SSDP: A Simple Evolutionary Approach for Top-K Discriminative Patterns in High Dimensional Databases

Tarcísio Pontes, Renato Vimieiro and Teresa Ludermir
Centro de Informática
Universidade Federal de Pernambuco
Recife-PE, Brasil,
Caixa Postal 7851 – 50732-970
Email: {tdpl,rv2,tbl}@cin.ufpe.br

***Abstract*—It is a great challenge to companies, governments and researchers to extract knowledge in high dimensional databases. Discriminative Patterns (DPs) is an area of data mining that aims to extract relevant and readable information in databases with target attribute. Among the algorithms developed for search DPs, it has highlighted the use of evolutionary computing. However, the evolutionary approaches typically (1) are not adapted for high dimensional problems and (2) have many nontrivial parameters. This paper presents SSDP (Simple Search Discriminative Patterns), an evolutionary approach to search the top-k DPs adapted to high dimensional databases that use only two easily adjustable external parameters.**

I. Introduction

Knowledge discovery in high dimensional databases is a challenge for companies, governments and researchers. Microarray databases are an important example of high dimensional problem. Microarray is a technology that allows measuring the expression thousands of genes in one experiment. Finding some combination of genes whose expression levels can distinguish some groups of patients (cancer vs. healthy, for example). Since microarray technology has developed databases for several important studies in Bioinformatics [1] [2] [3] [4] [5]. Microarray databases is having a revolutionary impact on molecular biology [2].

Discriminative Patterns (DPs) aims to find humanly interpretable subgroups where the presence of a label vs. others is exaggerated. From this, is possible to generate insights about a problem or just explain it in a simple way [5]. DPs have evolved rapidly with different terminologies (Subgroups Discovery [6] [7], Emerging Patterns [8] and Contrast Sets [9]).

However, mining best DPs in high dimensional databases is often computationally infeasible. In this way it highlights the development of heuristic algorithms based on *Evolutionary Computing* [10] [11] [12] [13] [14] [15] [16] and *Beam Search* [17] [18] [19] [20]. But none of evolutionary approach has been developed with focus on very high dimensional problem. Besides that, they use complex parameters and the user has no control over the amount of returned DPs.

This paper presents the SSDP (Simple Search Discriminative Patterns), a DPs mining approach focused on high dimensional problem based on *Evolutionary Computing* and *Beam Search*. SSDP uses simple parameters and returns the top-k DPs, where k is chosen by user. SSDP was developed in special for microarray databases, but it is a general solution for high dimensional problem.

Thus, we hope this work contributes to knowledge discovery task in Bioinformatics and other high dimensional problems. This paper is organized as follows. Section II summarizes the main DPs concepts. The Section III presents some related work, followed by Section IV, where the SSDP approach is described in detail. Section V shows the experiments and Section VI the results. Finally, Section VII presents the conclusion.

II. Discriminative Patterns (DPs)

The DPs problem can be defined as follows. Let D be a database where D^+ are positive examples (research target) and D^- the negative (other examples). DPs aim to find groups where the presence of positive examples is disproportionate in relation to negative. A DP is formed by one or more items (features). Each item consists of a pair $(attribute, value)$. The universe of all possible items of D is given by $I = \{i_1, i_2, ..., i_{|I|}\}$. A three dimensionality DP, for example, can be represented as follows: $dp_3 = \{i_a, i_b, i_c\}$, where $dp_3 \subseteq I$.

The analysis of all possible DPs for a given problem is usually an infeasible task. Thus, during the search process, the DPs are evaluated automatically (using one or more evaluation metrics). There are several types of evaluation metrics, but there is no consensus about the best one. This choice often depends on the problem or specialist convictions. In this way, it is important that the DPs search algorithms accept different options of evaluation metrics to meet user needs.

The metrics used to evaluate this work are described in Table I, where TP and FP are true positives and false positives DPs, k is the number of returned DPs and $|D|$, $|D^+|$ and $|D^-|$ are number of the total, positive and negative examples. Several other evaluation metrics can be found in [5] and [7].

