Towards a Framework for Agent-Based Simulation of User Behaviour in E-Commerce Context

Chapter *in* Advances in Intelligent Systems and Computing · June 2018

DOI: 10.1007/978-3-319-61578-3_3

Towards a Framework for Agent-based Simulation of User Behaviour in E-Commerce Context

Duarte Duarte[†], Hugo Sereno Ferreira[†‡], João Pedro Dias[†], and Zafeiris Kokkinogenis[†]

[†]Department of Informatics Engineering,
Faculty of Engineering of University of Porto
[‡]INESC Tec, Porto, Portugal
{ei11101,hugosf,jpmdias,kokkinogenis}@fe.up.pt

Abstract. In order to increase sales and profits, it is common that e-commerce website owners resort to several marketing and advertising techniques, attempting to influence user actions. Summarizing and analysing user behaviour is a complex task since it is hard to extrapolate patterns that never occurred before and the causality aspects of the system are not usually taken into consideration. There has been studies about characterizing user behaviour and interactions in e-commerce websites that could be used to improve this process. This paper presents an agent-based framework for simulating models of user behaviour created through data mining processes within an e-commerce context. The purpose of framework is to study the reaction of user to stimuli that influence their actions while navigating the website. Furthermore a scalability analysis is performed on a case-study.

Keywords: agent-based simulation, behaviour mining, e-commerce

1 Introduction

Customers interact with e-commerce websites in multiple ways and the companies operating them rely on optimizing key performance indicators (KPIs). In order to increase profits, retailers resort to target-marketing techniques, changing the web content and how it is displayed to the users [8]. Modelling user behaviour on the web is not new in e-commerce; it has been applied, *eg.*, for the improvement of search engines, for influencing purchase patterns, and for recommending related pages or products [6]. The typical approach consists in predicting which page has an higher probability of being the next one in the user path. This requires extensive use of historical datasets which might not expose all the causality aspects of the system.

In this work we present an agent-based framework that simulates users behaviours in an e-commerce context and assesses their interaction with artefacts that influence their navigation experience. The frameworks purpose is to be used as a "plug-and-play" test-bed where the modeller simply integrate a pre-built customer model that implements a set of actions. Particular emphasis is given on the scalability of the framework.

The remaining of the paper is organized as follows: Section 2 describes the implementation of the proposed simulation model and Section 3 discusses some validation aspects of the simulation. Some related work is presented in Section 4, and we finally draw our conclusions in Section 5.

2 Architecture & Implementation

In the following paragraphs we discuss in detail the requirements and the implementation aspects of the framework.

2.1 Requirements

Having considered several e-commerce web platforms, we have analysed the features and characteristics common to all of them in order to better assess what the framework should be able to represent and model.

Website Representation: A website consists of a collection of web pages and hyper-links between them. The common entry point is named homepage and pages have tag that describes it purpose. Product pages have, at least, the product name, its description and price [3, 4].

A virtual shopping cart is used as a staging area for the products that are going to be bought and the checkout is the act of taking all the products in the shopping cart and effectively buying and paying them.

Navigation Agents: Navigation agents are representations of customers interacting with a website. Some common interactions are: browsing, exiting the website, adding a product to the shopping cart, checking out, rating a product, writing a review or comment, bidding on a product, filling out forms and comparing two products side by side.

Website Agents: Website agents can modify any page before it is *served* to a user/customer. Example use cases are the recommendation of products to the user based on its preferences or observed browsing behaviour, targeted promotions and A/B testing analysis.

Simulation Engine: Given a website, the type of navigation and website agents and pretended simulation time, the simulation can be started, stopped and store its state and calculated metrics in a database. A simulation run can have thousands of navigation agents entering the simulation at each step, and each run can have one or more website agents.

Reporting: Once a simulation run ends, it can be analysed by taking a look at its results, metrics and other previously stored characteristics. At this end, at least two simulation runs can be put side and by side so they can be quickly compared. The comparison can be use it to infer the fitness of the website agent in providing engaging recommendations to the defined navigation agent.

