT

Considering the uncertain variables involved in practical scenarios, stochastic programming is a method to tackle optimization problems. Even if Sample Average Approximation is applied to solve the two-stage Stochastic programming (2SP) deterministically, it is not computationally efficient to solve the problem directly. Hence, Benders decomposition is introduced to solve the problems, which divides the 2SP into a master problem and one subproblem for each scenario. Through passing the candidate decision into subproblems, cuts are generated to either constrain the surrogate variables within certain values or to bound the first-stage decisions depending on feasibilty in the subproblems. In this paper, we cluster all scenarios into several groups using different clustering techniques. We compare the performance of the hybrid method and the scenario reduction method using different clustering techniques. Our results show that Affinity Propagation clustering provides performance improvements for problems with complicated uncertainty distributions.

Keywords Stochastic Programming · Benders Decomposition · Machine Learning

1 Introduction

While traditional optimization methods are flexible and conform to a wide variety of problems, many of them may contain data element that are best described by random variables and unknown elements. In these cases, stochastic programming (SP) is a powerful modeling framework that can help decision makers account for these probabilistic elements in their formulations.

One modeling paradigm with stochastic programming is the Two-Stage Stochastic program (2SP), often called a recourse model. These involve two sets of decisions, ones made here-and-now (first-stage decisions) and recourse decisions made after the uncertainty is realized (second-stage decisions). These programs often aim to optimize the value of the first stage decision along with the expected value of the second stage decision (though other summarising measures can be adopted as well).

Sample Average Approximation (SAA) is often used when creating two-stage stochastic programs to create a finite number of realizations of the random variables involved in a SP (Kleywegt et al., 2002). These realizations (scenarios) can then be leveraged to deterministically solve the stochastic program. SAA is not without its downsides however. As the number of scenarios introduced increases, the computational load of solving the SP increases tremendously as well as each realization introduces a new copy of the second-stage decision variables and any constraints involving these variables. To offset this load, decomposition algorithms are often used.

One such decomposition algorithm that can be used for two-stage linear stochastic programs is Benders Decomposition (Benders, 1962). Benders Decomposition divides the 2SP model into a master problem and one subproblem for each scenario. The main problem contains the first-stage decisions as well as a surrogate variable representing the second-stage objective value for each scenario. Through passing the candidate decision into subproblems, cuts are generated to either constrain the surrogate variables within certain values or to bound the first-stage decisions depending on feasibilty in the subproblems. For minimization problem, the master problem presents the lower bound. When more constraints are added to the master problem, the lower bound will monotonically increase and finally coverage to the upper bound given by solving the dual subproblems.

Although Benders composition can significantly decrease the amount of time required to solve a 2SP (Zverovich et al., 2012), the master problem can get more complex as the algorithm runs due to more and more constraints being added to it. To tackle this problem, variations of Benders have been proposed to limit the number of cuts per iteration, namely single-cut Benders where there is only one surrogate variable that represents the expected value of all scenarios (Van Slyke and Wets, 1969) and hybrid-cut Benders where there is a surrogate variable assigned to groups of scenarios (Sohn et al., 2012). Additionally, methods such as scenario reduction techniques are often used to further improve the run-time of said algorithms.

One method for grouping scenarios that has shown success in prior literature (Khatami et al., 2015; Li et al., 2018; Zhou et al., 2019) is leveraging clustering algorithms. By leveraging data driven methods of scenario grouping, the resultant groups or representative scenarios may provide more information.

In this paper, we aim to compare various clustering methods and the performance improvement they provide to solving two-stage stochastic problems using Benders Decomposition. More specifically:

1. We aim to provide insights into whether the clustering algorithm chosen has a significant impact on the runtime of a Hybrid-cut Benders approach.
2. We aim to compare and contrast the capabilities of various clustering algorithms to intelligently subset scenarios.

All experiments can be found at https://github.com/upadhyan/Clustering-in-Benders.

