Combining Machine Learning and Active Objects for Parallel Data Mining

F.Medjahed and G.Fernández

Dep. Ingeniería de Sistemas Telemáticos,
E.T.S.I Telecomunicación, Ciudad Universitaria,
2804 Madrid, Spain.

e-mail: medjahed,gfer@gsi.dit.upm.es

1 Introduction

Nowadays, the necessity and usefulness of the field of Data Mining (DM) and Knowledge Discovery in Databases (KDD) are largely established by both the scientific and industrial communities; and a number of real applications have already been developed in domains ranging from space data to financial analysis [1]. However, the need for scaling up DM algorithms is a natural requirement of the more and more huge size of real world Databases (DB).
In this work, we are interested in the task of classification based on rule induction technique. The paper presents the system IFORD (Induction of First Order Rules from Databases), designed for learning first order rules from relational DB. The serial version of IFORD is explained in section 2. The next section explains how active objects are used to allow the parallel execution of the mining process. We conclude the paper by presenting our first experiments and future directions.

2 Conceptual Description of IFORD

The aim of IFORD algorithm is to generate a logical definition in term of a set of first order rules that explain a concept or a relation. The induced description can be used to better understand a concept or to classify new tuples.
Example:

competitive(x) ← intelligently_managed(x), reputable(x)
competitive(x) ← intelligently_managed(x), powerful(x)

This rules define the competitiveness of companies. If we execute this rule on a deductive DB system, storing previously unseen data such as the names of companies, whether they are intelligently managed or not, powerful or not, and reputable or not, then the result will be the name of competitive companies.

IFORD algorithm is a natural extension of the sequential-covering algorithms to first order representations [4]. In order to generate a rule, the system needs:

1. The objective or target relation indicating that we would like to learn rules describing it.
2. All kinds of background relations that the user considers as useful for the definition.
3. Training examples, formed by a set of examples known to be true for the target relation and a set of examples known to be false for the target relation. The former are usually referred to as positive tuples, and the latter are known as negative tuples.

IFORD algorithm is based on the strategy of learning one rule, removing the positive tuples it covers, then iterating the process with the remaining examples.
To lean one rule, IFORD's search begins the rule generation process with a rule of empty antecedent. In each iteration the rule is specialized by adding some condition. The process of candidate rule generation is taken from the work of Quinlan[8] and the choice the best specialization is determined by a heuristic measure inspired from the probabilistic rule based heuristic [6].

2.1 IFORD´s heuristic

A probabilistic rule associated to two propositions A and B, is an if-then statement (noted ($A \rightarrow B$), for short) whose effect is that if the proposition A occurs, then there is a probability p that the proposition B is true. So the interest in a probabilistic rule ($A \rightarrow B$) can be computed as a function of the probabilities of A, B, and A&B.

Considering the numbers of examples T_A, T_B satisfying A, B respectively, and the number of examples $T_{A\&B}$ simultaneously satisfying A and B. A straightforward set of principles that a good Rule Interest (RI) measure must achieve are:

- $RI(A \rightarrow B) = 0$, if the propositions A and B are statistically independent.
- $RI(A \rightarrow B)$ monotonically increases with the $T_{A\&B}$ when T_A and T_B remain the same.
- $RI(A \rightarrow B)$ monotonically decreases with T_A (or T_B) when $T_{A\&B}$ remains the same.

Accordingly, the simplest RI heuristic could, intuitively, be stated as:

$$RI(A \rightarrow B) = (T_{A\&B} - T_A/T_B)/T \qquad (1)$$

Using the expression of linear correlation between two variables, the author normalized the RI metric in the [-1 +1] interval:

$$RI(A \rightarrow B) = \frac{T_{A\&B} - \frac{T_A T_B}{T}}{\sqrt{T_A T_B \left(1 - \frac{T_A}{T}\right)\left(1 - \frac{T_B}{T}\right)}} \quad (2)$$

To use RI with IFORD Algorithm, we first must specify a meaningful probabilistic rule. Being P the target relation and L_k k=1..i-1 the different conditions added to the rule after i iterations, the current state of the rule is:

$$P \leftarrow L_0, L_1, \ldots L_{i-1} \quad (3)$$

Then when we will looki for the literal L_i, a probabilistic rule can be defined as:

$$L_i \rightarrow \left[P \leftarrow L_1, L_2, \ldots L_{i-1}\right] \quad (4)$$

then by analogy and defining:

T_i^+ as the size of positive tuples covering the current rule.

