Volume 5, Issue 12, December 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Optimal Resource Planning Based on Data Matching Performance and Monetary Cost

[1]George Anderson, [2]Audrey Masizana, [3]Dimane Mpoeleng
[1, 2] Department of Computer Science, University of Botswana, Botswana
[3]Department of Computer Science, Botswana International University of Science and Technology, Botswana

Abstract— ***This article addresses the problem of planning the organization of exam administration in a university in terms of exam group sizes. The problem is modelled as a multi-objective optimization problem, with two categories of objectives: data matching performance and monetary cost. Data matching is used in this context to match student registration records with exam records, with some measure of accuracy, precision, recall, and other related data matching metrics. When students are organized into smaller exam rooms, data matching performs better, but monetary cost is greater due to the higher number of exam supervision staff required. In this study, we model this problem to come up with a set of optimal exam room configurations.***

Keywords— Data Matching; Multi-objective Optimization; Machine Learning

I. INTRODUCTION

Most courses in universities require an end of term exam. Organization of such exams involves scheduling, which can be complex, depending on number of students registered. Students have to be divided into groups that fit into existing exam rooms. Another problem that has arisen is, after the exam, matching student exam records with course registration records. This is a challenge if the student exam records contain some errors in attributes such as surname, student ID number, etc. We proposed a data matching solution to this problem in an earlier study [1]. Data matching involves using two lists of records, A and B, and considering which record in list A matches which record in list B. Data matching solves the problem of identifying and matching records from disparate databases that refer to the same real-world entities [2].The matches are usually not exact matches; the most likely matches are approximate matches identified based on scores. The data matching process is usually not perfect and therefore there are metrics associated with it, such as precision and recall. The higher the scores, the fewer errors there are in the process and the less manual work has to be done to match records. In this study, we propose a multi-objective optimization solution for the exam group size problem. We minimize the cost of exam invigilation (proctoring) and maximize the data matching performance in two objectives. We observe that smaller exam groups mean more rooms and therefore more monetary cost for invigilation, but better data matching performance. Larger exam groups means less invigilation cost but worse data matching performance. Our study involves using machine learning to fit a model to our data matching data, in order to be able to predict data matching performance over several metrics.

To the best of our knowledge, no prior work exists which designs a process, such as exam scheduling, in order to ensure ease of data matching. Also, therefore, no prior work combines maximization of data matching performance with minimization of monetary cost, in a multi-objective optimization problem. This is therefore our main contribution.

The rest of this paper is organized as follows: Section II discusses related work. Section III discusses the exam and environment our study is based on. Section IV discusses our models and problem formulation. Section V discusses our experiments and results. Section VI concludes.

II. RELATED WORK

To the best of our knowledge, there is no discussion in the literature of resource allocation with a view to optimizing data matching; nor this objective in conjunction with reducing monetary cost of exam administration. We try to summarize literature related to timetabling, which one can argue is related to finding optimal group sizes. We also discuss work using machine learning to model systems for prediction of numeric quantities. We then discuss multi-objective optimization. We also discuss efforts to improve performance of data matching.

Our focus problem is associated with many models of timetabling found in literature. Bolaji et al. [3] use natural phenomena of bee colony combined with a hill climbing optimizer to optimize university timetabling. They found that artificial bee colony (ABC) algorithm has been successfully used for tackling examination and course allocation problem. Thepphakorn et al. [4] also used an ant colony based approach to timetabling. New variants of Ant Colony Optimisation (ACO) called the Best-Worst Ant System (BWAS) and the Best-Worst Ant Colony System (BWACS) were embedded in ANCOT. They yielded 75% better result is producing the optimal timetable. Other timetabling problems are solved using event based algorithms combining them with genetic algorithms. Events in this case are lectures, tutorials, laboratories, and seminars. Dorneles et al. [5] used heuristic algorithms to optimise timetable problem; a comparison with results reported in the literature shows that the proposed fix-and-optimize heuristic outperforms state-of-the-art techniques for the resolution of the problem.

Rai et al. [6] used linear regression and other machine learning models to predict numerical quantities. These involved fitting models that relate a dependent variable y to some independent variables, x_1, …, x_n. One then arrives at a function $y = f(x_1,...,x_n)$. Their focus was in the area of operating systems; predicting Central Processing Unit behaviour in computers, in order to make better operating system scheduling decisions. Machine learning models such as linear regression abound in the literature. For example, Bishop [7] discusses various machine learning algorithms and models.

