Modelling the Initialisation Stage of the ALKR Representation for Discrete Domains and GABIL Encoding

María A. Franco
ASAP Research Group
School of Computer Science
University of Nottingham,
Jubilee Campus
Nottingham NG8 1BB
mxf@cs.nott.ac.uk

Natalio Krasnogor
ASAP Research Group
School of Computer Science
University of Nottingham,
Jubilee Campus
Nottingham NG8 1BB
nxk@cs.nott.ac.uk

Jaume Bacardit
ASAP Research Group
School of Computer Science
University of Nottingham,
Jubilee Campus
Nottingham NG8 1BB
jqb@cs.nott.ac.uk

ABSTRACT

Models in Genetic Based Machine Learning (GBML) systems are commonly used to gain understanding of how the system works and, as a consequence, adjust it better. In this paper we propose models for the probability of having a good initial population using the Attribute List Knowledge Representation (ALKR) for discrete inputs using the GABIL encoding. We base our work in the schema and covering bound models previously proposed for XCS. The models are extended to (a) deal with the combination of ALKR+GABIL representation, (b) explicitly handle datasets with niche overlap and (c) model the impact of using covering and a default rule in the representation. The models are designed and evaluated within the framework of the BioHEL GBML system and are empirically evaluated using first boolean datasets and later also nominal datasets of higher cardinality. The models in this paper allow us to evaluate the challenges presented by problems with high cardinality (in terms of number of attributes and values of the attributes) as well as the benefits contributed by each of the components of BioHEL's representation and initialisation operators.

Categories and Subject Descriptors

F.2.0 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity—*General*; I.2.6 [**Artificial Intelligence**]: Learning—*Concept Learning, Induction*

General Terms

Algorithms, Theory, Experimentation

Keywords

Evolutionary Algorithms, Learning Classifier Systems, Rule Induction, ALKR, GABIL

1. INTRODUCTION

Facetwise analyses [9] have been performed in the past over GBML systems[5, 13, 14, 16] to understand their domains of competence and the requirements that should be fulfilled for their correct functioning. In most of these analyses the authors derived models suitable for the ternary representation {0,1,#}. Thus, there is a need to extend these models to other representations so other systems take further advantage of this knowledge. In this work we develop models which correspond to the covering and schema bounds proposed for the XCS system[5], that is, the probabilities that an initial population covers the whole search space and that it contains representatives from all niches in a problem, respectively. These models are developed for a specific framework: the BioHEL GBML system[3, 17], which employs an Iterative Rule Learning paradigm[18], and its Attribute List Knowledge Representation (ALKR)[3] which, for discrete variables, uses the GABIL encoding[11]. Moreover, the models explicitly take into account other characteristics from BioHEL such as its covering operator or the use of an explicit default rule. Finally, specific models are developed for problems where there is overlap between some of its niches (rules).

These models are empirically evaluated, in a first stage, using binary problems with and without niche overlap: the multiplexer problem and randomly generated kDNF problems[6], respectively. The experiments show the models are accurate in these two types of scenarios. Moreover, the models also show how the covering and default rule mechanisms of BioHEL increase the probability of having a good initial population. In a second stage we present a generalisation of those models for x-ary attributes. The models show how the problem gets harder with the increase of the cardinality of the problem attributes and the number of classes. Nevertheless, they also show how the GABIL encoding can be made more robust by decreasing the probability of generating unmatchable rules, a known weakness of this representation[12].

These models are useful, not only because they show insights about the strengths and weaknesses of BioHEL, but also because they are the start point to design principled methodologies to automatically adjust some of the parameters of the system. This adjustment is particularly desirable when handling large scale datasets, since it helps avoiding the high computational costs involved in preliminary experimentation.

2. BACKGROUND WORK

Since Genetic Algorithms (GA)[10] were presented by Holland in the 60's there was a need to develop a theory that

GECCO'11, July 12–16, 2011, Dublin, Ireland.

formalises this technique. Goldberg[9] proposed an approach to formalise the design and application of GAs by employing a facetwise methodology in which the different aspects of the system are analysed individually, assuming that the rest of the components are working correctly. This methodology was adapted by Butz et al.[5] to the specific context of Michigan Learning Classifier Systems (LCS)[19]. In recent years, this methodology has been applied also to specific, challenging, learning scenarios such as problems with class imbalance[13]. Moreover, this analysis has also been extended to continuous domains in [14, 16].

Furthermore, other formal analyses of LCS have been proposed from a more probabilistically model-based perspective[7, 8]. In this work the authors model LCS as a Mixture of Experts (MoE), showing how this metaphor can generate a very similar prediction models and explain how LCS works from a machine learning point of view.

