ANT COLONY SYSTEM ALGORITHM FOR EXTRACTING MATHEMATICAL RELATIONS FROM DATABASE

R. F. Marques[1] and L. H. A. Monteiro[1,2]
[1]*Escola de Engenharia, Universidade Presbiteriana Mackenzie, Rua da Consolação 896, 01302-907, São Paulo, SP, Brazil*
[2]*Escola Politécnica, Universidade de São Paulo*
Av. Prof. Luciano Gualberto (travessa 3) 380, 05508-900, São Paulo, SP, Brazil

Keywords: Ant colony system, Breast cancer, Data mining, Swarm intelligence.

Abstract: An algorithm, inspired on the strategy employed by ant colonies for seeking food, was developed in order to extract mathematical formulas from data. This algorithm, called Formula Miner, is applied in a breast cancer diagnosis database and its performance is compared to the performances of some well-known data mining algorithms.

1 INTRODUCTION

Ant colonies do not present an evident centralized management. A social organization working without explicit command may seem incomprehensible, because no one has individual conscience about what is needed to do in order to accomplish the vital activities for the society, such as walk to find food. According to Deneubourg et al. (1993), ant walk is essentially a probabilistic behavior, which can be verified by studying the strategy that ants adopt in seeking food. Usually, some ants randomly go out the way and discover new sources of food. These new sources require the recruitment of new ants in order to explore them. Such a behavior gives them adaptive advantage, allowing fast adaptation when the environment changes.

When an ant moves, a chemical substance, called pheromone, is deposited on the floor, making a trail. Another ant finding this pheromone trail can decide to follow it, leaving along this trail its own pheromone. Thus, ants communicate among them, sharing information, and they can find the right way towards sources of food (Beckers et al., 1992).

This collective behavior, known as swarm intelligence, emerges from an autocatalytic process: as more ants follow a pheromone trail, more attractive this trail becomes to be followed. Such a process is characterized by positive feedback, which reinforces itself (Dorigo et al., 1996). Hence, the probability of an ant choosing a way increases with the number of ants that previously have chosen this same way.

Such an ant colony framework have inspired the development of evolutionary algorithms in order to numerically solve complex computational tasks. Usually, artificial ants are employed, in a collaborative manner, to find optimum solutions in large search spaces. The original idea was due to Dorigo et al. (1996) and Dorigo and Gambardella (1997). They showed how the simple behavior of following pheromone trails can be used to solve the traveling salesman problem. Their algorithms were based on the observation that ants are able to create the shortest path from their nest to the food source. Similar approaches have been developed by several authors: Gambardella et al. (1997) to solve the quadratic assignment problem; Rajesh et al. (2001) to optimize chemical process design; Parpinelli et al. (2002) to extract classification rules from data; Gómez et al. (2004) to plan the topology of power systems; Samrout et al. (2005) to minimize the preventive maintenance cost of series-parallel systems; Rahhal and Abu-Al-Nadi (2007) to determine the optimum configuration antenna array; Mirabedini et al. (2008) to find out effective routing in communications networks.

Here, we present an ant colony system algorithm for obtaining mathematical formulas from data. This algorithm, called Formula Miner, is applied in a breast cancer diagnosis database and its performance is compared to the performances of other data mining algorithms.

Marques R. and Monteiro L. (2009).
ANT COLONY SYSTEM ALGORITHM FOR EXTRACTING MATHEMATICAL RELATIONS FROM DATABASE.
In *Proceedings of the International Conference on Agents and Artificial Intelligence*, pages 204-207
DOI: 10.5220/0001656402040207

2 THE ALGORITHM FORMULA MINER

Bonabeau et al. (1999) gave tips about how to develop an algorithm based on ant colony behavior:

1. choosing an appropriated representation of the problem, where artificial ants can incrementally build or modify the solutions through a probabilistic rule based on the amount of pheromone deposited in a trail (a valid solution) and on other local heuristics;
2. defining a heuristic function η measuring the quality of a pheromone trail (an ant way) in the search space and a procedure for reinforcing this trail.