Multi-objective optimization involves having more than one objective function, with one trying to minimize or maximize each one. Usually there is some trade-off: improving one objective function leads to worsening another. It is discussed extensively in applied mathematics, engineering, and computer science literature; for example [8]. The concept of pareto front comes into play, where there is no single “best” solution, but a user is presented with a set of solutions for which none is absolutely better than any other in the set. We make use of a pareto front in our study.

There are many publications on improving data matching available in the literature. For example Christen [2] gives an overview of all the major issues in data matching and implications on performance of various algorithms. Christen also contributed a number of important algorithms towards the field. For example, classification for unsupervised record linkage [9].

Our study is focused on optimal allocation of examination room supervisors against room sizes. Data matching is used to match student registration records with exam records, with some measure of accuracy, precision, recall, and other related data matching metrics. Smaller examination rooms mean data matching performs better but it requires more staff to carry out supervision.

III. PROBLEM CONTEXT

In this section, we discuss the Computing Skills Fundamentals course and its exam administration, as well as exam administration in general in the University of Botswana.

Computing Skills Fundamentals I and II are computer literacy courses for which most academic programmes in the University of Botswana require enrolled students to register for in their first year of studies. These courses are taught by the Department of Computer Science. Every semester, approximately 2000 to 4000 students register for one of the courses, depending on admission numbers into the university. The courses have a continuous assessment component as well as an exam component. The exam is a multiple choice exam, where students make use of special forms which are read by an Optical Mark Recognition (OMR) scanner when marking. Students shade the correct answer for 100 multiple choice questions. They also shade their identification details, such as student ID number, surname, initials, and faculty. While marking is very fast with the OMR scanner, a problem arises due to students making mistakes when shading their identification details. For example, instead of a student shading his/her student ID number as “333345678”, it might be shaded as “333345687” (the last two digits swapped). Or, due to the scanner only accepting a single digit per student ID number position, and using a “*” if two digits have been shaded in one position, “333345678” might appear as “3333456*” (with a star for two digits). Another common error is a missing character, for example the surname “Morapedi” might appear as “Mor pedi” (with a missing “a” due to accidental omission).

Various reasons exist for such mistakes. For example, exam pressure, time pressure, as well as older students who might have poor eye sight. These problems can be corrected manually, by going through all exam records, which are stored in CSV (Comma Separated Value) files (which can easily be imported into a spreadsheet application), and matching them with student registration records. However, this process is time consuming and errors can also be introduced if the person doing this gets tired. Data Matching is an approach which provides an automated solution to this problem. A combination of algorithms exist which can take a course registration list, and an exam registration list, and produce the most likely matches for records in the two lists. We have proposed and implemented a solution to this problem in a previous publication [1].

Another issue in exam administration in universities is cost of administration. Typically, it is cheaper to run exams in large rooms, than in small rooms, due to a minimum number of exam invigilators (proctors) who have to be present even in smaller rooms. Every hourly exam invigilator is payed for a duration of two hours for exams for the courses discussed in this paper. The more rooms are used, the higher the total amount which has to be paid. It is also observed that the fewer rooms are used, with larger group sizes per room, the bigger the impact on data matching performance. This was shown in a previous publication [10]. Various metrics are used to evaluate data matching performance, such as precision (the fraction of identified matches which are true positives), recall (the fraction of actual matches which are identified as matches), and pairs quality (the fraction of the total number of comparisons which are actual matches).

IV. OBJECTIVE FUNCTIONS AND OPTIMIZATION MODELS

In this section we discuss our approach to formulating objective functions for multi-objective optimization.

We model the problem as a multi-objective optimization problem, with two objectives: 1) monetary cost of exam resource usage, which depends on number of invigilators due to room sizes and number of rooms; 2) data matching performance, which is obtained by fitting a machine learning model to predict data matching performance for a particular room size.

A. Monetary Cost

We note that, every exam room requires at least two invigilators, irrespective of the number of students, in case of eventualities such as a student going to a restroom, who must be escorted, or a student falling ill, as well as to perform tasks in pairs, such as head counts, or counting the number of exam scripts after the exam. Also, in the Department of Computer Science, which offers the courses we are considering, the rule of thumb is to have one invigilator for every 30 students. However if there are 900 students in a room, 30 invigilator would not be required, since in large rooms less than 30 per student would be sufficient to oversee the whole room. We also assume that each invigilator would be paid BWP 300 for the exam (BWP is Botswana Pula; we just need to use an x amount for our purposes, hence have chosen to use 300 as this x amount, without disclosing actual university salary structure). The formula we use for total monetary amount is

If RoomSize ≤ 60

$$TotalCost = 2 \times CostPerInvigilator \times NumRooms \tag{1}$$

Else

$$TotalCost = CostPerInvigilator \times \left\lceil 2 + \frac{1}{2} \times \frac{(RoomSize - 60)}{30} \right\rceil \times NumRooms \tag{2}$$

