Integrating an Association Rule Mining Agent in an ERP System: A Proposal and a Computational Scalability Analysis

Rafael Marin Machado de Souza[1], Fabrício Gomes Vilasbôas[2], Pollyana Notargiacomo[1] and Leandro Nunes de Castro[2]

[1]Games, Learning, Simulation, Systems and Signals Laboratory (JAS3), Graduate Program in Electrical Engineering and Computing, Mackenzie Presbyterian University, Rua da Consolação 930, São Paulo, Brazil

[2]Natural Computing and Machine Learning Laboratory (LCoN), Graduate Program in Electrical Engineering and Computing, Mackenzie Presbyterian University, Rua da Consolação 930, São Paulo, Brazil

Keywords: Software Agents, Data Mining, Association Rule Mining, Apriori, Erp Systems, Computational Performance.

Abstract: Deployment flexibility, low development cost, and value-adding tools are some of the features that developers are looking for in ERP systems. Modularization through software agents is one way of achieving these objectives. In this sense, the present paper proposes the planning, implementation and integration of a software agent for association rule mining into an ERP system. The development and use of tools for all Knowledge Discovery in Databases (KDD) phases (pre-processing, data mining and post-processing), will be presented. This includes input data, file loading for the agent processing, use of the Apriori association rule mining algorithm, generation of output files with association rules, use of agent outputs for database storage and use of the stored data by the item recommendation tool. Experiments were carried out focusing the assessment of the running profile for databases of different sizes and using different computational architectures.

SCITEPRESS
SCIENCE AND TECHNOLOGY PUBLICATIONS

1 INTRODUCTION

The use of agents, more specifically intelligent agents, in the development of ERP systems has been discussed in the literature and the benefits of adopting this technology are demonstrated both in the business and developer sides (Botta-Genoulaz et al., 2005 Bih-Ru et al., 2006; Kishore et al., 2006). Benefits include lowering development costs, flexibility in business matching (made possible by weak coupling), increased use of ERPs, and the integration of intelligent decision-making processes into the system itself. These benefits bring a high success rate in deployments and make the systems accessible to a wide range of businesses (Yi and Lai, 2008; Al-Mudimigh and Saleem, 2008; Al-Mudimigh et al., 2009; Botta-Genoulaz et al., 2005).

As described by Russell and Norvig (2010), software agents or software robots (softBots) are programs capable of interacting with the proposed environment using sensors to gather information and return their processing by means of actuators, according to previously specified performance measures.

This paper proposes the development of a data mining software agent (Lea, Gupta and Yu, 2005; Papazoglou, 2001; Fox, Barbuceanu and Teigen, 2000), called RecBot, to perform association rule mining using the Apriori algorithm (de Castro and Ferrari, 2016, Kotsiantis and Kanellopoulos, 2006, Kantardzic, 2003, Agrawal, Imielinski and Swami, 1993). Association rules allow the agents to create intelligent recommendations for products and services within the ERP itself, facilitating the decision-making process of its users, increasing the conversion rate and maximizing results. To illustrate this development, the proposed agent will be integrated into a real ERP and applied to a database of business transactions in the health sector. Furthermore, a study on the computational performance of RecBot will be made to investigate how this application behaves i) as a function of the growth of the transactional database, and ii) when run in different platforms.

Marin Machado de Souza, R., Vilasbôas, F., Notargiacomo, P. and Nunes de Castro, L.
Integrating an Association Rule Mining Agent in an ERP System: A Proposal and a Computational Scalability Analysis.
DOI: 10.5220/0007483307780786
In *Proceedings of the 11th International Conference on Agents and Artificial Intelligence (ICAART 2019)*, pages 778-786
ISBN: 978-989-758-350-6
Copyright © 2019 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

The paper is organized as follows. In Section II a review of association rule mining is made and the proposed RecBot agent will be presented. Section III presents the materials and methods for evaluating the proposal and Section IV brings an analysis of the execution profile. The work is concluded in Section V with a general discussion on the proposal and future work perspectives.

2 ASSOCIATION RULE MINING AND THE RECBOT AGENT

There are several practical applications in which the objective is to find relationships among attributes (or variables), not objects. The association analysis, also known as association rule mining, corresponds to the discovery of association rules that present attribute values that occur concomitantly in a database (Agrawal, Imielinski and Swami, 1993; de Castro and Ferrari, 2016; Han, Kamber, and Pei, 2012). This type of analysis is typically used in marketing actions and for the study of transactional databases. There are two central aspects in the mining of association rules: the efficient construction of association rules and the quantification of the significance of the proposed rules. That is, a good association rule mining algorithm needs to be able to propose associations of items that are statistically relevant to the universe represented by the database. More formally, association rules have the form X → Y:

A_1 and A_2 and ... and $A_m \rightarrow B_1$ and B_2 and ... and B_n,

where A_i, $i = 1, ..., m$, and B_j, $j = 1, ..., n$, are pairs of attribute values.

