CSC 591 Project - Group 7

Dev Kumar
Department of Computer Science
North Carolina State University
Raleigh, North Carolina 27606
Email: dkumar6@ncsu.edu

Rohan Sinha
Department of Computer Science
North Carolina State University
Raleigh, North Carolina 27606
Email: rsinha3@ncsu.edu

Palash Pravin Rathod
Department of Computer Science
North Carolina State University
Raleigh, North Carolina 27606
Email: prathod@ncsu.edu

I. Introduction

[1] Software engineering is a challenging and ever-changing field that requires its practitioners to solve difficult problems with goals that may conflict with each other. To achieve the desired results, software engineers must address various complex issues and questions, such as managing project timelines and budgets, reconciling conflicting stakeholder requirements, ensuring software quality and security, integrating new technologies, and balancing software development goals with business objectives. To succeed in software engineering, one needs to have a diverse set of skills and stay up-to-date with the latest developments in the field. Additionally, being adaptable, resourceful, and innovative is crucial to effectively address these challenges. Among the most pressing of these are the following:

- What is the smallest set of test cases that cover all program branches?
- What is the set of requirements that balances software development cost and customer satisfaction?
- What sequence of refactoring steps takes the least effort while most decreasing the future maintenance costs of a system?

Upon closer examination, it becomes clear that these questions are all optimization problems, which means that they require the application of specialized tools and techniques in order to arrive at satisfactory solutions. One such approach is known as Search Based Software Engineering (SBSE), which leverages sophisticated optimization algorithms to assist in the identification of optimal solutions to complex problems in software engineering.

There is no one algorithm that fits for every engineering problem. Depending on the optimization goals, one algorithm might work better than the other and vice-versa. Algorithms like Simulated Annealing and Genetic algorithms have been used for quite some time. However, a new and more effective algorithm was proposed by [2] which was SWAY (Sampling Way). Sway generates a large number of random candidates, evaluates a small number of representative candidates using some clustering algorithm and gets rid of the poor candidates. The clustering method proposed by [2] is that of Fastmap clustering. The authors of [2] came to the conclusion that Sway is better than most of the state-of-the-art evaluation algorithms and was faster than them.

The primary objective of this research paper is to explore the potential of Sway, a baseline optimizer proposed by [2], as a tool for facilitating multi-objective semi-supervised learning. In order to achieve this goal modifications have been made to the original Sway method. The results obtained from this modified version will be compared to those obtained using the baseline optimizer. To provide a comprehensive evaluation of the efficacy of the modified Sway algorithm, it is tested on 11 different datasets, all of which are thoroughly described in Section 3 of the paper.

The structure of the paper is as follows: Section 2 discusses the prior work done in the field of Search Based Software Engineering. Section 3 elaborates the methods that have been used to make changes to the original Sway method and also describe the datasets on which it has been tested. Section 4 discusses the results obtained. Section 5 states the conclusion of this study and section 6 lists the references used.

II. Related Work

Before implementing any changes, it is crucial to gain a comprehensive understanding of the existing work in the field. As stated in [4], software engineering problems often rely on metaheuristic search techniques, such as genetic algorithms and simulated annealing. To advance the field of Search-Based Software Engineering (SBSE), these search techniques need to be refined and reformed. Metaheuristic techniques, as described in [5], are a powerful tool for generating optimal solutions and lie at the heart of SBSE. The idea of SBSE [6] aims to address the challenges of optimization that arise when trying to balance multiple competing objectives in software engineering. Several challenges, such as test generation [7], requirement prioritization [8], and test optimization [9], can be solved using SBSE. These problems are often multi-objective in nature, and to tackle them, Multi-Objective Search Algorithms (MOSAs) are commonly employed.

