Applying model checking technique to a recommender system

About

In recent years, recommender systems have been extremely widely used in various applications and websites, which can generate user profiles by collecting user data to recommend customized services for users. This paper discusses the application of model checking in data cleaning after data collection of recommender systems from the perspective of data cleaning.

1. Introduction

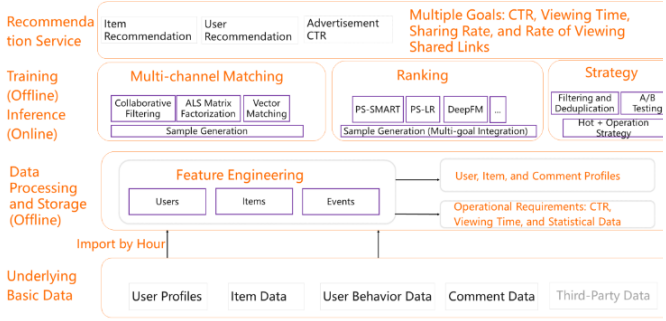
From the data storage layer to the recall layer, to the fusion filtering layer and the sorting layer, the candidate set decreases layer by layer, but the accuracy requirement is getting higher and higher, so it also brings the increase of computational complexity layer by layer, which is the biggest challenge of the recommender system. Data cleansing and processing is often required between the data source and the data storage layer. Data plays a very important role for the whole recommender system. Apply model checking in the data cleansing phase to better make the data work for the entire architecture.

2. Recommender System

RS are software tools and related technologies that suggest projects that are considered relevant to specific users, often in scenarios or applications where the project space is very large and project search and selection is difficult for the user.

In general, RSs follow a process that encompasses three main steps: 1). Collect related user information. 2). Learning from the collected information to build user profiles; 3). Applying a function or a previously built model to select and rank the items that the user is most likely to prefer.

(1) Overall Architecture of a Recommender System



(2) layers of Recommender System

- Data sources: the various data sources on which the recommender algorithm relies, including item data, user data, behavior logs, other available business data, and even data from third parties.
- Computing platform: responsible for cleaning and processing of various data at the bottom, off-line and real-time computing.
- Data storage layer: storage of data processed by the computing platform, which can be landed in different storage systems as needed.
- Recall layer: including various recommendation strategies or algorithms, such as classic collaborative filtering, content-based recall, etc.
- Fusion filtering layer: triggering multi-way recall, because each recall source of the recall layer will return a candidate set, so this layer needs to be fused and filtered.
- Sorting layer: Use machine learning or deep learning models, and richer features for reordering to filter out a smaller, more accurate set of recommendations to return to the upper layer of business.

3. Formal model

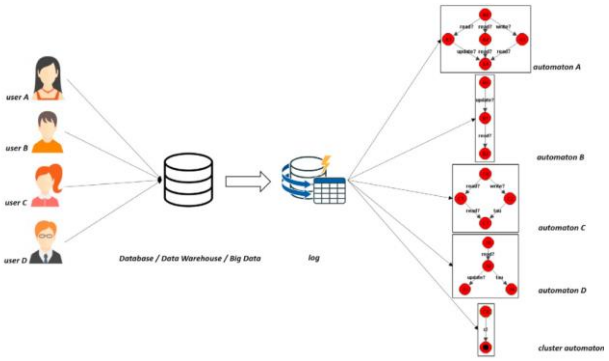
Using the Calculus of Communication Systems of Milner (CCS), a widespread process algebra.

CCS defines operators to build finite processes, to describe parallel composition, choice between actions and scope restriction but also some notion of recursion aimed to catch infinite behavior.

A CCS process p is formally defined using the structural operational semantics, i.e., a set of conditional rules describing the transition relation of

the automaton corresponding to the behavior expression defining p. The considered automaton is called standard transition system of p.

(1) Data Anomaly Detection CSS formal model



A set of four users (user A, user B, user C and user D) has been considered. Each user is able to perform read, update and write operation on data. Each user operation is stored through log: from the log are gathered a series of information enabling the proposed method to build an automaton for each user. In the considered example (automaton A, automaton B, automaton C and automaton D, representing the user behaviors, have been generated. Furthermore, a cluster automaton is built, with the responsibility to guarantee the synchronization between the other automata.

4. Model Checker

As model checker in this paper we consider TAPAs , a tool for specifying and analyzing concurrent systems.

TAPAS is a tool for specifying and analyzing concurrent systems. Its aim is to support teaching of process algebras. Systems are described as process algebra terms that are then mapped to labeled transition systems (LTSs). Properties can be verified by checking equivalences between concrete and abstract system descriptions or by model checking.