2 Related Work

The two-stage stochastic programming(2SP) is widely used to solve the optimization problem with uncertainty. The general objective function for 2SP,

$$\begin{aligned}&\underset{x\in\mathcal{X}}{\text{minimize}}\ c^T x + \mathbb{E}_\xi[Q(x,\xi)]\\ &\text{subject to } Ax \le b\end{aligned} \tag{1}$$

$$\begin{aligned}&\underset{x,y}{\text{minimize}}\ c^T x + \sum_{s=1}^{S} p_s(q^s)y^s\\ &\text{subject to } Ax \le b\end{aligned} \tag{2}$$

$$\begin{aligned}Q(x,\xi) = {}&\text{minimize}_y\ q(\xi)^T y\\ &\text{subject to } W(\xi)y \le h(\xi) - T(\xi)x\end{aligned} \tag{3}$$

,consists of two parts, here-and-now decisions(1) and recourse decisions(3). In the first stage, we assume the parameter is fixed. While scenarios are various and follow some distribution, recourse decisions take the scenarios into consideration to meet the requirements for cases and optimize the problem to obtain optimal solutions. However, directly solving expectation problem in stochastic programming is usually intractable. Hence, we apply SAA, which decomposes the expectation into several scenarios and solves the deterministic optimization instead. The typical way to sample the scenarios is Monte Carlo sampling, which randomly selects k scenarios following the given distribution with equal probability $\frac{1}{k}$.

As aforementioned, 2SP is a mathematical optimization to deal with uncertainty scenarios. Therefore, 2SP is widely applied in various fields, including disaster management(Grass and Fischer, 2016),resource distribution(Tajeddini et al., 2014)(Huang and Loucks, 2000)and finance. In disaster management, 2SP can be used for transportation management for first-aid commodities for earthquake(Barbarosoğlu and Arda, 2004)(Noyan, 2012). Barbarosoğlu and Arda build a 2-stage stochastic programming for the pre-disaster phase to minimize the loss of life and maximize the efficiency of operation considering uncertain variables, such as the magnitude of the earthquake and impact.

Benders decomposition, proposed by Benders (1962), is a technique to efficiently solve large-scale mixed-integer linear programming(MILP). It has been proved that Benders decomposition is successful in a wide range of difficult practical applications, including power system(Canto, 2008)(Shahidehopour and Fu, 2005), traffic netwrok(Fontaine and Minner, 2014), and facility location(Fischetti et al., 2016).

Machine learning has brought some improvement in Benders Decomposition algorithm. Lee et al. (2020) uses cut optimality indicator, cut violation, cut repeat, cut depth, and cut orders as input for classification prediction to check whether the cut is useful. Apart from classification, clustering is a typical approach to reducing scenarios. Khatami et al. (2015) design a novel reverse supply chain network and utilize Clolesky's factorization method to extract information from demands to build the probability distribution function for scenarios. To maintain the balance of accuracy and time efficiency, the paper applies the clustering technique to reduce 1000 scenarios to 100 scenarios with a time improvement of 85% and an objective error of 0.444%.

However, determining the optimal cluster number is challenging. A state-of-art algorithm called density peaks clustering is proposed by Li et al. (2018). Instead of manually selecting the cluster number, the idea of density peak clustering is to pick the scenarios group with high local or global density as a center, and distribute the remaining scenarios to these centers. The algorithm only takes less than 1% of the time CPLEX takes when it is tested on chance-constrained transmission expansion planning. Similarly, (Zhou et al., 2019) apply affinity prorogation (Frey and Dueck, 2007), which iteratively updates exemplar for each data point depending on their similarity matrix, to tackle the cluster number selection problem.

Considering different candidate decisions raising different violations, Vandenbussche et al. (2019) proposed a dynamic clustering algorithm for hybrid Benders Decomposition, which clusters dual variables during each iteration. The cut added to the MP is given by the outer parallelization of the Benders algorithm, which determines the subsets used to generate the cuts. Instead of clustering directly on scenario variables, Keutchayan et al. (2021) reduce the scenarios by evaluating possible objective values for each scenario, which gives a relatively accurate result compared to the typical scenario reduction methods.