In recent times, Search-Based Software Engineering (SBSE) has garnered a lot of attention, and many researchers are actively involved in exploring this field. In one particular study conducted by authors in [10], they highlight the difference between single and multiple objective algorithms and explain how each of them works. The first step involves identifying multiple non-dominated points near the Pareto-optimal front, while also striking a balance between different objectives. In the second step, a point is selected from the set of solutions obtained, based on higher-level information. Another noteworthy research effort was carried out by researchers in [11], where they implemented a multi-objective optimization model to group patients by optimizing the cost function of k-medians clustering, with the aim of recognizing similar patterns. The results obtained indicated that the multi-objective approach outperformed single objective ones. Moving on, authors in [12] proposed a semi-supervised learning algorithm that utilized recursive K-means clustering based on the divide and conquer strategy. This paper served as a valuable source of information, providing insights into

how recursive k-means can be used to cluster data when some parts are labeled, and some are not. Lastly, the research work in [13] focused on Loyd's k-means for clustering large datasets, but with sampling. The authors pointed out that the distance measure used in clustering is time-consuming, especially for large multi-dimensional datasets. Hence, they employed a sampling function that randomly samples n data points after every iteration until the entire dataset is classified into k clusters. The results obtained for 3-dimensional datasets revealed that their algorithm with sampling was faster than the standard k-means algorithm.

A wide range of studies have been conducted to investigate the potential of utilizing different optimization algorithms to handle multi-objective problems. While creating a new algorithm from scratch is a possibility, it is not the only approach to achieve better results. Sometimes, even minor modifications to existing techniques can lead to significant improvements in the outcomes. As mentioned earlier, [12] executed a k-means clustering algorithm to address certain problems, while [13] integrated sampling into the algorithm and obtained remarkable results for high-dimensional datasets. It is worth noting that this research paper aims not only to comprehend the current approaches, but also to explore the possibility of manipulating them to evaluate whether novel techniques or modifications can enhance their performance.

III. Methods

A. Algorithms

This paper aims at modifying the Sway function proposed by [2] to see if better results are achieved or not. Firstly, let's understand how Sway works originally. The original Sway method implements a multi-objective optimization algorithm for semi-supervised learning problems that uses distance-based clustering to divide the data and recursively optimize subsets of the data until a stopping condition is met.

It takes data as input and applies the Sway optimization algorithm to find multiple non-dominated points close to the Pareto-optimal front, with a wide trade-off among objectives. It recursively divides the data set into two sets, using distance-based clustering. The split for continuous decision spaces is implemented via Fastmap heuristic [14] where a randomly selected point is projected on the line which connects two extreme candidates using the Euclidean distance measure. This way all the candidates are projected on this line and based on it they are split into two parts. In order to compare the representatives of two halves of candidates, Sway uses Zitzler's binary domination for individual comparisons. If one representative dominates the other, Sway prunes the other half and vice-versa.

Now, we propose a modification to the original Sway method. Ahead in the paper, the modified Sway will be referred to as Sway2. In Sway, two points that are very far from each other are considered. Choosing extreme points as remote points can be more effective in some cases because they are likely to be far apart, which can lead to a better separation of the data. Extreme points are points that have the highest or lowest values for a particular feature or set of features. For example, if we are clustering data points based on their location in two-dimensional space, extreme points could be points that are located at the edges of the space. By using extreme points as remote points, we are effectively defining the boundaries of the clusters, which can be helpful in cases where the clusters are well-separated. However, extreme points may also be outliers or not representative of the entire dataset, which can lead to suboptimal clustering. Outliers are data points that are significantly different from the other data points in the dataset and can skew the clustering results. If we use extreme points that happen to be outliers, then we may end up with clusters that are not representative of the entire dataset.

Therefore, the approach that we are proposing for Sway2 is that instead of choosing two distant points, choose two points randomly as remote points. Choosing random points as remote points can be more representative of the overall data, as they are more likely to be spread out across the entire dataset. This can be helpful in cases where the dataset is more evenly distributed and there are no clear extreme points. By using random points as remote points, we are essentially taking a more holistic approach to clustering and trying to capture the overall structure of the data. Along with this, the distance measure used to find the distance between two data points is Euclidean distance.

