Optimized Association Rule Mining with Genetic Algorithms

Peter P. Wakabi–Waiswa* and Venansius Baryamureeba†
Makerere University, Kampala, Uganda.
*pwakabi@hotmail.com, †barya@cit.mak.ac.ug

Karunakaran Sarukesi
Hindustan University, Chennai - 603 103, India
profsaru@yahoo.com

***Abstract*—The mechanism for unearthing hidden facts in large datasets and drawing inferences on how a subset of items influences the presence of another subset is known as Association Rule Mining (ARM). There is a wide variety of rule interestingness metrics that can be applied in ARM. Due to the wide range of rule quality metrics it is hard to determine which are the most 'interesting' or 'optimal' rules in the dataset. In this paper we propose a multi–objective approach to generating optimal association rules using two new rule quality metrics: syntactic superiority and transactional superiority. These two metrics ensure that dominated but interesting rules are returned to not eliminated from the resulting set of rules. Experimental results show that when we modify the dominance relations new interesting rules emerge implying that when dominance is solely determined through the raw objective values there is a high chance of eliminating interesting rules. Keywords: optimal association rules, genetic algorithms, multi–objective interestingness metrics**

I. Introduction

The Association Rule Mining (ARM) approach to data mining was introduced in [1] as a structured mechanism for unearthing hidden facts in large datasets and drawing inferences on how a subset of items influences the presence of another subset. Quite a number of approaches to mining association rules do exist. There are those that generate all rules that satisfy a given hard constraint such as predictive accuracy while others search only promising regions of the database [2]. Another category of association rule miners are those that target specific sets or 'optimal' rules [3] according to some chosen rule interestingness metrics. The ARM problem is normally modeled as a multi–objective optimization problem when faced with several metrics.

The most popular approach to multi–objective ARM is to use the weighted sum method [4] where the fitness value of a candidate rule is derived using a linear transformation formula. This approach is widely employed because of its simplicity [5] but it is also faced with several drawbacks [6], [7]. The most suitable approach to ARM with multiple interestingness metrics is the use of the concept of Pareto–dominance. The Pareto–dominance approach can be explained as follows. Given a problem with several incommensurate objectives to be maximized, a solution a is said to dominate another solution b, if and only if, a is better than b in at least one of the optimization objectives, and a is not inferior to b in any of the optimization objectives. Formally put, a general multi–objective optimization problem is a vector function that maps decision variables to a number of objectives. Let y be a vector of objectives and x a decision vector. Then the multi–objective optimization problem is to minimize/maximize:

$$y = f(x) = (f_1(x), f_2(x), \ldots, f(x_n)) \quad (1)$$

where $x = (x_1, x_1, \ldots, x_m) \in X$ and $y = (y_1, y_1, \ldots, y_n) \in Y$.

The set of solutions of a multi–objective optimization problem contains all decision vectors that cannot be improved in any of the objectives without degrading any other objective. These decision vectors cannot be dominated by another solution in the search space and they are referred to as *non–dominated* or *Pareto–optimal*. All non–dominated solutions are known as the *Pareto–optimal set* or *Pareto–optimal front*, and it describes the trade–off surface of the multiple objectives of the problem.

In this paper, we propose two metrics; *transactional superiority* (T_s) and *syntactic superiority* (S_s), to be used as the objectives for optimizing the set of interesting association rules. Transactional superiority is used to decide dominance if there is a difference in the records that support competing rules and syntactic superiority measures the difference in rules in the attribute space.

The rest of the paper is organized as follows. We describe the algorithm in Section II. We evaluate the performance of the algorithm and study its performance sensitivity to parameters in Section III. Finally, we discuss our findings and conclude the paper in Section IV.

A. Mining Optimal Association Rules

A popular approach to mining optimal rules is to establish a partial ordering of the rules based on a set of metrics [8], [9]. The are many rule interestingness metrics including support, confidence, conviction, lift, Laplace, gain, gini, and the chi–square value [10], [11]. It was proved in [12] that the 'best' rules according to the different interestingness metrics are Pareto−optimal with respect to support and confidence partial ordering but it has been shown that using support–confidence optimality criteria results in rules with the same support–confidence combination being excluded from the results set [9], [13]. We therefore, propose new metrics to avoid that pitfall.