3 Methodology

To evaluate the performance improvements that may result from clustering scenarios, we will first generate a large set of synthetic problem instances, varying the number of variables and scenarios in each instance. We will then run various modifications to the Benders algorithm such as varying the clustering methods, leveraging Dimension Reduction, and using a subset of scenarios.

3.1 Data Generation

In order to generate instances to evaluate our various methods, we leveraged the package Generators for Combinatorial Optimization, also known as `GeCO` (Jonas Charfreitag, 2022). This code provides the capabilities for users to generate synthetic Mixed Integer Programs (MIP) instances of their preferred size through procedures defined by other publications.

We can define a linear program through 3 components that describe the objective and constraints around the first stage decision variable x: $c \in \mathbb{R}^{N_1}, A \in \mathbb{R}^{M_1 \times N_1}$, and $b \in \mathbb{R}^{M_1}$ where $N_1 = |x|$ and M_1 is the number of first stage constraints. Given that SAA is leveraged to solve the 2SP, the second stage problem can be described with a set of 4 components per scenario that describe the objective and constraints around the second stage decision variables y^s and their relations to the first-stage decision variable x. Each scenario can be defined by $q^s \in \mathbb{R}^{N_2}, W^s \in \mathbb{R}^{M_2 \times N_2}, h^s \in \mathbb{R}^{M_2}$, and $T^s \in \mathbb{R}^{M_2 \times N_1}$ where $N_2 = |y|$ and M_2 is the number of second-stage constraints (equation 3). The realization of randomness for each scenario can be represented by a summary vector ξ^s.

For each instance, the first stage problem is a packing problem generated through procedures used by Tang et al. (2020) with a linear relaxation. For the second stage problems of each instance, we first generated a single packing problem and extracted the A, b, and c components for the problem. For each scenario s the objective and the left-hand side is kept constant (referencing equation 3, $W^s = A, q^s = c;\ s = 1, 2, \ldots, K$) and the right-hand side is where the randomness occurs. For each scenario, $h^s = b + \xi_h^s, \xi_h^s \in \mathbb{R}^{M_2}$ and $T_s = \xi_T^s, \xi_T^s \in \mathbb{R}^{M_2 \times N_1}$. The values that make up ξ_h^s and ξ_T^s are sampled from some random distribution. The summary vector for each scenario is equal to ξ_h^s plus the row-wise sum of ξ_T^s. For each instance, $M_1 = 2N_1, M_2 = 2N_2$.

When generating instances, the distribution the randomness follows was varied between a normal distribution, a uniform distribution, and a bimodal distribution. The mean of each distribution was kept constant. We also varied the number of first-stage variables ($N_1 \in \{50, 100, 150, 200\}$) and second stage variables ($N_2 = \{50, 100, 150, 200\}$). Each instance also had a different number of scenarios sampled ($K \in \{100, 200, 300, 400\}$). By taking the combination of these parameters, a total of 192 instances were created.

3.2 Clustering Procedure

We aim to test 4 different clustering algorithms - K-means, Hierarchical (Agglomerative) Clustering, Spectral Clustering, and Affinity Propagation - and evaluate their impact on the performance of Benders Decomposition.

The scenarios will be clustered on their cluster summary variable ξ^s. While Affinity Propagation will automatically determine the number of clusters within the data, the other clustering methods require the number of clusters ($|C|$) to be specified by the user. For these methods, we clustered on ξ^s multiple times, varying the number of clusters to be between 1% of the number of scenarios to 20% of the number of scenarios ($|C| \in [K * 0.01, K * 0.2]$). For each set of clusters, we evaluated the silhouette score of the assignments and choose the set of clusters with the lowest silhouette score. This is an approach that has been shown to get the optimal number of clusters (Shahapure and Nicholas, 2020).