Along with Sway2, there is a rule generation algorithm that has been implemented to generate a set of rules based on input data that can be used to classify or predict outcomes for future data. This will be referred to as Xpln2 in the paper henceforth. The algorithm proceeds as follows:

- First, it collects all the ranges for each attribute into a single flat list.
- It sorts the ranges by their value.
- It then passes all the ranges to a function which selects the top N ranges based on their value.
- Along the way, the algorithm keeps track of the maximum number of ranges per attribute.
- It skips over any rules that would result in no matches.
- It passes the selected set of ranges to a scoring function, which generates a rule based on the selected ranges and evaluates its performance on a given data set.

The function which selects the top N ranges takes in a list of sorted ranges, a scoring function, and returns the best rule and its score. It works by trying out the first range, then the first two, then the first three, and so on up to the maximum number of useful ranges. It keeps track of the best rule seen so far and returns it at the end. We also define a function that only considers the ranges that are useful and skips the rest. For Xpln2, the thresholds of the useful function have been modified to make it more susceptible to select good ranges and skip over the rest.

The modified - Sway2 and Xpln2 - is now used to perform the task of a semi-supervised multi-objective search problem over 11 datasets that have been described in the next section.

B. Data

The success of any data-driven project heavily relies on the quality and relevance of the data used. In this section, we describe the data files and demonstrate our understanding of the data used for analysis of the multi-objective semi-supervised learning system.

Our goal was to first develop a clear understanding of the data used. This is important because our existing work was not fit to handle the different kinds of data sets provided to us. This is often the case in the real world and our focus was to improve our

existing work so that it is capable of handling different forms of data.

Following is a list of all the domains and its corresponding data sets that have been used for the project.

- Car Design: auto2, auto93
- Software Project Estimation: china, coc1000, coc10000
- Software Effort + Detects Estimation: nasa93dem
- Issue Close Time: healthCloseIsses12mths0001-hard, healthCloseIsses12mths0011-easy
- Agile Project Management: pom
- Computational Physics: SSM, SSN

During the time when we were analyzing the data for the project, we made certain observations for these data sets individually. Below is an attempt to give a deeper insight about the same.

1) auto93: The 'auto93' data set has been used throughout the semester to test the efficacy of the programs written. It belongs to the 'car design' domain and concerns city-cycle fuel consumption in miles per gallon. [15] Consisting of 398 rows, the data can be used to predict the "mpg" attribute in terms of 3 multi-valued discrete and 5 continuous attributes. [16]

2) auto2: The 'auto2' data set is yet another data set belonging to the 'car design' domain. It is a more comprehensive version of the previously discussed 'auto93' data set. With 93 different samples, it consists of 23 attributes - each giving information about various features of the car. Apart from the attributes that can be found in the auto93 data set, some of the new attributes are RPM, Passenger Capacity, Rear Seat Room, Luggage Capacity and Airbags. Such comprehensive information could really help in determining a more precise correlation between different features of the cars and the target variable.

3) china: This data set belongs to the 'Software Project Estimation' domain consisting of 499 samples measured across 19 different attributes including but not limited to Input, Output, Interface and Resource. The data set can be used to predict the software development effort which is the target attribute here (labelled as 'Effort'). This attribute is measured in number of months.

4) coc1000: This data set again belongs to the 'Software Project Estimation' domain and is part of the research that is relevant to the COnstructive COst MOdel (COCOMO) model. [17] This model is used to calculate the engineering or development effort for a project. Given its nomenclature, the dataset consists of 1000 samples and is 1/10th the size of its variant 'coc10000' which is also used in the project and is discussed below.

5) coc10000: As previously mentioned, this data set is same as above with the only difference that it consists of 10000 samples measured across 25 attributes with the 'Effort' attribute being the target attribute.

