An Adaptive Discretization in the ACDT Algorithm for Continuous Attributes

Urszula Boryczka and Jan Kozak

Institute of Computer Science, University of Silesia, Będzińska 39,
41–200 Sosnowiec, Poland
Tel.:/Fax: (+48 32) 291 82 83
{urszula.boryczka,jan.kozak}@us.edu.pl

Abstract. Decision tree induction has been widely used to generate classifiers from training data through a process of recursively splitting the data space. In the case of training on continuous-valued data, the associated attributes must be discretized in advance or during the learning process. The commonly used method is to partition the attribute range into two or several intervals using single or a set of cut points. One inherent disadvantage in these methods is that the use of sharp cut points makes the induced decision trees sensitive to noise. To overcome this problem this paper presents an alternative method called adaptive discretization based on Ant Colony Decision Tree (ACDT) approach. Experimental results showed that, by using that methodology, better classification accuracy has been obtained in both training and testing data sets in majority of cases concerning the classical decision tree constructed by ants. It suggests that the robustness of decision trees could be improved by means of this approach.

Keywords: ACDT, Ant-Miner, Data Mining, Ant Colony Decision Tree, Ant Colony Optimization.

1 Introduction

In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. In data mining, a decision tree describes data but not decisions; rather the resulting classification tree can be an input for decision making. A decision tree induction is a very popular technique in machine learning. It has a lot of beneficial aspects in comparison to other machine learning methods like, e.g., computational complexity, high accuracy and readability of data. A decision tree can be used to classify an example by starting at a root of the tree and moving through it until a leaf node, which provides the classification of the instance. The decision tree induction is a typical inductive approach to learn knowledge on classification. The decision tree induction algorithms were firstly introduced in 60's and as a general scheme of recursive partitioning is nowadays a quite attractive approach and still worth further developments [5,12,14].

P. Jędrzejowicz et al. (Eds.): ICCCI 2011, Part II, LNCS 6923, pp. 475–484, 2011.

There exist a plenty of different algorithms for decision tree induction and their implementations. All the approaches described here were implemented as variants of Ant Colony Optimization in data mining tasks. The most popular of them is the Ant-Miner algorithm. This approach firstly proposed by Parpinelli at al. applies an Ant Colony Optimization heuristic [6,7] to the classification task of data mining to discover an ordered list of classification rules [10,11]. The Ant-Miner algorithm was modified several times and deeply analyzed. Many of these modifications are described in [3]. In this article the population size of ants is discussed, new pheromone updating rules, exploration and exploitation rate and new pruning procedures and heuristic functions are tested. The heuristic function is a crucial point in this approach, so some extensions have also been proposed, mainly based on algorithms for construction decision trees such as CART [2]. In order to improve the processing time of the Ant-Miner algorithm the parallel Ant-Miner algorithm is proposed and analyzed [4]. The classical algorithm proposed by Parpinelli was not adapted to the continuous attributes, but some augmentations or extensions to different Ant-Miner versions of algorithms are applied for example the cAnt-Miner algorithm, cope with continuous attributes during the rule construction process [8,9,15]. The ACDT algorithm presented in this paper is an approach which is a different approach when compared it to the predecessor with respect to the decision tree construction. This approach is not suitable to discover list of rules. The ACDT algorithm, firstly proposed in 2010 [1] is a new technique to construct decision trees. The original version undergoing the twoing criterion or another splitting rule did not cope with the continuous attributes.

Our motivation is to show a new method of coping with the continuous attributes during the process of construction the decision tree. In the context of real-world classification problems described by nominal as well as continuous attributes it is important to overcome this problem because of appropriate interpretation these values. In order to improve our method we incorporate new splitting rule for better analyzing this real-world data sets and it seems to be one of proper ways.

In the present thesis we would like to discuss a series of important sections. The first section is devoted to an introduction to the subject of this paper. Section 2 describes Ant Colony Decision Tree approach, especially the splitting rules. Section 4 focuses on the presented new version of the algorithm cACDT. Section 5 presents the experimental study that has been conducted to evaluate the performance of cACDT, taking into consideration eleven data sets. The last section concludes obtained results and discusses the future evolution of the presented approach.