T_i^- as the size of negative tuples covering the current rule.

T_i^{++} as the size of positive tuples covered by the literal L_i

T_i^{-+} as the size of positive tuples covered by the literal L_i

we obtain the following metric:

$$RI(L_i) = \frac{T_i^- T_i^{++} - T_i^+ T_i^{-+}}{\sqrt{T_i^- T_i^+ (T_i^{++} + T_i^{-+})(T_i^+ + T_i^- - T_i^{++} - T_i^{-+})}} \quad (5)$$

The greatest positive value of RI gives the best literal. We can observe from the formula that :

- RI increases when T_i^{++} does and others parameters remain the same. Hence the probability to select a literal increases when its positive covering increases.
- RI decreases when T_i^{-+} does and other parameters remain the same. Thus, the probability to reject a candidate literal increases when its negative covering increases.

2.2 More IFORD´s Features

As told before, IFORD is conceptually based on existing machine learning algorithms. However during its design we kept in mind its use for DM. Thus, we cite the following main improvements:

1. As it happens in most learning algorithm, the search criterion that is used depends only on the number of examples and counter examples covered by the candidate. With such a criterion, extensions that are not equivalent can be so from the learning algorithm's point of view. To remedy to this problem, we use meta-rules in form of a dependency graph to assign a credit to each potential literal. These meta-rules, given by the expert, are used to separate weak extensions from strong extensions.
2. To resolve the problem of space complexity, most of first inductive learners made use of hill climbing strategy at the expense of completeness. IFORD, instead uses a beam search strategy by keeping at each step all candidates specializations of RI value greater than a specified threshold.

3 Parallel IFORD

Many diverse techniques have been proposed and implemented for scaling up inductive algorithms and parallel DM is one of the most promising. The strategy of IFORD makes it a natural candidate for a parallel execution [3]. Our model is based on the simple observation that an object, having a well-defined interface and a hidden realization, resembles a server process that is handling requests from others processes. Parallel execution can be achieved by a set of objects, dispersed among several computers connected via network (see fig.1).

Objects in our system are autonomous and active entities, capable to carry out work upon request from the users, other objects, or on their own behalf [5]. They become active when they migrate dynamically to an idle machine to perform some work. Each entity of the system has a degree of power in function of its grade. The grade supervisor gives the power to delegate tasks to the object “worker” or to communicate with the user to better understand his recommendations and problems. The objects “worker” carry out the work assigned by their supervisor. The object worker decide when a migration can take place and where (which idle machine to use). The objects “counter” perform candidate rule evaluation by concurrently counting different rule covering.