TABLE I
DISCRIMINATIVE PATTERNS EVALUATE MEASURES.

Equation	Description
$Q_g = \frac{TP}{FP-g}$, default $g = 1$	*Trade off* between TP and FP [18]
$WRAcc = \frac{TP+FP}{\lvert D \rvert}(\frac{TP}{TP+FP} - \frac{\lvert D^+ \rvert}{\lvert D \rvert})$	Relative DP accuracy [21]
$DiffSup = \lvert \frac{TP}{\lvert D^+ \rvert} - \frac{FP}{\lvert D^- \rvert} \rvert$	Difference between positive and negative support [9]
$supp = \frac{TP}{\lvert D^+ \rvert}$	Average positive support [12]
$conf = \frac{TP}{TP+FP}$	Confidence [7]
$SUPP = \frac{1}{k} \sum_{i=1}^{k} supp^*$	Positive support by set of DPs (D^+ covered percentage) [12]
$size$	Average size of top-k DPs

The DPs search algorithm usually return the best DPs in one of two ways: (1) based on constraints, where it returned DPs with some constraint, as minimum support and minimum confidence and (2) based on top-k, where it returned the k best DPs determined according to a given quality function. Both options have their relevance depending on the analysis goals, but the top-k approach provides more flexibility for users [6].

There are several algorithms for DPs mining [5] [7]. The use of thresholds parameters are often in these approaches. However, setting values as minimum support and confidence is not a simple task. If it is too large, the algorithm can not return any results, if it is small can not represent a useful constraint.

III. RELATED WORK

There are several DPs mining approaches based on Evolutionary Computing [10] [11] [12] [13] [14] [15] [16]. However, most of the performance tests on evolutionary approaches were directed to problems with less than 40 attributes and none of them was validated to thousand dimensionality order.

Some important features, as initial population, show that some evolutionary approaches would have difficulty in high dimensional databases. In [10] [11] [12] 75% of individuals are generated up to 25% of items $i \in I$. Already [16] uses between 1% to 50% of the attributes. This type of initialization can be problematic in high dimensional databases. A problem where $|I| = 10000$, for example, an individual using 5% of I possibilities represent a DP with 500 dimensions. This hardly represents a valid solution and may hinder the algorithm convergence.

The individual representation is another example. In evolutionary approach it is often the use of fixed size individuals equal to $|I|$ [10] [11] [12] [14]. But in high dimension problems the items that are not used by best DPs is often more than 99%, the most genes is zero. Other approaches using dynamic size tree generated by grammars [15] [16], but to build grammars can not be a simple process.

Another feature present in some evolutionary approaches is the number and complexity of the parameters. Table II summarizes some of the parameters required by six evolutionary approach. The definition of such parameters is not a trivial task and may hinder the use of these algorithms. It is also common in current evolutionary approaches the user has no control over the amount of DPs returned.

TABLE II
SUMMARY OF PARAMETERS USED BY 6 EVOLUTIONARY TECHNIQUES TO SEARCH DISCRIMINATIVE PATTERNS.

Parameter	SDIGA [10]	MESDF [11]	NMEEF [12]	EDER [14]	GP3 [15]	FuGeP [16]
Fitness	X	X	X	X	X	X
Linguistic labels	X	X	X			X
Crossover		X	X			X
Mutation	X	X	X			X
Population	X	X	X	X	X	X
Elite size		X				
Evaluations	X	X	X			
Generations				X	X	X
Confidence	X		X		X	X
Support				X		
Sensitivity						X
Total	6	7	7	4	4	8

Finally, few studies have considered the efficiency of evolutionary methods with respect to processing time. In high dimensional databases context, time is often critical. In the next section is explained in detail SSDP algorithm, an evolutionary approach that has as main features: (1) focused on high dimensional problems, (2) uses only k and the metric evaluation as external parameters and (3) it allows the user to choose the number of DPs want to receive.