Formula Miner intends to construct analytical formulas, extracting knowledge from database. For example, in the Cartesian plane $x_1 \times x_2$ the algorithm can find the (non-linear) function $x_2 = F(x_1)$ that divide the plan in two regions, thus classifying the data.

The algorithm uses a symbolic representation to build formulas by appending terms to an initial formula. Here, we impose that all formulas begin with the number zero, and zero occurs only at this first term. An operator followed by a variable or a constant composes each term. We use only the four basic arithmetic operators $(+,-,\times,\div)$ plus parentheses. The constants are integer numbers pertaining to the interval $(1,9)$. The variables in a n-dimensional space are x_1, x_2, ..., x_n. Three examples of terms: "$-x_1$", "$\div 3$" and "$\div 3)$": here "$\div 3$" means that the last term in the expression will be divided by 3; "$\div 3)$" means that all previous expression will be divided by 3. The formula is represented by $x_n = F(x_1, x_2, ..., x_{n-1})$.

The trail (a partial formula) generated by the first ant released in the search space is stored. Then, the pheromone value corresponding to each term composing the trail is updated according to the rule described below. Thus, it is possible to create conditions for the next ants follow a previous trail, allowing the convergence of the algorithm. The higher the pheromone in a trail, the more attractive this trail to the next ants. New ants can originate new formulas too; that is, they do not necessarily follow the trails that already exist.

Let Q_k be the quality criterion of the formula F_k, where k is the label of the ant. The value of Q_k is used to update the amount of pheromone τ on the corresponding trail. We adopt the same expression for Q_k suggested by Parpinelli et al. (2002). Thus, Q_k is given by:

$$Q_k = \left(\frac{TP}{TP+FN}\right)\left(\frac{TN}{TN+FP}\right) \tag{1}$$

where TP is the number of true positives, TN the number of true negatives, FN the number of false negatives, FP the number of false positives. Notice that $0 \le Q_k \le 1$.

The pheromone update at the iteration t is performed for each term i composing the formula F_k according to:

$$\tau_i(t+1) = \tau_i(t)(1+\gamma Q_k) \tag{2}$$

where γ is a positive number. For simplicity, a factor for pheromone evaporation is not taken into account.

We choose the heuristic function η_k^i defined by the ratio between the number of correct classifications performed by the formula F_k^i after including the term i, and the number of database records. Thus, this heuristic function is written as:

$$\eta_k^i = \frac{TP+TN}{TP+TN+FP+FN} \tag{3}$$

3 FORMULA CONSTRUCTION

Artificial ants are used to generate a formula, which is modified by attaching pairs of operator and variable-or-constant. The ant k starts with an empty formula (number zero) and produces the formula F_k^i by appending terms at the current formula, where i is the label of the new term. Each appended term corresponds to a partial trail walked by the ant, and this fragment receives pheromone according to the quality criterion achieved by the final formula.

The probability P_k^i of taking by chance the term i depends on the heuristic function and the pheromone amount deposited on the trails. Thus, the probability P_k^i that the ant k will pick the term i to be attached to the formula can be given by (Dorigo et al., 1996):

$$P_k^i = \frac{(\tau_i(t))^\alpha(\eta_k^i)^\beta}{\sum_{i=1}^N(\tau_i(t))^\alpha(\eta_k^i)^\beta} \tag{4}$$

where N is the number of trail options offered to the ant, and α and β are parameters to control the relative weight between the pheromone amount on the trail and the heuristic function.

In order to find the term $i+1$ to be included in F_k, the heuristic function η_k^{i+1} is calculated to all trail options (all combinations between operator and variable-or-constant, with or without parentheses). These options form the trail matrix. Then, the probabilities P_k^{i+1} related to the offered options are calculated. The probability P_k^{i+1} of each term $i+1$ is taken into account in the draw determining the term that will be added to the formula. With this new term, the quality criterion Q_k^{i+1} is calculated and compared with Q_k^i, corresponding to the formula without it. If $Q_k^{i+1} > Q_k^i$ such a term is accepted and added to the formula F_k. If not, this term is discarded and a new draw is performed. A tentative counter is incremented in this case. When a term is added to F_k, the tentative counter is reset and the process for choosing terms begins again. The parameter M controls how many times the algorithm must insist to improve the quality criterion by attaching terms. When the counter reaches a preset value, the algorithm records the formula F_k obtained up to this moment, and begins a new cycle with the next ant, $k+1$. Another criterion used to stop the algorithm is the maximum number of terms L that the formula can have.