The X → Y association rules are interpreted as follows: database records that satisfy the condition in X also satisfy the condition in Y.

The significance of the proposed rules is established on the basis of statistical arguments. Rules that involve mutually exclusive items or that cover a very small number of transactions are of little relevance. Thus, it is possible to objectively propose measures of interest that evaluate such features of the rules, such as support and trust (Agrawal, Imielinski and Swami, 1993).

The *support*, or *coverage*, of a rule is an important measure, since rules with very low support values occur only occasionally. Rules with low support are also of little interest from the business perspective, since it does not make much sense to promote items that customers buy little together. For this reason, support is typically used to eliminate uninteresting rules.

The support of an association rule, A → C, indicates the frequency of occurrence of the rule, that is, the probability of this rule being found in the total set of transactions of the base:

$$Sup(A \rightarrow C) = P(A \cup C) = \frac{\sigma(A \cup C)}{n} \qquad (1)$$

where $\sigma(A \cup C)$ is the rule support count, which corresponds to the number of transactions that contain a particular set of items, and n is the total number of transactions in the base.

Mathematically the support count of a set of items A is given by:

$$\sigma(\mathrm{A}) = |\{t_i \mid A \subseteq t_i, t_i \in T\}|$$

The *confidence*, or *accuracy*, verifies the occurrence of the consequent part of the rule in relation to the antecedent:

$$Conf(A \rightarrow C) = P(C \mid A) = \frac{\sigma(A \cup C)}{\sigma(A)} \qquad (2)$$

where $\sigma(A)$ is the support count of the antecedent.

While confidence is a measure of the rule’s accuracy, support corresponds to its statistical significance. Together, these are the most commonly used measures of interest in the association rule mining literature (Al-Mudimigh and Saleem, 2008; Al-Mudimigh, Saleem and Ullah, 2009; Han, Kamber, and Pei, 2012; de Castro and Ferrari, 2016). During the association rule mining process, criteria based on minimum values of support and confidence are established so that a rule is part of the final set of rules. However, many potentially interesting rules can be eliminated by a minimum support criterion, just as confidence is a measure that ignores the support of the set of items.

One way to reduce the computational cost of association rule mining algorithms is to decouple the support and confidence requirements from the rules. Because rule support only depends on the item set, infrequent item sets can be deleted early in the process without having to calculate their confidence. Thus, a common strategy adopted by association rule mining algorithms is to decompose the problem into two subtasks: generation of the frequent itemset; and rule generation.

Table 1: PEAS (Performance, Environment, Actuators, Sensors) description of RecBot.

Agent Type	**Predictive Performance**	**Computational Performance**	**Environment**	**Actuators**	**Sensors**
Association rule mining	Know and attend the customers needs, increase sales and maximize return	Computational performance in different platforms	Sales, customers and salesmen	Output files	Input files

The work of Agrawal (1993), and Agrawal and Srikant (1994) were pioneers in proposing an association rule mining algorithm, named Apriori, whose objective is to discover product associations in large transactional databases. The Apriori algorithm is the best-known method for association rule mining and employs depth-first search (Russell and Norvig, 2010) to generate candidate itemsets of *k* elements from sets of items with *k*–1 elements. The non-frequent candidate items are deleted, and the entire database is tracked and the frequent itemsets obtained from the candidate itemsets. This paper uses the Apriori algorithm to generate the RecBot rules.

Table 1 presents the PEAS (Performance, Environment, Actuators, Sensors) description of the implemented RecBot agent.

Each KDD task, or even its sub-tasks, can be performed by agents and in the most varied forms of interaction, ranging from competition - where each agent competes for information and the winner forces the loser to become symbiotic - to mutual cooperation, which leads to the evolution of all agents of the system (Steels 1998, Eguchi, Hirasawa, Hu and Ota, 2006).

The integration of the proposed model occurs in a cooperative way, using sales data preprocessed by the ERP system and in specific format as the agent data entry. The output of the agent processing also occurs in a specific format presented in the following sections, and all post-processing (such as persistence and use of association rules) will be performed again by the ERP system (Fig. 1).

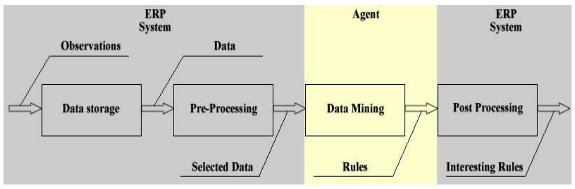


Figure 1: KDD responsibilities. Adapted from (Liu and Motoda, 2002).