Where

$$NumRooms = \left\lceil \frac{TotalNumberStudents}{RoomSize} \right\rceil \tag{3}$$

B. Data Matching Performance

Data matching process has several metrics associated with it, which include: Accuracy, Precision, Recall [2]. In order to calculate these, the concepts of TP (True Positives i.e. records pairs identified as matches which are actual matches), FP (False Positives i.e. record pairs identified as matches which are not actual matches), TN (True Negatives i.e. records pairs which are identified as non-matches, which are actual non-matches), FN (False Negatives i.e. record pairs identified as non-matches which are actual matches), are used. Accuracy = (TP+TN)/(TP+FP+TN+FN) shows how accurate the classification of record pairs is, considering how many TP and TN are detected out of all results produced. Precision indicates the accuracy of identification of TP: TP/(TP+FP). Recall measures the proportion of true matches that have been correctly classified: TP/(TP+FN). In addition, Reduction Ratio calculates the reduction in number of comparisons that have to be carried out, and Pairs Quality refers to the percentage of record pairs generated that correspond to TP (calculations for the previous two have been omitted to save space).

In order to implement the data matching performance as an objective function, we develop a machine learning model which maps exam group size, also known as room size, to data matching performance. We make use of linear regression [7], [11], [12] as our machine learning model. This is because of its simplicity and the fact that we got good results with it. Our data matching performance data was generated using 4116 student records for Computing Skills Fundamentals I for the period 2007-2013 semester 1. We made use of FEBRL [13],[14] (Fully Extensible Biomedical Record Linkage Systems). FEBRL is an application with several implementation for all components in a data matching pipeline: pre-processing, blocking/indexing (breaking up of lists into groups in order to reduce the number of comparisons), matching (computing scores for all pairs of records considered for matching), classification (for deciding whether a record pair is a match or not, based on a score vector), and analysis (computing scores for data matching performance and visualizing them as well). Given a course registration list with M records and an exam list with N records, there would have to be N×M comparisons and score computations, with large room for error. If however, the M records were broken up into two lists, and M records into two lists, and we know that we only need to match M1 with N1 and M2 with N2, there are fewer comparisons to make, and therefore less mistakes, and better data matching performance. This is the whole idea behind improving data matching performance by using smaller group (exam room) sizes.

Before the exam, students are assigned to exam rooms. So, if 50 students are assigned to room A, we keep that short list of course registration records separate. When processing exam records, we also keep the exam records for the 50 students separate (which is easy to do). Therefore when doing data matching, instead of matching more than 4000 course registration records with more than 4000 exam records, we match many groups of 50 course registration records with 50 exam records, thus improving data matching performance.

To fit our linear regression model, we used a set of 395 group sizes, with 300 kept for training, and 95 used for testing to computer an R^2 score (the Coefficient of Determination). This score is a real number which varies from 0 to 1, with 1 being a perfect fit (so perfect prediction properties), while a score close to 0 is a very poor fit. We chose an appropriate metric for data matching performance, which has both a good model fit as well as usefulness as a measure for data matching quality which can reduce manual work in data matching.

V. EXPERIMENTS AND RESULTS

In this section we discuss the implementation of our framework and results based on our data.

A. Machine Learning Model

We made use of the Python scikit-learn machine learning library [15] for our linear regression model fitting. We used a data set of 395 feature vectors (group size) from group size of 5 up to group size 1975, in increments of 5. For each group size, we had metrics for data matching performance for Accuracy, Precision, Recall, F-Measure, Reduction Ratio, and Pairs Quality. These metrics are explained in detail in [2]. Table I shows R^2 values for the various metrics. Figures 1 and 2 show plots for Precision vs Group Size and F-Measure vs Group Size. The blue circles are data points while the red line is the regression line (the model). For the purpose of our study, we opted to use F-Measure because it has a fairly good fit, and tries to balance precision and recall, which is important [16].