2 Ant Colony Decision Trees

In the context of decision tree construction or discovering classification rules in data mining, Ant Colony Optimization (ACO) approaches have been widely applied to different tasks. In essence, Ant Colony Decision Trees (ACDT) is a first approach for constructing decision trees and is competitive with the well known CART algorithm.

In ACDT each ant chooses the appropriate attribute for splitting in each node of the constructed decision tree according to the heuristic function and pheromone values (fig. 1). The heuristic function is based on the Twoing criterion, which helps ants divide the objects into two groups, connected with the analyzed attribute values. In this way, the attribute, which well separate the objects is treated as the best condition for the analyzed node. The best splitting is observed when we classified the same number of objects in the left and right subtrees with the maximum homogenity in the decision classes. Pheromone values represent the best way (connection) from the superior to the subordinate nodes – all possible combinations in the analyzed subtrees. For each node we calculate the following values according to the objects classified using the Twoing criterion of the superior node.

A decision tree is built in accordance with splitting rule that performs the multiple splitting of learning sample into smaller parts. Data in each node have to be divided into two parts with maximum homogeneity in the decision class.

Twoing criterion will search for two classes that will make up together more then 50% of the data. Twoing splitting rule will maximize the following change-of-impurity measure which implies the following maximization problem for nodes m_l, m_r:

$$\underset{a_j \leq a_j^R, j=1,\ldots,M}{\arg\max} \left(\frac{P_l P_r}{4} \left[\sum_{k=1}^{K} |p(k|m_l) - p(k|m_r)| \right]^2 \right), \tag{1}$$

where:

$p(k|m_l)$ – the conditional probability of the class k provided in node m_l,
P_l – the probability of transition objects into the left node m_l,
P_r – the probability of transition objects into the right node m_r,
K – number of decision class,
a_j – j–th variable,
a_j^R– the best splitting value of variable a_j.

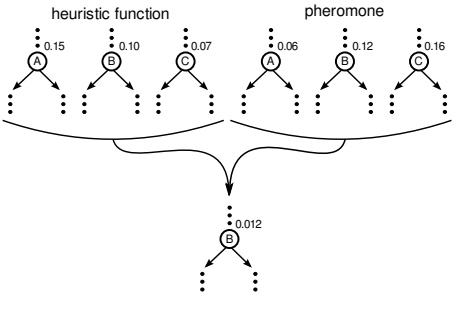

Fig. 1. Choice of splits in ACDT

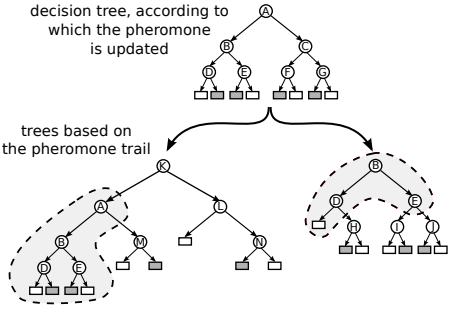

Fig. 2. Building the decision tree with pheromone

Although Twoing splitting rule allows us to build more balanced trees, this algorithm works slower than the Gini rule. For example, if the total number of classes is equal to K, then we will have 2^{K-1} possible splits.

The pseudo-code of the proposed algorithm is presented below. Lines 2–13 describe one iteration of this algorithm. At the beginning of its work, each ant builds one decision tree (lines 4–11). At the end of the loop, the best decision tree is chosen and then the pheromone is updated according to the splits performed during the process of construction the decision tree, iteratively. While constructing the tree, agents–ants are analyzing previous structures and some modifications are performed in the single node. This process is performed till the best decision tree is obtained. The process of building the decision tree is presented in Figure 2.