Also, other analysis over the initialisation stage of the GABIL representation have been performed in the past[2] in the context of the GAssist[1] Pittsburgh LCS. In this work the probability of covering the search space with the GABIL representation was modelled in order to propose smart initialisation strategies for GAssist. Moreover, the GABIL representation has also been analysed in terms of scalability in [12]. In this work the authors point out weaknesses of the GABIL representation, such as generating rules that are incapable of matching any example. Here the authors show how the number of unmatchable rules increase exponentially with the problem size, presenting scalability problems for the systems that use this representation.

3. BIOHEL, ALKR AND GABIL

BioHEL[3] is a GBML system specially designed to cope with large scale datasets[17, 4]. This system is inspired by a previous Pittsburgh LCS called GAssist[1], but instead of following the same paradigm it follows an Iterative Rule Learning Approach[18]. In this paradigm the rules are learnt, one at a time, in sequence until all the training set has been covered. As a consequence of this learning paradigm, the rule sets generated by BioHEL follow a decision list structure[15], the same rule set structure that GAssist employs and thus its explicit default rule mechanism[1] is inherited directly. For a complete description of BioHEL, please see [3].

BioHEL employs a rule representation called Attribute List Knowledge Representation (ALKR) [3], designed to cope with high dimensionality problems. This representation automatically identifies which are the relevant attributes of a rule and discards all others. Hence, its match process is more efficient, as irrelevant attributes are simply not present in the rule, but also, its exploration process is more focused, as the rule only contains data about attributes considered to be relevant. In order to explore the space of attributes to identify the relevant ones, this representation has two extra operators, called generalise and specialise. This operators drop or add attributes from/into the rule, respectively. For discrete attributes this representation employs the GABIL encoding[11], while intervals[20] are employed for continuous attributes.

Figure 1 shows an example of an individual using this representation. Each classifier condition is formed by five structures: a) an integer with the number of attributes represented b) a list of the identifiers of the represented attributes, c) a list of values for the represented attributes (nominal and continuous), d) a list with the positions where each attribute can be found in this classifier and e) the class of the classifier.

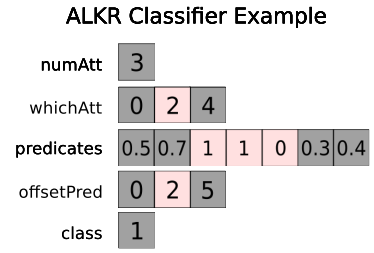


Figure 1: Representation of a classifier using ALKR

We will describe the GABIL encoding as it is employed in the models proposed in this paper. In GABIL the attributes are represented by binary strings of fixed length. The length correspond to the number of possible values the feature can have. For example if the attribute F1 may have the values (A,B,C), F2 the values (O,P), and F3 the values (W,Z,X,Y) a possible condition string for each one of the attributes would look like:

F1	F2	F3
100	01	1101

Each attribute is read as a disjunctive clause between all the values that have their bit on. For example, this condition can be interpreted as *if F1 is A and F2 is P and F3 is W or Z or Y*.

The initialisation of rules using the ALKR+GABIL representation works as follows. First, since the ALKR list usually do not represent all the attributes, the probability of an attribute to appear in a randomly generated classifier depends on $ExpAtts$. This is a user defined parameter that determines the expected value of the number of relevant attributes in a rule. Based on this value and the number of attributes d we can calculate the probability l_d of an attribute to appear in the attribute list as follows:

$$l_d = \begin{cases} 1 & d <= ExpAtts \\ \frac{ExpAtts}{d} & d > ExpAtts \end{cases} \quad (1)$$

After an attribute is selected to appear in the list, the conditional structure for this attribute is determined. The construction of it depends on whether the system is using covering, the default rule mechanism and the type of attribute.

If the attribute is nominal we will set each one of the positions of the GABIL representation in 1 with probability p. When covering is used, for each attribute the bits corresponding to the instance's values are set to one and the rest of the positions will be defined probabilistically depending on p. If a default class is employed, the rules in the population are not allowed to cover this class. This means that (a) the class attribute of the rule cannot take the value for the predefined default class and (b) the covering operator will not sample instances from that class.

4. PROBABILISTIC MODELS

There are two characteristics a good initial population needs to comply with, according to [5]. First, all the building blocks of the problem should be present in the initial

population, meaning that there should be representatives of each niche. That is rules that belong to a niche and do not mis-classify. Moreover, the whole search space should be covered. These two requirements are referred as the schema and covering bounds respectively.

In this paper we will focus on calculating the probabilities on which the bounds are based. Afterwards, this probabilities can be used as in [5] to specify restrictions or bounds over system parameters.

In the following sections we calculate the probabilities of fulfilling these two requirements independently for the binary case. In Section 5 we generalise the models for x-ary attributes. The derivation of problem parameters based on the bound formulas will be done as a further work.

