The formalization of building software system user segments using relational interactive logic

D A Naumov¹ and D V Zakharov²

¹ Institute of Information Technology and Data Science, Irkutsk National Technical Research University, 83 Lermontov St., 664074 Irkutsk, Russia
² Department of Probability Theory and Discrete Mathematics, Irkutsk State University, 1 Karl Marx St., 664003 Irkutsk, Russia

E-mail: zakharov-danyl@mail.ru

Abstract. This paper proposes an approach to formalizing the process of constructing segments of software system users by relational interactive logic. The segment description is specified as a system of logical equations and inequalities. The role of variables in such a system is played by relational terms, each of which represents some qualitative or quantitative characteristic of the user. To automatize this approach, a software solution architecture diagram is proposed, which provides the construction and interpretation of a system of logical equations.

1. Introduction

The current trend in the sphere of customer relationship is a personalized approach [1]. Segmentation of buyers, site visitors or application users allow companies to gain significant competitive advantages by building more effective communications, as well as tailoring their offers to the needs of specific customers. Examples include the formation of targeted mailings, personalized pricing policies, or even the adaptation of content on web pages depending on the user's preferences [2].

User segmentation is of particular importance in the field of software development management. Dividing users into segments may be necessary when testing hypotheses about the influence of any user parameters on his behavior when using the product, forming samples for conducting qualitative research, or, for example, optimizing the line of pricing plans according to the criteria of the paying capacity of user segments and the functionality necessary for these segments.

In the general case, a separate segment of users can be represented as a set of parameters with restrictions on these parameters. Such parameters can be gender, age, type of hardware platform used, presence of certain types of events, or type of a pricing plan (figure 1).
Figure 1. An example of user parameters for building a segment.

Typically, information about users is accumulated and stored in one or more data stores. These can be relational databases, NoSQL databases, or external analytical systems with their own dialects of query languages, for example, such as Google Analytics or Yandex.Metrica. However, for the purposes of this article, we will assume that all data is stored in a relational database.

Regardless of the goals of forming a segment, the process of collecting and uploading data is always approximately the same and consists of the following stages:

- determination of a set of characteristics and parameters for which data will be sampled
- definition of limiting values for the selected parameters
- recording one or more queries to data stores
- data uploading or reporting

A wide variety of people may have a need for user sampling: product managers, marketers, technical support specialists, or data analysts.

Note that for a specialist who does not have the necessary technical skills, and also has no idea about the structure of data storage, the task of forming a segment of users can become very sophisticated. In this case, it becomes necessary to attract data specialists—data engineers and analysts, which ultimately leads to an increase in various costs for the initial task.

Taking into account the above, we can conclude that it is necessary to create a methodology that provides description and subsequent formation of a segment of users in some formalized way available for use by both technical specialists and specialists of other areas of expertise.

Within the scope of this work, it is proposed to use relational interactive logic (RIL) as a formalized way of describing user segments. RIL allows you to write down conditions that limit the data sampling for the formation of a segment as systems of logical equations and inequalities, in which the attributes of users of the software product act as variables.

It is worth noting that a similar approach can be used to generate samples on the so-called “manual”, deterministic basis, when the set of parameters and the values limiting them are known in advance.

Below, in the next paragraph, we will consider alternative approaches to building user segments.
2. The approaches to building segments

2.1. Segmentation based on predefined conditions.
The most significant limitation in applying this approach is the lack of flexibility when choosing segmentation parameters. The introduction of new parameters automatically means the need to improve the software product. With a large number of parameters, filter systems become inconvenient for perception, which leads to errors when constructing segments.

2.2. Segmentation based on clustering by parameter set.
The main disadvantage of this approach is that it is necessary to preprocess the data and adjust the clustering algorithms in order for them to more accurately reflect the picture of the subject area. If we are talking about segmentation on small amounts of data, then classifying sub-clusters with excessively fine pitch can be formed. Such results are more difficult to interpret because subcluster data can be subject to random noise. For example, clustering has shown that users who use 50-60% and 80-100% of the functionality are most prone to make additional purchases. A sub-cluster with 60-80% of functionality is not necessarily unlikely to make additional purchases, as the result could be influenced by random noises, and the true result is: “users that are prone to additional purchases use more than 50% of functionality”. To configure clustering according to the condition “> 50%”, deep knowledge of clustering algorithms and data-engineering skills are required, while building such a segment using RIL does not require substantial knowledge in computer science. Another significant problem of segmentation through clustering is the need to normalize the input data, as the influence of a parameter with low dispersion may be “absorbed” by the influence of a parameter with high dispersion.

