Analysing Collaborative Filtering algorithms in a Multi-Agent Environment

Tiago Cunha, Rosaldo J. F. Rossetti, Carlos Soares

Faculdade de Engenharia da Universidade do Porto
email: tiagodscunha, rossetti, csoares@fe.up.pt

KEYWORDS
Data Mining, Machine Learning, Recommender Systems, Social Simulation, Agents

ABSTRACT

The huge amount of online information deprives the user to keep up with his/hers interests and preferences. Recommender Systems appeared to solve this problem, by employing social behavioural paradigms in order to recommend potentially interesting items to users. Among the several kinds of Recommender Systems, one of the most mature and most used in real world applications are known as Collaborative Filtering. These methods recommend items based on the preferences of similar users, using only a user-item rating matrix. In this paper we explain a methodology to use Multi-Agent based simulation to study the evolution of the data rating matrix and its effect on the performance of several Collaborative Filtering algorithms. Our results show that the best performing methods are user-based and item-based Collaborative Filtering and that the average algorithm performance are surprisingly constant for different rating schemes.

Introduction

The Internet has become one of the most important tools for the XXI century, allowing the access to a high volume of information, which would be difficult or even impossible to achieve in any other ways. Businesses also felt the need to adapt to this paradigm and increasingly more companies have an online business position. However, the ability for the user to access a lot of information made difficult for each online business to meet the user's expectations and demands, due to the sheer quantity of purchasable items. Therefore, it was a necessity to filter the user's preferences among the vast choice of items for sale and recommend these in a easier and controlled fashion.

To solve this task, a new research field appeared: Recommender Systems. A Recommender System attempts to solve the problem of information overload and recommends potentially interesting items to users (Yang et al. 2014, Bobadilla et al. 2011). These systems are inspired by human social behaviour, where it is common to take into account the tastes, opinions and experiences of our acquaintances when making all kinds of decisions (Bobadilla et al. 2013). Among the several strategies available, Collaborative Filtering appears as the most mature and most used in real world applications.

Agent-based Modeling and Simulation (ABMS) is a technique that models a world environment and includes autonomous interacting agents in order to study a specific phenomenon (Al-Sharawneh and Williams 2009). This type of simulation aims to solve the problems that traditional simulation techniques cannot solve. Therefore, there are applications for several areas, including business, commerce, economy and biology.

In this work we will join the areas of Recommender Systems and ABMS, by evaluating several Collaborative Filtering algorithms using a simulation basis. The simulation aims to reproduce the rating process of users to different items in a website and evaluate it continuously using several Recommender Systems evaluation metrics. The goal for this work is to study the evolution of different Collaborative Filtering algorithms in a evolving rating matrix with different rating schemes. We would like to assess the influence of each rating strategy on the algorithm performance in order to understand how it evolves over time.

This paper is organized as follows: Section focus on Collaborative Filtering methods, integrations between Data Mining techniques and ABMS and related work in using ABMS on Recommender Systems. In the Section the developed methodology is explained in detail, while Section presents the main results and discussion for our approach. Finally, Section states the final conclusions of this work and task for future work.

Related Work

As we have previously seen, one of the most mature and used strategy in Recommender Systems is known as Collaborative filtering recommendations. These are based on the premise that a user must like the favorited items of a similar user. It uses the feedback from each individual user to recommended items among similar users (Yang et al. 2014). In this case, Adomavicius et al. (Adomavicius and Tuzhilin 2005) state that the utility $u(c, s)$ of item s for user c is estimated based on the utilities $u(c_j, s)$ assigned to item s by those users $c_j \in C$