3 ERP RECBOT INTEGRATION

To illustrate the KDD and its use in business, as also described by Bendoly (2003), a data preprocessing interface was initially developed. This step can also be performed by an agent, but here it was done by the ERP system itself, which is responsible for making the RecBot agent data file available and running it. The agent developed here has the task of receiving sales data previously processed by the ERP system and mining the association rules, allowing the recommendation of items.

RecBot was developed using the Object Pascal language (Lazarus/Delphi), aiming to run on Linux platforms. The development of this agent took approximately 26 days, since it was necessary to develop all the methods that involve data processing.

To reduce the development and total execution time, a version of this agent was developed in C++ with support from the Data Analytics Acceleration Library (DAAL), which offers computationally optimized methods for all stages of the KDD process. Its implementation, maintenance and optimization is done by Intel® and can be purchased together with Intel Parallel Studio or in a free version available on Github. With the support of this library it was possible to reduce the development time, which previously was 26 days, to only 3 days. The results generated by these implementations are presented and discussed in Section IV. The input files and outputs, such as item groups, association rules, and runtimes, for parameterizations used in both implementations of the same agent, are available for online access in the RecBot Project within the Mendeley Data platform, in directories Inputs and Outputs, respectively (Souza, 2018).

As this is a software agent, the communication pattern between agents has been defined through execution parameters that must meet the following sequence: Number of Load Threads, Confidence Factor, Support Factor, File with input data and folder for output files. An example of an agent call is shown below:

```
./recbot 10 0.03 0.3 /data/inputs/db3k.csv/data/outputs/
```

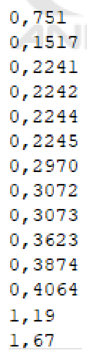
SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

In this example, 10 threads are defined for input file processing, a confidence factor of 3%, a support factor of 30%, the input file is db3k.csv, which is in the /data/inputs/directory, and the directory to store the output files is /data/outputs/.

There is a peculiarity in the agent developed in the C++ language. The DAAL library dynamically defines the number of threads that will be used during the agent processing, so the first parameter is omitted for this version to run.

The algorithm performs a combinatorial analysis of the items to construct the rules, counts the transactions containing the items in order to determine the support (frequency of occurrence) and confidence (accuracy) of the rules. The aim is to find rules that satisfy minimum values of support and confidence, pruning all candidate rules that have not reached the pre-defined minima (Agrawal, Imielinski and Swami, 1993; Nath, Bhattacharyya1 and Gosh, 2012; de Castro and Ferrari, 2016).

An interface for transactional data processing was developed in the ERP to select the filtered transactions and allow the export of the file used for communication with the agent. The data entry files must be in the Comma-Separated Values (CSV) standard and contain the preprocessed transactional data in the "id, item" format, where id is a sequential code used to represent the transaction and item is the transaction, as illustrated in Figure 2.



```
0,751
0,1517
0,2241
0,2242
0,2244
0,2245
0,2970
0,3072
0,3073
0,3623
0,3874
0,4064
1,19
1,67
```

Figure 2: Partial Input file structure.

The agent starts by loading the minimum support and confidence values chosen for the execution and loading the input data file. From this a matrix composed of *N* transactions is fed, indicating 1 when the item is present in the transaction, and 0 otherwise. Thus, the frequency of occurrence of each item is calculated and those that meet the minimum support are evaluated in pairs, until there are no possibilities of combinations that meet the minimum support (de Castro and Ferrari, 2016; Kantardzic, 2003). The support calculation is done by Eq. (1). All subsets with support less than the minimum support are excluded (pruned) from the next step, and so the algorithm continues until there are no more possible combinations.

With the subsets that meet the minimum support properly selected, the next step is to use this result to build the rules $a \rightarrow c$ (if a, then c), and these new combinations must meet the minimum confidence (Eq. (2)). Rules that do not meet the minimum confidence will be pruned and the process is repeated until all rules have been evaluated. The complete process is shown in Figure 3.