The proposed algorithm generates an optimal set of rules comprised of only those that are Pareto–optimal with respect

978-1-4244-9953-3/11/$26.00 ©2011 IEEE

to *transactional superiority* (T_s) and *syntactic superiority* (S_s). These rules are generated by placing a partial ordering, $\leq_{T_s} S_s$, on rules, r_1 and r_2, where $r_1 \leq_{T_s S_s} r_2$ if and only if the following conditions hold [12]:

- $T_s(r_1) \leq T_s(r_2) \wedge S_s(r_1) < S_s(r_2)$
- $T_s(r_1) < T_s(r_2) \wedge S_s(r_1) \leq S_s(r_2)$

and, $r_1 =_{T_s S_s} r_2$, if and only if, $T_s(r_1) = T_s(r_2) \wedge S_s(r_1) = S_s(r_2)$.

The partial ordering criteria for ARM is faced with the problem that if two rules have the same combination of the two optimality objectives, only one of the rules may be kept in the Pareto–optimal set [14]. There are circumstances when the two rules are different either in attribute space or they could be describing a different subset of records [15], [16]. So the Pareto–optimal rule set may not be suitably representative of the most interesting rules underlying the database. We, therefore, make further modification to the dominance relations to ensure that rules that are different in attribute space (syntactic difference) or cover different sets of rules are always maintained in the set of records returned to the user.

II. The Proposed Algorithm

We propose the Mining Optimized Association Rules Algorithm (MOAR) which maintains two populations: the internal population P, and a Pareto−store, $\ddot{P}$. Within each generation individuals are selected from the set of the non–dominated solutions for reproduction using crossover and mutation before updating the Pareto–store.

A. Representation of the Rules

We adopted a modified Michigan approach proposed in [17] whereby the encoding/decoding scheme associates two bits to each attribute in the database. If these two bits are 00 then the attribute next to these two bits appears in the antecedent part and if it is 11 then the attribute appears in the consequent part. The other two combinations, 01 and 10 indicate the absence of the attribute from the rule. Following this approach the algorithm can handle variable length rules with more storage efficiency, adding only an overhead of $2k$ bits, where k is the number of attributes in the database.

B. Initialization

We initialize the individuals by selecting a training instance to act as a "seed" for rule generation as proposed in [18]. In this approach, a seed is a data instance lying in the middle of a cluster of examples with a common consequent. The training examples are stored in an array and iteratively for each example we calculate the fraction, ρ, of those that cover the consequent against those that negate the coverage of that consequent. The training example with the highest ρ is selected as the seed.

C. Reproduction

The reproduction mechanism involves rule selection and the application of the crossover operators. The rule selection method used by this algorithm follows the "universal suffrage" approach proposed in [19]. With this approach each association rule is represented by a single individual. The individuals to be mated are elected by training data instances. Each instance votes for a rule that it covers in a stochastic fitness–based way.

1) The Crossover Operator: We modified the standard crossover operator to either generalize the crossover operator if the rule is too specific, or to specialize it if the rule is too general. A rule is considered too specific if it covers too few data instances i.e. when too few data instances satisfy both the antecedent and the consequent of the rule. In contrast, a rule is considered too general when it covers too many data instances. We make use of the bitwise OR and the bitwise AND to implement generalization and specialization, respectively. The bitwise OR and the bitwise AND are applied to the antecedent part of the rule since this determines the consequent part [19].

The generalization/specialization crossover procedure first constructs an index, $I = \{i_1, ..., i_n\}$, of pointers to the positions in the chromosome where the corresponding bits in the two parents have different values. Then, for every element $i_k \in I$ the following procedure is repeated. If the rule is too general, the algorithm replaces the value of the bits $b(i_k)$ with the logical OR of the corresponding bits in the parents, otherwise if the rule is too specific the algorithm replaces the value of the bit $b(i_k)$ with the logical AND of the corresponding bits in the parents.

2) The Mutation Operator: We adopted the $non - uniform - mutation$ operator proposed in [20]. The non-uniform mutation operator adapts to the environment by varying as the fitness of the individuals in the population change. We made a modification to the non–uniform mutation operator to enable it to generalize and/or specialize a *rule condition*. The resulting mutation operator randomly selects a condition from the rule. If that condition involves a nominal attribute, then the value is randomly changed from one value to another. If the attribute is continuous, the operator randomly changes the condition's interval values.

D. Replacement

We use an elitist individual replacement approach that ensures that more fit genotypes are always introduced into the population.