In addition to clustering methods, we also use a random grouping method as a baseline. For this method, we randomly choose a number of groups between 1% of the number of scenarios to 20% of the number of scenarios. We then randomly assign scenarios to each group. This imitates the naive hybrid cut Benders and the naive scenario selection method.

Our experiments are split up into two main pieces: evaluating clustering use in grouped multi-cut Benders (which we will refer to as hybrid-cut Benders) and the use of clustering for intelligent scenario reduction (which we will call dropout cut Benders).

3.2.1 Hybrid Cut Benders

The traditional hybrid cut Benders attempts to find a balance between the single-cut version and the multi-cut version of the algorithm. Where single-cut Benders has a single decision variable that summarizes the objective value of all scenarios and multi-cut has a decision variable for all K scenarios, hybrid-cut Benders assigns a decision variable to groups of scenarios so that there are H decision variables representing the objective of the subproblems with $1 < H < K$.

We hypothesize that if one were to take the naive approach of hybrid cut and randomly group scenarios together, a large amount of information may be lost as scenarios that are extremely dissimilar may be put in groups together. As a result, the Benders algorithm may take longer to converge as each cut may cover a larger area of the underlying probability distribution, effectively making the cuts weaker.

By clustering the scenarios beforehand and assigning groups based on the clusters, we hope to generate cuts that each cover a local area in the probability distribution, therefore providing more information to the master problem when they are used.

For each instance, we will cluster the scenarios using the four methods chosen and then run the hybrid-cut version of Benders. Since our goal is to optimize the expected value of our subproblem, we maintain information about the probabilities of each individual scenario occurring by treating the probability of a cluster ($\mathbf{p}(C_n)$) as the sum of the scenarios in the cluster. Due to this, the Benders formulation becomes:

$$\underset{x\in\mathcal{X}}{\text{minimize}}\ c^T x + \sum_{n=1}^{|C|} \mathbf{p}(C_n)\theta_n \tag{4}$$

$$\text{subject to } Ax \leq b \tag{5}$$

$$\mathbf{p}(C_n) = \sum_{s\in C_n} p_s \ \ \forall C_n \in C \tag{6}$$

As a baseline, we will compare our clustered version of hybrid-cut against single-cut Benders, multi-cut Benders, and naive hybrid-cut Benders.

3.2.2 Dropout Cut Benders

In Benders decomposition, scenario reduction can be a useful technique for improving the efficiency of the optimization process. While this approach may lead to a sub-optimal solution, the reduced complexity of the optimization problem makes this approach feasible when limited computational power is available. However, naively choosing scenarios to represent the problem may result in sampling scenarios that skew the representation of the problem space. Clustering may help resolve this problem as scenarios in different clusters are more dissimilar from each other than clusters within the same scenario.

For each instance, we will first cluster all the scenarios using the four methods chosen. For each cluster, We will then choose the scenario who's ξ vector is closest to the averaged ξ vector of the cluster to be the representative scenario of the cluster and we will "dropout" the rest. The representative scenario is the one with the shortest Euclidean distance between the mean center of the scenario group.

We will also adjust the probability of the scenario to be the probability of the cluster as defined in equation 6.

We will compare the quality of the solutions and the time it took to compute the solutions to the naive dropout cut baseline.

3.3 Experimental Configuration

All of our experiments leveraged Python as our main language and Gurobi (Gurobi Optimization, LLC, 2022) as our optimizer. The implementations of the various clustering algorithms used were all from `sci-kit learn` (Pedregosa et al., 2011). All experiments were run on an Intel i7-12700 h @ 2.10 GHz with 32.0 GB of RAM. Additionally, every combination of instance and solution method was repeated 3 times in order to mitigate the impact of randomness. The results reported are based on the averages of these 3 runs. Each run also has a time limit of 300 seconds.