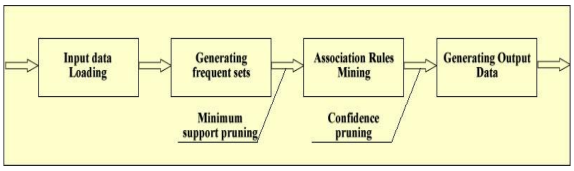


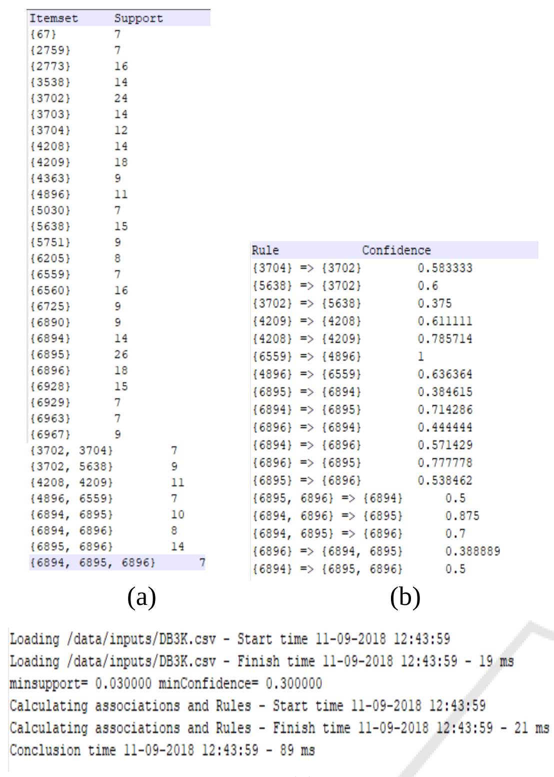
Figure 3: How the agent works.

Thus, three outputs are created by the agent: 1) one with the subsets that meet the minimum support and its respective support value (outpuItemSet file), as shown in Figure 4; 2) another file with the association rules that meet the minimum confidence (outputRules file); and 3) one with the running time of each agent process, as described in Figure 4.

The association rules are loaded by the ERP system and stored in a database for later use. Determining the most flexible format for recording these data in a relational database has been one of the challenges of this work, because it was necessary to consider various combinations of items made in the sale (antecedents), and to use SQL queries to find the items to recommend (consequents). As a proposed solution, two tables were created, one to record the antecedent items and their confidence values and another to record the consequent items. Each rule is assigned an ID.

Thus, to select the most relevant rules, a query is made with the existing items of the current transaction and all rules that have existing combinations are selected. The consequent items of these rules that do not exist in the current transaction are selected and presented to the user as recommendations.

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS



```
Itemset        Support
{67}           7
{2759}         7
{2773}         16
{3538}         14
{3702}         24
{3703}         14
{3704}         12
{4208}         14
{4209}         18
{4363}         9
{4896}         11
{5030}         7
{5638}         15
{5751}         9
{6205}         8
{6559}         7
{6560}         16
{6725}         9
{6890}         9
{6894}         14
{6895}         26
{6896}         18
{6928}         15
{6929}         7
{6963}         7
{6967}         9
{3702, 3704}        7
{3702, 5638}        9
{4208, 4209}        11
{4896, 6559}        7
{6894, 6895}        10
{6894, 6896}        8
{6895, 6896}        14
{6894, 6895, 6896}      7
```

(a)

```
Rule                 Confidence
{3704} => {3702}          0.583333
{5638} => {3702}          0.6
{3702} => {5638}          0.375
{4209} => {4208}          0.611111
{4208} => {4209}          0.785714
{6559} => {4896}          1
{4896} => {6559}          0.636364
{6895} => {6894}          0.384615
{6894} => {6895}          0.714286
{6896} => {6894}          0.444444
{6894} => {6896}          0.571429
{6896} => {6895}          0.777778
{6895} => {6896}          0.538462
{6895, 6896} => {6894}      0.5
{6894, 6896} => {6895}      0.875
{6894, 6895} => {6896}      0.7
{6896} => {6894, 6895}      0.388889
{6894} => {6895, 6896}      0.5
```

(b)

```
Loading /data/inputs/DB3K.csv - Start time 11-09-2018 12:43:59
Loading /data/inputs/DB3K.csv - Finish time 11-09-2018 12:43:59 - 19 ms
minsupport= 0.030000 minConfidence= 0.300000
Calculating associations and Rules - Start time 11-09-2018 12:43:59
Calculating associations and Rules - Finish time 11-09-2018 12:43:59 - 21 ms
Conclusion time 11-09-2018 12:43:59 - 89 ms
```

(c)

Figure 4: (a) Frequent itemset. (b) Association rules. (c) Running time.

4 PERFORMANCE EVALUATION

This section presents an analysis of the execution profile of the Apriori algorithm implemented in two versions: 1) one in Object Pascal (Lazarus/Delphi); and 2) another in C++ with DAAL library support, as described previously.

Two computer systems of shared memory architecture and with the support of SIMD (*Single Instruction Multiple Data*) type flows were used. The first computer system consists of two 1.83 MHz Dual Core Xeon processors and 12 GB of RAM, two SATA 2.0TB and Linux Ubuntu Server 18.04 LTS. The second computer system consists of two Intel Xeon Platinum 8160 @ 2.10 GHz processors, each with 24 physical cores (48 logical) and 33 MB cache memory, 190 GB RAM, two Intel S3520 Series 1.2Gb and 240TB SSDs GB capacity and CentOS 7 operating system with kernel version 3.10.0-693.21.1.3l7.x86_64. For reasons of ease of identification, the first computer system was named Woodcrest and the second one Skylake.