1) Uniqueness testing: The application of the genetic operators on the parent population may result in identical genotypes in the population. The algorithm first tests to ensure the new offspring do not duplicate any existing member of the population. The adoption of a replacement strategy with genotype uniqueness enforced preserves genetic diversity within the population [21]. Genetic diversity is significant as crossover–type operators depend on recombining genetically dissimilar genotypes to make fitness gains. Uniqueness of the

genotypes permits the algorithm to find several very good genotypes in a single run.

E. Fitness Assignment and Dominance Calculation

Our algorithm adopts the Pareto dominance approach proposed in [22] for maintaining multiple stable niches. This ensures that the individuals in the Pareto store are uniformly distributed near the Pareto−optimal front. To determine the dominance of an individual all individuals in the internal population and the Pareto store are evaluated. For every generation, t, each individual in both the internal population, P_t, and Pareto store $\ddot{P}_t$, is assigned a dominance strength.

Two rules could have the same values of the objectives to be optimized but one rule may be more interesting than the other rule. We term the more interesting rule *superior*. This superiority may be caused by a rule providing more information to the user or when the rule underlies more records than the other rule does. To establish superiority when a rule, r_1, dominates another rule, r_2, we calculate the difference between the rules. This difference is a measure of *'superiority'*. So if rule r_1 provides enough superiority it is permitted to remain non–dominated.

The proposed dominance relation is derived from the number of transactions *(transactional superiority)* that a rule covers as well as its syntactic composition *syntactic superiority*. *Transactional superiority* is a measure of the number of records that satisfy a given rule, *r*, and it is calculated as follows. We count the number of records, r_a, that satisfy a given antecedent and those that satisfy the consequent, r_c, of a given rule r. We then calculate the number of records that satisfy both the antecedent and the consequent, denoted by r_X. The support set of r, $S(r)$, is the set of all records that satisfy r and it is given by $S(r) = S(r_a \wedge r_c)$. The support for r, is given by $Support(r) =\mid S(r) \mid$ and the confidence of r, $confidence(r)$, is calculated as $confidence(r) = \frac{Support(r)}{Support(r^a)}$. The transactional superiority, T_s, of rule r_1 over r_2, is calculated as follows:

$$T_s(r_1, r_2) = \frac{|S(r_1) - S(r_2)|}{|r_c|}. \quad (2)$$

Transactional superiority will be used to decide dominance if there is a difference in the records that support either rule. In the event that there is no difference in records that support both rules, we use *syntactic superiority*. *Syntactic superiority* measures the difference in rules in the attribute space. When calculating syntactic superiority we establish the number of attribute tests (items) in which a field participates and we take their conjunction. We measure *syntactic superiority* as follows. Let R_1A, be the event that a record matches the antecedent of rule r_1, let R_2A be the event that a record matches the antecedent of rule r_2 and let C be the event that a record matches the consequent of both rules r_1 and r_2. Let R_1A_i be the conjunction of all *items* in the antecedent of r_1 and let R_2A_i be the conjunction of all *items* in the antecedent of r_2. Then syntactic superiority, $S_s(r_1, r_2)$, is given as (3):

$$S_s(r_1, r_2) = \sum_{i=0}^{n} (P(R_1A_i \mid C) - P(R_2A_i \mid C)) \quad (3)$$

where *n* is the number of attributes in the database. The overall superiority of rule r_1 over r_2 will be as (4):

$$\alpha(r_1, r_2) = T_s(r_1, r_2) + S_s(r_1, r_2) \quad (4)$$

With these rule superiority measures we now define a mechanism through which we'll determine which rule remains non–dominated. Applying the overall superiority on the objectives can now permit us to determine which rule remains non–dominated. This value is determined by the minimum amount of improvement in at least one of the objective values to ensure that a rule is no longer dominated. The minimum amount of improvement required is calculated as (5):

$$\gamma(r_1, r_2) = min\,(T_s(r_1) - T_s(r_2), S_s(r_1) - S_s(r_2))) \quad (5)$$