The defined metrics should be relevant to the business context. Some examples are [11]: bounce rate, conversion rate, total/average order value, average order value, items per order, new visitor conversion rate, shopping cart sessions, shopping cart conversion rate, shopping cart abandonment rate, average session length, number of browsing sessions, page views per session and product views per session.

Limitations: In order to keep simplicity without disregarding the validity of the results of the framework, some limitations were assumed. Namely, adding a product to the cart and the checkout are a single step, there is no customer accounts and related interactions (e.g. *login*), visual information about the pages and products is not represented, it's not possible to remove an item from the cart, interactions with the website are limited and not extensible and the metrics gathered during the simulation are limited.

2.2 Architecture

The simulation framework encompasses two different kinds of agents, navigation agents and website agents, as shown in Figure 1.

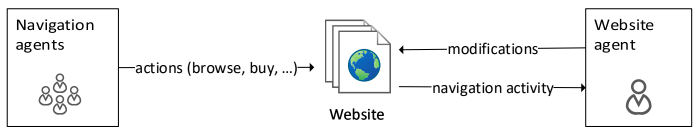

Fig. 1. Agent interaction with the environment

Navigation agents represent users interacting with the website. They have a limited view of the system, namely, access to the website pages and hyperlinks and know the current page they are visiting. Each simulation step, the framework asks each navigation agent which action will they pick next. The action may be to visit another page (`BrowseToAction`), exit the website (`ExitAction`), add aproduct to the cart (`AddToCartAction`), finish the purchase (`CheckoutAction`) or simply do nothing (`IdleAction`). Also related to the navigation agents subsystem, an implementation of `NavigationAgentFactory` is used to decide how many navigation agents are added to the system in each step, number that can be fixed *a priori* or, for example, follow a Poisson distribution model [5].

Website agents are able to modify the pages before they are presented to users. In [4] is presented an approach for behaviour mining of customers in e-commerce context. Here, profiles of regular base users are created using static and dynamic data. Every new user can be classified into one of the pre-built clusters. Considering this mining approach, it is possible to create archetypical user models that the website agent can be based on. Website agents have broader view of the system than navigation agents and are notified of all the actions that navigation agents do. The most common use case of the website agents is to recommend products to the users: before the page is served to a user, a website agent can modify a section of the page to display a custom list of products, based on user profiles.

The framework does not assume how these agents behave however the interactions between them are limited. The agents do not send messages between each other and may only interact indirectly, through the framework. While a simulation run might have hundreds or thousands of navigation agents, to simplify, each run has only one website agent instance.

Simulation Engine: The simulation engine follows a fairly standard and simple discrete event simulation architecture. The domain model we are dealing with allows certain simplifications of the simulation:

– The event list only contains events scheduled for the next step;
– There are no conditional events
– All the events happen instantaneously;
– The events do not depend on other events, they do not require synchronization and may be implemented in a single-threaded engine.

In our approach, in each simulation loop, the engine starts by calling NEWNAVIGATIONAGENTS() which adds new navigation agents to the simulation. The number and type of these agents are decided by the `NavigationAgentFactory`. Afterwards, each navigation agent currently active picks the next action. Depending on the action that was picked, the engine updates its internal state. The simulation state is represented by `WebsiteState` and contains statistics and other performance metrics. Whenever the picked action implies presenting the navigation agent a page from the website, the website agent can modify that page before it is presented, by calling MODIFYPAGE(NAVAGENT, PAGE). The website agent is also notified about all actions that the navigation agents do (NOTIFY(NAVAGENT, ACTION)). The simulation is configured to end after a fixed number of steps, otherwise it could run forever. This process is illustrated in figure 2.