After the ant k has completed its formula F_k, the pheromone amount τ_i for each position i of the formula F_k is updated following expression (2).

Then, when the next ant $k+1$ starts to construct its own formula, it considers the deposited pheromone on the trails formed by the previous ants. This process is repeated until a maximum number of ants is reached, specified by the parameter R.

A simple description of the algorithm is the following:

```
Set the initial trail matrix ($i = 0$)
Calculate $\eta_0^i$ for each term appearing at the trail matrix
Calculate $P_0^i$ for each term appearing at the trail matrix
For each ant $k = 1$ until $R$
    Do until tentative counter $\leq M$
        Choose at random the term $i$ considering $P_k^i$
        Calculate $Q_k^i$ to $F_k$ with the randomly picked term
        If $Q_k^{i+1} > Q_k^i$
            Attach the term $i$ to $F_k$
            Increment $i$
            Reset the tentative counter
            Calculate $\eta_k^i$ for each term at the trail matrix
            Calculate $P_k^i$ for each term at the trail matrix
        Else
        Increment the tentative counter
    End Do
    Update the pheromone in each segment forming $F_k$
    Reset the tentative counter
    Reset the term position ($i = 0$)
    Reset $\eta_k^i$
    Calculate $P_k^i$ for each term at the trail matrix
End Loop
```

4 NUMERICAL EXPERIMENT

Formula Miner was tested on the Wisconsin Diagnostic Breast Cancer WDBC dataset, which contains 559 cases, 2 classes and 30 numerical attributes. According to Mangasarian et al. (1995), 3 of these 30 attributes are more relevant for diagnostics. Thus, we selected them among the other attributes to search for a mathematical relationship that separates the cases of malignant and benignant breast cancer. These attributes are: mean texture (x), worst smoothness (y), and worst area (z) of the analyzed cells (Mangasarian et al., 1995). Due to the dimensional differences among these attributes, it was necessary to include scale factors. Thus: $x_1 \equiv x$, $x_2 \equiv 10y$, $x_3 \equiv z/10$.

Then, Formula Miner was applied to search a mathematical relationship for these three attributes, in order to correctly classify malignant and benignant cases. The goal is to find a formula that expresses one attribute in terms of the two others. The attributes are related by the function $x_3 = F(x_1, x_2)$

The algorithm accuracy was evaluated using a ten-fold cross-validation method (Stone, 1974). The database is divided in ten equal parts. One part is set apart and the algorithm is applied in the other nine parts. The resulting formulas are tested at the part that was removed of the database. In this procedure, all cases are used only once as test and nine times to run the Formula Miner.

The accuracy rate of each evaluation is defined as the quotient between the number of correctly classified cases by the total number of tested cases, using the heuristic function given by expression (3). The final accuracy rate is the arithmetical average of the accuracy rate of the nine rounds, followed by the corresponding standard deviation.

We chose the following parameter values: maximum number of ants R = 15; maximum number of tentatives M = 5; $\alpha = 4$; $\beta = 5$; $\gamma = 3$. The number of registries used in each turn was 512; the number of registries used in the validation procedure was 57.

Our results were compared with the ones obtained by applying well-known algorithms, for the same database (the same 3 attributes) and all using a ten-fold cross-validation procedure. The algorithms and their respective performances are: Ant Miner (Parpinelli et al., 2002): (96.0 ± 1.0)%; C4.5 (Quinlau, 1993): (95.0 ± 0.3)%; Formula Miner: (81.4 ± 7.6)%; MSM-T (Mangasarian et al., 1995): 97.5%.

