An Incremental Approach for Niching and Building Block Detection via Clustering

Leonardo Ramos Emmendorfer
Numerical Methods in Engineering, PhD Program
Federal University of Paraná, Brazil
leonardo@inf.ufpr.br

Aurora Trinidad Ramirez Pozo
Department of Computer Science
Federal University of Paraná, Brazil
aurora@inf.ufpr.br

Abstract

Diversity preservation has already been established as an important concern for evolutionary computation. Clustering techniques were, among others, successfully applied to this purpose. Another important aspect of the research on evolutionary computation is related to linkage learning – the detection of the problem structure avoiding disruption of building blocks when new individuals are generated. This paper presents a novel approach which is a new estimation of distribution algorithm (EDA) where clustering plays two roles: diversity preservation and linkage learning. Initial empirical investigations illustrate the behavior of the algorithm when solving two benchmark optimization problems.

1. Introduction

Two aspects are recognized as determinant for the success of the Genetic Algorithm (and evolutionary computation in general): (i) the maintenance of a critical level of diversity in the population and (ii) the ability of detecting the structure of the problem. Both concerns were, up to this time, investigated through parallel threads without much contact among them.

The first theoretical model about substructure learning was the building-block (BB) hypothesis, proposed by [11], which discusses the preservation and recombination of building blocks – highly fit substructures of the problem being solved or optimized. After much work with the Genetic Algorithm (GA) and other similar approaches, a design decomposition theory was presented [5], motivating the interest on properly learning the structure of the problem. While some approaches rely on the underspecification of individuals like the messy GA [6] and the Linkage Learning GA [8], others attempt to learn the dependencies among the genes by constructing probabilistic models from the best solutions. This last class of GA comprises the EDAs - estimation of distribution algorithms [13]. Some of the most representative EDAs are the Extended Compact Genetic Algorithm (eCGA) [9] and the Bayesian optimization algorithm (BOA) [16]. Each of them is characterized by the kind of model learned at each generation; eCGA searches for marginal product models (MPMs) while BOA adopts a Bayesian network representation. This task of successively learning, at each generation, a statistical model can, potentially, be very computationally expensive; searching for the structure of a Bayesian network, as in [16], is an example. If independence among genes is assumed, then the model search is avoided, and very simple statistical models are learned at each step [2]. Unfortunately, simpler EDAs which rely on simpler statistical models are known to be unable to solve hard problems, since they cannot detect the problem structure.

Another important research thread has been also carried out in the GA literature: niching and diversity preservation techniques [12][7]. The maintenance of a critical degree of diversity of the population empirically avoids premature convergence and allows for the detection of multiple local and global optima [12]. Therefore, niching methods have been successfully applied to solve multimodal problems [20][18]. Additionally, it has been shown that the diversity preservation can also aid linkage learning even for unimodal problems [8] since diversity is a necessary condition for the success of selectorcombinative algorithms [4] (which are based on selection and combination). Clustering algorithms are natural candidates for the niching mechanism of a GA [15][20][10].

This paper applies clustering techniques in the GA context; however, the scope of this application is expanded beyond its conventional purpose of diversity preservation. The aim of this paper is to present (and empirically verify) the hypothesis that both niching and linkage learning could be effectively solved by the same mechanism, based on clustering. The underlying hypothesis is based on applying a clustering algorithm to the population of individuals of the evolutionary algorithm and obtaining clustering concepts,

0-7695-2976-3/07 $25.00 © 2007 IEEE
DOI 10.1109/ISDA.2007.84

which describe the diversity of the population. High-level concepts would reveal some important information about the hidden structure of the problem; additionally, each concept could be interpreted as a building block, therefore a recombination process could avoid building block disruption. Empirical results suggest that this hypothesis is acceptable and the discussion attempts to point out some insights on possible applications and limitations of this approach.

The next section describes an algorithm based on the ideas mentioned above. Sections 3 and 4 conduct a more detailed description of the method. The empirical evaluations are presented in section 5. Section 6 discusses the results and implications of the proposal.

2. Solving linkage learning via clustering techniques

Diversity preservation is crucial for linkage learning even for unimodal problems [12] since the detection of a given BB is only attainable if a critical number of individuals in the population possess that BB. Clustering techniques were already applied for diversity preservation in evolutionary computation [15][20][10]. However, diversity preservation alone has not been suitable for building block detection or problem structure learning – there remains the need for some kind of linkage learning.