6) healthCloseIsses12mths0001-hard: Like all the data sets discussed till now, this data also helps in dealing a kind of regression problem where the goal is to predict a continuous numerical target variable. In this data set, the target attribute seems to be the max depth of a Decision Tree used as Extra-Tree in the Extra Trees Classifier. The data set consists of 10000 values measured for 8 attributes each.

7) healthCloseIsses12mths0011-easy: This data set appears to be exactly like the one discussed above with all the same attributes as well as number of rows. One of the few differences that was observed between the two was with the standard deviation for the 'PRED40+' attribute. It is much higher in this data set given that the minimum value as well as the three quartiles (lower, median and upper) are 0.00. It was also observed that the 'ACC+' attribute only consists of values less than 0.00 which is not the case in the previous data set.

8) nasa93dem: The 'nasa93dem' data set is also focused on software development effort and its estimation. With 29 different attributes (including the target variable 'Months'), the data has 93 samples - the values of which can help in estimating the effort required for software development.

9) pom: The 'pom' data set deals with problems pertaining to Agile Project Management. This data can aid in studying about the idle rate in Agile Projects and its affects over different components of Agile Project Management. POM3 Model is a tool that helps in overcoming challenges in Agile Project Management. In the agile space, projects terminate after achieving an X% completion rate and this renders the team to be idle. This results in the resources not being optimized efficiently and is a big concern among stakeholders. [18] The 'pom' data consists of 10000 rows and 13 attributes. These attributes consist of a variety of components that determine the success of an Agile project. Some examples include team size, cost, inter-dependency and criticality.

10) SSM: The 'SSM' data is about the configuration space of Trimesh - a library to manipulate triangle meshes. [19] It consists of 15 attributes (13 config. options) with values resulting in 239,260 different configurations. It belongs to the 'Mesh Solver' family in the 'Computational Physics' domain.

11) SSN: The 'SSN' data is about the configuration space of an X-264 video encoder. [19] The data consists of 19 attributes (17 config. options) with values resulting in 53,662 different configurations. It belongs to the 'Video Encoder' family in the 'Computational Physics' domain.

C. Data preprocessing

The 11 data sets are of different kind and have different types of values. To facilitate the working of our modified functions on all the data sets, it is important to preprocess them and make them uniform in the sense that the functions will not run into unwanted errors.

To clean the data, we have filled in the missing values with the mean value of that particular column. In case a '?' is encountered, then depending on whether that column had numeric or categorical type data, it is removed and filled up. Certain columns have categorical data which need to be converted into numeric for which encoding has been done. In a nutshell, several data cleaning and transformation operations, including filling missing values and converting categorical data into numeric data have been performed.

D. Performance Measures

In order to assess the performance of the modified functions we need to take into consideration certain measures. For SBSE, there are many performance measures that can be considered depending on the requirements. Some of these measure are runtime, number of evaluations, Generational Distannce (GD), Generated Spread (GS), Pareto front size (PFS) and Hypervolume (HV).

This paper considers number of evaluations as the performance measure for Sway2 and Xpln2. Since for each dataset, 20 iterations are run, we have considered the average number of evaluations for 20 runs as the final performance measure. When we consider all the y variables, the number of evaluations are equal to the number of rows in the dataset. However, when we use Sway, Sway2, Xpln or Xpln2 functions, the number decreases significantly making it faster and effecient.

E. Summarization Methods

1) Significance testing: Significance testing is a commonly used statistical method to determine if the difference or relationship between two variables is statistically significant or not. Parametric tests, such as t-tests, assume that the data is normally distributed and calculate the difference between two variables in terms of their means. Non-parametric tests, on the other hand, do not assume any specific distribution of the data and use rank or permutation tests to determine significance. However, these methods may not always be appropriate for small sample sizes or non-standard data distributions.