that are similar to user c.
Collaborative Filtering strategies can also be divided in memory-based and model-based (Bobadilla et al. 2013, Yang et al. 2014, Lü et al. 2012). Memory-based methods act only on the matrix of user ratings of items to execute recommendations, whereas model-based methods induce a model from such matrix and use this model recommend items. Traditional approaches use user-based nearest neighbour, item-based nearest neighbour and association rules to mine the recommendations.
Since Collaborative Filtering is the most used Recommender System strategy used in real-world applications, there is a large number of works published in the literature (Jiang et al. 2011) and it is very difficult to keep track of all of them. Some notable publications can be found in (Hu et al. 2008, Zheng et al. 2009, Rong et al. 2009).
Evaluating Recommender Systems can be seen as a generic data mining evaluation problem. This means that we must split the data into training and testing datasets, using strategies such as hold-out, leave-one-out or k-fold cross-validation. The main difference is that the evaluation metrics must suit the problem at hand. In order to evaluate Recommender Systems, several metrics are proposed (Bobadilla et al. 2013, Lü et al. 2012, Yang et al. 2014). The most used evaluation metrics to asses the rating accuracy are MAE (Mean Absolute Error), RMSE (Root of Mean Square Error) and NMAE (Normalized Mean Average Error). Lu et al. (Lü et al. 2012) provides an extensive list of evaluation metrics for Recommender Systems.
The integration between Data Mining and ABMS has been studied by several researchers. Baqueiro et al. (Baqueiro and Wang 2009) state that there are typically two approaches to achieve this symbiosis: either apply Data Mining techniques in ABMS research or using the ABMS results in a Data Mining research. The first approach is used mainly to provide statistical expertise into the verification and validation steps of the simulation. On the other hand, the second strategy enables to provide extra data for a mining process for situations where it is insufficient. It even enables to model the data properties in the data generation step, in order to reduce errors and missing values within.
Remondino et al. (Remondino and Correndo 2006) conceptualized the existence of two kinds of Data Mining and ABMS combinations: endogenous and exogenous. While endogenous modelling focus on providing an agent with intelligent behaviour mined from past simulation experiments, exogenous modelling involves analysing the results of a simulation experiment to extract interesting patterns that can improve the behavioural model of the entire system. A combination of both conceptualizations is available at (Arroyo et al. 2010).
There has been effort made into approaching these two areas, especially in the area of distributed Data Mining computing using agents (Kargupta et al. 1997; 1999, Albashiri et al. 2008, Mateo and Lee 2010). In the context of Recommender Systems, there are a few publications that use ABMS to study the performance of Collaborative Filtering algorithms.
Yamashita et al. (Yamashita et al. 2007) evaluate the effect of community characteristics on recommender system, using multi-agent based simulation. They model users preferences and items characteristics in a random vector fashion and apply a utility function to calculate whether the user may be interested in the item. The algorithms used are (1) random recommendation (for a baseline indicator), (2) recommend popular items and (3) a user-based nearest-neighbour collaborative filtering with Pearson's correlation. They conclude that if the number of ratings is low, then the popular items recommender performs best. On the other hand, if there are a lot of ratings, collaborative filtering methods are better. In our proposal, we increase the number of algorithms and rating schemes, but evaluate the influence of time rather than the number of ratings.
Saga et al. (Saga et al. 2011) proposed a a software simulator based on a small world network that allows the evaluation of algorithms for recommender systems. The agents implemented are users (who evaluate the items based on the agents' rating algorithm and the attributes of each item and agent), items (has attributes used in the recommendation), a recommender (recommends items to users based on the recommendation algorithm), a controller (handles the simulation flow) and a recorder (obtains the results of the rating and evaluation measurements for the recommendation and outputs the evaluation metrics). This simulator is able to: (1) build an evaluation environment for the recommender system, (2) enable the comparison of collaborative or content-based filtering algorithms and (3) output the results of evaluations in order to compare them. The evaluation metrics used were MAE, recall, precision, novelty, diversity, and discovery. The implementation of this proposal can be found in (Saga et al. 2013). The simulation recommends 5, 10 and 20 items to users and measures the preference of each user to a certain item. Their results show that their implementation is valid for the problem of Collaborative Filtering and that the best recommendation strategy is to recommend 20 items, since only in this situation the precision rises to acceptable values.
Our approach differs from the previous by not providing the recommender system as an agent, but by using a combination concept as the ones developed in (Remondino and Correndo 2006, Arroyo et al. 2010). We use off-the-shelve ABMS and Data Mining tools: NetLogo and R, respectively. This facilitates the reproduction of experiments by other researchers and since the tools have proven to be efficient for each research area, we are able to run experiments for a larger population size. However, we use fewer evaluation metrics

and our rating schemes are all random based strategies. Ultimately the difference is that our work is oriented to building and evaluating an evolving rating matrix, while their strategy is based on modelling users and algorithms as agents in a small world network model and evaluating the recommendation performance by assessing the feedback of each user to the recommended items.