4.1 Schema Bound

In a binary domain, a *schema* is a sequence of ternary symbols that represent some of the bits in the string (i.e 1*0*11) [10]*. These correspond to the inner structure or rules of the problem. A *representative* is a classifier that represents at least† all the bits in the schema (i.e 110#11).

The schema bound constraints the parameters of the system so the probability of having representatives of each schema (a rule that covers all/part of a schema and does not misclassify) in the initial population is high enough[5]. This is important since having good building blocks in the initial population of a GA is essential for the learning process[9]. It is also possible to generate building blocks along the learning process. However, this is modelled by the reproductive opportunity and sustenance bounds which we will adapt to BioHEL as a further work.

The schema bound is based on modelling the probability of generating a rule that is a representative of a schema. The probability of a representative was already calculated for the ternary alphabet in [5] based on the global specificity of the population and without considering overlapping. However, the probability for BioHEL is different since this system uses the ALKR+GABIL encoding. The probability in this case depends on the methodologies used to create new classifiers (covering and default class) and two user defined parameters p and $ExpAtts$.

Considering the covering and the default rule mechanisms in BioHEL, there are four ways in which a rule can be created: (a) not using covering or default rule (base case), (b) using default rule only, (c) using coverage only and (d) using coverage and default rule.

For each one of these cases we derived the probability of having a representative in problems that do not have overlapping. Afterwards, based on this probability we handle the niche overlapping case. The models generated in this stage are based on the multiplexer problem and its more general case, the kDNF family of problems[6]. In subsequent sections we show a generalisation of the models for problems with x-ary attributes.

To calculate the probability of the representative we need to consider two types of attributes:

Fully mapped attributes. Attributes for which all its possible values are specified within the schemata.

Partially mapped attributes. Attributes for which only one of its possible values is specified in the schemata, either 1 or 0.

We will refer to the number of fully mapped attributes as k_f and the number of partially mapped attributes as k_p. If the schemata have k attributes, then $k_f + k_p = k$.

Considering that BioHEL uses the GABIL representation immerse in a ALKR list of attributes, an attribute is relevant if it was selected to be in the attribute list. This occurs with probability l_d as shown in Equation 1. Moreover, an attribute can also be relevant if the attribute is fully mapped and the strings generated are 01 or 10, or if the attribute is partially mapped and the "right" string is generated.

In the following sections we will show the final probability formulas for each one the the cases mentioned before.

4.1.1 Base case (No covering and no default rule)

Considering that a representative should have at least the number of bits in the schema represented k, the size of the problem d, and the number of possible actions n, the probability of a rule becoming a representative is:

$$P(rep) = \frac{(r_d^p)^{k_p} \left(r_d^f\right)^{k_f} (1 - l_d P(00))^{d-k}}{n} \quad (2)$$

where r_d^p is the probability of relevance of a partially mapped attribute and r_d^f is the probability for a fully mapped attribute. Moreover, the last term avoids the string 00 in the $d-k$ attributes that are not relevant. If one of the attributes has this string the classifier would not match any instance, and consequently a representative would not be formed.

Considering that $P(00)$, r_d^p and r_d^f can be calculated as:

$$r_d^p = l_d P(01 \vee 10) = 2 l_d p(1-p) \quad (3)$$

$$r_d^f = l_d P(01) = l_d p(1-p) \quad (4)$$

$$P(00) = (1-p)^2 \quad (5)$$

we can construct the probability of having a representative without using covering or default rule mechanism as shown in Equation (6).

$$P(rep) = \frac{2^{k_f} \left(l_d p(1-p)\right)^k \left(1 - l_d (1-p)^2\right)^{d-k}}{n} \quad (6)$$

4.1.2 Default class case (no covering)

When there is a default class, there are only $n-1$ possible actions for a new classifier. Including this assumption, the probability of a representative in this case would be:

$$P(rep) = \frac{2^{k_f} \left(l_d p(1-p)\right)^k \left(1 - l_d (1-p)^2\right)^{d-k}}{n-1} \quad (7)$$

4.1.3 Covering case (no default rule)

To model the usage of covering we need to consider four new aspects over Equation (2):

1. The number of possible actions will be equal to 1. Since the action will be copied from the example there is no choice besides using that action.
2. The probability of having a string 01 or 10 will depend on the probability of the attribute in the instance being 0 or 1.

*We use the ternary representation to explain the concept of schema as it is more compact. However, the representation used in this paper is the GABIL representation.

†It might be a more specific than the schema.

3. Having an attribute 00 is not possible through covering anymore.
4. Despite the fact that the classes might be imbalanced the covering opportunities for all the classes are the same.