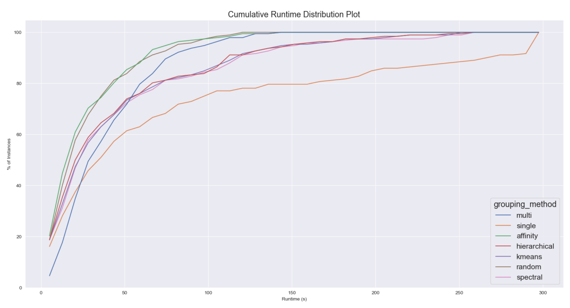
For each experimental run, we collect the number of cuts applied, number of iterations the algorithm went through, the average time it took to solve the master problem across iterations, the average total time it took to solve all the subproblems across iterations, the time it took to cluster the scenarios, and the total runtime of the whole procedure. We also collect information on the primal gap if the algorithm did not converge after 300 seconds. Additionally, the optimal first-stage found in each run is evaluated against an extensive form formulation that included all scenarios (this is collected for both hybrid and dropout cut, but only really matters for dropout cut)

4 Results

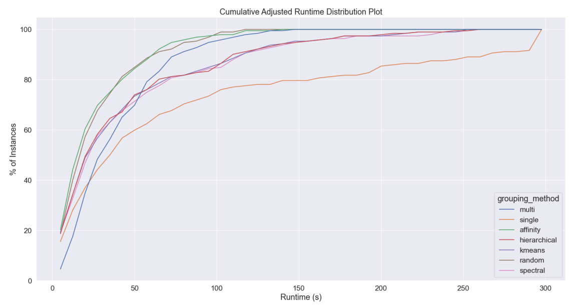
4.1 Hybrid Cut Analysis

When looking at the runtime distributions of different variations of Benders (figure 1a), we can clearly see that the hybrid-cut approach outperforms single-cut Benders for all problem sizes. However, the performance of most clustering algorithms, namely K-means, Spectral Clustering, and Hierarchical Clustering, were outperformed by multi-cut Benders on larger sized instances. In addition, these clustering methods were outperformed by randomly grouping scenarios regardless of problem size. Affinity propagation (AP) was the only clustering method that could compete with random grouping, and even then the runtime reduction provided by clustering scenarios using AP was less than the runtime reduction from random groups on a small portion of the instances.

These results hold even when only taking problem solve time into account (shown in figure 1b). The time required to cluster the the scenarios in all the instances was extremely small and had an insignificant impact on the total runtime of the algorithm. As such, adjusting the runtime to exclude the clustering time did not change how the different methods performed when compared to each other.



(a) Cumulative Runtime Distribution

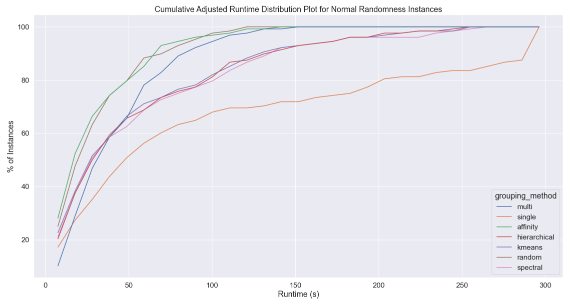


(b) Cumulative Adjusted Runtime Distribution

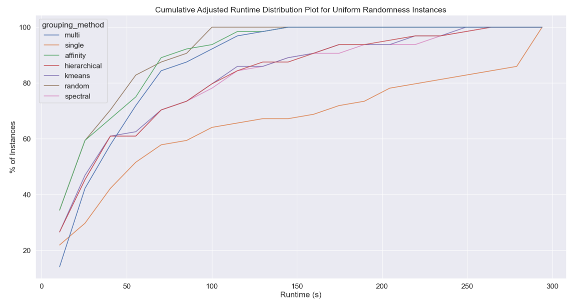
Figure 1: Cumulative Runtime/Adjusted Runtime Distributions comparing different clustering methods

In addition to the overall performance of each method, we noticed a difference in the impact would help when ξ was sampled from different types of distributions. Clustering had a much smaller runtime improvement impact in instances with uncertainty from symmetric distributions (uniform and normal distributions) as seen in figure 2. The difference is most pronounced when looking at the instances with uniform randomness (figure 2b). While Affinity Propagation still results in lower runtimes for most instances compared to normal multi-cut, the performance of the two is much more similar when compared to their runtime impacts on normal uncertainty instances (figure 2a). In contrast, grouping the scenarios randomly resulted in equal or better performance for a large portion of the instances for the uniform randomness instances.