We have used the bootstrap procedure that can be used to estimate the sampling distribution of a statistic by resampling from the original dataset with replacement. It allows to calculate confidence intervals and p-values for the observed statistic without relying on the assumptions of traditional significance tests. The procedure involves building a large number of resampled datasets, computing the statistic of interest on each dataset, and then using the distribution of those statistics to estimate the probability of observing the observed statistic by chance.

2) Effect size tests: Effect size tests are statistical methods used to determine the magnitude of the difference or relationship between two variables, rather than just whether there is a significant difference or relationship. The Cohen's-D parametric effect-size method assumes that the data is normally distributed and calculates the difference between two variables in terms of the number of standard deviations. If the data does not meet the normality assumption, the non-parametric Cliff's Delta effect size test can be used.

We have used Cliff's Delta to calculate the probability of a randomly selected value in one distribution being larger than a randomly selected value in another distribution and classify the effect size as negligible, small, medium, or large.

Effect size tests provide a more informative measure of the strength of the relationship or difference between variables than simply reporting the p-value. They are particularly useful when analyzing data from multiple datasets, as they allow researchers to compare the effect sizes across datasets and determine if there is a consistent pattern.

We have also used Scott-Knott procedure for ranking. The Scott-Knott method sorts treatments based on their median scores and recursively divides them into two sub-lists at the point where the expected value of the difference in median performance before and after the division is maximized. The sub-lists are then tested for significant differences using effect size and significance tests. If no significant differences are found, the recursion stops. One of the advantages of using the Scott-Knott method is that it can be made fully non-parametric by selecting non-parametric effect size and significance tests, which reduces the number of potential errors in statistical analysis. Additionally, this method overcomes a limitation of other multiple-comparison statistical tests like the Friedman test, where treatments are assigned to multiple groups, making it difficult to distinguish the real groups to which the means should belong.

IV. Results and discussions

The results for our modified functions over all the 11 datasets are shown below in the images. Each image is the output for one dataset where it contains the result for all, Sway1, Sway2, Xpln1 and Xpln2. As stated earlier, the average number of evaluations have also been shown as a performance measure for this study. The bottom table shows the conjunction of effect size test and significance test which is known as prudence check.