We modify the dominance relation with respect to the dominance improvement values to discover superior rules. The overall superiority multiplied by a constant value, δ, must be greater the dominance improvement. Therefore, rule r_1 dominates rule r_2, if and only if, $r_1 >_{S_sT_s} r_2$ and

$$\delta.\alpha(r_1, r_2) > \gamma(r_2, r_1). \quad (6)$$

F. Pareto Store Update

Our algorithm maintains a constant size Pareto store. Populating the Pareto store starts by copying all non-dominated individuals in the internal population (P_t) into the Pareto store $\ddot{P}_t$ in the first generation. In subsequent generations the population in the archive and the Pareto store are merged into a single one−dimension, $(1-D)$, array whose size is equal to $\mid P_t \mid + \mid \ddot{P}_t \mid$. The one $(1-D)$ array is sorted according to the fitness values ($r_i \leq_{S_sT_s} r$,) of the individuals. After sorting, the first $\mid P_t \mid$ members of the array are copied into the new Pareto store, $\ddot{P}_{t+1}$. This is an adaptation of the population archival procedure in [22].

G. Mating Selection

The selection mechanism employed in this research is a binary tournament selection method whereby two individuals i and j are randomly selected from the a Pareto store. Their fitness are compared. If individual i is more fit than j then i is selected otherwise, j wins the tournament and is selected for reproduction.

H. Solution evaluation

The algorithm iteratively evaluates a solution by first decoding the bit string into a rule which is compared to the data in the database. For all records that cover a given consequent the values are compared against the consequent to calculate the supports of the antecedent and the consequent. After the dataset scan, the measures of strength used as the objectives are calculated for each rule.

III. Evaluation of the Algorithm

We assessed the performance of our proposed algorithm in comparison with MOGAMAR [23], a multi–objective ARM algorithm modified to find S_sT_s-optimal rules.

A. Datasets

To evaluate our algorithm we used Adult, Connect-4, Chess, Diabetes, DNA and Mushroom datasets from UCI repository of machine learning databases [24]. We also used the Motor Vehicle Licensing System (MVLS) database from the Uganda Revenue Authority (URA) which we processed for use in experiments. The MVLS database contains data pertaining to motor vehicle licensing fees, vehicle ownership and registration, transfer of ownership, accidents, usage, country of origin, date or year of manufacture. The MVLS has over 4 million records but we randomly selected 441,327 transactions for these experiments. The summary of the datasets is given in Table I.

Dataset	No. of Instances	Attributes
Adult	48,842	14
Chess	3,196	36
Connect-4	67,557	42
Diabetes	768	20
DNA	3190	62
Mushroom	8,124	22
MVLS	441,327	32

TABLE I
SUMMARY OF THE DATASETS

B. Evaluation Metrics

The performance characteristics that we specifically analyzed included the following: execution time of the algorithm; the number of optimal rules returned by the algorithm; the spread of the results; and the effect of modifying the the dominance relation.

Dataset	MOAR		MOGAMAR	
	No. of Rules	Time (Secs)	No. of Rules	Time (Secs)
Adult	31	0.35	29	0.34
Connect-4	805	12	795	19
Chess	100	0.01	105	0.015
Diabetes	19	0.015	21	0.017
DNA	33	0.23	33	0.21
Mushroom	15	0.03	17	0.04
MVLS	900	900	905	903

TABLE II
NO. OF RULES GENERATED AND RUN TIMES WITH STANDARD S_sT_s-DOMINANCE

C. Using standard $_{S_sT_s}$ dominance

We first study the relative performance of our algorithm using the standard $_{S_sT_s}$–dominance relation. Table II contains the number of optimal rules generated by the algorithms together with the execution times. It is evident that the algorithms

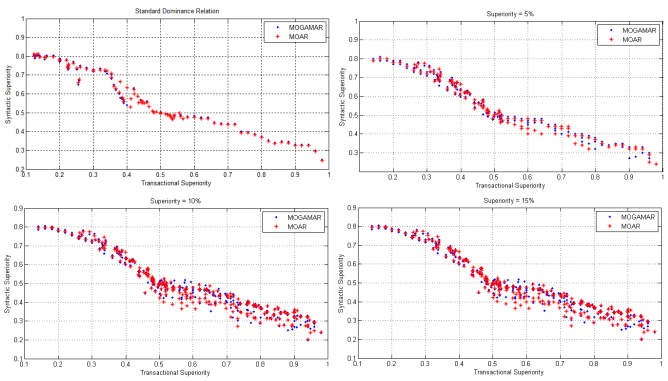

Fig. 1. Rules obtained with standard $_{S_sT_s}-$dominance (top–left), $\delta = 0.1$ (top–right), $\delta = 0.2$ (bottom–left), $\delta = 0.3$ (bottom–right)