Therefore, to improve the use of the computational systems available for our experiments the algorithm was transcribed from the Object Pascal language into the C++ language using the DAAL library methods. The two computer systems have microarchitecture processing units developed by Intel that have made efforts to apply computational optimization and parallelism techniques to both the C++ language compiler and DAAL, both available in the Intel Parallel Studio XE (Intel, 2018a). In the shared memory systems, DAAL supports thread-level parallelism using Threading Building Blocks (Intel, 2018b), which is a runtime-based parallel programming model for C/C++ code, and supports vectorization, which is the programming model for the SIMD architecture. Thus, there is the guarantee of using all resources available by the system.

Table 2: Number of itemsets and rules in RecBot.

Dataset size ×10^3	**Support 2% Confidence 20%**		**Support 3% Confidence 30%**	
	Item sets	**Rules**	**Item sets**	**Rules**
3	178	91	70	18
6	142	38	55	9
12	150	41	60	10
24	139	45	61	10
48	126	36	56	7
96	127	37	55	10
192	124	37	54	8
384	118	30	53	6
768	111	26	49	6
1536	121	30	55	7
3072	131	39	61	8

Samples of varying sizes from the same real-world database were used for the experiments. The initial sample has a size of 3,000 (3K) records, and it doubles up to 3,072,000 records (Table 2). We also tested the following minimum support (minsup) and minimum confidence (minconf) values: minsup = 2%, minconf = 20%; and minsup = 3%, minconf = 30%. As previously explained, the higher the support and confidence, the greater the correlation between the items. Their configuration depends on business demand of data, since the higher the value of these parameters, the less itemsets will be generated and, therefore, the less items will be recommended to the user application.

Table 2 shows the number of items in the frequent itemset and the number of rules generated for each dataset size. It can be noted that increasing the database does not imply increasing the number of frequent items or the number of rules. While this may seem counterintuitive, the explanation lies in

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

Table 3: Performance evaluation of the RecBot agent implementations in the Woodcrest and Skylake architectures for exponentially growing databases.

Dataset size ×10^3	**Server**	**C++**		**Object Pascal**	
		***minsup* 2% *minconf* 20%**	***minsup* 3% *minconf* 30%**	***minsup* 2% *minconf* 20%**	***minsup* 3% *minconf* 30%**
		ET	ET	ET	ET
3	**Woodcrest**	112	89	200	127
	Skylake	45	31	57	31
6	**Woodcrest**	121	99	376	244
	Skylake	47	35	114	67
12	**Woodcrest**	160	119	938	425
	Skylake	51	37	242	133
24	**Woodcrest**	182	155	2040	957
	Skylake	53	39	516	323
48	**Woodcrest**	244	226	4465	2319
	Skylake	59	48	1140	762
96	**Woodcrest**	420	370	9670	4600
	Skylake	71	64	3246	1889
192	**Woodcrest**	729	661	31615	13355
	Skylake	100	91	8110	4835
384	**Woodcrest**	1345	1270	110160	70671
	Skylake	150	141	17933	10721
768	**Woodcrest**	2540	2421	225943	154344
	Skylake	254	236	36104	21856
1536	**Woodcrest**	5032	4768	459691	309819
	Skylake	461	432	80123	47075
3072	**Woodcrest**	10122	9583	-	-
	Skylake	826	815	237129	191673

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

the fact that the increase in the database also increases the denominator of the coverage calculation and the support of a rule, causing fewer frequent items to meet the minimum support criterion and fewer rules satisfying the minimum confidence level.

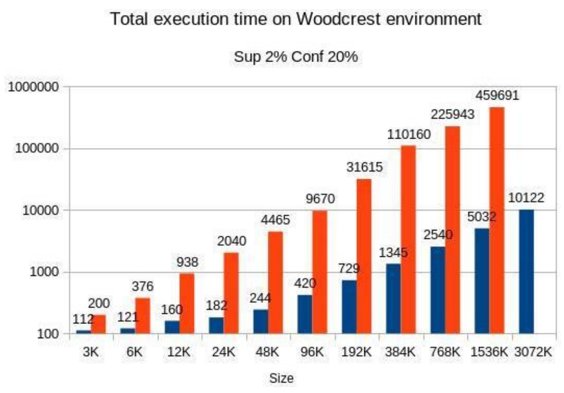
As the Apriori algorithm is deterministic, a single experiment was performed for each configuration and the results presented in Table 3 are the total Execution Time (ET) of the two implementations, for the different combinations of minsup and minconf, and increasing dataset sizes.

Figure 5 shows that the C++ version provided a 98.46% reduction in execution time when compared with the Object Pascal implementation. It can also be observed that for the 3072K dataset the Object Pascal version could not converge, because there was not enough RAM space to store the data structures and the program was shut down by the operating system. By contrast, the C++ version finished execution and the running time still complies with the previously observed pattern.