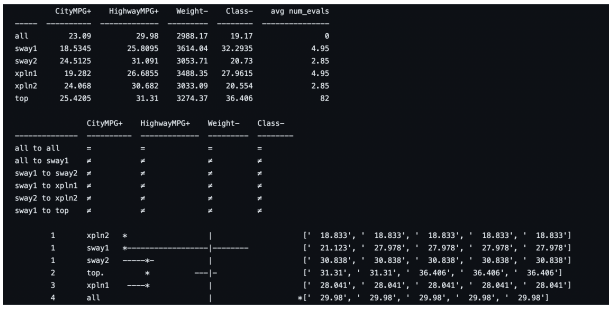

Fig. 1. Auto2 dataset

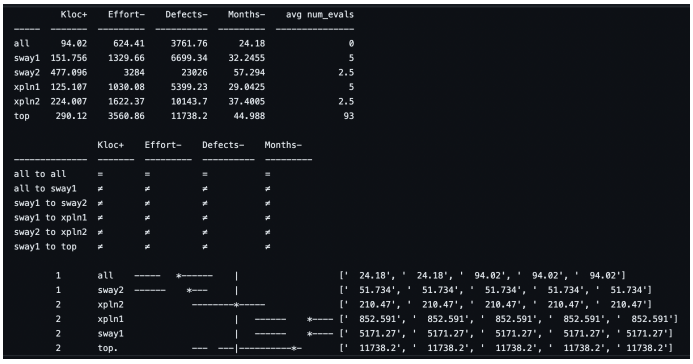

Fig. 2. Nasa dataset

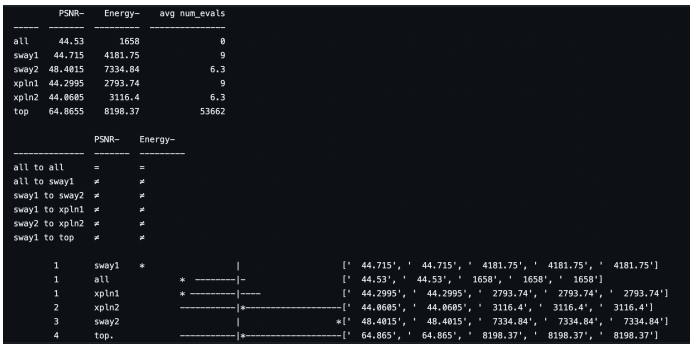

Fig. 3. SSN dataset

V. Conclusion

Our research aimed to improve the baseline optimizer Sway by developing a semi-supervised multi-objective system. In contrast to the original optimizer, which relied on 2 extreme points in the dataset to create clusters, we implemented a novel approach that selects 2 random points in the dataset. This modification allows for a more diverse and representative clustering, which can ultimately lead to better optimization outcomes.

Choosing extreme points as remote points seem more effective as they are likely to be far apart, which can lead to a better separation