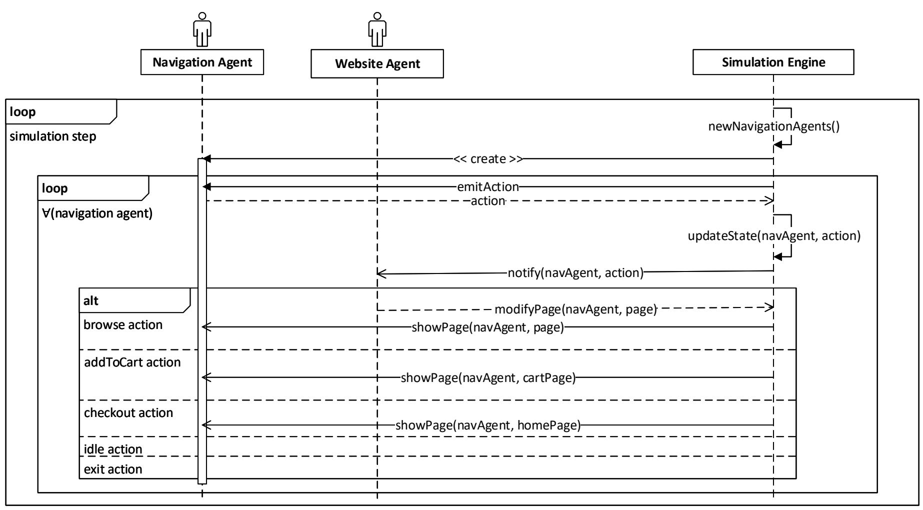

Fig. 2. Sequence diagram for the simulation engine

Scalability: To assess the scalability and performance of the simulation engine, some benchmarks were made and they are described next. The tests were ran in a Windows 10 laptop with a Intel® Core™ i7-4710HQ CPU @ 2.50GHz (8 CPUs) processor and the Benchmark.scala[1] library. The focus is not necessarily

[1] https://github.com/balagez/Benchmark.scala

in the raw speed of the engine but rather in the variation of the simulation time when the number of agents in the system or the number of steps of the simulation are increased. The test performed consists of running the same simulation with an increasing number of navigation agents and number of simulation steps, set up in the following way:

- **Website**: Sample website with 9 pages and 32 total links between pages (1 homepage, 1 cart page, 3 product list pages and 4 product pages);
- **Website agent**: Sample agent, does not modify any page;
- **Navigation agent**: Sample agent implementation which picks the next action randomly. Configured with a chance of exiting the website of $\frac{1}{3}$ and a change of adding a product to the cart of $\frac{1}{20}$;
- **Number of navigation agents**: From 1000 to 10000 (increments of 1000);
- **Number of simulation steps**: From 100 to 1000 with increments of 100.

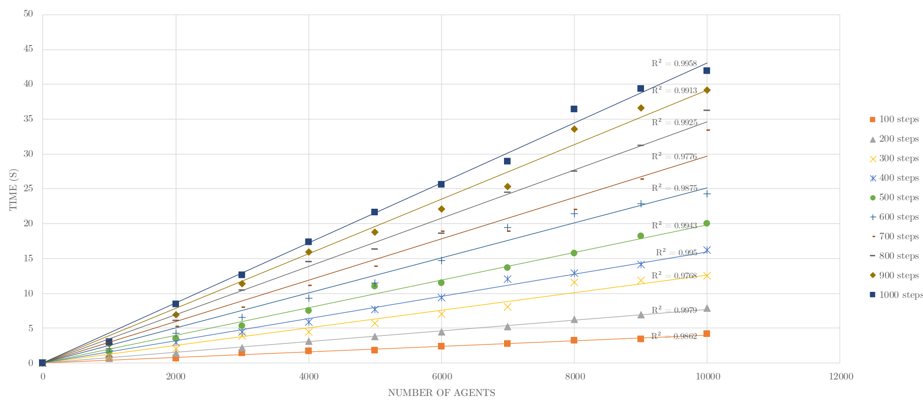

Fig. 3. Simulation running time for different number of navigation agents and simulation steps.