Methodological Approach

We conceptualize the world of simulation as a 2D space within a website, with several agents randomly moving around. The agents act as users (who rate items) and items (that are rated). The rating step occurs depending on the proximity of both types of agents at each time tick. A limit is imposed to allow only one neighbour item to be rated per each user at each time tick. A user parameter *frequency* controls at which time interval the rating matrix is sent into R for processing and waits for the response, that provides the evaluation metrics for each algorithm. This process iterates until the time tick value reaches 1000, at which point the simulation stops. This closed-world methodology enables to evaluate the performance of several Recommender Systems algorithms on an evolving rating matrix, by using several rating schemes.

In order to evaluate Recommender Systems using ABMS, we use off-the-shelve tools for both research areas: R[1] and NetLogo[2]. The R package used for Recommender Systems is recommenderlab[3], which provides several Collaborative Filtering algorithms and already has implementations for validation and evaluation for this type of Data Mining algorithms. The algorithms chosen were POPULAR (recommends popular items), Item-based CF, User-based CF, SVD and RANDOM (baseline indicator of performance). The package also provides two typical Data Mining validation techniques, namely k-fold cross-validation and split validation and both are available in our simulation tool. Lastly, the evaluation metrics available in the package and used in this simulator are error metrics, namely MAE, MSE and RMSE. These metrics are gathered at each *frequency* interval and reported to the user through three graphics in the simulation tool. The interface RServe[4] was used to exchange data between R and NetLogo. Figure 1 shows the entire simulation tool, with the user parameter controls on the left, the simulation space in the middle and the graphics results on the right.

The simulation tool has several user-defined parameters, namely the size of users and items populations, which rating scheme must be used in the simulation instance, whether there is always rating in each encounter between user and item, which are the algorithms chosen