IV. SSDP: SIMPLE SEARCH DISCRIMINATIVE PATTERNS

SSDP uses important concepts of different search algorithms, they are:

- In [22] was presented an evolutionary algorithm to search Diverse-Frequent Pattern (a type of patterns similar to DPs) in high dimensional databases. The algorithm includes to the next generation the best individuals from old population P_{old} and others newly created by genetic operators ($P_c \leftarrow crossOver(P_{old})$ and $P_m \leftarrow mutation(P_{old})$), where the size of populations are equal ($|P_{old}| = |P_c| = |P_m|$). That is, $P_{new} \leftarrow best(P_{old}, P_c, P_m)$. SSDP uses this process to generate new populations.
- *Beam Search* is an efficient search strategy used in some DPs algorithms, like Subgroup Miner [17], SD [18], CN2-SD [19] and RSD [20]. There are two important features in *Beam Search* algorithm. One is to initialize the search from all one dimension DPs. This ensures that all items $i \in I$ are considered in the search. The other feature is that the searches in the dimension d are made from the best DPs smaller than d. In SSDP the initial population is formed by all one dimension DPs and the genetic operators expand the search to other dimensions.

- SD [18] is an algorithm that ensures that all DPs stored along the search are relevant. A solution dp_a is considered irrelevant to a set DP if there is $dp_b \in DP$ that dp_a covers a subset of the positive samples and all the negative examples of dp_b. With this concept the algorithm eliminate redundancies among the top-k DPs. SSDP algorithm uses this concept only for k best DPs.

The most important parts of the SSDP algorithm are described below:

A. Representation

The individuals have variable size and represent only items used by DP. Thus, each individual is represented by integers (or index) that is the item position i in I. For example, $dp = \{2043, 213\}$ is a dp_2 composed by items at position 2043 and 203 from I.

B. Initialization and population size

The initial population is composed of all one dimentional possible DPs. That is, for each $i \in I$ an individual is created (dp_1), where I is all possible items (attribute value pairs) in the database. It represents a new way for initial population in evolutionary approach for DP problem.

C. Genetic operators

- Crossover: there are two possibilities: (1) *crossOverAND*, when two individuals unite their genes creating a new individual (used only in the first generation) or (2) *crossOverUniform*, where two individuals generate two new by uniform crossover with 50% mixing ratio.
- Mutation: there are two possibilities: (1) a new item is selected and added to the individual or (2) an old gene is replaced by new item. Both options with 50% probability.
- Selection: by binary tournament.

In each generation n new individuos are generated exclusively by crossover and other n exclusively by mutation. That is, SSDP considers the same importance to mutation and crossover operators. This is because, besides providing diversity, mutation is used to find unlikely DPs.

D. Stopping criterion

The algorithm stops when there are no changes in the top-k DPs for three consecutive generation.

E. Parameters and fitness

SSDP does not use some common parameters of other evolutionary DPs mining approaches, as mutation and crossover rate, population size and minimal support. It uses only two easily adjustable external parameters, they are:

- k: number of DPs returned to the end of the process. The k allows the user to have control over the amount of information that he wants to receive. It is also an intuitive parameter and does not require technical knowledge.
- Evaluating measure: function to evaluate DPs quality. The more functions, the more the algorithm becomes flexible for the user. SSDP theoretically allows the use of any objective function. Currently SSDP implementation has the following possibilities: Q_g, $WRAcc$ and $DiffSup$. The genetic algorithm uses the evaluating measure as fitness.

F. Algorithm

SSDP works with five population, where P, P_c, P_m and P_* size are $|I|$ and P_k size is k. They are:

- P: current population.
- P_c: generated from P by crossover.
- P_m: generated from P by mutation.
- P_*: generated by best individuals of P, P_m e P_c. It does not require that individuals are unique.
- P_k: keeps the best k individuals that are relevant. An individual is considered irrelevant in relation to P_k if it is a subset of positive and superset of negative examples for any $dp \in P_k$.

SSDP algorithm starts for all dp_1 possibilities and the genetic operators expand the search to larger dimensions. Thus, at first, the searches tend to be directed to larger dimension as best fitness individuals are found. In a second moment the individuals are becoming very specific, then, the fitness tends to worsen and the algorithm can return the searches for smaller dimension or converge.

