Simulating Strategic Interaction on Online Marketplaces

[How to Survive Dynamic Pricing Competition]

Marvin Bornstein, Johanna Latt, Jan Lindemann, Nikolai J. Podlesny, Sebastian Serth, Jan Selke

ABSTRACT

@TODO:

1. INTRODUCTION

While for the last couple of centuries prices are defined through rule-based actions, more and more companies base their pricing calculations and strategies on technology and data-driven processes. Not only the actual computation of profit and cost is nowadays algorithm driven but also its update and review strategy. One of the most competitive and advanced fields is the algorithmic trading or high-frequency trading on stock exchanges. But also each one of us experiences nowadays technology driven price calculation on online marketplace like amazon, which we will hereinafter referrer to as dynamic pricing.
Currently, pricing strategies and algorithms exist but handler of those mechanism lack of the possibility to test them appropriately before releases them into the real world where they can create huge loses [1] [2] [3] [4]. We picked up the challenge to create such an environment imitating different market situations and therefore testing how pricing strategies react and interact based on influences and against each other.

Contribution: In this work, we briefly elaborate our progress of building a distributed and scalable platform to imitate market situations and simulate dynamic pricing algorithms and their effects with potential real world settings. Therefore, the following Chapter 3 contains a short introduction into the underlying architecture of the platform. The choreography of different services is described in 4. Chapter 5 will provide insights in the already implemented algorithms and their behaviors. An user facing interface on top of the RESTful API is delineated in 6 and finally Chapter 9 concludes this elaboration.

2. RELATED WORK

(only own section if we find enough)

Check literature review in http://faculty.chicagobooth.edu/workshops/omscience/pdf/Spring%202016/Popescu.pdf

3. ARCHITECTURE

Confronted with the challenge of creating a high-performance and expandable infrastructure for simulating a marketplace with different merchants and consumers, a microservice architecture was created allowing the user to scale, exchange or add single service ad-hoc and on demand. Each service within our architecture implements one business artifact. This architecture pattern comes with the cost of a communication overhead and requires farsighted API design.

Figure 1 describes the underlying architecture modeling as FMC[1] diagram. Avoiding ..
@TODO:

- add some words to microservice architecture / SOA in general
- explain our setup

4. SERVICE CHOREOGRAPHY

Like previously discussed, the architecture consists out of several services communicating with each other. This communication is done via well-defined RESTful APIs using JSON objects[2]. Due to the micro-service architecture and the design goal to allow real competition of different merchants without cheating, all important routes are secured by authorization tokens. For authenticating all participants in the simulation without the need of a centralized authentication server, a hash-based token and identification system was introduced which additionally enables the id based logging of event message corresponding to one merchant or consumer. One of the causes to implement this decentralized authentication system was to reduce the amount of requests during the simulation. Described in the previous section, the gain of flexibility and scalability of services withing a microservice architecture goes along with the cost of an increased communication overhead. With this knowledge and experiences of a first prototype, the goal was to reduce the

[1]http://www.fmc-modeling.org/
[2]https://hpi-epic.github.io/masterproject-pricewars/

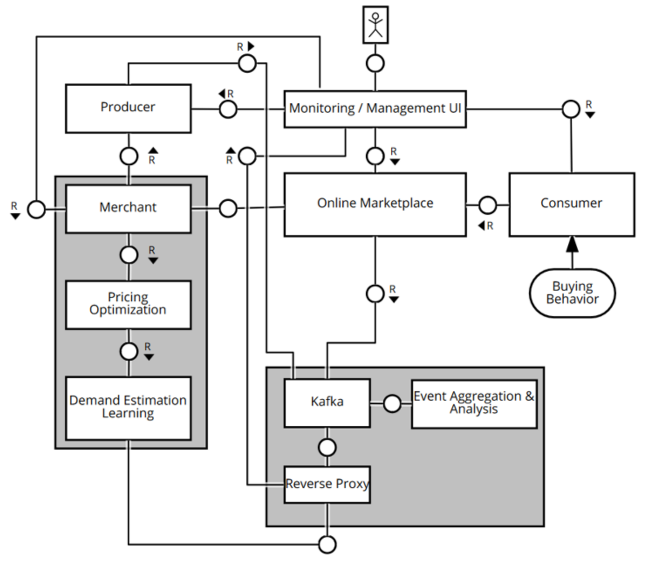

Figure 1: FMC diagram of the Price Wars architecture

amount of request being sent over the network for streamlining a simulation and its necessary communication. In the following, we will briefly outline some of the main challenges which were solved.

4.1 Decentralized Authorization

4.2 Event log analysis with Kafka and Flink

4.3 Challenges with high-density inter-service communication

4.4 Simulation Limits and bottleneck evaluation

@TODO: how do the services interact, how do we secure some major challenges in short sentences.
No ticks or such, but completely free and dynamic, every merchant can check or update prices at any time -> close to real life (unlike eg http://www.informsrmp2017.com/description-challenge.pdf)

- event logs, kafka etc
- fraud / cheating (and explain, why the consumer is required to send the price when buying products)
- (inter service communication (via REST and connection pools))
- (where are limits / bottlenecks?)

@Todo: include data Simulation 1 Stunde, 4000 Produkte à 4 Qualitäten, max 1000 Aktionen (Updates / Käufe) pro Minute, 750 Offer pro Merchant 06: 25 MB 05: 2,3 GB 04: 24 MB 03: 4 MB 02: 2 MB 01: 60 KB

5. BEHAVIORS

The addressed solution provides in its default setting already several behaviors for merchants as well as the consumers. The former include known rule-based strategies like the "Gas Station strategy", "Be the n-cheapest", "fix pricing" as well as a first data-driven approach implementing a pricing strategy based on logistic regression [5].