The result of the 10 simulation runs is shown in figure 3. A quick analysis shows that the simulation time scales linearly ($\bar{R^2} = 0.99149, \sigma = 0.00648$) with both the number of agents and the number of simulation steps. For instance, a simulation with 1000 steps and 10000 navigation agents (entering the system each step) took 41.95 seconds. These initial results are very satisfactory, however, they should be improved, especially when the number of steps is increased, so that simulations that span a longer period of time can be evaluated.

3 Tests & Validation

To validate the framework two experiments were conducted. The first was a collection of small fabricated test cases, sanity checks, where we compare the output of multiple simulation runs to the expected results to prove the correctness of the simulation engine. The second case deals with a real use of the framework, applied to an on-line store.

This second test case uses data from a real on-line store that sells electronics and computers products. This website presents a fairly standard on-line store, mostly consisting of product listing and product pages. There are three places where it is possible to recommend products: the homepage has two sections, one with product highlights and another with product promotions and each product page has a tab to show related products.

Input data and configuration: The website consists of 2540 pages with 343201 links between pages, spanning 25 base product categories and 103 subcategories. There are 750 product list pages, 1748 product pages, 1 cart page and 41 uncategorised/generic pages.

To simulate users and customers (the `NavigationAgents`) interacting with this particular website, a model based on affinities was built. This model is composed by the *affinities* themselves (a mapping between product categories and the likelihood of the user liking or having interest on products of that category), the probability of buying a product, the probability of exiting the website and the arrival rate.

Because real usage website data is not available for this website, a sample profile was created with the following properties: the affinities were set up as displayed in Table 1, probability of buying set to 5%, probability of leaving the website of 15% and a rate of arrival to the website following a *Poisson* distribution with $\lambda = 500$.

Simulation: The simulation was configured as described in the subsection above. For validation purposes, in this case, all the navigation agents use the same profile. However, the framework is extensible in order to enable the use of different user profiles for different agents. The "thought" process for each agent is fairly simple: at each step, they try to buy a product and exit the website in accordance to the probabilities defined *a priori* or navigate to a different page based on their categories, with preference as stated by the affinity table. The simulation was run for 30 steps.

Results: The results of a sample simulation run are summarized in the Tables 2 and 3. They are expected: the number of unique users is 14894 and the expected value is 15000 (500×25); the bounce rate is 14.58% and the prior leaving rate is 15%; and the conversion rate is 4.77% and the prior buy rate is 5%.

4 Related work

Peer-Olaf et al. [10] describes agent-based simulations as a "well suited to modelling systems with heterogeneous, autonomous and pro-active actors, such as human-centred systems".

The Multi-agent systems metaphor has been applied to e-commerce context mostly in two distinct areas, namely, recommendation systems and negotiation.

Table 1. Affinities for a sample user.

Category	PCs	MSI	Pens	Laptops	Intel 2011	Mem. Cards	Brand	Processors
Weight	14.29%	14.29%	7.14%	14.29%	14.29%	7.14%	14.29%	14.29%

Table 2. Visits per category for a sample simulation run.

Category	Sub-category	Count
Memory Cards	Pen Drives	6492
	SD Cards	1203
	Card Readers	1199
	Compact Flash	1158
	-	37
Laptops	MSI	14326
	HP	2623
	Asus	2584
	-	263
Processors	Intel 2011	7097
	Intel 1151	2752
	Intel 1150	2379
	AMD	2234
	-	240
Computers	Brand	19802
	-	188
Motherboards	Intel 2011	4917

Table 3. Metrics/info regarding a sample simulation run.

Field	Value
Unique users	14894
Bounce rate	14.58%
Conversion rate	4.77%
Purchases	676
NavAgentFactory	AffinityFactory
NavAgent	AffinityUser
WebsiteAgent	DummyWebsiteAgent
Start time	00:00:36
End time	00:00:39

Few relevant works exist in the literature regarding simulation of user behaviour in e-commerce context and even fewer using the agent-based paradigm.