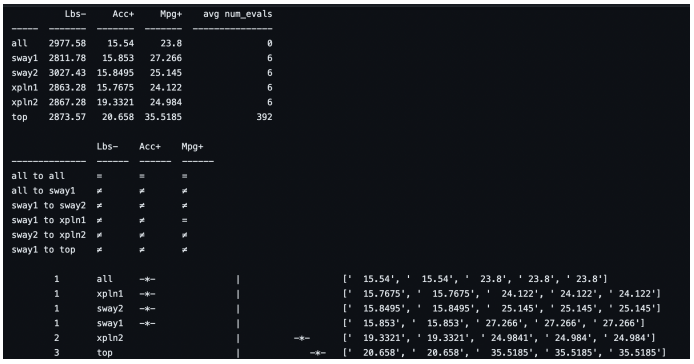

Fig. 4. Auto 93 dataset

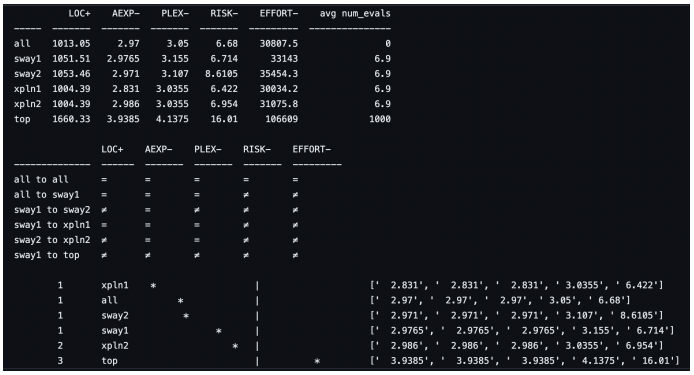

Fig. 5. coc1000 dataset

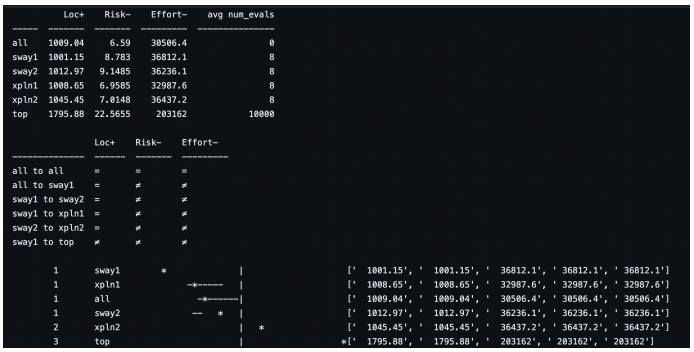

Fig. 6. coc10000 dataset

Fig. 7. china dataset

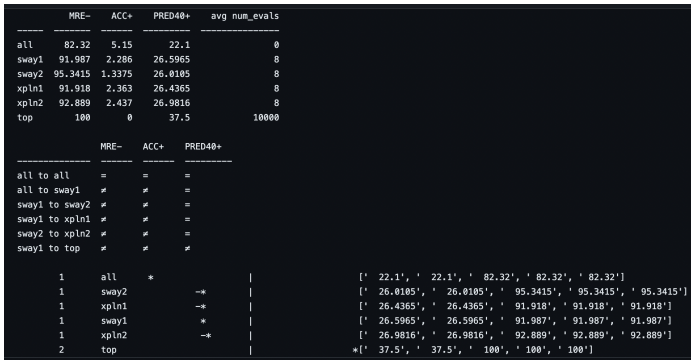

Fig. 8. healthCloseIsses12mths0001-hard dataset

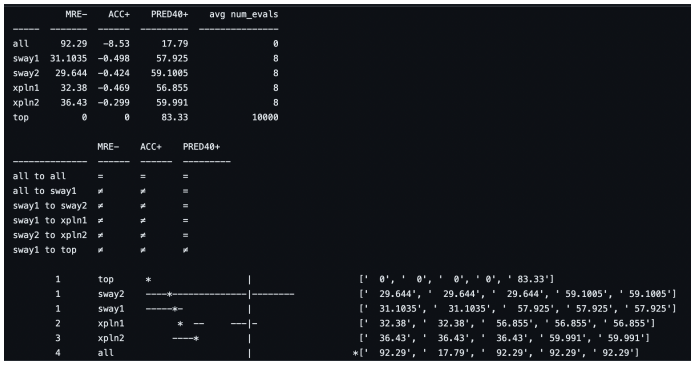

Fig. 9. healthCloseIsses12mths0011-easy dataset

Fig. 10. Pom dataset

of the data. However, extreme points may also be outliers or not representative of the entire dataset, which can lead to suboptimal clustering. Using outliers for clustering can result in clusters that are not representative of the entire dataset.

On the other hand, choosing random points as remote points can be more representative of the overall data, as they are more likely to be spread out across the entire dataset. This can be helpful in cases where the dataset is more evenly distributed and there are no clear extreme points. By using random points as remote points, we are essentially taking a more holistic approach to clustering and trying to capture the overall structure of the data.

However, random points may also be closer together and not provide as clear of a separation as extreme points. This can lead to

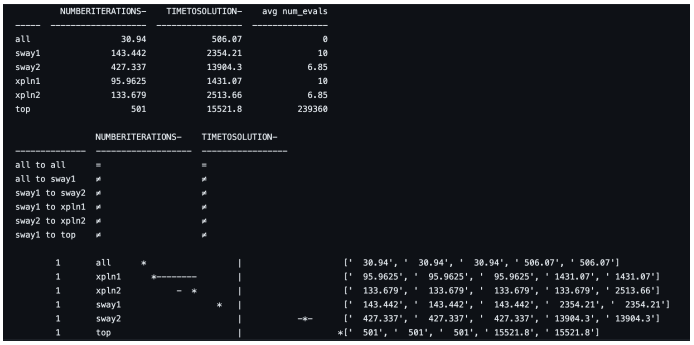

Fig. 11. SSM dataset

suboptimal clustering if the clusters are not well-separated.

In general, the choice between extreme points and random points as remote points should be made based on the specific characteristics of the data being used. If the dataset has clear extreme points, then using them as remote points may lead to better clustering results. If the dataset is more evenly distributed, then using random points as remote points may be more effective.

ACKNOWLEDGMENT

We would like to express our sincere thanks to Professor Tim Menzies and the TAs of the course CSC 591 for their invaluable guidance and support throughout the course of this project. Their feedback, insights, and encouragement have been instrumental in shaping our ideas and improving our work.