Affinity Propagation does however triumph as the winner when looking at instances that have more complicated distributions, namely bimodal distributions (figure 3). When using Affinity Propagation to cluster the scenarios of



(a) Randomness from Normal Distribution



(b) Randomness from Uniform Distribution

Figure 2: Cumulative Adjusted Runtime Distribution Comparing Different Clustering methods on instances with Symmetric Shapes

bimodal randomness instances, all the instances solved in less than 100 seconds. While random grouping is still a close second, Affinity Propagation yields higher runtime improvements on every instance size.

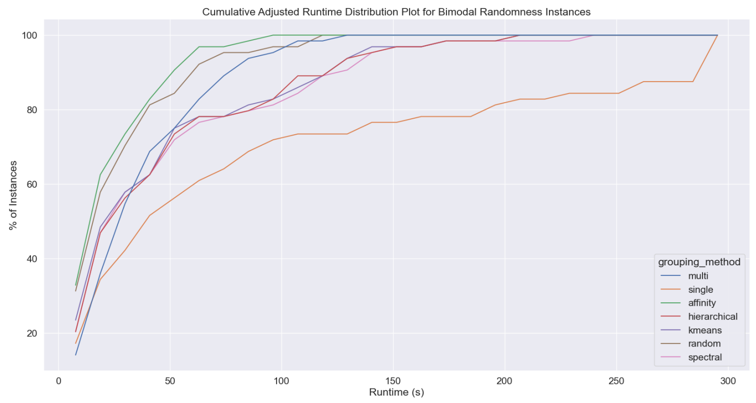


Figure 3: Cumulative Adjusted Runtime Distribution Comparing Different Clustering methods on instances with randomness from Bimodal Distributions

4.2 Dropout Cut Analysis

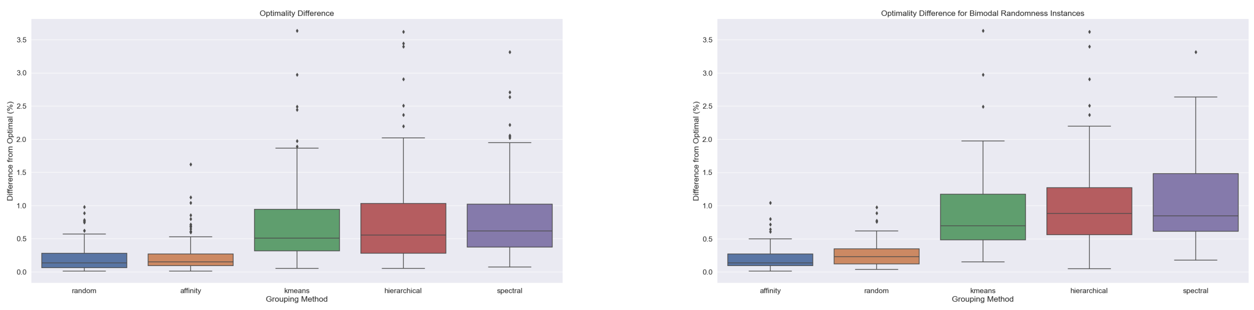
While runtime is a valuable metric for evaluating the performance of various clustering methods on hybrid-cut Benders, it is misleading when using to evaluate the performance of clustering on scenario clustering. Instead, we look at how close the selected scenarios were able to get to the true optimal. The method that is able to select the most scenarios will have the lowest optimality difference. In figure 4a, we can see that random grouping and affinity propagation results in a significantly lower optimality difference than the other clustering methods. These results closely mirror the runtime results from hybrid-cut with random and affinity propagation competing closely, while K-Means and the other clustering algorithms yield significantly lower performances.