In this case the values for r_d^p and r_d^f will be the same as shown in (8).

$$r_d^p = r_d^f = ld(1-p) \tag{8}$$

The probability for fully mapped attributes being relevant in this case is reduced by half. This is because the string generated does not depend only on p but on the string that we are actually copying from the example.

However the whole probability $(r_d^p)^{k_p}$ of the partial attributes being relevant depends on the relation of classes mapped m over the total number of classes n. This is because all the classes have the same probability of being selected for covering, but only the classes represented in the schemata will produce representatives. Substituting this assumptions we obtain the probability of having a representative using covering as shown in Equation (9).

$$P(rep) \quad = \quad \frac{m}{n}\left(l_d\left(1-p\right)\right)^k \tag{9}$$

4.1.4 Covering and Default class case

The usage of a default rule mechanisms limits the number of classes that can be used for covering to $n-1$. Also the number of mapped classes m would not include the default class. Including this changes in Equation (9) we obtain the probability using covering and default class:

$$P(rep) \quad = \quad \frac{m}{n-1}\left(l_d\left(1-p\right)\right)^k \tag{10}$$

4.1.5 How does the overlapping affects?

A randomly generated kDNF problem has a very high probability that some of its rules overlap among themselves, specially if the number of rules is very high and the number of specified attributes is small. In this sense, the multiplexer problem is a extremely rare case of kDNF since none of the rules overlap with each other. We will extend our models now to consider the overlapping between rules.

To calculate this probability we first have to calculate the probability that a rule belongs to a specific niche $P(niche)$. For a problem with no overlapping this probability consists in dividing the probability of a representative by the number of niches or rules.

$$P(niche)_{no} = \frac{P(rep)}{r} \tag{11}$$

When there is overlapping each rule covers the same amount of examples as before. However, the covered space shrinks because there are instances covered by more than one rule. Then the percentage of space covered by one niche can be generalised by dividing the amount of examples covered by a niche EN among the total number of examples covered EC.

For a randomly generated kDNF problem we have a probabilistic estimation of the amount of positive examples (examples covered) with the following formula:

$$EC = 2^d\left(1-\left(1-2^{-k}\right)^r\right)$$

Moreover, we know the amount of examples covered by each rule is:

$$EN = {}^{2^d}\!/_{2^k}$$

Substituting the value ${}^1\!/_r$ by EN/EC we obtain that the probability of having a representative of certain niche is:

$$P(niche) = \frac{P(rep)}{2^k\left(1-\left(1-2^{-k}\right)^r\right)} \tag{12}$$

Using (12) we can calculate the probability of having a representative under overlapping conditions $P'(rep)$ by forcing that at least one of the niches has a representative.

$$P'(rep) = \left(1-\left(1-P(niche)\right)^r\right) \tag{13}$$

This model is based on the assumption that the rules of a problem cover random subsets of the problem attributes. If the distribution of niches is not uniform, the model does not fully hold.

4.2 Covering bound

The covering bound assures that at least each instance of the problem space is covered by one classifier in the population. In this way, we will map the whole problem domain into a initial population.

To calculate the covering bound we need to calculate the probability of matching an instance with a randomly generated classifier. Although, this was already calculated for the ternary alphabet in [5], in this work we adapted this to the ALKR+GABIL representation. In the following sections we will calculate this probability using and not using the covering mechanism. The default class mechanism in this case do not affect the results obtained in terms of matching. Therefore, no specific formulas are presented for these cases.

4.2.1 Base case (No covering)

There are three ways in which a randomly generated attribute can match an example: a) the attribute does not appear in the list with probability $1-l_d$, b) the attribute appears in the list and its value is 11 or c) the attribute appears in the list and has the correct value. Cases b and c occur with probability p of setting the right bit on. Therefore, the probability of matching is:

$$P(match) = (1-l_d+l_dp)^d \tag{14}$$

4.2.2 Covering Case

The usage of covering affects the probabilities of setting the right bit on. We need to consider two cases: a) the instance used for covering is similar to the one we want to match or b) the instance used for covering is different. If the instance is similar the probability is 1 and if the instance is different the probability is p. Then the probability of setting the right bit on considering these two cases is ${}^{(1\,+\,p)}\!/_2$.