Nevertheless, clustering algorithms could obtain some explanation for the diversity in the population. This approach follows a hypothesis that high-level concepts, which describe each cluster, could also be interpreted as substructure descriptors, since these concepts describe rules for the similarities observed among individuals of the population. This leads to a simple evolutionary algorithm, which relies on clustering the population and then learning clustering concepts, which would describe the most important aspects of the problem structure.

This new evolutionary algorithm, which allows an empirical validation of the hypothesis described above, is proposed in this section. It can be considered as an EDA (Estimation of Distribution Algorithm) with additional niching and linkage learning features. The algorithm is based on three core procedures (i) learning clustering concepts (ii) statistical modeling of the subpopulation of each cluster (iii) a new recombination operator, which combines the concepts found in (i) with statistical models found in (ii). Some other mechanisms, like selection, will be described later. It is important to note that this algorithm is restricted to allow binary attributes only.

A single individual is generated at each time without the adoption of a succession of "generations" as other EDAs often do. Therefore, the clustering hypothesis and the statistical models for each subpopulation will be just updated and not fully relearned whenever a new individual is generated and selected.

Before a more detailed description of the algorithm these three core procedures should be discussed.

Clustering. As discussed previously, clustering and concept learning would, hypothetically, reveal some information about the problem structure. In this first implementation, a two-stage process was adopted: initially, clusters are obtained by k-means, which is a simple and fast partitioning algorithm, and then clustering concepts are detected by an additional procedure. It is also important to note that k-means will just update the current centroids, except for the first run. Some restarts may occur since some clusters can become empty during the incremental update of the algorithm. There are some other choices for approaching this problem, as reducing the number of centroids k or inferring the best value of k at each iteration; these alternatives could be tested in future implementations.

Statistical modeling. Some evolutionary algorithms generate new individuals from statistical models, which were learned from previously selected individuals. The most simple statistical models employed for this purpose assume independence among the genes [2]. For binary variables, therefore, simple binomial proportions are learned hence the probabilistic model for all genes is a simple vector, called probabilistic vector (PV). Here, a similar mechanism is adopted: every new individual is generated by sampling from one of the existing PV except when recombination is applied. If subpopulations change (e.g. selection of the new individual) then PVs are updated. Since only binary variables are considered then each centroid can be directly interpreted as a probability vector (PV) of binomial proportions, where each position j of a centroid i corresponds to the proportion $\hat{\pi}_{i,j}$ of 1s in the corresponding position j for the subpopulation of that cluster i.

A new recombination operator. Since each concept contains a piece of information related to the structure of the problem then recombination should not disrupt these concepts. The key idea is that two clusters should be selected as parents and proportions from both PVs would be used when generating a new individual. However, the concept descriptions could guide the choice of which parent will give the value of each position of the new PV. A measure $\hat{w}_{i,j}$, of how informative is each parent cluster i for each variable j, is computed for both parents, and then the parent with the greatest $\hat{w}_{i,j}$ is selected (actually, all $\hat{w}_{i,j}$s are calculated, and stored in a matrix $\hat{W}$). Each $\hat{w}_{i,j}$ is the variation of the entropy of the distribution of a variable j, before and after observing a given cluster i.

Algorithm 1 shows the method proposed here. Bino-

Algorithm 1 φ-PBIL

(1) *Initialization:* Generate an initial random population of size N_0, compute the fitness of the individuals and select only $N_w < N_0$ of the best.

(2) *Learning:* Learn clusters from the population. For each cluster, a probability vector (PV) of binomial proportions is obtained. This leads to two matrixes: one for binomial proportions $\hat{\Pi} = (\hat{\pi}_{i,j})$ and another for information measures $\hat{W} = (\hat{w}_{i,j})$.

while convergence criteria were not met **do**

(3) *Breeding:* Generate a new individual H, by randomly choosing from one of the two following possible procedures: (i) generate an individual by sampling from one of the PVs, or (ii) apply cg-recombination (better described on section 3), performing a combination of the PVs of two parents, guided by their respective information measures stored in $\hat{W}$, and then sample from the resulting temporary PV in order to obtain a new individual.

(4) *Selection:* Compute F_H, the fitness of the new individual H. If H is not worse than the worst individual currently stored, then delete this worst individual and insert H in the population.