generate optimal rules in less than a second for most of the datasets except Connect–4 and MVLS datasets. The algorithm finds these two datasets particularly difficult because of two primary reasons: The first reason is that they have substantially much more records and attributes than the others. Secondly, rules in these datasets with a high confidence have a very low support. This greatly affects the search mechanism because the algorithm tries to find diverse rules that meet these conditions thereby taking quite a bit of time. The most immediate solution to this is to reduce support and confidence, which results in tremendous reduction in execution time. We plot the $_{S_s}-T_s-$ border to get the spread of the rules generated using the standard $_{S_sT_s}-$ dominance shown in top–left of Figure 1. It can be seen that the results obtained by MOAR and MOGAMAR generally have a common trend.

D. Effects of modifying the dominance relations

We then study the effect of modifying the dominance relations using rule superiority with varying values of superiority weight (δ). Absolute superiority was experiments with by varying its value from 10% in increments of 10 up to 30%. The spread of the Pareto–optimal rules produced by the algorithm is shown in Figure 1. It can be seen that as rule superiority weight (δ) increases, the number of optimal rules produced increases implying that some rules that could have been removed due to $_{S_sT_s}-$dominance do appear. This shows that modifying the dominance relations with respect to superiority has a positive impact on the discovery of optimal rules. It is evident that new patterns emerge as the value of δ increases, suggesting that probably completely new rules may be found. The main discovery here is the use of standard $_{S_sT_s}-$dominance optimality criteria gets some rules dominated and consequently left out. It is clear that modifying the dominance relations results in lots more rules than when using the standard $_{S_sT_s}-$dominance (see Table III). Furthermore, the higher the δ, the more execution time is required to find an optimal set of rules. We therefore, need to find a mechanism for summarizing the rules in such a way that the user is presented with user friendly results. This is an area we have identified for possible research efforts.

Dataset	MOAR		MOGAMAR	
	Rules(#)	Time(Secs)	Rules(#)	Time(Secs)
Adult	231,336	721.45	198,456	724.69
Connect-4	759,300	12,065.00	840,075	13,021.43
Chess	282,423	554.17	294,573	531.93
Diabetes	2,310	12.54	1,820	13.48
DNA	123,624	25.50	120,435	24.86
Mushroom	600	0.67	645	.68
MVS	2,436,280	18,765.42	2,730,450	20,136.03

TABLE III
AVERAGE NO. OF RULES GENERATED AND RUN TIMES WITH MODIFIED S_sT_s-DOMINANCE

IV. DISCUSSION AND CONCLUSION

We have proposed a new multi–modal association rule mining algorithm that incorporates partial ordering in the search mechanism for rules. We have studied the impact of dominance relations to the number of rules found. Results from our studies show that using the standard $_{S_sT_s}-$dominance relations causes some interesting rules to remain unearthed to the user. We have observed that when we modify the dominance relations new rules in large numbers are found. This implies that when dominance is solely determined through support and confidence, there is a high chance of eliminating interesting rules. With more rules emerging it implies there should be a mechanism for managing their large numbers and also to significantly improve the response time of the algorithm. When we compare the spread of the rules for different levels of the superiority weight, δ, we observe that for increasing levels of δ more rules appear in bands. Experimental results show that the performance of our algorithm is good in comparison to an existing popular algorithm achieving better CPU performance in some instances. The introduction of the superiority measure causes more rules to be discovered which requires better presentation of the results to the user and it takes the algorithms longer to generate optimal rules. These are areas we have identified for future research works.

REFERENCES

[1] R. Agrawal, T. Imielinski, and A. Swami, "Mining association rules between sets of items in large databases," in *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C.*, 1993, pp. 26–28.

[2] V. Chaoji, M. A. Hasan, S. Salem, and M. J. Zaki, "An integrated, generic approach to pattern mining: data mining template library," *Data Mining and Knowledge Discovery*, vol. 17, pp. 457–495, December 2008.

[3] F. Stahl, M. Bramer, and M. Adda, "Parallel induction of modular classification rules," in *Twenty-eighth SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence*, 2008.

[4] D. Corne and J. Knowles, "Techniques for highly multiobjective optimisation:some nondominated points are better than others," in *Dirk Thierens, editor, 2007 Genetic and Evolutionary Computation Conference (GECCO2007)*, vol. 1. ACM Press, London, UK, 2007, pp. 773–780.