Algorithm 1. Pseudo-code of the ACDT algorithm

```
1  initialization_pheromone_trail(pheromone);
2  for number_of_iterations do
3      best_tree = NULL;
4      for number_of_ants do
5          new_tree = build_tree(pheromone);
6          pruning(new_tree);
7          assessment_of_the_quality_tree(new_tree);
8          if new_tree is_higher_quality_than best_tree then
9              best_tree = new_tree;
10         endIf
11     endFor
12     update_pheromone_trail(best_tree, pheromone);
13 endFor
14 result = best_constructed_tree;
```

The value of the heuristic function is determined according to the splitting rule employed in CART approach (see formula (1)), in dependence on the chosen criterion. Whereas the probability of choosing the appropriate test in the node is calculated according to a classical probability used in ACO:

$$p_{i,j} = \frac{\tau_{m,m_{L(i,j)}}(t)^{\alpha} \cdot \eta_{i,j}^{\beta}}{\sum_{i}^{a} \sum_{j}^{b_i} \tau_{m,m_{L(i,j)}}(t)^{\alpha} \cdot \eta_{i,j}^{\beta}} \tag{2}$$

where:

$\eta_{i,j}$ – a heuristic value for the test of the attribute i and value j,

$\tau_{m,m_{L(i,j)}}$ – an amount of pheromone currently available at time t on the connection between nodes m and m_L, (it concerns the attribute i and value j),

α, β – the relative importance with experimentally established values 1 i 3.

The initial value of the pheromone trail, similarly to the Ant–Miner approach is established in dependence on the number of attribute values. While

the pheromone updates are performed (3) by increasing the previous values on each pairs of nodes (parent–child):

$$\tau_{m,m_L}(t+1) = (1-\gamma) \cdot \tau_{m,m_L}(t) + Q(T) \tag{3}$$

where $Q(T)$ determines the evaluation function of decision tree (see formula (4)), and γ is a parameter representing the evaporation rate, equal to 0.1.The evaluation function for decision trees will be calculated according to the following formula:

$$Q(T) = \phi \cdot w(T) + \psi \cdot a(T,P) \tag{4}$$

where:
$w(T)$ – the size (number of nodes) of the decision tree T,
$a(T,P)$ – the accuracy of the classification object from a test set P by the tree T,
ϕ and ψ – constants determining the relative importance of $w(T)$ and $a(T,P)$.

3 ACDT Algorithm for Continuous Attributes

Most empirical learning systems are given a set of pre-classified cases, each described by a vector of attribute values, and construct from them a mapping from attribute values to classes. The attributes used to describe cases can be grouped into continuous attributes, which values are real numbers, and discrete attributes with unordered nominal values [14].

As mentioned in the introduction, the previous version of the Ant Colony Decision Tree approach does not cope with continuous attributes directly. It requires continuous attributes to be discretized in a preprocessing step. In classical version of the ACDT algorithm splitting the data due to the default equality (equivalence) test (5) or adjunction test (6) applying to the discrete attributes (fig. 3).

$$T_e(x) = \begin{cases} 1 & \text{, if } a_j(x) = v \\ 0 & \text{, otherwise} \end{cases} \tag{5}$$

$$T_a(x) = \begin{cases} 1 & \text{, if } a_j(x) \in V \\ 0 & \text{, otherwise} \end{cases}, \tag{6}$$

where:
T – a decision tree,
x – an object,
v – attribute value,
V – set of the attribute values,
a_j – a conditional attribute.

Many researchers have recently noted that the equality or adjunction rules in the decision tree construction are weaker in domains with a preponderance of continuous attributes than for learning tasks that have mainly discrete attributes. Discussing the effect of replacing continuous attributes by discrete attributes,

it can be observed that this approach causes limitation of the prediction new cases [8,9,13].

The proposed modification of the ACDT algorithm coping with continuous attributes incorporates an inequality test (7). For discrete attributes we still perform the default adjunction test in this new version of ACDT:

$$T(x) = \begin{cases} 1 & \text{, if } a_j(x) \leq c \\ 0 & \text{, otherwise } (a_j(x) > c) \end{cases}, \tag{7}$$

where c is a threshold value (cut-off point).
The effects of these changes influence on the precision and prediction of splitting the continuous values. The observations are as follows: the cutting points are performed according to the values occurring in the analyzed attributes in the testing set. It can be assessed empirically in series of experiments with a substantial number of learning tasks. For attribute a_j each cutting point is performed due to the concrete value derived from the testing set and finally is evaluated according to the chosen criterion (e.g. Twoing criterion (1)).