The Algorithm 1 describes the SSDP approach. In it, the *kBestRelevants* function returns the best relevant individuals. Already the *best* function accepts repeated and not relevant individuals as a way to reduce the computational cost.

Algorithm 1 SSDP pseudocode

```
Require: $k$, $ObjectiveFunction$
  $P \leftarrow$ all dp1 possibilits ($i \in I$)
  $P_k \leftarrow kBestRelevants(P)$
  while $P_k$ not improve three times in a row do
    if generation == 1 then
      $P_c \leftarrow crossOverAND(P)$
      $P* \leftarrow best(P, P_c)$
    else {generation > 1}
      $P_c \leftarrow crossOverUniform(P)$
      $P_m \leftarrow mutation(P)$
      $P* \leftarrow best(P, P_c, P_m)$
    end if
    $update(P_k, P_*)$
    $P \leftarrow P*$
  end while
  return $P_k$
```

V. Experiments

The experiments start from 21 original microarray databases, described in Table III. Such databases are available in the package *datamicroarray* [4] from R software [23]. For each database the majority class was considered the target of searches (p) and other examples were labeled as negative (n). The attributes of databases are all numeric. They have been

TABLE III
MICROARRAY DATABASES DESCRIPTION

Name	N^o Examples	N^o Attributes	N^o Labels
alon	62	2,000	2
borovecki	31	22,283	2
burczynski	127	22,283	3
chiaretti	111	12,625	2
chin	118	22,215	2
chowdary	104	22,283	2
christensen	217	1,413	3
golub	72	7,129	3
gordon	181	12,533	2
gravier	168	2,905	2
khan	63	2,308	4
nakayama	105	22,283	10
pomeroy	60	7,128	2
shipp	58	6,817	2
singh	102	12,600	2
sorlie	85	456	5
subramanian	50	10,100	2
sun	180	54,613	4
tian	173	12,625	2
west	49	7,129	2
yeoh	248	12,625	6

discretized using methods based on frequency and width by 2, 4 and 8, totaling 126 discretized databases.

Each experiment was repeated 30 times, with the objective function Q_g ($g = 1$) and $K = \{5, 10, 20, 50\}$. SSDP performance was compared to the following algorithms:

- Random1M e Random2M: one and two million DPs randomly generated. The purpose of this comparison is to validate SSDP heuristic.
- ExaustiveK: DPs with highest fitness among all combinations of up to four dimensions, but using only the k best items. The purpose of this comparison is to validate the SSDP ability to find non-trivial DPs.
- SD-adapted: SD algorithm was adapted to search the same types of rules of SSDP approach. The SD is based on *Beam Search*. The aim is to confront SSDP with a competitive classical SD mining approach. SD used the following parameters: $beamWidth = 2 * k$ and $minimumSupport = \frac{\sqrt{|Dp|}}{|D|}$, as indicate by author [18].

VI. RESULTS

The results were divided into two parts. In first part the aim is to evaluate the SSDP search strategy. In the second the aim is to evaluate SSDP performance.

A. Validation SSDP search strategy

SSDP starts the search considering all items possibilities $i \in I$. Table IV shows the average size frequency of top-50 DPs from 126 databases. In 18 of them the top-50 DPs were found exclusively in the first dimension. At the same time, in 15 of them the average size was above 3. This shows that is unpredictable to know what dimensions are the best DPs. In this context, boot searches by the size of $d = 1$ and evolve into other dimensions d prioritizing the well evaluated DPs seems to be an effective strategy.

TABLE IV
AVERAGE SIZE FREQUENCY OF TOP-50 DPS IN 126 *microarray* DATABASES

Average size	**Frequency**
[1;1]	18
(1;2]	54
(2;3]	39
(3;4]	14
(4;5]	1

Figure 1 shows the evolution of DPs average size in populations P and P_k, for $k = 50$ from *West* database. So, in the first generation P and P_k are just dp_1. After that poorer quality dp_1 are replaced by higher best quality DPs. The P behavior shows that SSDP tends to evolve searches for larger dimensions but it can change the direction to smaller dimensions when required.