[5] F. M. Belém, E. F. Martins, J. M. Almeida, M. A. Gonçalves, and G. L. Pappa, "Exploiting co-occurrence and information quality metrics to recommend tags in web 2.0 applications," in *Proceedings of the 19th ACM international conference on Information and knowledge management*, ser. CIKM '10. New York, NY, USA: ACM, 2010, pp. 1793–1796.

[6] R. U. Kiran and P. K. Reddy, "An improved multiple minimum support based approach to mine rare association rules," in *CIDM*, 2009, pp. 340–347.

[7] F. Stahl, M. Bramer, and M. Adda, "Pmcri: A parallel modular classification rule induction framework," in *Proceedings of the 6th International Conference on Machine Learning and Data Mining in Pattern Recognition*, ser. MLDM '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 148–162.

[8] X. Yan, C. Zhang, and S. Zhang, "Confidence metrics for association rule mining," *Applied Artificial Intelligence: An International Journal*, vol. 23, pp. 713 – 737, 2009.

[9] R. U. Kiran and P. K. Reddy, "Mining rare association rules in the datasets with widely varying items frequencies," in *DASFAA*, 2010, pp. 49–62.

[10] F. E. B. Otero, A. A. Freitas, and C. G. Johnson, "A hierarchical multi-label classification ant colony algorithm for protein function prediction," *Memetic Computing*, vol. 2, no. 3, pp. 165–181, 2010.

[11] A. Salleb-Aouissi, C. Vrain, and C. Nortet, "Quantminer: A genetic algorithm for mining quantitative association rules," in *Proceedings of the 20th international joint conference on Artifical intelligence*, 2007.

[12] R. Bayardo and R. Agrawal, "Mining the most interesting rules," in *Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining, (KDD 99)*. AAAI Press, 1999, pp. 145–152.

[13] W. Y. Chen and J. Han, "Re-examination of interestingness measures in pattern mining: a unified framework," *Data Mining and Knowledge Discovery*, pp. 49–62, 2010.

[14] S. Dehuri, S. Patnaik, A. Ghosh, and R. Mall, "Application of elitist multi-objective genetic algorithm in classification rule generation," *Applied Soft Computing Journal*, vol. 8, pp. 477–487, 2008.

[15] M. A. Hasan and M. J. Zaki, "Musk: Uniform sampling of k maximal patterns," in *Proceedings of the 9th SIAM International Conference on Data Mining*, 2009.

[16] A. Surana, R. U. Kiran, and P. K. Reddy, "Selecting a right interestingness measure for rare association rules," in *15th International Conference on Management of Data COMAD 2010*, December 2010, pp. 8–10.

[17] A. Ghosh and B. Nath, "Multi-objective rule mining using genetic algorithms," *Information Sciences*, vol. 163, pp. 123–133, June 2004.

[18] A. A. Freitas, *Data mining and knowledge discovery with evolutionary algorithms*. Springer-Verlag Berlin Heidelberg, 2002.

[19] A. Giordana, C. Anglano, A. Giordana, G. L. Bello, and L. Saitta, "A network genetic algorithm for concept learning," in *Proceedings of the Sixth International Conference on Genetic Algorithms*. Morgan Kaufmann, 1997, pp. 436–443.

[20] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, New York, 1999.

[21] C. F. Lima, M.Pelikan, D. Goldberg, K. S. O G. Lobo, and M. Hauschild, "Influence of selection and replacement strategies on linkage learning in boa," in *Evolutionary Computation, 2007. CEC 2007. IEEE Congress*, 2008, pp. 1083–1090.

[22] E. Zitzler, M. Luamanns, and L. Thiele, "Spea2: Improving the strength pareto evolutionary algorithm," Computer Engineering and Networks Laboratory (TIK), Tech. Rep., 2001.

[23] P. P. Wakabi-Waiswa and V. Baryamureeba, "Mining high quality association rules using genetic algorithms," in *Proceedings of the twenty second Midwest Artificial Intelligence and Cognitive Science Conference*, April 2011, pp. 73–78.

[24] A. Frank and A. Asuncion, "Uci machine learning repository," School of Information and Computer Science, 2010. [Online]. Available: http://archive.ics.uci.edu/ml