(5) *Learning:* Update clusters, binomial proportions in $\hat{\Pi}$ and information measures in $\hat{W}$.

end while

mial proportions are learned from the population for each gene, as in the PBIL algorithm [2]. Additionally, since the greatest extension here is to learn and use high-level concepts, therefore this new EDA will be called Concept-guided Population-Based Incremental Learning (φ-PBIL).

In the *Initialization* step a number N_0, $N_0 > N_w$ of individuals is generated and only the N_w best are selected. After that, during the evolutionary process, the population is kept at constant size N_w.

Next, in the *Learning* step, a clustering algorithm is applied to the population, which results subpopulation labels for all individuals. Probability vectors are also obtained, and the $\hat{w}_{i,j}$s are calculated for all variables and clusters.

In *Breeding*, the cg-recombination operator (see section 3) may or not be applied. The decision of applying cg-recombination or not at each moment is randomly chosen, with a given probability p_c, which is a parameter of the algorithm. The selection of parents is also random, but with probabilities proportional to the mean fitness of the individuals belonging to each corresponding cluster.

Finally, *Selection* calculates f_H - the fitness of the new individual H - and replaces the worst individual of the population for H (if H is not even worse). A *learning* step updates cluster hypothesis and information measures $\hat{W}$.

It should be noted that the new individual is inserted into the population, not into a subpopulation directly; actually, this new individual will only have a subpopulation label after the next clustering update.

Cluster A (receptor):	$\hat{\pi}_{A.} = [0.5, 0.7, 0.7, 0.3]$
	$\hat{w}_{A.} = [0.01, 0.1, 0.6, 0.4]$
Cluster B (donator):	$\hat{\pi}_{B.} = [0.1, 0.8, 0.1, 0.2]$
	$\hat{w}_{B.} = [\mathbf{0.6}, \mathbf{0.4}, 0.1, -0.01]$
Resulting PV:	$v = [\mathbf{0.1}, \mathbf{0.8}, 0.7, 0.3]$

Figure 1. The cg-recombination.

3. A concept-guided recombination operator

The previous section presented a high-level description of a new EDA, which involves the adoption of a clustering algorithm as the core mechanism for learning the problem structure. A new recombination operator, called concept-guided recombination, is proposed and described in this section.

The operator is based on blending two PVs into a new PV by choosing, for each variable j, the most informative parent. The measure of how informative each parent cluster i for each variable j is denoted by $\hat{w}_{i,j}$. It is the difference between the entropy of the distribution of each variable j before and after observing a given cluster i.

After computations of $\hat{w}_{i,j}$ are done this procedure is able to decide from which parent a proportion should be taken in a cg-recombination. The key idea is to pick only those positions of a parent's PV which are more informative than the respective positions of the other parent. Here, PV (probability vector) denotes both a k-means centroid and its respective binomial model since both represent the same information (binomial proportions).

The following procedure is adopted: in a cg-recombination, after the receptor (A) and the donator (B) clusters have been set, build the new probability vector (PV) by taking some $\hat{\pi}_{i,j}$ from B and some $\hat{\pi}_{i,j}$ from A, respecting the following criterion: if $\hat{w}_{B,j} > \hat{w}_{A,j}$, and $\hat{w}_{B,j} > 0$ then choose $\hat{\pi}_{B,j}$, else choose $\hat{\pi}_{A,j}$. Figure 1 shows an example of cg-recombination, using the concept detecting heuristic herein proposed.

Computation of $\hat{w}_{i,j}$ must be described now. Let $\hat{\Pi} = (\hat{\pi}_{i,j})$ be a matrix representing all the PVs; one PV on each row. The columns are associated to the variables, therefore, $\hat{\pi}_{i,j}$ represents the j-th variable of the i-th PV. Since a PV is a model for the individuals of a cluster, so each $\hat{\pi}_{i,j}$ can be interpreted as the proportion of 1s in gene j, considering all the individuals belonging to cluster i.

The entropy-reduction measure $\hat{w}_{i,j}$ allows a comparison among its corresponding $\hat{\pi}_{i,j}$ and the other $\hat{\pi}_{l,j}$s, $l \in [1, 2, ..., k], l \neq i$. It is the difference between $h_{i,j}$, which is

the entropy of the distribution of the binary variable j when cluster i is considered minus the entropy of the same variable when cluster i is excluded from the calculation, $h'_{i,j}$. All clusters are assumed to have equal numbers of individuals as if the population were equally distributed.