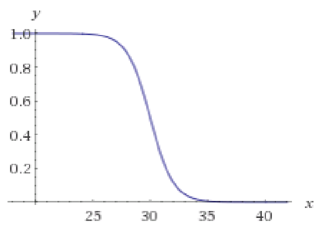

Figure 2: Sigmoid distribution as consumer behavior

Additionally, a consumer is included in the deployment implementing several buying behaviors which can be chosen, weighted and reconfigured on-the-fly. Those behaviors range from very subtle approaches like buying the n-cheapest, first, most expensive or simple random offers on the marketplace up to more sophisticated methods trying to imitate more complex consumer situations. A sigmoid distribution with twice of the producer price as mean (see fig. 2) is available through the default settings as well as a logit behavior.
The logit behavior implements a logistic regression with feature scaling and calculates for each offer the buying probability based on the feature coefficients provided in the behavior settings. Based on this buying probability for each offer, the consumer will actually choose an offer to buy. In this way, potential consumer behavior can be learned on real world data and imitated in the simulation solution.
The default settings for the logit behavior holds coefficients which were extracted from a real world use case provided by a big book retail company. For deeper insights in the concepts of logistic regression, the reader is kindly referred to Hosmer Jr et al. [5] elaboration.

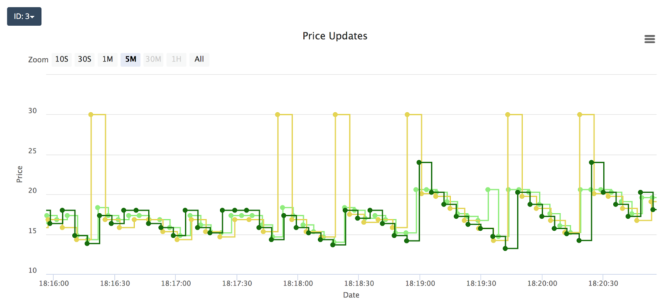

Figure 3: Price Graphs

5.1 Logistic Regression

Logistic regression is a regression model where the dependent variable – in our case the selling of an offer – is categorical. This categorization outcome must be discrete and should be dichotomous in nature simply expressed by a boolean whether a purchase happened or not. To determine this, this behavior consumes features and their coefficients can be altered within runtime and will then be applied to the next calculation iteration taking place. The describing hashmap of features and their coefficients contains only available features which are already implemented otherwise they will be ignored.

6. USER INTERFACE

Johanna, feel free to add anything which I missed

The "Management User Interface" enables the user to configure, operate and orchestrate the different microservices all in one place. Therefor, its implementation is based on angularJS consuming the RESTful APIs exposed by each individual service. Additionally, we used the socket.io[3] technology which imitates a websocket connection based on HTTP to realize real time graphs and KPIs for an ongoing simulation. This socket.io interface is provided by the kafka-reverse-proxy as consumer of the Apache Kafka[4] instances.

7. EVALUATION

@TODO: Short evaluation with a few basic strategies to show that our system indeed enables to compare these strategies ...

8. FUTURE WORK

We are aware that the current solution is far from being perfect. One is already able to simulate different market situations based a variety of consumer behaviors and competing merchants, however, this solution can be extended to cover even more possibilities. Currently, the producer may provide goods without any expiration date. But when thinking about plane or festival tickets, perishables or any other short-life products, those can be also included in a later step as simulation content. If so, the producer may include an expiration date and optionally a cap of items (for the air plane case) which consequently has to be checked and verified by the marketplace. Additionally, the behaviors by the consumer as well as within the merchants may need to react one those additional attributes.

Another very interesting case to cover in the near future could be consumer ratings and how they influence pricing strategies. Having now such an environment to simulate different market situations and consumer demands, different consumer ratings may also influence pricing strategies significantly.

9. CONCLUSION

In this work, we presented a distributed and scalable platform to imitate market situations and simulate dynamic pricing algorithms and their effects with potential real world settings. With this toolkit, handler may be enabled to test their pricing strategies appropriately before releases them into the real world where they can create huge loses. Simultaneously, students and researchers may now implement and test how pricing strategies react and interact based on influences and against each other.

The source code and its technical documentation will be publicly available at github.com [5] while a screencast is accessible under youtube[6].

Acknowledgements: As part of this elaboration, special thanks goes to Dr. Rainer Schlosser and Martin Boissier for their continuous support and supervision during the workshop. Also, thank belongs to all members of the Enterprise Platform and Integration Concepts research group for their fruitful discussions during the intermediate and final presentation.

Acknowledgments

As part of this elaboration, special thanks goes to Dr. Matthias Uflacker, Dr. Rainer Schlosser and Martin Boissier for their continuous support and supervision. Also, we thank all attendant members of the EPIC research group for their fruitful discussions.

References

[1] Matthias Uflacker, Rainer Schlosser, and Christoph Meinel. Ertragsmanagement im wandel–potentiale der in-memorytechnologie. *Handel 4.0: Die Digitalisierung des Handels-Strategien, Technologien, Transformation*, page 177, 2016.

[2] Rainer Schlosser and Martin Boissier. Optimal price reaction strategies in the presence of active and passive competitors. 2016.

[3] Rainer Schlosser. Stochastic dynamic multi-product pricing with dynamic advertising and adoption effects. *Journal of Revenue and Pricing Management*, 15(2): 153–169, 2016.

[4] Rainer Schlosser, Martin Boissier, Andre Schober, and Matthias Uflacker. How to survive dynamic pricing competition in e-commerce. 2016.

[5] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.

[3]https://github.com/socketio/socket.io
[4]https://kafka.apache.org/
[5]https://github.com/hpi-epic/masterproject-pricewars
[6]https://www.youtube.com/watch?v=bqXSi5cv8cE