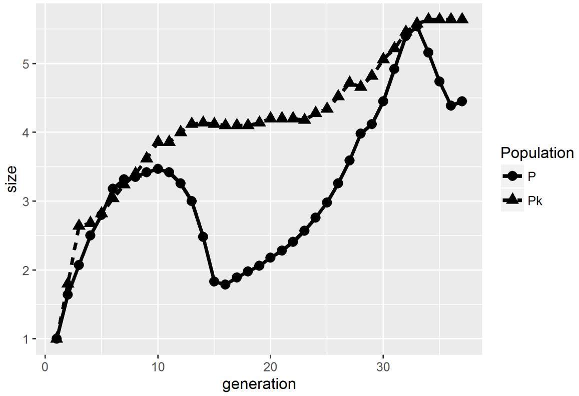

Fig. 1. DPs average size evolution in populations P and P_k, for $k = 50$ from *West* database.

B. Performance Analysis

Figure 2 and Figure 3 show respectively the average Q_g and time from SSDP, SD, *Random1M*, *Random2M* and *ExaustiveK* for $K = \{5, 10, 20, 50\}$ in 126 microarray databases. SSDP and SD obtained better average Q_g then random approach for all k values. The SSDP processing time is close to *Ramdom2M* for all k value, while SD used more time them *Ramdom2M* for $k = \{20, 50\}$. So, at first analysis it is possible to validate the heuristic SSDP. SSDP obtained better results than random approaches with closed time processing.

At second analysis it is possible to validate the SSDP regarding the ability to find nontrivial relevant DPs. Figure 2 shows that SSDP obtained better average Q_g then *ExaustiveK* for all k value. This feature also applies to the SD algorithm.

Finally, the comparison with the SD approach shows that SSDP is a promising approach in the context of top-k DPs for high dimensional databases. This is because the SSDP got considerably better DPs for all k values with time process slightly higher to $k = \{5, 10\}$ and a bit less for $k = \{20, 50\}$.

It is still applied the *Wilcoxon test* to evaluate if the performance between SSDP and SD was statistically significant. The *Wilcoxon* is a non-parametric test that has been indicated and used for performance analysis between two algorithm [24]

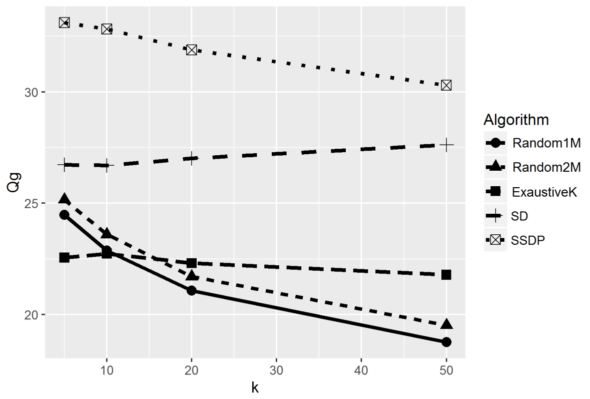

Fig. 2. Qg average for SSDP, SD, *ExaustiveK*, *Random1M* and *Random2M* in 126 microarray databases for different k values.

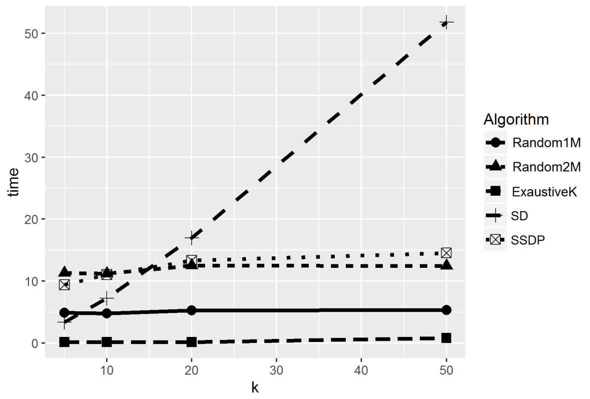

Fig. 3. Time average for SSDP, SD, *ExaustiveK*, *Random1M* and *Random2M* in 126 microarray databases for different k values.