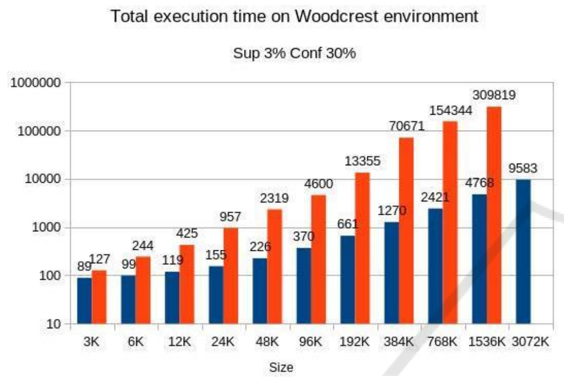
Figure 5 plots the execution time (ET) of the Woodcrest architecture for both implementations: Object Pascal and C++. It is noted that the ET increase is exponential, like the dataset increase, but the increase for Object Pascal occurs at a significantly higher rate than the increase for C++.

A similar behavior can be observed in Figure 6 for the Skylake architecture. The difference between the execution time of the C++ version in relation to the Object Pascal version comes to be 190858ms faster for largest dataset (3072K objects), a 99.57% reduction in execution time. The average growth rate of the C++ runtime is approximately 1.37 and for the Object Pascal version it is approximately 2.32. This shows that by doubling the dataset size, the Object Pascal time more than doubles, while the C++ version has an increase of approximately 44%.

Figure 7 summarizes the gain in performance when comparing both implementations in both architectures. It can be observed that the C++ version had a gain of 286.94 times to *minsup* = 2% and *minconf* = 20% and a gain of 235.18 times to *minsup* = 3% and *minconf* = 30% in the Skylake for

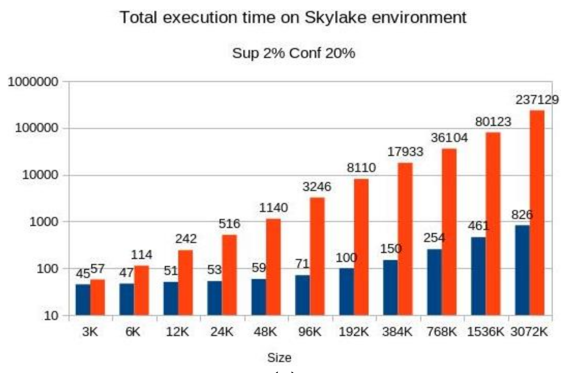


(a)

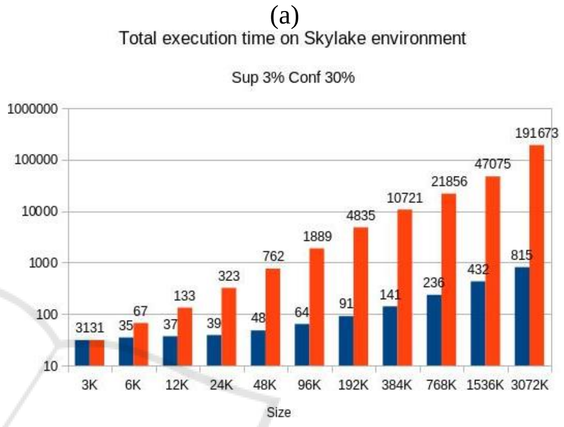


(b)

Figure 5: Growth rate comparison between each implementation in the Woodcrest archtiecture. First Column of dataset: C++; Second Column of dataset: Object Pascal. (a) *minsup* = 2% and *minconf* = 20%; (b) *minsup* = 3% and *minconf* = 30%.



(a)



(b)

Figure 6: Growth rate comparison between each implementation in the Skylake archtiecture. First Column of dataset: C++; Second Column of dataset: Object Pascal. (a) *minsup* = 2% and *minconf* = 20%; (b) *minsup* = 3% and *minconf* = 30%.