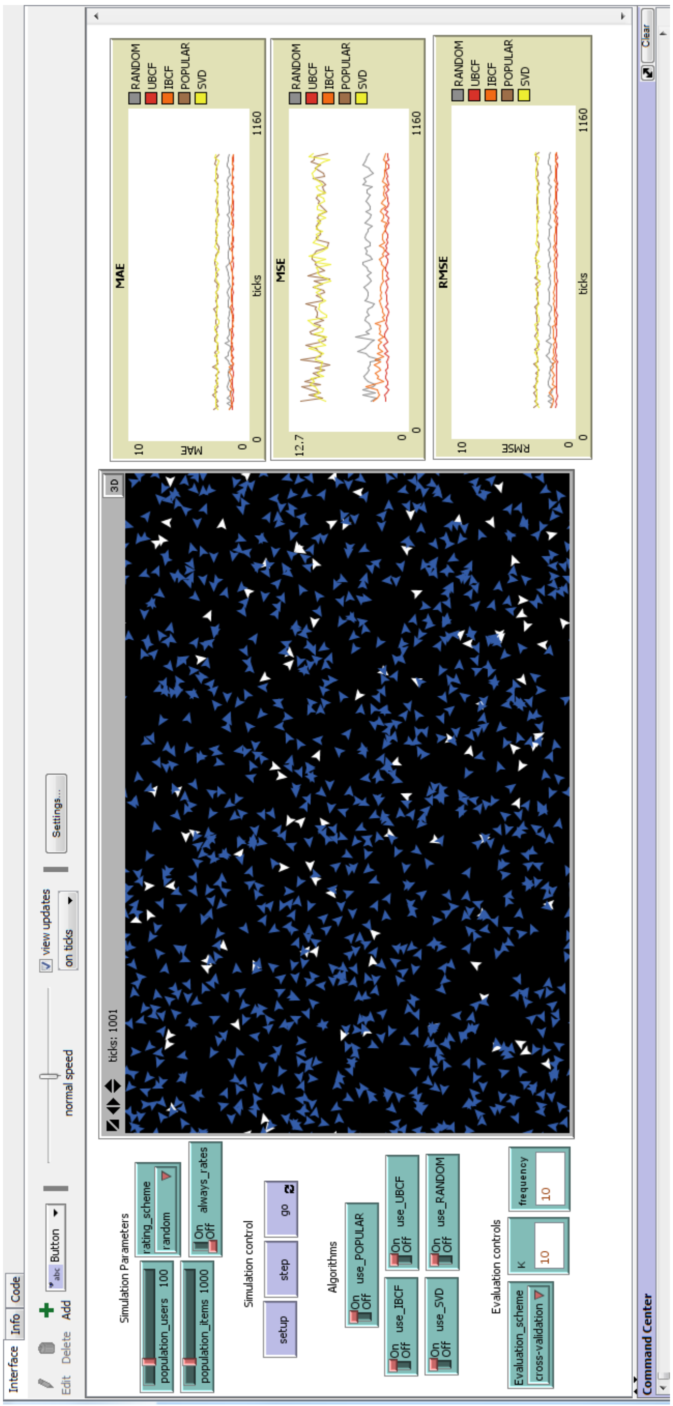

Figure 1: Simulation tool

[1]http://www.r-project.org/
[2]https://ccl.northwestern.edu/netlogo/
[3]http://cran.r-project.org/web/packages/recommenderlab/recommenderlab.pdf
[4]http://rserve-ext.sourceforge.net/

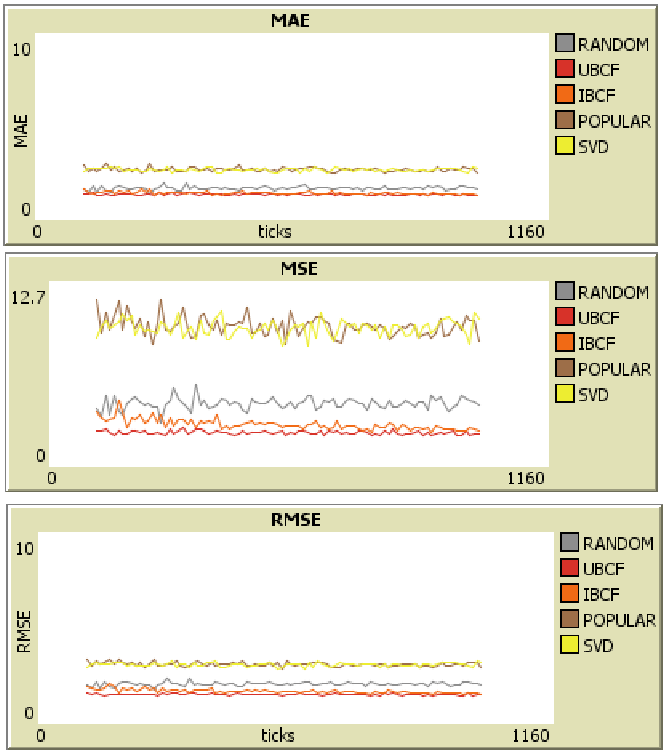

Figure 2: Experiment 1: Performance metrics for random scheme.

to be evaluated, the validation scheme and the already mentioned parameter *frequency*, that controls the data flow. The five rating schemes available are all random-based on a scale of 1 to 5, but with different approaches: the NetLogo random command and random approaches based on four probability distributions were used (normal, poisson, exponential and gamma). The parameter that controls whether a user always rated a neighbour item means to reproduce the well-known effect in Recommender Systems, which states that users do not rate all the items they encounter. When the simulator has this parameter at false, it chooses randomly when a user rates a neighbour item.

Preliminary Results and Analysis

Having the goal to evaluate Recommender System algorithms through simulation of an evolving rating matrix, we tested all kinds of rating schemes and compare the performance of all the algorithms. The results are presented in Figures 2 - 6. The simulation parameters were the same for every experiment, excluding the rating scheme: 100 users, 1000 items, 10-fold cross-validation and 10 tick interval between evaluations.

When we compare the results for the several experiments, it is visible that the algorithms SVD and POPULAR always have the worst performance, indepen-