REFERENCES

[1] Harman, Mark and Mansouri, Afshin and Zhang, Yuanyuan. (2009). Search Based Software Engineering: A Comprehensive Analysis and Review of Trends Techniques and Applications.

[2] Chen, J., Nair, V., Krishna, R., & Menzies, T. (2019). "Sampling" as a Baseline Optimizer for Search-Based Software Engineering. IEEE Transactions on Software Engineering, 45(6), 597–614. https://doi.org/10.1109/tse.2018.2790925

[3] https://arxiv.org/pdf/2301.06577.pdf

[4] Harman, M., & Jones, B. D. (2001). Search-based software engineering. Information & Software Technology, 43(14), 833–839. https://doi.org/10.1016/s0950-5849(01)00189-6

[5] Dutta, P., & Mahanand, B. S. (2022). Affordable energy-intensive routing using metaheuristics. In Elsevier eBooks (pp. 193–210). Elsevier BV. https://doi.org/10.1016/b978-0-323-85117-6.00013-3

[6] Wu, J., Arcaini, P., Yue, T. et al. On the preferences of quality indicators for multi-objective search algorithms in search-based software engineering. Empir Software Eng 27, 144 (2022). https://doi.org/10.1007/s10664-022-10127-4

[7] McMinn P (2004) Search-based software test data generation: a survey. Softw Test Verif Reliab 14(2):105–156

[8] Achimugu P, Selamat A, Ibrahim R, Mahrin MN (2014) A systematic literature review of software requirements prioritization research. Inf Softw Technol 56(6):568–585

[9] McMinn P (2011) Search-based software testing: Past, present and future. In: 2011 IEEE Fourth International Conference on Software Testing, Verification and Validation Workshops. IEEE, pp 153–163

[10] Deb, Kalyan. (2001). Multiobjective Optimization Using Evolutionary Algorithms. Wiley, New York.

[11] Akbarzadeh Khorshidi H, Aickelin U, Haffari G, Hassani-Mahmooei B. Multi-objective semi-supervised clustering to identify health service patterns for injured patients. Health Inf Sci Syst. 2019 Aug 29;7(1):18. doi: 10.1007/s13755-019-0080-6. PMID: 31523422; PMCID: PMC6715760.

[12] Gowda, H., Suhil, M., Guru, D. S., & Raju, L. N. (2016). Semi-supervised Text Categorization Using Recursive K-means Clustering. In Communications in computer and information science (pp. 217–227). Springer Science+Business Media.

[13] Bejarano, J., Bose, K., Brannan, T., Thomas, A. C., Adragni, K. P., Neerchal, N. K., & Ostrouchov, G. (2011). Sampling Within k-Means Algorithm to Cluster Large Datasets. In OSTI OAI (U.S. Department of Energy Office of Scientific and Technical Information). U.S. Department of Energy Office of Scientific and Technical Information. https://doi.org/10.2172/1025410

[14] Christos Faloutsos and King-Ip Lin. Fastmap: a fast algorithm for indexing, datamining and visualization of traditional and multimedia datasets. In ACM SIGMOD international conference on Management of data, 1995.

[15] Quinlan,R. (1993). Combining Instance-Based and Model-Based Learning. In Proceedings on the Tenth International Conference of Machine Learning, 236-243, University of Massachusetts, Amherst. Morgan Kaufmann.

[16] https://archive.ics.uci.edu/ml/datasets/auto+mpg

[17] T. Menzies, Y. Yang, G. Mathew, B. Boehm, and J. Hihn, "Negative results for software effort estimation," Empirical Software Engineering, vol. 22, no. 5, pp. 2658–2683, Nov. 2016, doi: https://doi.org/10.1007/s10664-016-9472-2.

[18] https://arxiv.org/pdf/1608.07617.pdf

[19] https://arxiv.org/pdf/1801.02175.pdf