In TAPAs, concurrent systems are described by means of processes, which are nondeterministic descriptions of system behaviors, and process systems, which are obtained by process compositions. Notably, processes can be defined in terms of other processes or process systems. Processes and process systems are composed by using the operators of a given process algebra.

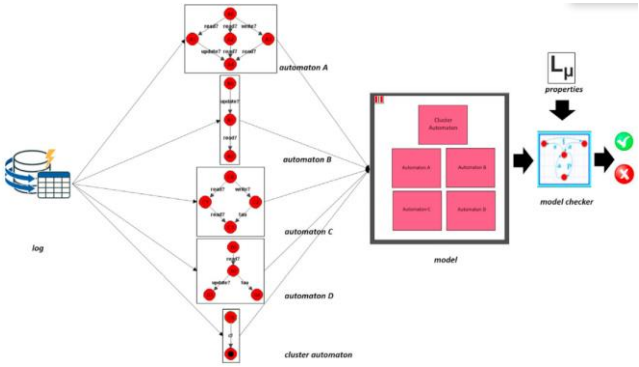
5. Temporal Logic Properties and Model Checking

(1) Temporal logic formula

$$\phi = \nu X.[action\_A]\, ff \wedge [-action\_A, action\_B]\, X$$

The ϕ property is describing following behavior: “it is not possible to perform the action A whether action B is not previously done”. This property will be considered in the case study to detect the data anomalies.

(2) Model checking



In this phase a CSS model invoking in parallel all the automata is built (i.e., model in Figure 2) with a set of restricted action to guarantee the synchronization. Thus, the model checker verifies whether the built model satisfies a set of properties expressed in mu-calculus: whether the model checker outputs true the model is compliant with respect to the legitimate behavior, otherwise (i.e., the model checker outputs false) the property under analysis is not verified on the model and this is symptomatic that a deviation from the legitimate behavior is occurred.

(3) Data anomaly detection temporal logic properties and Model Checking results

Anomaly_1	ν X. ([read_g_Registry_A?]false ∧ [-(read_g_Registry_A?, write_f_Account_M?)]X)	Yes	0.0 s
Anomaly_2	ν X. ([read_f_Account_F?]false ∧ [-(read_f_Account_F?, write_f_Account_M?)]X)	Yes	0.0 s
Anomaly_3	ν X. ([read_c_Registry_A]false ∧ [-(read_c_Registry_A?, write_g_Registry_A?)]X)	Yes	0.005 s
Anomaly_4	ν X. ([write_g_Registry_A?]false ∧ [-(write_g_Registry_A?, read_g_Registry_F?)]X)	Yes	0.0 s
Anomaly_5	ν X. ([read_f_Account_F?]false ∧ [-(read_f_Account_F?, write_f_Account_M?)]X)	Yes	0.0 s
Legitimate_1	ν X. ([read_b_Project_A?]false ∧ [-(read_b_Project_A?, write_g_Registry_A?)]X)	No	0.0 s
Legitimate_2	ν X. ([write_e_Budget_A?]false ∧ [-(write_e_Budget_A?, write_g_Registry_A?)]X)	No	0.006 s
Legitimate_3	ν X. ([write_e_Budget_A?]false ∧ [-(write_e_Budget_A?, read_e_Budget_F?)]X)	No	0.003 s
Legitimate_4	ν X. ([read_c_Registry_A?]false ∧ [-(read_h_Movement_M?, read_c_Registry_A?)]X)	No	0.0 s
Legitimate_5	ν X. ([write_e_Budget_A?]false ∧ [-(write_e_Budget_A?, write_g_Movement_M?)]X)	No	0.005 s

Reference

1. documentation,eTAPAS,Introduction to CCSP. http://etapas.sourceforge.net/?page_id=185
2. Madalina G. Ciobanu, Fausto Fasano, Fabio Martinelli, Francesco Mercaldo, Antonella Santone, Model Checking for Data Anomaly Detection, Procedia Computer Science,Volume 159,2019,Pages 1277-1286,ISSN 1877-0509, https://doi.org/10.1016/j.procs.2019.09.297

3. Almonte, L., Guerra, E., Cantador , I. et al. Recommender systems in model-driven engineering. Softw Syst Model 21, 249-280 (2022). https://doi.org/10.1007/s10270-021-00905-x
4. Basic Concepts and Architecture of a Recommender System, https://www.alibabacloud.com/blog/basic-concepts-and-architecture-of-a-recommender-system_596642
5. TAPAs model TAPAs model. https://en.wikipedia.org/wiki/TAPAs_model_checker