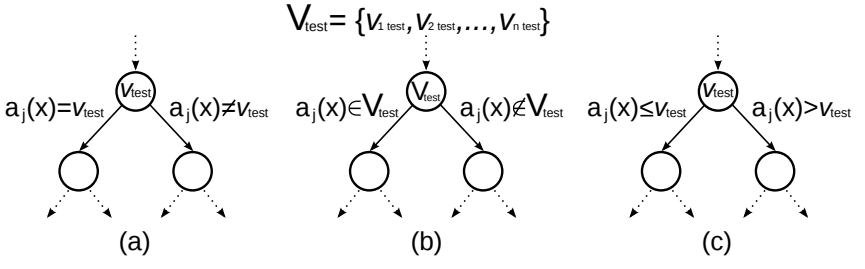

Fig. 3. Tests on the attributes of the ACDT and cACDT algorithm: (a) equality (equivalence) test, (b) adjunction test, (c) inequality test

4 Experiments

A variety of experiments were conducted to test the performance and behavior of the proposed algorithm. First we describe our experimental methodology and explain its motivation. Then we present and discuss our results. In this section we will consider an experimental study (see Table 2 and 3) performed for the following adjustments. We have performed 100 experiments for each data set. Each experiment included 25 generations with the population size of ant colony equal to 10.

For this purpose, a post–pruning phase was applied to the decision trees that were generated by ACDT to form the lookahead sample, and the size of the pruned decision trees will be considered. Previous comparative studies did not find a single pruning method that is generally the best and conclude that different pruning techniques behave similarly. Therefore, during the analysis we used the Twoing criterion and Error–Based Pruning, and examine post–pruning of the final trees. We have performed the experimental study for two approaches:

ACDT – with quality coefficient calculated accordingly to with the learning set, when the algorithm is evaluated based on the testing set. We also used the Twoing criterion. We evaluated the effectiveness of the approach and then updated the pheromone values for the best quality decision tree. The splitting was performed in each node accordingly to with the adjunction test.

cACDT (ACDT with continuous attributes) – with quality coefficient calculated accordingly to with the learning set, when the algorithm is evaluated based on the testing set. We also used the Twoing criterion. We evaluate the effectiveness of the approach and then update the pheromone values for the best quality decision tree. The splitting in each node is performed accordingly to the adjunction test, and for continuous attributes we apply inequality test.

4.1 Data Sets

Evaluation of the performance behavior of ACDT was performed using 11 public–domain data sets from the UCI (University of California at Irvine) data set repository available from at: `http://archive.ics.uci.edu/ml/`. Table 1 shows the main characteristics of the data sets, which are divided into two groups in a random way: training ($\frac{2}{3}$ of all objects) and testing ($\frac{1}{3}$ of all objects) sets, appropriately. In order to obtain reliable performance estimates train–and–test were carried out to produce each of the statistics presented in the tables (see below). The experiments were carried out on the Intel Celeron 2.27 GHz Computer with 3 GB RAM.

Table 1. Original parameters in data sets

Dataset	Number of instances	Number of attribute		Decision Class
		nominal	continuous	
australian	690	8	6	2
bands	512	19	20	2
bupa	345	0	6	2
crx	690	9	6	2
ecoli	336	0	7	8
glass	214	0	9	7
hepatitis	155	13	6	2
ionosphere	351	0	34	2
iris	150	0	4	3
spam	4601	0	57	2
yeast	1484	0	8	10

4.2 Results

Results of experiments prove that dynamic, adaptive discretization relies on the inequality test increase the classification accuracy of the constucted decision trees.

Table 2. Comparative study accuracy rate (standard deviations in parentheses)

Dataset	ACDT algorithm		cACDT algorithm		Ant-Miner algorithm [8]	cAnt-Miner algorithm [8]
australian	0.8650	(0.0013)	0.8603	(0.0083)	0.8552	0.8660
bands	0.6813	(0.0294)	**0.7370**	(0.0374)	—	—
bupa	0.5827	(0.0218)	0.5843	(0.0324)	—	—
crx	0.8565	(0.0000)	**0.8817**	(0.0112)	0.8532	0.8556
ecoli	0.4231	(0.0114)	**0.9505**	(0.0174)	—	—
glass	0.4940	(0.0289)	**0.8615**	(0.0269)	0.5148	0.6569
hepatitis	0.7107	(0.0283)	**0.7844**	(0.0280)	0.7461	0.8489
ionosphere	**0.8637**	(0.0136)	0.8495	(0.0213)	0.9068	0.9000
iris	0.7506	(0.0604)	**0.8984**	(0.0123)	—	—
spam	0.9178	(0.0040)	**0.9393**	(0.0087)	—	—
yeast	0.4092	(0.0138)	**0.4341**	(0.0167)	—	—