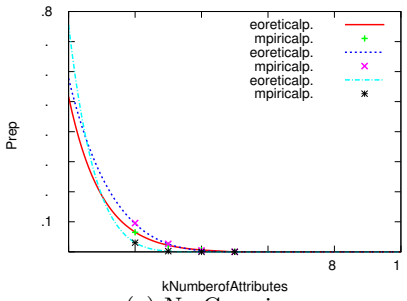
Applying this changes on equation (14) we obtain:

$$P(match) = \left(1-l_d+l_d\left(\frac{1+p}{2}\right)\right)^d \tag{15}$$

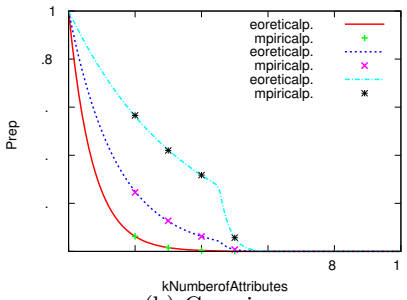
For the worst case, matching an instance when the example used for covering is different happens with probability ${}^1\!/_2$. If the number of instances is smaller than the number of classifiers this factor might grow up to 1.

4.3 Model validation

To validate the previous models we used kDNF and multiplexer problems. The kDNF problems used are of size

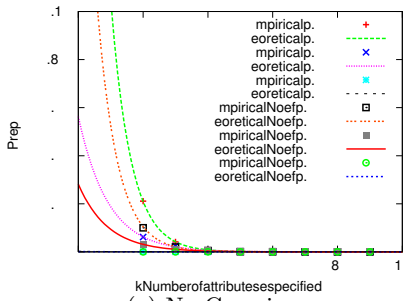


(a) No Covering

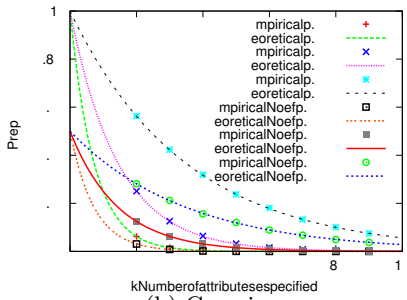


(b) Covering

Figure 2: Validation of the probability of a representative using the multiplexer problem.



(a) No Covering



(b) Covering

Figure 3: Validation of the probability of a representative using the kDNF problem with one rule

$d = 10$, with $k = \{2 - 10\}$ and $r = \{1, 5, 10, 20, 40\}$. On the other hand, the multiplexer problems used were of size 3, 6, 11 and 20. For the kDNF problems we generated 5 different instances of each configuration and run each one of them with 25 different seeds. In the case of the multiplexer problems we run each one of them with 125 seeds. For each run we generated a random population of 500 individuals and calculated the average number of classifiers that had the bits in the schema represented, and the average number of classifiers that match each one of the instances in the problem. For all experiments the value for the *ExpAtts* parameter was 15 (or the number of attributes in the problem, if less than 15).

Figures 2 and 3 show the validation of the schema bound for non-overlapping problems using the multiplexer and the kDNF problems respectively. In particular Figure 2 only shows the base case and the covering case, since the usage of the default rule does not produce changes in the probability for the multiplexer problem. For the kDNF models we only used problems with one rule to guarantee that there is no rule overlap. In these figures we can see that the models adjust to the empirical data. While the number of attributes increase the probability of having a representative decreases. It can be observed in the figures that the mechanisms of default rule and covering increase the chances of creating representatives, which demonstrate the benefits of these techniques. Moreover, as expected, using a p too large decreases the probability of having a representative because it raises the chances that a rule misclassifies. However, when not using covering a larger p is beneficial.

To validate the models for problems with niche overlap we used the kDNF problems with more than one rule. Figure 4 shows the validation of these models. In this figure we can see that the models fit the empirical data. Moreover, the probability of generating a representative increases with the number of rules. Once again we can observe the benefits of the usage of covering and default rule mechanisms.

Figure 5 shows the validation for the probability of match using the multiplexer problem[‡]. We can see that while the problem size increases the probability of match decreases. Moreover, we can see that when the problem is very small the empirical probability is higher, this is because the problem has less instances than classifiers in the population, so the probability of generating classifiers with the same instance increases. We can also observe a plateau in the models when the number of dimensions in the problem is larger than *ExpAtts*, as attributes that do not appear in ALKR's list are considered as irrelevant, hence matching any value.