We must notice the differences among the approaches and the purposes of the algorithms in relation to Formula Miner in order to properly comment these results. The MSM-T (Multisurface Method Tree) looks for multiples planes (that can not form a function), while Formula Miner looks for a unique mathematical relationship that defines just one continuous surface to classify the diagnosis. In the algorithms C4.5 and Ant Miner, the attributes were used as discrete range of values, and the their results depend on the adopted ranges. Formula Miner deals with continuous attributes.

The algorithms C4.5 and MSM-T are based on decision-tree; Ant Miner is based on rule construction. In fact, these three algorithms accept discontinuities in a possible classification rule.

The Formula Miner goal is to obtain a continuous function for better classifying the cases. Our best formula presented performance of 92.98%. It is given by:

$$\begin{aligned} x_3 &= x_1(x_2^2+x_2+1)/x_2 \\ &\quad +(x_2^2+17)/x_2+1/x_1+5x_2 \end{aligned} \qquad (5)$$

Observe that the performance of this formula is comparable to the ones of the other data-mining algorithms.

5 CONCLUSIONS

This paper presented a way of extracting analytical formulas from database, inspired in the ant colony behavior. The aim is to separate data in a multidimensional search space by a continuous function. These formulas written in a literal way can be useful to classify and to develop analytical mathematical models from a database.

ACKNOWLEDGEMENTS

LHAM is partially supported by CNPq.

REFERENCES

Beckers, R., Deneubourg, J. L., & Goss, S. (1992). Trails and u-turns in the selection of the shortest path by the ant Lasius Niger. *Journal of Theoretical Biology, 159,* 397-415.

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm intelligence: From natural to artificial systems*. New York: Oxford University Press.

Deneubourg, J. L., Pasteels, J. M., & Verhaeghe, J. C. (1983). Probabilistic behaviour in ants: a strategy of errors? *Journal of Theoretical Biology, 105,* 259-271.

Dorigo, M., Maniezzo, V., & Corloni, A. (1996). The ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics B, 26,* 29-41.

Dorigo, M., & Gambardella, L. M. (1997). Ant colonies for the travelling salesman problem. *BioSystems, 43,* 73-81.

Gómez, J. F., Khodr, H. M., de Oliveira, P. M., Ocque, L., Yusta, J. M., Villasana, R., & Urdaneta, A. J. (2004). Ant colony system algorithm for the planning of primary distribution circuits. *IEEE Transactions on Power Systems, 19,* 996-1004.

Gambardella, L. M., Taillard, E., & Dorigo, M. (1997). Ant colonies for QAP. *Technical Report IDSIA97-4* Lugano, Switzerland.

Mangasarian, O. L., Street, W. N., & Wolberg, W. H. (1995). Breast cancer diagnosis and prognosis via linear programming. *Operations Research, 43,* 570-577.

Mirabedini, S. J., Teshnehlab, M., Shenasa, M. H., & Rahmani, A. M. (2008). Flar: An adaptive fuzzy routing algorithm for communications networks using mobile ants. *Cybernetics and Systems, 39,* 684-702.

Parpinelli, R. S., Lopes, H. S., & Freitas, A. A. (2002). Data mining with an ant colony optimization algorithm. *IEEE Transactions on Evolutionary Computation, 6,* 321-332.

Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. San Francisco: Morgan Kaufmann.

Rahhal, J. S., & Abu-Al-Nadi, D. I. (2007). A general configuration antenna array for multi-user systems with genetic and ant colony optimization. *Electromagnetics, 27,* 413-426.

Rajesh, J., Gupta, K., Kusumakar, H. S., Jayaraman, V. K., & Kulkarni, B. D. (2001). Dynamic optimization of chemical processes using ant colony framework. *Compututers & Chemistry, 25,* 583-595.

Samrout, M., Yalaoui, F., Chatelet, E., & Chebbo, N. (2005). New methods to minimize the preventive maintenance cost of series-parallel systems using ant colony optimization. *Reliability Engineering and System Safety, 89,* 346-354.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions. *Journal of Royal Statistical Society B, 36,* 111-147.