3. Relational interactive logic
Before moving on to practical examples, let us consider the subject and method of relational interactive logic.

The main difference between RIL and other formal logics is that, in addition to textual notation, formulas can also be represented as relational tables and SQL queries. Execution of a query against the database over a set of tables that define the domain is an act of logical inference [3].

Let us consider a number of key concepts of RIL in the context of the topics discussed in this article.

3.1 Relational table.
In the context of RIL terms, let us agree that the concept of a relational table fully corresponds to the concept of “relational relation” from relational algebra and satisfies all its necessary conditions and constraints. [4]. In the context of this work, a table is an array of tuples, each of which contains a set of user data.

3.2 Relational term.
A relational term is generally an expression of the form [table]. [Field] and is used as a variable when constructing logical expressions. On the other hand, the term specifies a link to refer to a specific field of a given table. Taking into account the fact that in the considered context the field contains information about some user parameter, then we can regard each term as a user attribute. For example, the “age” parameter will be specified by the term [user_info]. [age]. As an object of a logical expression, a relational term exists exclusively in the context of an RIL.

3.3 Constant.
A constant is some fixed value. In general case, constants are used to impose constraints on the values of relational terms. For example, a constant can be used to form a condition limiting the sample of users by age: [user_info]. [Age] <35. For this expression, an integer constant 35 was used.
3.4 Formula.
A formula can be atomic – indivisible or composed of atomic formulas. An example of the simplest atomic formula is the boolean constants true and false. Atomic formulas are constructed using logical functions and comparison operators, from relational terms and constants. On the basis of atomic formulas, more complex formulas are built using logical connectives (NOT, AND, OR, IMP, XOR, EQV). Since RIL is a grammar that can be interpreted in the WHERE clause of a SQL query, it seems possible to introduce fuzzy query operators to implement the fuzzy queries mechanism based on Zadeh's fuzzy set theory. [5, 6] It should be noted that the classical SQL dialect does not provide the ability to work with fuzzy queries and most RDBMS do not implement this feature in the provided dialects. In this work, formulas will be used to record individual conditions that specify the sample limitation for a segment of users.

3.5 Logical equation.
A logical equation is a formula of the form: A EQV B, where A and B, in turn, are other logical formulas, which can be atomic or consist of other formulas. The EQV operator defines an equivalence relation.

3.6 Logical inequality.
By a logical inequality we mean a formula written as: C IMP D, where C and D are formulas, and IMP defines the relation of the corollary. Logical inequalities are used to write conditions in the event that there is a cause-effect relationship between user attributes.

3.7 System of logical equations and inequalities.
By an ordinary system of logical equations and inequalities (SLEI) we mean a finite set of logical equations and inequalities united by the AND operator, which is the equivalent of the logical conjunction operator Ʌ. The conjunction operation ensures that conditions are fulfilled simultaneously. Conditions are set by equations and inequalities included in the system. Ultimately, it is SLEI that completely describes a separate segment of users.

The next paragraph will consider examples of building user segments using RIL.

4. Building user segments using RIL
Let's consider an example of the description of a user segment using RIL.

Let's suppose that for the research it is necessary to select a set of users under 35 years old from Russia who have subscribed to a premium subscription. Let's write the segment description as a logical equation.

\[ \text{[user_info].[age]} < 35 \text{ AND [user_info].[country]} = 'RU' \text{ AND [user_info].[subscription_type]} = 'premium' \text{ EQV TRUE} \]

In this logical equation, three relational terms were used to denote the corresponding attributes, as well as constants to select the required values. An SQL query built on the basis of this equation will return user records for which this equation becomes an identity. This expression describes a fairly simple condition for the sample of users; we will supplement it with a number of additional constraints. String constants and algebraic comparison operators were also used to write constraints on attribute values.