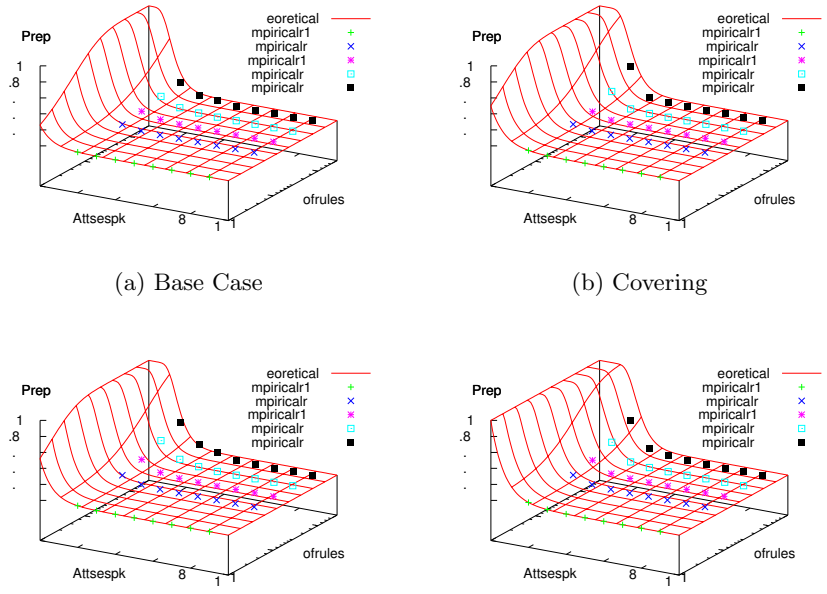
5. TOWARDS A GENERALISED MODEL FOR X-ARY ATTRIBUTES

As a small step towards general models we generalised the formulas presented previously to work with nominal attributes with more than 2 values. This generalisation will show how the problem becomes more difficult when we increase the domains of each attribute. Also it will show how the GABIL representation gets more robust by decreasing the probability of generating unmatchable rules (rules with all the bits set to 0) which is an issue that had been identified in the literature as a weakness of this representation[12]. In the following sections, we present the generalised formulas for the schema and covering bound for x-ary attributes.

5.1 Schema bound

Let's call the number of possible values of a attribute t and the number of specified values in a attribute e. A rep-

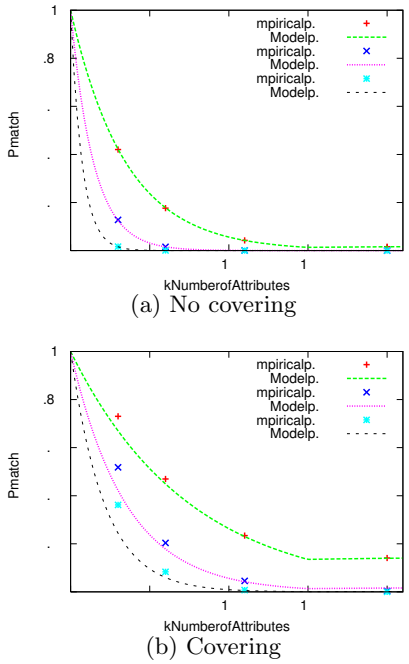
[‡]In this case the kDNF problems were not used because all of them had the same number of attributes



(a) Base Case (b) Covering

(c) Default Class (d) Covering and Default Class

Figure 4: Validation of the probability of a representative for kDNF problem with rule overlap and $p = 0.75$. The graphics show the probability of having a representative considering all the niches



(a) No covering

(b) Covering

Figure 5: Validation of the probability of match using the multiplexer problems.

resentative will be created if we set to 1 e bits and to 0 the rest of $t - e$ bits. Considering this, the formula for the base case can be generalised as follows:

$$P(rep) = \frac{t^{k_f}\left(l_d p^e (1-p)^{t-e}\right)^k \left(1 - l_d(1-p)^t\right)^{d-k}}{n} \quad (16)$$

In this formula, we can see that the probability of creating unmatchable rules decreases while t increases. Figure 6 shows how the probability of generating a matchable rule $\left(l_d(1-(1-p)^t)\right)^{d-k}$ compares to the probability of generating the right string $\left(l_d p^e (1-p)^{t-e}\right)^k$. We can see that while t and k increase it is less likely that we generate rules that do not match. However, the probability of generating the right string decreases when t and k increase.

Furthermore, the probability for the default class case is similar to (16) but subtracting 1 to the number of classes.

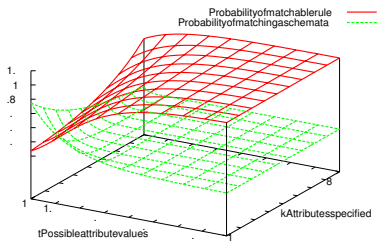
On the other hand, the usage of covering can be generalised by assuming that the right string will be created if we put the rest of the $t - e$ bits in 0.

$$P(rep) = \frac{m}{n}\left(l_d\,(1-p)^{t-e}\right)^k \quad (17)$$

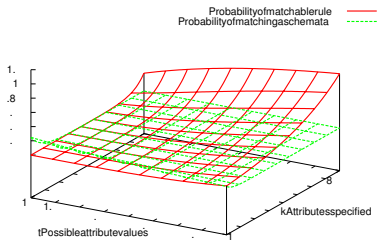
Similar as the case with no covering, the introduction of the default class only restricts the classifiers considered for covering. Therefore, to calculate the probability for this case we only need to substitute $^{m}/_{n}$ for $^{m}/_{n-1}$ in (17).