4. Some improvements to the core algorithm

After some initial tests with φ-PBIL (not shown here), it was noticed that some fashion of local search should be included. Also, a better allowance for overlapping and hierarchical building blocks would be beneficial. Two additional features were then empirically tested and incorporated to this first version of φ-PBIL:

Perturbation via the Wilson point estimator. The maximum likelihood estimator (MLE) of a binomial proportion is the simple mean of successes. In our case, it is interpreted as the proportion of ones at each locus, or

$$\frac{number\_of\_ones}{number\_of\_individuals\_in\_cluster}$$

and calculated for all clusters independently. Unfortunately, when all individuals in a cluster have the same value (all 0s, or all 1s) in a certain locus, this estimator saturates at one of the extremes (0% or 100%). It wipes out the chance of the alternative value, 1 or 0, to be generated; it is noticeable that this behavior would be harmful to local search. A desirable degree of uncertainty is allowed by simply adopting the Wilson point estimator [1][21], which is defined, in our case, as

$$\frac{number\_of\_ones + 2}{number\_of\_individuals\_in\_cluster + 4}$$

For example, even when all 100 individuals of a given cluster posses a 1 at given locus, there would still remain a probability of approximately 2% of a 0 to be generated at that same locus for a new individual.

A mutation operator could also be applied, probably achieving similar effects on local search.

Memory of old BBs. When smaller BBs start blending, new BBs emerge. This causes the loss of older clustering hypotheses consequently inhibiting those initial BBs in new individuals. This is a consequence of the well known race between selection and innovation [4]. For overlapping and/or hierarchical building blocks this constitutes a great problem. A potential solution is the maintenance of some old BBs (therefore, old clustering hypothesis) which can be used to generate new individuals. Both the current and an old clustering hypotheses compete when breeding. This mechanism works as follows: both $\hat{W} = (\hat{w}_{i,j})$ and $\hat{\Pi} = (\hat{\pi}_{i,j})$ matrixes are stored for some old well-performing clustering hypotheses. The cg-recombination selects randomly from one of two clustering hypotheses: the old set $\{\hat{\Pi}_{old}, \hat{W}_{old}\}$ and the new, recently updated set $\{\hat{\Pi}, \hat{W}\}$. It also computes performance records about the both sets: if the new individual, generated by one of the sets, is selected, then the performance variable of the corresponding set is updated (increased by one). If the new set overperforms the old one, then the old set and its performance variable are updated, and the performance variable of the new set is set to zero.

5. Empirical results

The main goal of the experiments is to evaluate the efficacy of the method when solving multimodal, deceptive and/or hierarchical benchmark problems. Some parameters were fixed for all experiments: the probability p_c, of applying cg-recombination during a breeding step, was set to 50%. The size of initial population was set to be $15N_w$, where N_w is the population size of the remaining process. Finally, the Wilson point estimator and the MLE were set to be applied with equal chances on every breeding.

The first experiment intends to show that φ-PBIL can solve a massively multimodal and deceptive optimization problem. For that purpose, the M7 function [12] was chosen; it has 32 global maxima of value equal to 5 and several million local maxima. The φ-PBIL is compared to a diversity preservation approach, called Clearing [18], which performs very well on this class of problem. The Clearing procedure is proposed as a niching technique for GA, where all the resources are supplied only to the best individuals of each subpopulation, instead of being shared among all individuals, as in other traditional niching techniques. Since Clearing was incorporated to a GA, then the resulting method is called here GA+clearing.

In figure 2, the GA+clearing, as reported in [18], is compared to φ-PBIL, both optimizing M7 with 30 binary variables. In both cases, results are averaged for 100 runs and population sizes are set to 800. The number of clusters k, specific for φ-PBIL, was set to 40. The algorithms were able to detect and stably maintain all 32 global maxima, but the GA+clearing was much faster than φ-PBIL.