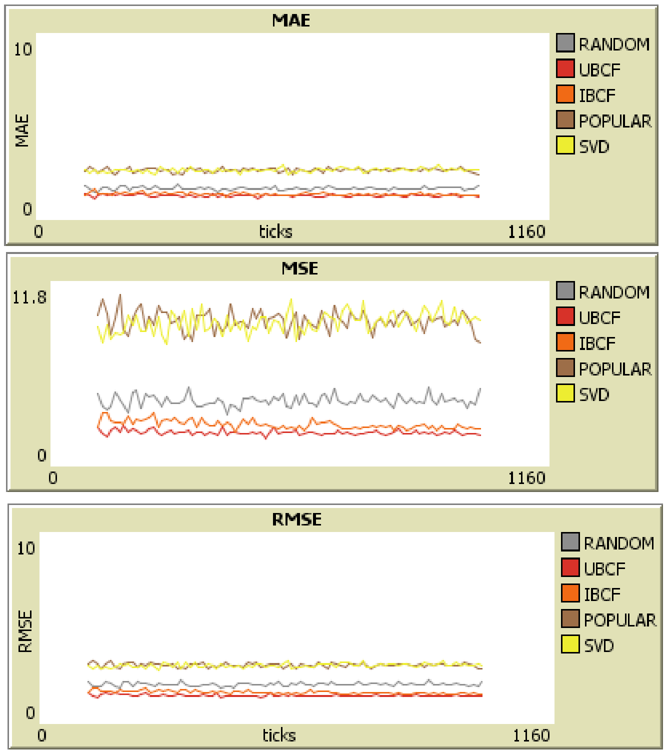

Figure 3: Experiment 2: Performance metrics for random-normal scheme.

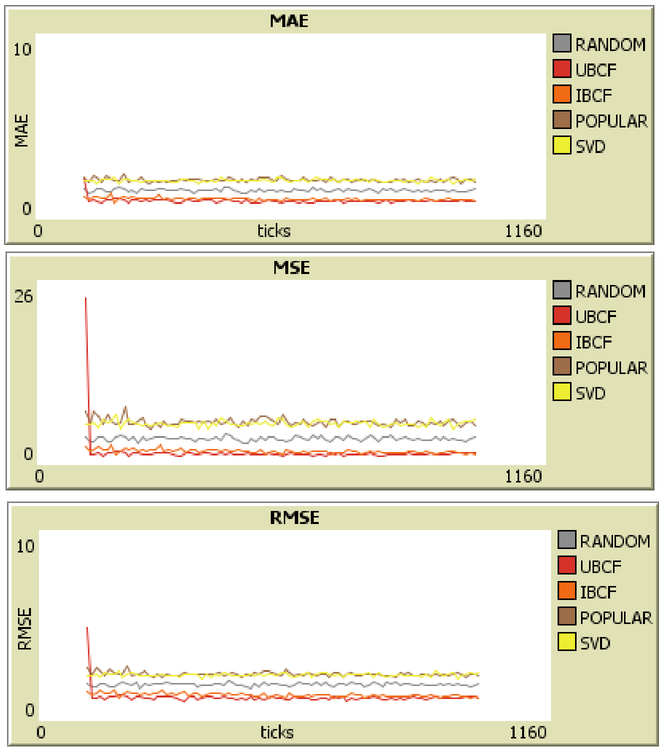

Figure 4: Experiment 3: Performance metrics for random-poisson scheme.

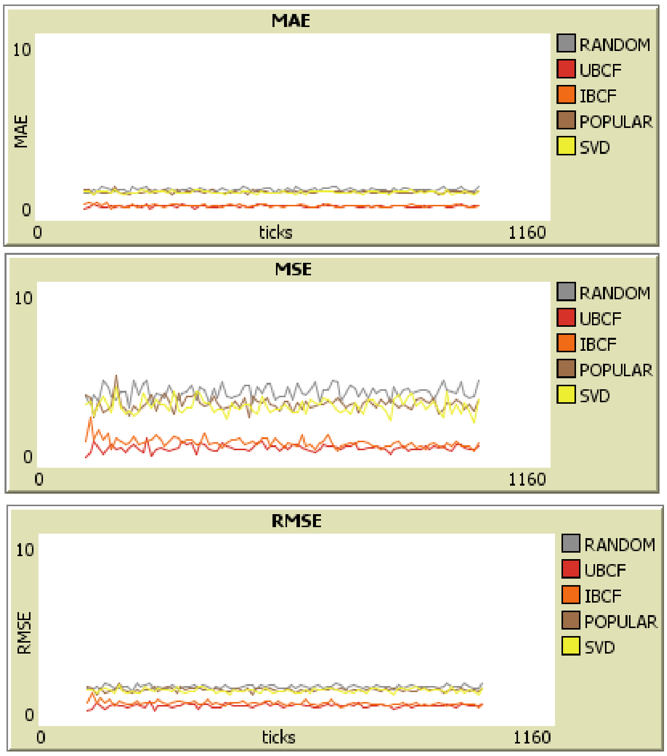

Figure 5: Experiment 4: Performance metrics for random-exponential scheme.