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

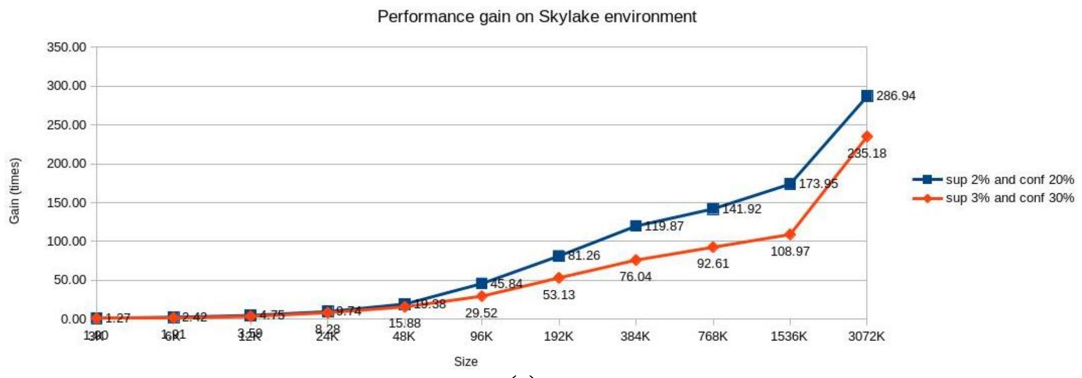
the dataset with 3072K objects. The gain is obtained from the ratio between the execution time of the Object Pascal version and the execution time of the C++ version. By analyzing the pattern of gain growth, it can be observed that it is very close to stabilizing in the Woodcrest environment, whilst in the Skylake environment the gain still seems to be growing. This reinforces the conclusion that the C++ version is best suited for database scalability. This result also demonstrates the efficiency of the Skylake architecture since the operations performed by the algorithm are the same in all environments.

5 CONCLUSIONS

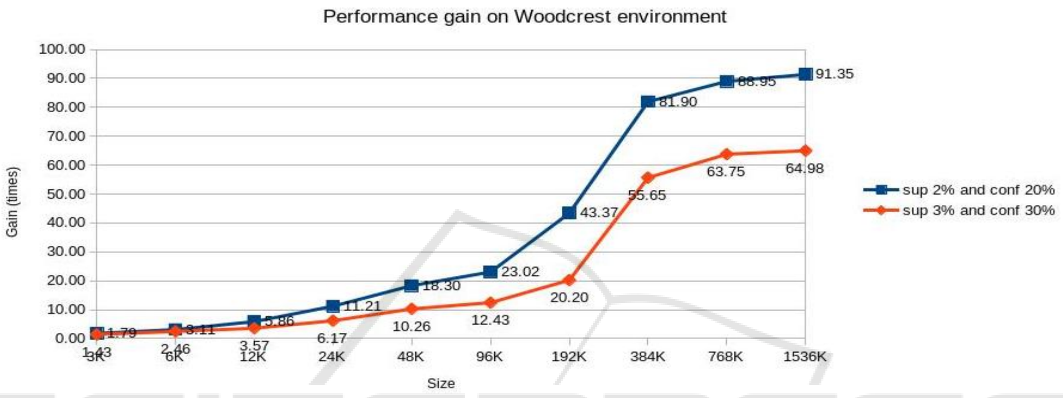
This paper introduced an association rule mining agent to be integrated with an ERP system as an alternative to implement a microservice structure. For this, the agent was developed in two different languages, Object Pascal (Delphi/Lazarus) and C++, making use of different programming techniques. In the first implementation (Object Pascal) the Apriori algorithm was fully coded using object orientation, which led to a programming time 8 times greater than the second one (C++), which was done using third party libraries (DAAL Intel). The use of Intel DAAL also provided considerable performance gains.

The performance of these agent implementations was compared in two distinct computational architectures: Woodcrest and Skylake. In the performance analysis presented, a real transactional database was used with 3,072,000 objects, presented in increasing subsets from 3,000 objects, doubling at each experiment. Two distinct configurations of minimum confidence and support thresholds were also tested, allowing empirically investigating the application scaling with the dataset size.

In the time analysis, it was obtained a reduction in the total execution time of 99.57% in the Skylake and 98.46% in the Woodcrest for the database with



(a)



(b)

Figure 7: Reduction in times of the execution time of the C++ version in relation to the Object Pascal in the (a) Skylake and the (b) Woodcrest architecture.

3072K objects. It has also been observed that the C++ version is more suited to dataset scalability, presenting a runtime increase rate of approximately 1.37 times as opposed to 2.32 times of the Object Pascal version.

With this, the efficiency of the presented technique is determined, since its flexibility of languages and environments demonstrated the viability of the solution for different operating platforms and computational architectures.

As further works it is proposed the use of the implemented agent in real-world ERP systems and the assessment of its business improvement, such as the number of sold items and billing volume (considering seasonalities and possible outliers).

ACKNOWLEDGEMENTS

The authors thank CAPES, CNPq, Fapesp, and Mackpesquisa for the financial support. The authors also acknowledge the support of Intel for the Natural Computing and Machine Learning Laboratory as an Intel Center of Excellence in Artificial Intelligence.

REFERENCES

Agrawal, R., Imielinski, T. and Swami, A. 1993. Mining Association Rules between Sets of Items in Large Databases. In: *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, Volume 22 Issue 2, pp 207-216.

Agrawal, R., Srikant, R. 1994. Fast Algorithms for Mining Association Rules in Large Databases. *Proceedings of the 20th International Conference on Very Large Data Bases* (VLDB), p. 487-499.