When looking at only the instances with bimodal randomness (figure 4b), we can see that affinity propagation again has the best performance and has selected the best instances. However, this difference is not very large.

4.3 Evaluation of Clustering Methods

For both hybrid-cut and dropout cut, all clustering methods apart from Affinity Propagation have significantly underperformed random grouping. This could be attributed to a few possible reasons.

One potential factor is that the cluster number determination process results in a suboptimal number of clusters for K-Means, Hierarchical, and Spectral Clustering. Since the distributions we have created are at most bimodal, the optimal number of clusters is relatively arbitrary even when using analytical methods to determine this. Additionally, if the number of clusters has a tendency to be very low, then the number of cuts introduced per iteration of Benders will be very low as well. In comparison, the random grouping method has a chance to introduce a lot more groups. In context of the dropout cut, this would mean that the solver has more scenarios to use in order to find a solution.



(a) All Instances (b) Bimodal Randomness Instances

Figure 4: Percent True Optimality Difference

This would naturally result in a better solution. This would be a problem for the hybrid-cut method as well where the optimal number of clusters determined through methods like silhouette score does not provide as much information as an arbitrarily larger number of clusters. In our testing, AP clustering has been able to get around this hurdle, but this requires further testing on whether the clusters chosen by AP are better or if AP just created more clusters.

The performance improvement of affinity propagation over random grouping on instances with complex uncertainty distributions does show a clear benefit to clustering. Whether it is approximating the optimal solution via scenario reduction, or attempting to reduce the runtime of Benders, AP has shown to consistently provide good results over other clustering methods and random grouping when looking at bimodal instances. This success also potentially implies that if a method for cluster amount determination that is more relevant to Benders is used, other clustering methods could potentially also provide significant runtime improvements.

Despite the success of clustering in improving runtimes and solution quality in bimodal randomness instances, clustering has struggled to outperform random grouping when looking at uniform and normal randomness instances. This is unsurprising as the underlying distributions do not have a complicated space that requires clustering to group.

This underperformance could also be attributed to the information compression method used to create ξ. Since the vector representing the randomness is a simple rowwise sum, a lot of information regarding the randomness of individual elements of the T^s matrix could be lost.

5 Conclusion

While most clustering methods did not significantly improve the runtime or solution quality of Benders Decomposition, we did find that Affinity Propagation clustering had a positive performance improvement when used to group scenarios drawn from a multimodal probability distribution.

Some adaptations and experiments are left for the future due to time constraints. First, we did not vary the number of constraints in our instances, instead we set them to twice the number of variables. In the future, we would like to vary the number of constraints independently from the number of variables to get a more representative test set. Additionally, we would like to also benchmark our methods across real-life instances such as the ones from SIPLIB (Ahmed et al., 2015) to prove our analysis generalizes well. Additionally, we primarily focused on adding randomness to the right-hand side of the sub-problems in these experiments. While the results may not significantly change, we also aim to add randomness to the left-hand side of the constraints and the objective function of the second-stage problems. One approach that was also not tested in this paper was the dynamic clustering method proposed by Vandenbussche et al. (2019).

References

Shabbir Ahmed, R Garcia, N Kong, L Ntaimo, G Parija, F Qiu, and S Sen. Siplib: A stochastic integer programming test problem library. *See http://www2. isye. gatech. edu/˜ sahmed/siplib*, 2015.

Gulay Barbarosoğlu and Yasemin Arda. A two-stage stochastic programming framework for transportation planning in disaster response. *Journal of the operational research society*, 55(1):43–53, 2004.

Jacques F Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische mathematik*, 4(1):238–252, 1962.

Salvador Perez Canto. Application of benders' decomposition to power plant preventive maintenance scheduling. *European journal of operational research*, 184(2):759–777, 2008.

Matteo Fischetti, Ivana Ljubić, and Markus Sinnl. Benders decomposition without separability: A computational study for capacitated facility location problems. *European Journal of Operational Research*, 253(3):557–569, 2016.