[16]. Table V shows the result. In this way the null-hypothesis that SSDP perform equally well as SD are rejected for all k values for level of significance $\alpha = 0.01$.

TABLE V
RESULTS OF THE WILCOXON TEST BETWEEN SSDP AND SD

K	p-value	Hypothesis
5	1.67E-13	Rejected by SSDP
10	5.42E-12	Rejected by SSDP
20	4.29E-10	Rejected by SSDP
50	0.0005933	Rejected by SSDP

An important differential of heuristics in DPs mining problem is the search capability in larger dimensions. Figure 4 shows the average size of top-k DPs for $k = \{5, 10, 20, 50\}$ from all algorithms. It shows the more successful of SSDP in larger dimension search for all k values. That is the probably explanation for more SSDP superiority over other algorithms.

Finally, Figure 5 shows the percentage of samples covered by top-k DPs for different values k. The tested approaches is not intended to cover all the positive examples, four of them obtained $SUPP > 80\%$ for $k = 5$ and $SUPP > 90\%$ for $k \geq 10$.

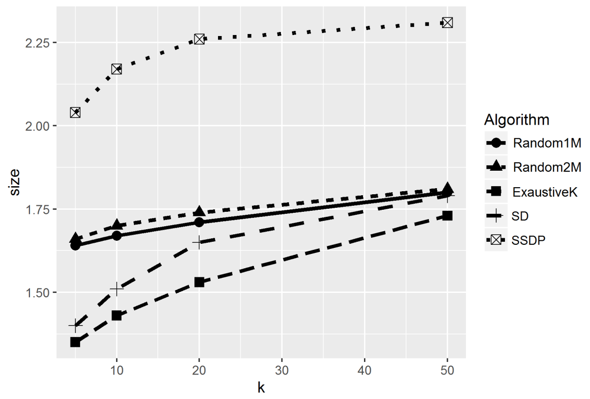

Fig. 4. Average size for SSDP, SD, *ExaustiveK*, *Random1M* and *Random2M* DPs in 126 microarray databases for different k values.

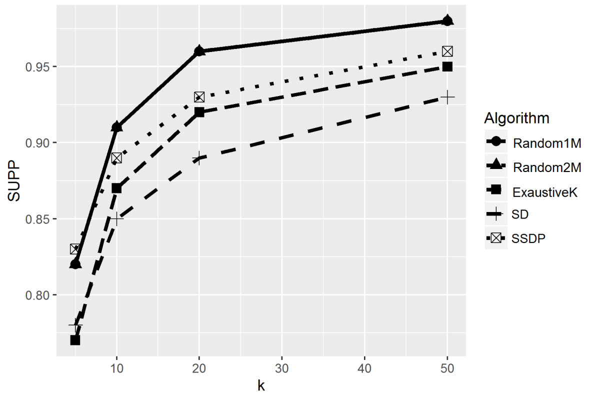

Fig. 5. Positive support by set of top-k DPs returned by *ExaustiveK*, *Random1M* and *Random2M* DPs in 126 microarray databases for different k values.

The exact values of Q_g, $time$, $size$, $SUPP$ average and other metrics as support and confidence average in all databases for $K = \{5, 10, 20, 50\}$ can be seen in Tables VI,VII,VIII and IX, respectively. It can be seen that SSDP also obtained DPs with greater confidence and support for all k values.

TABLE VI
$Q_g, time, size, supp, conf$ AND $SUPP$ AVERAGE FOR 126 MICROARRAY DATABASES FOR $k = 5$.

K = 5						
Algorithm	**Qg**	**time**	**size**	**conf**	**supp**	**SUPP**
SSDP	**33.12**	9.40	2.04	1.00	0.53	0.83
SD	26.73	3.34	1.40	1.00	0.45	0.79
ExaustiveK	22.57	0.14	1.35	0.99	0.42	0.78
Random1M	24.49	4.87	1.64	1.00	0.42	0.82
Random2M	25.16	11.32	1.67	1.00	0.43	0.83

VII. CONCLUSION

Microarray databases are having a revolutionary impact on molecular biology. But microarray databases are an high dimension problem. Discriminative Patterns (DPs) aims to find humanly interpretable subgroups where the presence of a label

TABLE VII

$Q_g, time, size, supp, conf$ AND $SUPP$ AVERAGE FOR 126 MICROARRAY DATABASES FOR $k = 10$.