TABLE I R^2 VALUES FOR VARIOUS DATA MATCHING METRICS

Metric	**Value**
Accuracy	0.11766
Precision	0.92939
Recall	0.78834
F-Measure	0.93361
Reduction Ratio	0.99688
Pairs Quality	0.07349

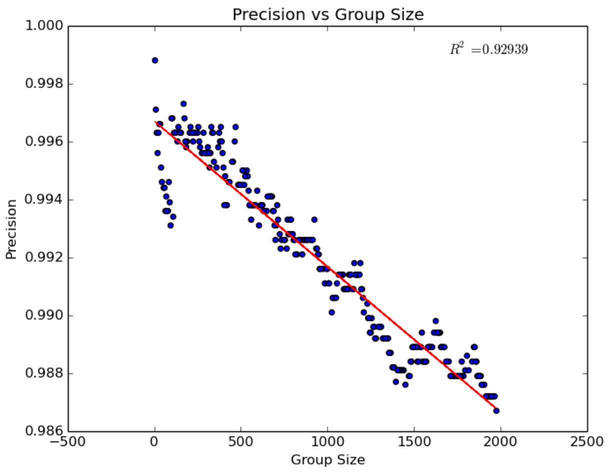

Figure 1: Precision

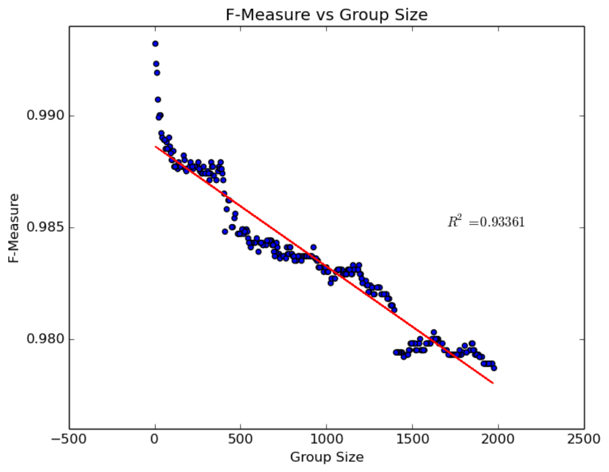

Figure 2: F-Measure

B. Multi-Objective Optimization

We made use of PyGMO (Python Parallel Global Multiobjective Optimization) library [17] for our multi-objective optimization. In the library, we made use of the VEGA (Vector Evaluated Genetic Algorithm) [18]. With Genetic Algorithms [19], a population of candidate solutions is randomly created, with each individual in the population being represented by a chromosome (a vector). A new generation of solutions is created through a process of mating and mutation, usually with a tendency to select the "fittest" individuals (according to the objective functions). After several (or many) generations (iterations), a good (or perfect) solution can be found. In VEGA [17], [20], the initial population is split into a number of objectives. Each sub-population is associated with one objective as a fitness function, and sub-populations are selected in a proportional selection process. The subpopulations are then put in a mating pool and the resulting population is mutated. VEGA produces a pareto front, which is a collection of candidate solutions, in which none is worse than the others in the set. Figure 3 shows a subset of the pareto front produced in our experiment. It is important to produce a pareto front because there is no single "best" solution. Each solution is better than others in one of the two objective functions, but worse in the other one.

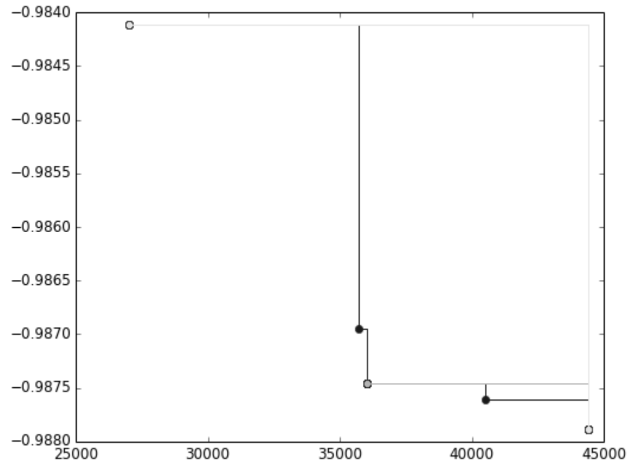

Figure 3: Pareto Front with 5 Solutions(F-Measure×-1 vs Monetary Cost)

C. Results

Table II shows the results for four members of the pareto set, including the monetary cost and F-Measure score. It is evident that, when looking at F-Measure and Monetary Cost, no solution (Group Size) dominates the others i.e. each solution is better than the others in one objective function, but worse in the other one. Note that F-Measure is multiplied by -1 to convert the problem from a maximization problem into a minimization problem. With such a pareto set, an administrator can decide which solution s/he wants to use base on how significant monetary cost is versus data matching performance. If monetary cost is very important, he will pick a solution with the lowest cost, at the expense of worse data matching performance, and therefore more manual work needed to match records. If data matching performance is more important, s/he will pick a solution which maximizes data matching performance, at higher monetary cost due to more exam rooms and a higher number of exam invigilators.