Pirmin Fontaine and Stefan Minner. Benders decomposition for discrete–continuous linear bilevel problems with application to traffic network design. *Transportation Research Part B: Methodological*, 70:163–172, 2014.

Brendan J Frey and Delbert Dueck. Clustering by passing messages between data points. *science*, 315(5814):972–976, 2007.

Emilia Grass and Kathrin Fischer. Two-stage stochastic programming in disaster management: A literature survey. *Surveys in Operations Research and Management Science*, 21(2):85–100, 2016.

Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2022. URL `https://www.gurobi.com`.

GH Huang and Daniel Peter Loucks. An inexact two-stage stochastic programming model for water resources management under uncertainty. *Civil Engineering Systems*, 17(2):95–118, 2000.

Jonas Charfreitag. Generators for Combinatorial Optimization, 2022. URL `https://github.com/CharJon/GeCO`.

Julien Keutchayan, Janosch Ortmann, and Walter Rei. Problem-driven scenario clustering in stochastic optimization. *arXiv preprint arXiv:2106.11717*, 2021.

Maryam Khatami, Masoud Mahootchi, and Reza Zanjirani Farahani. Benders' decomposition for concurrent redesign of forward and closed-loop supply chain network with demand and return uncertainties. *Transportation Research Part E: Logistics and Transportation Review*, 79:1–21, 2015.

Anton J Kleywegt, Alexander Shapiro, and Tito Homem-de Mello. The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 12(2):479–502, 2002.

Mengyuan Lee, Ning Ma, Guanding Yu, and Huaiyu Dai. Accelerating generalized benders decomposition for wireless resource allocation. *IEEE Transactions on Wireless Communications*, 20(2):1233–1247, 2020.

Yunhao Li, Jianxue Wang, and Tao Ding. Clustering-based chance-constrained transmission expansion planning using an improved benders decomposition algorithm. *IET Generation, Transmission & Distribution*, 12(4):935–946, 2018.

Nilay Noyan. Risk-averse two-stage stochastic programming with an application to disaster management. *Computers & Operations Research*, 39(3):541–559, 2012.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Ketan Rajshekhar Shahapure and Charles Nicholas. Cluster quality analysis using silhouette score. In *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 747–748, 2020. doi:10.1109/DSAA49011.2020.00096.

M Shahidehopour and Yong Fu. Benders decomposition: applying benders decomposition to power systems. *IEEE Power and Energy Magazine*, 3(2):20–21, 2005.

Hansuk Sohn, Bill Tseng, and Dennis L Bricker. A hybrid benders/genetic algorithm for vehicle routing and scheduling problem. *International Journal of Industrial Engineering*, 19(1):33–46, 2012.

Mohammad Amin Tajeddini, Ashkan Rahimi-Kian, and Alireza Soroudi. Risk averse optimal operation of a virtual power plant using two stage stochastic programming. *Energy*, 73:958–967, 2014.

Yunhao Tang, Shipra Agrawal, and Yuri Faenza. Reinforcement learning for integer programming: Learning to cut. In *International conference on machine learning*, pages 9367–9376. PMLR, 2020.

Richard M Van Slyke and Roger Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM journal on applied mathematics*, 17(4):638–663, 1969.

Baudouin Vandenbussche, Stefanos Delikaraoglou, Ignacio Blanco, and Gabriela Hug. Data-driven adaptive benders decomposition for the stochastic unit commitment problem. *arXiv preprint arXiv:1912.01039*, 2019.

Huansheng Zhou, JH Zheng, Zhigang Li, QH Wu, and XX Zhou. Multi-stage contingency-constrained co-planning for electricity-gas systems interconnected with gas-fired units and power-to-gas plants using iterative benders decomposition. *Energy*, 180:689–701, 2019.

Victor Zverovich, Csaba I Fábián, Eldon FD Ellison, and Gautam Mitra. A computational study of a solver system for processing two-stage stochastic lps with enhanced benders decomposition. *Mathematical Programming Computation*, 4(3):211–238, 2012.