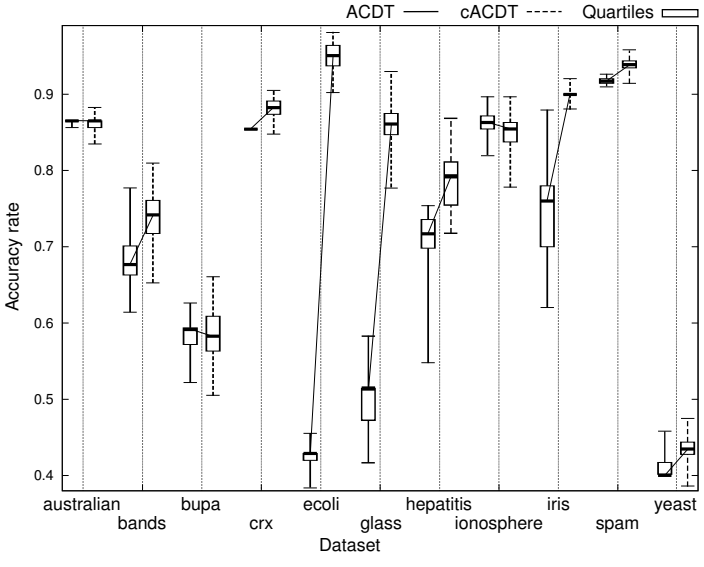

Fig. 4. Results (accuracy rate)

Results of these trials, summarized in tab. 2 show the accuracy of the classification for two approaches. The best results are presented in bold. This approach leads to improved accuracy on eight of the tasks (about 50%). Most of the improvements are significant - evidence of this situation is provided by fig. 4.

Please note, that for data sets with big number of decision classes (ecoli, glass and yeast), classical version of our approach (ACDT) obtains really low values of classification accuracy. The result confirms the clear trend which shows that the adaptive discretization degrades the performance of the presented approach

more as data set become larger, but can be beneficial for data sets with smaller clases. In general, it is finished after the only one split of data (the tree consists on 1 node), so it is unprofitably. The results of the performance of cAnt-Miner [8] highlights the promising effects of the proposed modification, leading to not very large trees and the structure of the tree is clarified. The above-mentioned results are depicted in Table 3. It is worth noting that it is natural.

The accuracy of cACDT approach is rivaled or surpassed the ACDT on the most of the datasets. The results show that the straightforward changes concerning the continuous attributes lead to an overall improvement in its performance on the five testable data sets. Only two data sets are significantly difficult during the analysis. The same dependence could be observed in the case of Ant-Miner and cAnt-Miner [8]. In this situation we highlight the fact of considerable improvement of the performance of our proposition. Statistical (The Wilcoxon Two Sample Test) evidence confirms that cACDT is better than ACDT algorithm.

Table 3. Comparative study number of nodes (standard deviations in parentheses)

Dataset	ACDT algorithm		cACDT algorithm	
australian	1.15	(0.87)	8.45	(6.54)
bands	10.44	(2.77)	18.27	(4.58)
bupa	4.56	(1.19)	10.53	(3.13)
crx	1.00	(0.00)	10.76	(2.50)
ecoli	3.64	(1.14)	7.04	(0.99)
glass	4.92	(1.23)	6.80	(1.24)
hepatitis	4.07	(1.17)	5.22	(1.89)
ionosphere	3.58	(1.11)	10.62	(2.60)
iris	8.87	(1.76)	3.20	(0.40)
spam	53.73	(12.43)	48.77	(7.82)
yeast	4.56	(5.47)	37.74	(11.40)

5 Conclusions

This study has focused on the extension to ACDT, named cACDT, which copes with continuous attributes during the decision trees construction. Our approach does not require a discretization method in a preprocessing step. The investigation of the variety of approaches concerning the splitting rule applied in ACDT and cACDT on several data sets has provided evidence for the following proposals. In test evaluation, the decrease of the number of levels the decision trees does not influence on the quality of this classification. Approaches that ignore the continuous attributes (ACDT) perform badly and lead to very inferior performance.