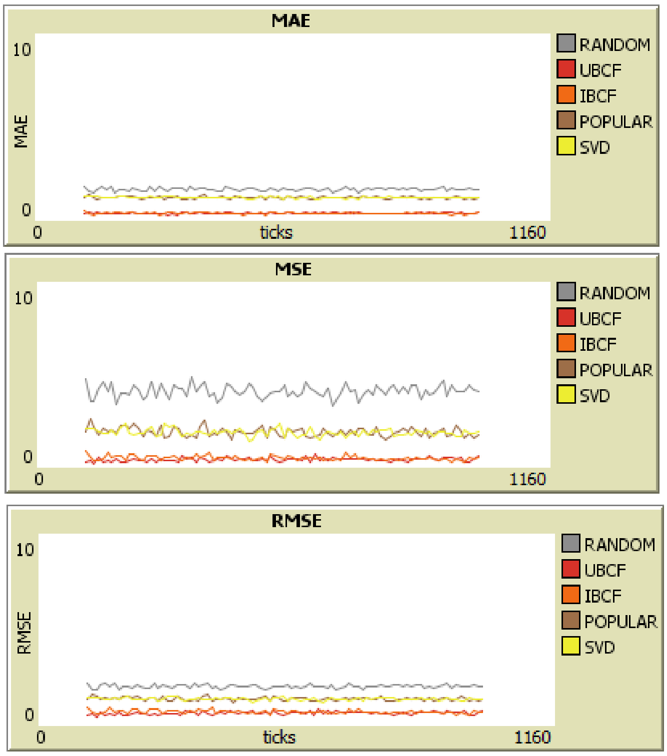

Figure 6: Experiment 5: Performance metrics for random-gamma scheme.

dently of the rating scheme. The exception is found in the random-exponential and random-gamma schemes, where the baseline provided by RANDOM is beaten. Although this behaviour is expected for the POPULAR algorithm since recommendations are not user-oriented, it should not be so in the case of SVD. In fact, there are several research works using SVD for recommendation and these results are unexpected at the least. However, several matrix factorization methods have been proposed after SVD, which outperform it both in accuracy and execution time.
On the other hand, UBCF and IBCF always have the best performance and their error is always lower than the baseline. Although IBCF requires higher processing time, the difference in performance results is not significant. In fact, all algorithms maintain an almost constant performance in each experiment. This may mean that the performance over time is not the correct way of comparing the algorithms. The assumption is that this may indicate that the performance depends not on the values assigned to each item, but rather on the matrix sparsity or the number of ratings for instance.
When we compare the different rating schemes, we observe that the metric that enables a better understanding is the MSE, because the variation is more accentuated. Using this metric for comparison among the rating schemes, it is visible that the random-exponential and random-gamma schemes provide the best average error results. In fact, only in the case of random-gamma we can observe that all algorithms outperform the baseline indicator.
Despite successfully providing a simulation tool to evaluate the performance of Recommender System algorithms, we must state that since the results are non deterministic (due to the several random rating schemes and simplistic problem statement) and therefore may not have significance in the real world. However, the proposed method can be helpful in evaluating Recommender Systems if suitable data is available to model the items and user preferences bias for a specific problem. We observe that the frameworks listed in the related work also have a great non deterministic aspect to them and that this may indicate that the evaluation should be executed for a specific context and not in a generic fashion.

Conclusions and Future Work

In this paper we presented a ABMS solution to evaluate Recommender System algorithms. We used off-the-shelve simulation and Data Mining tools to improve efficiency and assure implementation validity. The process involves deploying agents as users and items into the simulated world and use several strategies to rate items. The rating matrix is processed by several algorithms and a continuous evaluation of the algorithms executed, using MAE, MSE and RMSE error metrics. Our results show that the best performing algorithms in our context are user-based and item-based Collaborative Filtering, while the worst are recommendation of popular items and SVD. We observe also in our experiments that the performance is surprisingly constant in each recommendation scheme and that the random-exponential and random-gamma schemes provide the best average error results. For future work, there area several possibilities: (1) use better rating strategies that follow scientifically accepted social behaviours in order to increase model validity, (2) increase the number of algorithms, (3) diversify the evaluation metrics to try to extract more meaningful conclusions, (4) adapt to other kinds of Recommender Systems beyond Collaborative Filtering and (5) report the evaluation over population size and/or rating matrix sparsity instead of time to compare our results with others available in the literature.