5.2 Covering bound

When no covering is used the probability of matching an instance is the same, no matter the number of values an attribute can accept. However, when the covering mechanism is activated the probability of generating a matching individual varies. Therefore, this probability will be:



(a) p=0.75



(b) p=0.25

Figure 6: Probability of generating the schema attributes against the probability of generating a matchable rule

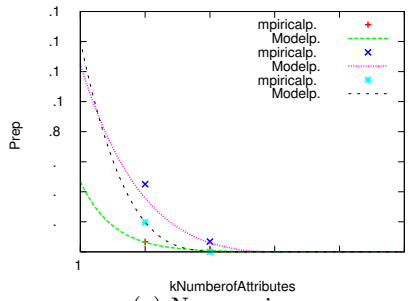
$$P(match) = \left(1 - l_d + l_d \left(\frac{e + (t-e)p}{t}\right)\right)^d \qquad (18)$$

This means that if the attribute is selected to be in the list we face t possible cases. The first e cases is that the instance used for covering is similar to the one that we want to match. Then the probability of matching is 1. The following $t - e$ cases refers to the cases where both instances are different and we will match with probability p.

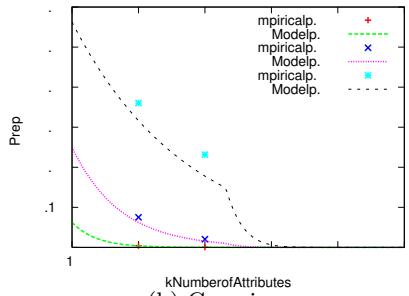
5.3 Model validation

To validate these models we generated ternary multiplexer problems, which instead of only accepting values 0,1, they accept an extra value. In this case the size of the problem string would be $k-1+3^{k-1}$, where k is the number of bits in the schemata. We generated ternary multiplexer problems of size 4 and 11. We generated initial populations with 25 different seeds for each one of these problems and calculated the empirical probabilities for a representative and match. Again, the population size was 500 and $ExpAtts$ was 15.

Figure 7 shows the validation for the generalised models of the schema bound. Since the multiplexer does not show any differences on the probabilities using the default rule, this cases are not shown. In this figures we can see that the models fit the empirical data. However, there is a subestimation of the models because the empirical data considers the cases where the address bits might map two values and the model does not. We can notice also that while the domains of the attributes expand the probabilities of generating a representative are lower. Moreover, while a smaller p is better when using covering, $p = 0.50$ gives the best results in the non-covering case, as expected.



(a) No covering



(b) Covering

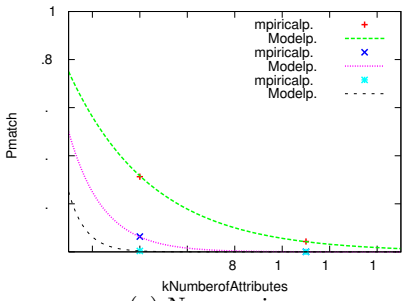
Figure 7: Validation of the probability of a representative using the ternary multiplexer problem.

Figure 8 shows the validation of the generalised models for the covering bound. We can see that in this case the models also fit the empirical data. Moreover, we can notice that the increase of t, the attribute domain, reduces the probabilities of matching. However, the covering mechanism, as it was shown before, slightly increases the chances of covering the whole search space.

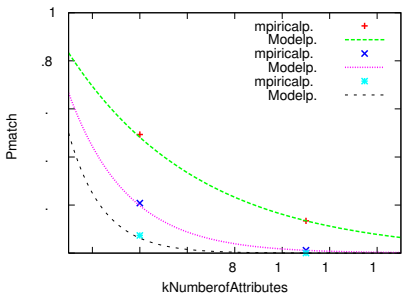
6. CONCLUSIONS AND FURTHER WORK

The models presented in this paper predict satisfactorily the probabilities of generating a good initial population in terms of: (a) covering the search space and (b) generating accurate representatives for each niche in a problem. The models were generated considering the ALKR representation with GABIL encoding for binary attributes. Moreover, we presented a generalisation of the models for x-ary attributes and validated it with ternary multiplexer problems. This generalisation showed how the GABIL representation becomes more robust with the increase of the cardinality of the attributes, in terms of generating unmatchable rules. However, the overall probability of having a good initial population decreases when the number of values per attribute increases, since it is more difficult to generate the right string. The models also show how the covering and default mechanisms introduced in BioHEL improve the chances of generating a better and more useful initial population for the system.