TABLE II CANDIDATE SOLUTIONS IN PARETO SET

Reference Number	**Group Size**	**-1×F-Measure**	**Monetary Cost**
0	358	-0.9867018425	29400.00
1	228	-0.9874019460	33000.00
2	278	-0.9871326754	32400.00
3	838	-0.9841168448	27000.00

VI. CONCLUSIONS

We have discussed a solution towards resource planning for university exam administration for large classes, such as those for Computing Skills Fundamentals I and II in the University of Botswana. Our approach is based on machine learning and multi-objective optimization, taking data matching performance into account. Our implementation shows that relevant tools are readily available and an entire system can be implemented by a small group of developers.

ACKNOWLEDGMENT

The authors wish to thank the University of Botswana and the Department of Computer Science for their efforts towards improving the administration of courses such as Computing Skills Fundamentals I and II.

REFERENCES

[1] G. Anderson, A.N. Masizana, and D. Mpoeleng, “An Exact and Inexact Data Matching Approach for Saving Time and Preventing Errors in Processing of Student Exam Results at the University of Botswana,” *International Journal on Information Technology (IREIT)*, vol. 1, no. 3, pp. 179-185, May 2013.
[2] P. Christen, *Data Matching*, Berlin, Germany: Springer, 2012.
[3] A.L. Bolaji, A.T. Khader, M.A. Al-Betar, and M.A. Awadallah, “University course timetabling using hybridized artificial bee colony with hill climbing optimizer,” *Journal of Computational Science*, vol. 5, no. 5, pp. 809-818, September 2014.
[4] T. Thepphakorn, P. Pongcharoen, and C. Hicks, “An ant colony based timetabling tool,” *International Journal of Production Economics*, vol. 149, pp. 131-144, March 2014.
[5] Á.P. Dorneles, O.C.B. de Araújo, and L.S. Buriol, “A fix-and-optimize heuristic for the high school timetabling problem,” *Computers & Operations Research*, vol. 52, pp. 29-38, December 2014.
[6] J.K. Rai, A. Negi, R. Wankar, and K.D. Nayak, “Characterizing l2 cache behavior of programs on multi-core processors: Regression models and their transferability,” in *Proc. World Congress on Nature and Biologically Inspired Computing, NaBIC 2009*, pp. 1673-1676.
[7] C.M. Bishop, *Pattern Recognition and Machine Learning*, New York, NY, USA: Springer Science+Business Media, 2006.

[8] T. Weise, *Global Optimization Algorithms – Theory and Applications*. [Online]. Available: http://www.it-weise.de/projects/.

[9] P. Christen, “A Two-Step Classification Approach to Unsupervised Record Linkage,” in *Proc. of the Sixth Australasian Data Mining Conference AusDM 2007*, pp. 111-119.

[10] G. Anderson, A.N. Masizana, and D. Mpoeleng, “Determining Optimal Blocking Sizes for Large Exam Groups When Data Matching,” *International Journal on Information Technology (IREIT)*, vol. 2, no. 3, pp. 80-86, May 2014.

[11] T. Hastie, R. Tibsharani, and J. Friedman, *The Elements of Statistical Learning*, 2nd Edition, Berlin, Germany: Springer, 2008.

[12] R.V. Kumar and R. Chandrasekaran, “Prediction of Software Development Cost and Effort Using Multiple Linear Regression,” *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 5 no. 5, pp. 338-346, May 2015.

[13] P. Christen, “FEBRL – A Freely Available Record Linkage System with a Graphical User Interface,” in *Proc. HDKM 2008*, Wollongong, NSW, Australia, 2008, pp. 17-25.

[14] P. Christen, *FEBRL: Freely Extensible Biomedical Record Linkage Manual Release 0.4.2*, http://sourceforge.net/projects/febrl/.

[15] 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*, vol. 12, pp. 2825-2830, 2011.

[16] C.D. Manning, P. Raghavan, and H. Schütze, *An Introduction to Information Retrieval*, Cambridge, UK: Cambridge University Press, 2008.

[17] D. Izzo and F. Biscani (2015) PyGMO: Python Parallel Global Multiobjective Optimizer. [Online]. Available: https://esa.github.io/pygmo/index.html.

[18] J.D. Schaffer, “Multiple Objective Optimization with Vector Evaluated Genetic Algorithms,” in *Proc. 1st International Conference on Genetic Algorithms*, 1985, pp. 93-100.

[19] B. Gupta and S. Dhingra, “Analysis of Genetic Algorithm for Multiprocessor Task Scheduling Problem,” *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 3 no. 7, pp. 339-344, July 2013.

[20] I.K. Gupta and J. Kumar, “VEGA and MOGA an Approach to Multi-Objective Optimization,” *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 5 no. 4, pp. 865-870, April 2015.