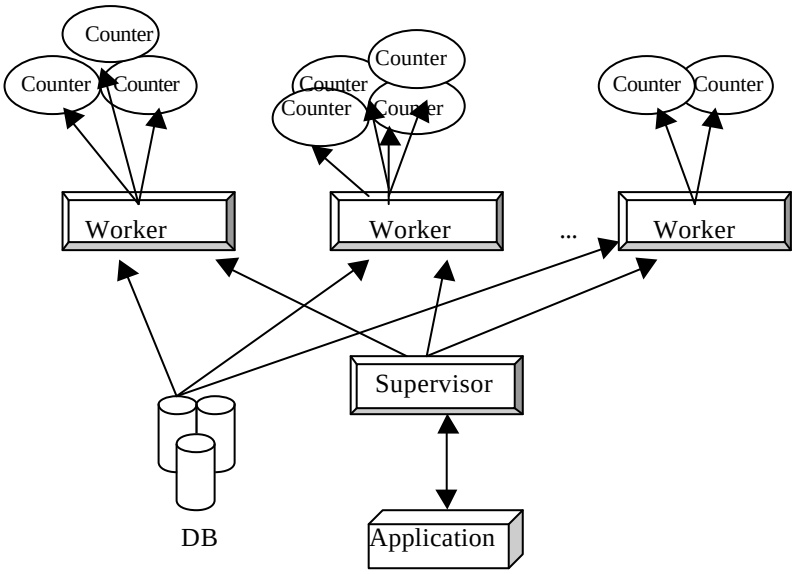

Fig. 1. Architecture of Parallel IFORD

4 Experimental results

To experiment our system, we used synthetic data of about 5000 tuples by relation. We mainly varied the number of background relations and their arity. Moreover, as the purpose of our experiments is not the leaning process but the behavior of the leaning algorithm, we fixed our experiments for learning one rule using mainly parallel counting.

The figure below shows the execution time required by a simulation of parallel IFORD for 6 pseudo applications. The experimentation was performed using 1, 2, 4, 8 and 12 processors, and shows a linear decrease of the execution time when increasing the number of processor.

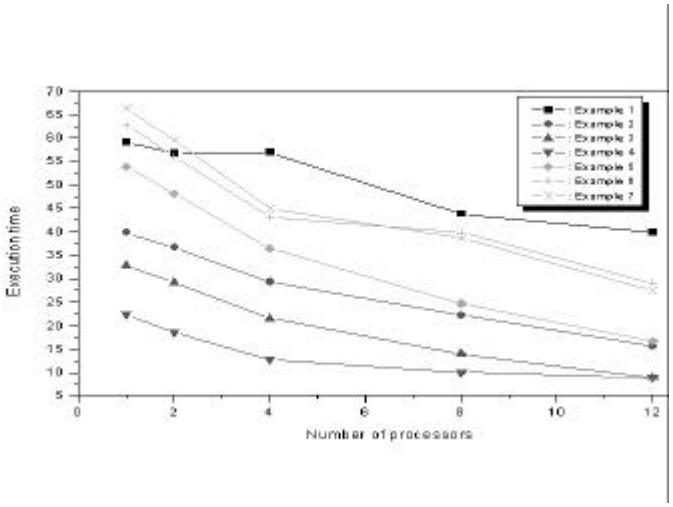

Fig. 1. Time to Learn on Synthetic Data Using Different Number of Processors.

5. Discussion

The work presented is in progress, and we will report more about it in the near future. As a first conclusion, we can say that the model is very promising but we have still to validate it using real workstation LAN first, then using direct access to DB, increasing the size of data significantly, and learning complete definition.

Bibliography

[1] U.M.Fayyad, G.Piatetsky-Shapiro, P.Smyth. From Data Mining to knowledge Discovery: An Overview. In [2], p1-34. 1996.

[2] U.M.Fayyad, G.Piatetsky-Shapiro, P.Smyth, R.Uthurusamy, editors. Advances in Knowledge Discovery and Data Mining. AAAI Press, Menlo Park, California, 1996.

[3] A.A.Freitas, S.H.Lavington: Mining very Large Databases with Parallel Processing. Kluwer Academic Publishers, Boston, 1998.

[4] T.M. Mitchell: Machine Learning. WCB/McGraw-Hill, 1998.

[5] O. Nierstrasz: A Language for Programming with Active Objects. Advances in Object-Oriented Software Engineering, Prentice-Hall, pp167-182, 1992.

[6] G. Piatetsky-Shapiro: Discovery, Analysis, and Presentation of Strong Rules. In [7], p229-248. 1991.

[7] G. Piatetsky-Shapiro, W.J.Frawley editors: Knowledge Discovery in Databases, AAAI Press, Menlo Park, California, 1991.

[8] J.R.Quilan: Learning Logical definitions from relations. Machine Learning,vol.5, num.3, 1990.