We have modelled with high accuracy several scenarios for the ALKR representation. However, these models were based on the assumption that we could estimate several parameters of the problems. In a further work, we would like to study how can we simplify these models so some of these assumptions can be removed without producing a large impact in the accuracy. Moreover, models for the reproductive opportunity, sustenance and learning time should be developed



(a) No covering



(b) Covering

Figure 8: Validation of the probability of match using the ternary multiplexer problems

in order to have an unified theory for the correct functioning of BioHEL with ALKR representation and, for instance, derive boundaries for the population size and other user-defined parameters. Furthermore, additional challenges or differences between Iterative Rule Learning and the Michigan approach should be identified in order to model specific limitations of this type of systems. Finally, based on this models, we are interested in proposing methods for the automatic adaptation of some of BioHEL's parameters.

7. ACKNOWLEDGEMENTS

We acknowledge the support of the UK Engineering and Physical Sciences Research Council (EPSRC) under grant EP/H016597/1. We are also grateful for the use of the HPC facility at the University of Nottingham.

8. REFERENCES

[1] J. Bacardit. *Pittsburgh Genetics-Based Machine Learning in the Data Mining era: Representations, generalization, and run-time*. PhD thesis, Ramon Llull University, Barcelona, Spain, 2004.

[2] J. Bacardit. Analysis of the initialization stage of a pittsburgh approach learning classifier system. In *Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 1843–1850. ACM, 2005.

[3] J. Bacardit, E. Burke, and N. Krasnogor. Improving the scalability of rule-based evolutionary learning. *Memetic Computing*, 1(1):55–67, Mar. 2009.

[4] J. Bacardit, M. Stout, J. D. Hirst, A. Valencia, R. E. Smith, and N. Krasnogor. Automated alphabet reduction for protein datasets. *BMC Bioinformatics*, 10:6, 2009.

[5] M. V. Butz. *Rule-Based Evolutionary Online Learning Systems: A Principled Approach to LCS Analysis and Design*, volume 109 of *Studies in Fuzziness and Soft Computing*. Springer, 2006.

[6] M. V. Butz and M. Pelikan. Studying XCS/BOA learning in boolean functions: structure encoding and random boolean functions. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 1449–1456. ACM, 2006.

[7] J. Drugowitsch. *Design and Analysis of Learning Classifier Systems: A Probabilistic Approach (Studies in Computational Intelligence)*. Springer Publishing Company, Incorporated, 1 edition, 2008.

[8] N. Edakunni, T. Kovacs, G. Brown, and J. A. Marshall. Modeling UCS as a mixture of experts. In *GECCO '09: Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 1187–1194. ACM, 2009.

[9] D. E. Goldberg. *The Design of Innovation: Lessons from and for Competent Genetic Algorithms*. Kluwer Academic Publishers, 2002.

[10] J. H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[11] K. D. Jong and W. M. Spears. Learning concept classification rules using genetic algorithms. In *Proceedings of the 12th international joint conference on Artificial intelligence - Volume 2*, pages 651–656. Morgan Kaufmann Publishers Inc., 1991.

[12] X. Llorà, K. Sastry, and D. Goldberg. Binary rule encoding schemes: A study using the compact classifier system. In *Learning Classifier Systems*, volume 4399 of *Lecture Notes in Computer Science*, pages 40–58. Springer Berlin / Heidelberg, 2007.

[13] A. Orriols-Puig. *New Challenges in Learning Classifier Systems: Mining Rarities and Evolving Fuzzy Models*. PhD thesis, Universitat Ramon Llull, Barcelona, Catalonia, Spain, 2008.

[14] A. Orriols-Puig, X. Llorà, and D. E. Goldberg. How xcs deals with rarities in domains with continuous attributes. In *GECCO'10*, pages 1023–1030, 2010.

[15] R. L. Rivest. Learning decision lists. *Machine Learning*, 2(3):229–246, 1987.

[16] P. O. Stalph, X. Llorà, D. E. Goldberg, and M. V. Butz. Resource management and scalability of the xcsf learning classifier system. *Theoretical Computer Science*, In Press, Corrected Proof:–, 2010.

[17] M. Stout, J. Bacardit, J. D. Hirst, and N. Krasnogor. Prediction of recursive convex hull class assignments for protein residues. *Bioinformatics*, 24(7):916–923, Apr. 2008.

[18] G. Venturini. SIA: a supervised inductive algorithm with genetic search for learning attributes based concepts. In P. B. Brazdil, editor, *Machine Learning: ECML-93 - Proceedings of the European Conference on Machine Learning*, pages 280–296. Springer-Verlag, 1993.

[19] S. W. Wilson. Classifier fitness based on accuracy. *Evolutionary Computation*, 3(2):149–175, June 1995.

[20] S. W. Wilson. Get real! XCS with continuous-valued inputs. In *Festschrift in Honor of John H. Holland*, pages 111–121. Center for the Study of Complex Systems, 1999.