Our proposition cACDT has been compared against the predecessor of the ACDT with respect to predictive accuracy and the structure of the tree. Regarding the predictive accuracy Ant Colony Decision Tree with adaptive discretization - cACDT significantly outperformed the previous version. Please note, that

this modification is not complicated and rather intuitive. Therefore the results obtained by cACDT are promising and worth further analysis.

Really good performance of cACDT with simple modification leads to the next application concerning regresion. As future research direction, it would be interesting to extend this approach to allow the creation a specific analysis for decision attribute with continuous values.

References

1. Boryczka, U., Kozak, J.: Ant colony decision trees – a new method for constructing decision trees based on ant colony optimization. In: Pan, J.S., Chen, S.M., Nguyen, N. (eds.) ICCCI 2010. LNCS, vol. 6421, pp. 373–382. Springer, Heidelberg (2010)
2. Boryczka, U., Kozak, J.: A New Heuristic Function in Ant–Miner Approach. In: ICEIS 2009, Milan, Italy, pp. 33–38 (2009)
3. Boryczka, U., Kozak, J.: New Algorithms for Generation Decision Trees – Ant–Miner and Its Modifications, pp. 229–264. Springer, Berlin (2009)
4. Boryczka, U., Kozak, J., Skinderowicz, R.: Parellel Ant–Miner. Parellel implementation of an ACO techniques to discover classification rules with OpenMP. In: MENDEL 2009, pp. 197–205. University of Technology, Brno (2009)
5. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Chapman & Hall, New York (1984)
6. Dorigo, M., Gambardella, L.M.: Ant Colony System: A cooperative learning approach to the Traveling Salesman Problem. IEEE Tr. Evol. Comp. 1, 53–66 (1997)
7. Dorigo, M., Birattari, M., Stützle, T., Libre, U., Bruxelles, D., Roosevelt, A.F.D.: Ant colony optimization – artificial ants as a computational intelligence technique. IEEE Comput. Intell. Mag. 1, 28–39 (2006)
8. Otero, F., Freitas, A., Johnson, C.: cAnt-Miner: An ant colony classification algorithm to cope with continuous attributes. In: Dorigo, M., Birattari, M., Blum, C., Clerc, M., Stützle, T., Winfield, A.F.T. (eds.) ANTS 2008. LNCS, vol. 5217, pp. 48–59. Springer, Heidelberg (2008)
9. Otero, F.E.B., Freitas, A.A., Johnson, C.G.: Handling continuous attributes in ant colony classification algorithms. In: CIDM, pp. 225–231 (2009)
10. Parpinelli, R.S., Lopes, H.S., Freitas, A.A.: An ant colony algorithm for classification rule discovery. In: Abbas, H., Sarker, R., Newton, C. (eds.) Data Mining: a Heuristic Approach, Idea Group Publishing, London (2002)
11. Parpinelli, R.S., Lopes, H.S., Freitas, A.A.: Data mining with an ant colony optimization algorithm. IEEE Transactions on Evolutionary Computation, Special issue on Ant Colony Algorithms, 321–332 (2004)
12. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco (1993)
13. Quinlan, J.R.: Improved use of continuous attributes in c4.5. Journal of Artificial Intelligence Research 4, 77–90 (1996)
14. Rokach, L., Maimon, O.: Data Mining With Decision Trees: Theory and Applications. World Scientific Publishing, Singapore (2008)
15. Schaefer, G.: Ant colony optimisation classification for gene expression data analysis. In: Sakai, H., Chakraborty, M.K., Hassanien, A.E., Ślęzak, D., Zhu, W. (eds.) RSFDGrC 2009. LNCS, vol. 5908, pp. 463–469. Springer, Heidelberg (2009)