REFERENCES

Adomavicius G. and Tuzhilin A., 2005. *Toward the Next Generation of Recommender Systems : A Survey of the State-of-the-Art and Possible Extensions. IEEE Transactions on Knowledge and Data Engineering*, 17, no. 6, 734–749.

Al-Sharawneh J. and Williams M.a., 2009. *ABMS : Agent-based Modeling and Simulation in Web Service Selection.* In *Proceedings of the International Conference on Management and Service Science.* 1–6.

Albashiri K.; Coenen F.; and Leng P., 2008. *Agent Based Frequent Set Meta Mining: Introducing EMADS. Artificial Intelligence and Practice II*, 276, 23–32.

Arroyo J.; Hassan S.; Gutiérrez C.; and Pavón J., 2010. *Re-thinking simulation: a methodological approach for the application of data mining in agent-based modelling. Computational and Mathematical Organization Theory*, 16, no. 4, 416–435.

Baqueiro O. and Wang Y., 2009. *Integrating data mining and agent based modeling and simulation. Advances in Data Mining*, 5633, 220–231.

Bobadilla J.; Hernando A.; Ortega F.; and Bernal J., 2011. *A framework for collaborative filtering recommender systems. Expert Systems with Applications*, 38, no. 12, 14609–14623.

Bobadilla J.; Ortega F.; Hernando A.; and Gutiérrez A., 2013. *Recommender systems survey. Knowledge-Based Systems*, 46, 109–132.

Hu Y.; Koren Y.; and Volinsky C., 2008. *Collaborative Filtering for Implicit Feedback Datasets. 2008 Eighth IEEE International Conference on Data Mining*, 263–272.

Jiang Y.; Liu J.; Tang M.; and Liu X., 2011. *An Effective Web Service Recommendation Method Based on Personalized Collaborative Filtering.* In *2011 IEEE International Conference on Web Services.* Ieee, 211–218.

Kargupta H.; Byung-Hoon; Hershberger D.; and Johnson E., 1999. *Collective Data Mining: A New Perspective Toward Distributed Data Analysis. Advances in Distributed and Parallel Knowledge Discovery*, 133–184.

Kargupta H.; Hamzaoglu I.; and Stafford B., 1997. *Scalable, Distributed Data Mining Using An Agent Based Architecture. Proceedings the Third International Conference on the Knowledge Discovery and Data Mining*, 211–214.

Lü L.; Medo M.; Yeung C.H.; Zhang Y.C.; Zhang Z.K.; and Zhou T., 2012. *Recommender systems. Physics Reports*, 519, no. 1, 1–49. ISSN 03701573.

Mateo R. and Lee J., 2010. *Data-mining model based on multi-agent for the intelligent distributed framework. International Journal of Intelligent Information and Database Systems*, 4, no. 4, 322–336.

Remondino M. and Correndo G., 2006. *Mabs validation through repeated execution and data mining analisys. International Journal of Simulation: Systems, Science & Technology*, 7, no. 6, 10–21.

Rong W.; Liu K.; and Liang L., 2009. *Personalized Web Service Ranking via User Group Combining Association Rule. 2009 IEEE International Conference on Web Services*, 445–452.

Saga R.; Okamoto K.; Tsuji H.; and Matsumoto K., 2011. *Proposal of a recommender system simulator based on a small-world model. Artificial Life and Robotics*, 16, no. 3, 426–429.

Saga R.; Okamoto K.; Tsuji H.; and Matsumoto K., 2013. *Evaluating Recommender System Using Multiagent-Based Simulator. Recent Progress in Data Engineering and Internet Technology*, 156, 155–162.

Yamashita A.; Kawamura H.; Iizuka H.; and Ohuchi A., 2007. *Effect of the Number of Users and Bias of Users' Preference on Recommender Systems. Intelligent Data Engineering and Automated Learning*, 4881, 1112–1121.

Yang X.; Guo Y.; Liu Y.; and Steck H., 2014. *A survey of collaborative filtering based social recommender systems. Computer Communications*, 41, 1–10. ISSN 01403664.

Zheng Z.; Ma H.; Lyu M.R.; and King I., 2009. *WSRec: A Collaborative Filtering Based Web Service Recommender System. 2009 IEEE International Conference on Web Services*, 437–444.