K = 10						
Algorithm	**Qg**	**time**	**size**	**conf**	**supp**	**SUPP**
SSDP	**32.82**	11.06	2.18	1.00	0.53	0.90
SD	26.71	7.25	1.51	1.00	0.44	0.86
ExaustiveK	22.74	0.14	1.44	1.00	0.41	0.87
Random1M	22.88	4.81	1.68	1.00	0.39	0.91
Random2M	23.61	11.23	1.71	1.00	0.41	0.92

TABLE VIII

$Q_g, time, size, supp, conf$ AND $SUPP$ AVERAGE FOR 126 MICROARRAY DATABASES FOR $k = 20$.

K = 20						
Algorithm	**Qg**	**time**	**size**	**conf**	**supp**	**SUPP**
SSDP	**31.89**	13.33	2.27	1.00	0.52	0.94
SD	27.03	16.97	1.65	1.00	0.45	0.90
ExaustiveK	22.31	0.17	1.54	1.00	0.39	0.92
Random1M	21.07	5.29	1.72	0.99	0.37	0.96
Random2M	21.71	12.53	1.74	0.99	0.38	0.97

TABLE IX

$Q_g, time, size, supp, conf$ AND $SUPP$ AVERAGE FOR 126 MICROARRAY DATABASES FOR $k = 50$.

K = 50						
Algorithm	**Qg**	**time**	**size**	**conf**	**supp**	**SUPP**
SSDP	**30.30**	14.55	2.31	1.00	0.49	0.96
SD	27.62	51.81	1.79	1.00	0.45	0.93
ExaustiveK	21.79	0.81	1.73	1.00	0.38	0.95
Random1M	18.77	5.36	1.80	0.99	0.33	0.99
Random2M	19.52	12.46	1.81	0.99	0.35	0.99

vs. others is exaggerated. However, mining best DPs in high dimensional databases is often computationally infeasible. In this context, several evolutionary approaches were developed, but with little focus on high dimensional databases. They also often use many complex parameters and the user has no control over the amount of returned DPs.

This paper presented the SSDP, an evolutionary approach to search the top-k DPs adapted to high dimensional databases that use only two easily adjustable external parameters and the user can control the number of DPs returned. SSDP has as main concepts features: (1) the evolutionary strategy using concepts of *Beam Search* and (2) the simple and efficient way to represent individuals.

SSDP was validated as heuristic and the ability to find nontrivial DPs. The proposed approach also is superior to SD, a classical and competitive algorithm based on *Beam Search*. This work also showed the SSDP ability to change the focus of the search for larger or smaller as needed.

Finally, this study is being expanded to: (1) evaluate SSDP in other types of problems, (2) compare performance with newer approaches and (3) further experiments with statistical tests.

ACKNOWLEDGMENT

The authors would like to thank FACEPE (project IBPG-0666-1.03/13) and CNPQ (project APQ-0585-1.03/14) for supporting this research.

REFERENCES

[1] J. Quackenbush, "Computational analysis of microarray data," *Nature reviews genetics*, vol. 2, no. 6, pp. 418–427, 2001.

[2] M. Molla, M. Waddell, D. Page, and J. Shavlik, "Using machine learning to design and interpret gene-expression microarrays," *AI Magazine*, vol. 25, no. 1, p. 23, 2004.

[3] M. de Souto, A. Lorena, A. Delbem, and A. de Carvalho, "Técnicas de aprendizado de máquina para problemas de biologia molecular," *Sociedade Brasileira de Computaçao*, 2003.