In Petrushin [9] a customer model is created using transaction and click stream data to generate shopping lists. Following the model is simulated under varying set-up conditions with the goal to develop business strategies to capture customer's interest in purchasing product outside of shopping list. Jagannathan et al. [7] have developed an analytical framework for pricing of on demand content in a monopolistic market. The framework models customer behaviour as well as resource constraints. Based on this framework they have coupled an algorithm that suggests prices to the content-provider. Yin et al. [12] work analyses the transaction process between business and customer with its social network, and applies signalling game theory to perceive the signals that affect the utilities of business and customer. An agent-based simulation model of business-customer game is developed for computing the utilities after repeated transactions. In [2] authors,consider user behaviour simulation in using digital services and investigated the impact of online news' attributes on a discrete choice model. Ahn [1] considers agent-based modelling and evolution strategy for evaluating customer aid functions at Internet stores. The method can be used to evaluate customer aid functions under various assumptions at much lower cost compared with previous attempts based on expensive tests or empirical experiments involving human participants with fully or partly-working systems.

As we mentioned previously, few works involving agents are described in e-commerce literature. Respect to existing works we have proposed a "plug-and-play" test-bed to assess user interactions with artefacts that can influence their navigation experience.

5 Conclusions

In this paper, we discuss the implementation of a framework capable of running agent-based simulations of users that interact with an e-commerce website. This approach exploits the agent metaphor, by representing users as navigation agents. On the other hand the e-commerce context is represented by a set of artefacts and a web-page agent that adapts the context and content according to navigation agent's configuration. The framework can be considered to assess recommendation engines performance and optimization algorithms in e-commerce platforms.

There are certain limitations and assumptions in developing the model that should be further considered, such as introducing parallelism in the simulation engine for scaling-up the set-up complexity (e.g large-scale simulation) and improving the data analytic process by considering more complex metrics on the navigation agents as well as in the website agents. Furthermore, we can consider a more extensive definition of the agents action-state space.

References

1. Ahn, H.J.: Evaluating customer aid functions of online stores with agent-based models of customer behavior and evolution strategy. Information sciences (2010)
2. Constantiou, I., Hoebel, N., Zicari, R.V.: How do framing strategies influence the user's choice of content on the web? Concurrency and Computation: Practice and Experience 24(17), 2207–2220 (2012)
3. Dias, J.P.: Reverse engineering static content and dynamic behaviour of e-commerce websites for fun and profit (July 2016)
4. Dias, J.P., Ferreira, H.S.: Automating the extraction of static content and dynamic behaviour from e-commerce websites. To appear in the Proceedings of the 8th International Conference on Ambient Systems, Networks and Technologies. (2017)
5. Gündüz, Ş., Özsu, M.T.: A poisson model for user accesses to web pages. In: Computer and Information Sciences-ISCIS 2003, pp. 332–339. Springer (2003)
6. J Srivastava, R Cooley, M.D.: Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data. ACM SIGKDD Explorations (2000)
7. Jagannathan, S., Nayak, J., Almeroth, K., Hofmann, M.: A model for discovering customer value for e-content. In: Proceedings of the 8th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining. pp. 532–537. ACM (2002)
8. Mobasher, B., Cooley, R., Srivastava, J.: Automatic personalization based on Web usage mining. Communications of the ACM 43(8), 142 – 151 (2000)
9. Petrushin, V.A.: eshopper modeling and simulation. In: Aerospace/Defense Sensing, Simulation, and Controls. pp. 75–83. International Society for Optics and Photonics (2001)
10. Siebers, P.O., Aickelin, U.: Introduction to Multi-Agent Simulation. Ecology (2010)
11. Sterne, J., Cutler, M.: E-Metrics: Business Metrics for the New Economy (2000)
12. Yin, Q., Zhi, K.: Study on multi-agent based simulation process of signaling game in e-commerce. In: International Conference on Information and Management Engineering. pp. 66–72. Springer (2011)