In another experiment the scalability of φ-PBIL is verified, using the concatenated trap-5 [16] problem, which is also deceptive. It posses possesses a single global optimum (the string fulfilled with 1s), with a fitness equal to the size of the problem. For each contiguous nonoverlapping block of five variables, two building blocks can be identified: 00000 and 11111, which contribute with 4 and 5 to the overall fitness, respectively. Single-variable statistics may lead apart from the value 1, what makes this problem deceptive. Several combinations of the two building blocks

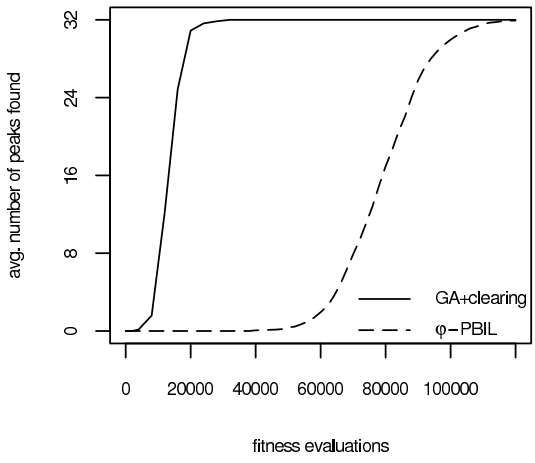

Figure 2. Average number of peaks found for M7, by φ-PBIL and a GA with clearing (GA+clearing).

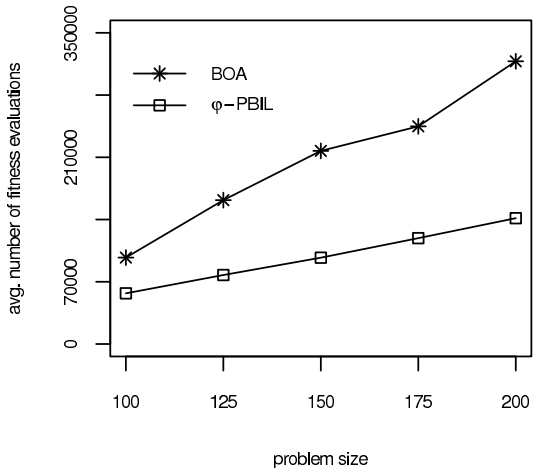

Figure 3. Scalability of φ-PBIL and BOA for some sizes of the trap-5 problem

lead to multiple suboptimal solutions.

A number of 30 successive runs were performed for some problem sizes λ: 100, 125, 150, 175 and 200. The population size N_w was set to be proportional to the problem size λ, as $N = 5\lambda$. The number of clusters k was also proportional to λ; $k = \lambda/5$. Termination criterion was set to be the loss of diversity inside PVs: the algorithm finishes when all $\hat{p}_{i,j}$s are > 0.95 or < 0.05.

Figure 3 shows the result of this experiment. In the same graph, a similar verification for BOA is also shown (from [14]), with the same intention of checking scalability . Results show that both algorithms scale up very well; actually, the hypothesis of (at most) linear scalability is supportable, since linear models for both curves resulted coefficient of determination $R^2 = 0.9992$ for φ-PBIL and 0.9846 for BOA.

6. Discussion and conclusion

Linkage learning has revealed a challenging goal for evolutionary computation, and several approaches have been proposed. Some of them are highly explicit on this intention: messy GA [6], for example, performs the identification and recombination of BBs in two separate phases. An EDA follows a diverse tactic, attempting to a learn a (potentially) complex statistical model in each generation. In that particular aspect, φ-PBIL is similar to the simple GA, since both rely on simple recombination operators - crossover in GA and cg-recombination in φ-PBIL.

However, an important difference should be pointed out, between φ-PBIL and GA (and also other evolutionary algorithms). φ-PBIL is insensitive to the ordering of the genes in the string. GA with one-point crossover, on the other hand, relies on a tight linkage of genes [8], therefore the ordering of genes is not immaterial. A genetic linkage is said to be tight when there is a correlation between gene proximity and gene dependence. This may be the reason for the great difference between the performance of GA+clearing and φ-PBIL, since the genes in the problem being addressed are tight linked, and GA takes advantage from that.

The application of unsupervised learning in the GA and EDA context is mostly related to the adoption of clustering algorithms as a niching mechanism [15][20][10]. Alternatively, in [17] a new EDA is proposed, which is based on unsupervised learning of Bayesian networks. That work extends the applicability of unsupervised learning since it shows, for the first time, that those techniques could be employed in the model learning stage of an EDA. In the work herein presented, the application of unsupervised learning in an EDA context follows a similar approach. However, here a much simpler statistical model is incrementally learned, guided by the clustering algorithm, without any explicit model for representing dependencies among genes.