Al-Mudimigh, A. S. and Saleem, F. 2008. A Framework of an Automated Data Mining Systems Using ERP Model. *International Journal of Computer and Electrical Engineering*.

Al-Mudimigh, A. S., Saleem, F. and Ullah, Z. 2009. The effects of data mining in ERP-CRM model: a case study of MADAR. *W. Trans. on Comp*. v. 8, n. 5, p. 831-843.

B. Nath, D. K., Bhattacharyya1 and A. Ghosh. 2013. Incremental association rule mining: a survey. *WIREs*

DataMining and knowledge Discovery, Volume 3, Issue 3, pp 157–169.

Bendoly, E. 2003. Theory and support for process frameworks of knowledge discovery and data mining from ERP systems. *Information & Management*, v. 40, pp. 639–647, 2003.

Bih-Ru, L., Gupta, M. C., Wen-Bin, Y. 2005. A prototype multi-agent ERP system: an integrated architecture and a conceptual framework. *Technovation*, 25(4), pp. 433-441.

Botta-Genoulaz V., Millet, P.-A. and Grabot, B. 2005. A survey on the recent research literature on ERP systems. *Computers in Industry*, v. 56, n. 6, pp. 510-522, 2005. ISSN 0166-3615, https://doi.org/10.1016/j.compind.2005.02.004

Castro, L. N. 2016. *Introdução a Mineração de dados: Conceitos, Algoritmos e aplicações*. São Paulo: Editora Saraiva.

Eguchi, T., Hirasawa, K., Hu, J. and Ota, N. 2006. A Study of Evolutionary Multiagent Models Based on Symbiosis. *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS*, Vol. 36, n. 1.

Flynn, M. J. 1972. Some computer organizations and their effectiveness. *IEEE transactions on computers*, IEEE, v. 100, n. 9, p. 948–960.

Fox, M., Barbuceanu, M. and Teigen, R. 2000. Agent-oriented supply-chain management. *International Journal of Flexible Manufacturing Systems*, v. 12, p. 165–188.

Han, J., Kamber, M. and Pei, J. 2012. *Data mining: concepts and techniques*. Amsterdam: Morgan Kaufmann.

Intel Corp. 2017. DAAL - Data Analytics Acceleration Library. Available: https://software.intel.com/pt-br/intel-daal

Intel Corp. 2018a. Intel® Parallel Studio XEAvailable: https://software.intel.com/en-us/parallel-studio-xe

Intel Corp. 2018b. Intel® Threading Building Blocks (Intel® TBB) Available: https://software.intel.com/en-us/intel-tbb

J. Yi, J. and Lai, C. 2008. Research on Reengineering of ERP System Based on Data Mining and MAS.In: *International Symposium on Knowledge Acquisition and Modeling*, IEEE Computer Society, pp. 180–184.

Kantardzic, M. 2003. *Data Mining: Concepts, models, methods and algorithms*. Piscataway: IEEE Press.

Kishore, R., Zhang, H., Ramesh, R. 2006, Enterprise integration using the agent paradigm: foundations of multi-agent-based integrative business information systems. *Decision Support Systems*, 42(1), pp. 48-78.

Kotsiantis, S. and Kanellopoulos, D. 2006. Association Rules Mining: A Recent Overview. *GESTS International Transactions on Computer Science and Engineering*, v.32, n. 1, p. 71-82.

Liu, H. and Motoda, H. 2000. *Feature Selection for Knowledge Discovery and Data Mining*, Kluwer Academic Publishers.

Lea, B-R. and Gupta, M. C. and Yu, W-B. 2005. A prototype multi-agent ERP system: an integrated architecture and a conceptual framework. *Technovation*. v. 25, n. 4, p. 433-441.

Papazoglou, M. 2001. Agent-oriented technology in support of e-business. *Communications of the ACM*, v.44, n. 4, p. 71–77.

Rajaraman, V. and Siva, R. M. C. 2016. *Parallel Computers Architecture and Programming*. PHI Learning Pvt. Ltd.

Russell, S. and Norvig, P. 2010. *Artificial Intelligence A Modern Approach*, 3nd ed. New York, NY, USA: Prentice Hall.

Steels, L. 1998. The Origins of Ontologies and Communication Conventions in Multi-Agent Systems. In: *Autonomous Agents and Multi-Agent Systems*, Vol. 1, Issue 2, pp 169–194.

Souza, Rafael Marin Machado de. 2018. *RecBot Inputs and Outputs*. Mendeley Data.

Tanenbaum, A. S. 2007. *Organização e Estrutura de Computadores*. 5. ed. [S.l.]: Pearson Prentice Hall.

SCITEPRESS
SCIENCE AND TECHNOLOGY PUBLICATIONS