[4] J. Ramey. (2016) The datamicroarray r package. [Online]. Available: https://github.com/ramhiser/datamicroarray

[5] X. Liu, J. Wu, F. Gu, J. Wang, and Z. He, "Discriminative pattern mining and its applications in bioinformatics," *Briefings in bioinformatics*, p. bbu042, 2014.

[6] M. Atzmueller, "Subgroup discovery," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 5, no. 1, pp. 35–49, 2015.

[7] F. Herrera, C. J. Carmona, P. González, and M. J. Del Jesus, "An overview on subgroup discovery: foundations and applications," *Knowledge and information systems*, vol. 29, no. 3, pp. 495–525, 2011.

[8] G. Dong and J. Li, "Efficient mining of emerging patterns: Discovering trends and differences," in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1999, pp. 43–52.

[9] S. D. Bay and M. J. Pazzani, "Detecting group differences: Mining contrast sets," *Data Mining and Knowledge Discovery*, vol. 5, no. 3, pp. 213–246, 2001.

[10] M. J. del Jesus, P. Gonzalez, F. Herrera, and M. Mesonero, "Evolutionary fuzzy rule induction process for subgroup discovery: A case study in marketing," *IEEE Transactions on Fuzzy Systems*, vol. 15, no. 4, pp. 578–592, Aug 2007.

[11] M. J. del Jesus, P. González, and F. Herrera, "Multiobjective genetic algorithm for extracting subgroup discovery fuzzy rules." in *MCDM*. Citeseer, 2007, pp. 50–57.

[12] C. J. Carmona, P. González, M. J. del Jesus, and F. Herrera, "Nmeef-sd: non-dominated multiobjective evolutionary algorithm for extracting fuzzy rules in subgroup discovery," *Fuzzy Systems, IEEE Transactions on*, vol. 18, no. 5, pp. 958–970, 2010.

[13] V. Pachón, J. Mata, J. L. Domínguez, and M. J. Maña, "Multi-objective evolutionary approach for subgroup discovery," in *Hybrid Artificial Intelligent Systems*. Springer, 2011, pp. 271–278.

[14] D. Rodríguez, R. Ruiz, J. C. Riquelme, and J. S. Aguilar-Ruiz, "Searching for rules to detect defective modules: a subgroup discovery approach," *Information Sciences*, vol. 191, pp. 14–30, 2012.

[15] J. M. Luna, J. R. Romero, C. Romero, and S. Ventura, *Discovering subgroups by means of genetic programming*. Springer, 2013.

[16] C. J. Carmona, V. Ruiz-Rodado, M. J. del Jesús, A. Weber, M. Grootveld, P. González, and D. Elizondo, "A fuzzy genetic programming-based algorithm for subgroup discovery and the application to one problem of pathogenesis of acute sore throat conditions in humans," *Information Sciences*, vol. 298, pp. 180–197, 2015.

[17] W. Klosgen and M. May, "Census data mining - an application." Proceedings of the 6th European conference on principles of data mining and knowledge discovery, 2002, pp. 65–79.

[18] D. Gamberger and N. Lavrac, "Expert-guided subgroup discovery: Methodology and application," in *J. Artif. Int. Res.* AI Access Foundation, 2002, pp. 501–527.

[19] N. Lavrac, B. Kavsek, P. Flach, and L. Todorovski, "Subgroup discovery with cn2-sd," in *Journal of Machine Learning Research*, S. Wrobe, Ed., 2004, pp. 153–188.

[20] F. Zelezny and N. Lavrac, "Propositionalization-based relational subgroup discovery with rsd," in *Machine Learning*, H. Blockeel, D. Jensen, and S. Kramer, Eds. Springer, 2006, pp. 33–63.

[21] N. L. Peter, P. Flach, and B. Zupan, "Rule evaluation measures: A unifying view," in *Proceedings of the 9th International Workshop on Inductive Logic Programming (ILP-99*. Citeseer, 1999.

[22] S. Khatun, H. U. Alam, and S. Shatabda, "An efficient genetic algorithm for discovering diverse-frequent patterns," 2015, vol. abs/1507.05275.

[23] J. Chambers. (2016) The r project for statistical computing. [Online]. Available: https://www.r-project.org/

[24] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.