The choice for k-means with an additional concept-detecting procedure as a first attempt on implementing φ-PBIL was successful. However, some other architectures should also be tested on future implementations. Conceptual clustering algorithms [3][19][22], besides just searching for good clustering hypotheses, directly produce high-level descriptions, or concepts, for each cluster found.

The method proposed here is shown to be able to solve

some relevant benchmark problems. Some questions arise from the experiments, as why φ-PBIL is much slower than GA+Clearing, or how does φ-PBIL extends to more difficult classes of problems. The choice of parameter values is done after empirical investigation, and will be better explained in future work. Some topics of future research include also a more complete investigation on the effects of adopting different levels of memory, and also a research on how φ-PBIL scales on hierarchical problems.

References

[1] A. Agresti and B. Coull. Approximate is better than exact for interval estimation of binomial proportions. *The American Statistician*, 58:119–126, 1998.

[2] S. Baluja and R. Caruana. Removing the genetics from the standard genetic algorithm. In *Int. Conf. on Machine Learning*, pages 38–46, 1995.

[3] D. Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 2:139–172, 1987.

[4] D. E. Goldberg. The race, the hurdle, and the sweet spot: Lessons from genetic algorithms for the automation of design innovation and creativity. Technical Report ILLIGAL Report No. 98007, University of Illinois at Urbana-Champaign, 1998.

[5] D. E. Goldberg, K. Deb, and J. H. Clark. Genetic algorithms, noise and the sizing of the populations. *Complex Systems*, 6:333–362, 1992.

[6] D. E. Goldberg, G. Korb, and G. Deb. Messy genetic algorithms: Motivation, analysis , and first results. *Complex Systems*, 3:493–530, 1989.

[7] D. E. Goldberg and J. J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the Second International Conference on Genetic Algorithms*, pages 41–49, 1987.

[8] G. R. Harik. *Learning gene linkage to efficiently solve problems of bounded difficulty using genetic algorithms*. University of michigan, 1997.

[9] G. R. Harik. Linkage learninig via probabilistic modeling in the ecga. Technical Report ILLIGAL Report No. 99010, University of Illinois at Urbana-Champaign, Urbana IL, 1999.

[10] C. Hocaoglu and A. C. Sanderson. Evolutionary speciation using minimal representation size and clustering. *Evolutionary Programming IV*, pages 187–203, 1995.

[11] J. H. Holland. *Adaptation in natural and artificial systems*. University of Michigan Press, Ann Arbor, MI, 1975.

[12] S. W. Mahfoud. *Niching methods for genetic algorithms*. Doctoral dissertation, University of Illinois at Urbana-Champaign,, Urbana, USA, 1995.

[13] H. Mühlenbein and G. Paass. From recombination of genes to the estimation of distributions: I binary parameters. In *Parallel Problem Solving from Nature III*, pages 178–187, 1996.

[14] M. Pelikan. *Hierarchical Bayesian Optimization Algorithm: Toward a New Generation of Evolutionary Algorithms*. Springer-Verlag, 2005.

[15] M. Pelikan and D. E. Goldberg. Genetic algorithms, clustering, and the breaking of symmetry. In *Parallel Problem Solving from Nature VI*, pages 385–394, 2000.

[16] M. Pelikan, D. E. Goldberg, and E.Cantu-Paz. BOA: The bayesian optimization algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99)*, pages 525–532, 1999.

[17] J. Peña, J. Lozano, and P. Larrañaga. Globally multimodal problem optimization via an estimation of distribution algorithm based on unsupervised learning of bayesian networks. *Evolutionary Computation*, 13(1):43–66, 2005.

[18] A. Petrowski. A new selection operator dedicated to speciation. In *Proceedings of the 7th International Converence on Genetic Algorithms*, pages 144–451, 1997.

[19] D. A. Simovici, N. Singla, and M. Kuperberg. Metric incremental clustering of nominal data. In *Proceedings of ICDM 2004*, pages 523–527, 2004.

[20] F. Streichert, H. Ulmer, and A. Zell. A clustering based niching EA for multimodal search spaces. In *Proceedings of the 6th International Conference on Artificial Evolution*, 2003.

[21] E. B. Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22:209–212, 1927.

[22] B. Zenko, S. Dzeroski, and J. Struyf. Learning predictive clustering rules. In *Proceedings of the 4th International Workshop on Knowledge Discovery in Inductive Databases*, pages 122–133, 2005.