From the sample formed above, we also need to select a group of users who have viewed the product card more than 20 times and registered after January 1, 2020. To form an additional SLEI, use the relational term [open_product_cart].COUNT and [registration_date]. Note that to count the number of required events, in addition to the relational term itself, you also need to specify the COUNT aggregation function, which returns the number of records with this event.

The combination of logical expressions and aggregating functions that allow performing additional calculations makes RIL a rather powerful tool for building user segments. For numeric attributes,
MAX(), MIN(), or AVG() can be used as aggregation functions. The applicability of aggregate functions depends on the implementation of the SQL dialect used by the particular DBMS.

Taking into account the above, we will write down the SLEI with the description of the refined segment.

\[
(((\text{[user_info].[age]} < 35) \text{ AND } (\text{[user_info].[country]} = 'RU') \text{ AND } (\text{[user_info].[subscription_type]} = 'premium') \text{ EQV TRUE}) \text{ AND } ((\text{[open_product_cart].COUNT} \geq 20) \text{ AND } (\text{[registration_date]} \geq '01-01-2020') \text{ EQV TRUE}) \text{ EQV TRUE})
\]

RIL allows you to define rather complex user segments from the point of view of conditions. A combination of logical operators such as XOR – exclusive OR or IMP – implication operator.

The use of attributes of a specific subject area allows various specialists to effectively operate them. Previously formed segment descriptions can be refined and supplemented by introducing new logical conditions into the sample description.

In turn, the formalized recording of segments allows you to automate the process of automatic sampling by generating an SQL query based on the SLEI.

The next section discusses the features of the software system for the formation of user segments. The issue of generating a dictionary of attributes based on a database diagram is considered separately.

5. Software implementation
The architecture diagram of the software system that enables user segmentation by applying RIL is shown in figure 2.

![Diagram of the segmentation system architecture](image)

**Figure 2.** Diagram of the segmentation system architecture.

As you can see from the figure, the system consists of two main parts: SLEI-Builder and SLEI-Interpreter.

SLEI-Builder is an interface part of the system that allows the user to enter the necessary SLEI in a visual form that is convenient for human perception. It should be noted that, as stated above, RIL is a context-free grammar for describing an SQL query. Thus, you can parse the LARL (1) expression described in terms of RIL with a parser. However, in the segmentation task, one of the requirements is the ability to use the tool without having deep knowledge of computer science and mathematics. To comply with this requirement, it is proposed to provide an interface for constructing an SLEI in the form of abstract syntax trees (ASTs) or structures that can be reduced to ASTs by trivial graph
algorithms. An example of the implementation of the Builder of segmentation conditions in the form of a WEB-application is shown in figure 3.

Figure 3. An example of a segmentation condition builder.

RIL terms can be generated both on the basis of the DDL DESCRIBE TABLE query, and on the basis of the fields contained in the table (including calculable ones), as well as the results of using aggregate functions, which are selected by a separate SELECT query during SLEI-Builder initialization.

SLEI-Interpreter is the interpreter of the resulting abstract syntax tree, building the SQL WHERE clause on its basis. In the process of interpretation, the terms and constants listed in the SLEI are selected. RIL statements are replaced with and applied to statements or SQL function calls.

Thus the above structure is the following SLEI:

\[((\text{events}.\text{actions\_count} \geq 5) \text{ EQV TRUE})\]
\text{AND}
\[((\text{e-comm-app}.\text{is\_use\_delivery} \text{ EQV TRUE }) \text{ EQV TRUE})\]

This SLUN is interpreted in the WHERE clause of the SQL query given below:

WHERE actions\_count (events.event)\geq 5 AND e-comm-app.is\_use\_delivery

It should be noted that since in this case the SLEI describes only the WHERE clause, the rest of the SELECT sections (including terms that are calculable fields and the results of using aggregate functions) must be prepared in advance.

6. Conclusion

The use of relational interactive logic in segmentation tools offers great opportunities for simplifying interaction with databases, which makes it possible to fully use all the filtering capabilities provided by SQL for non-database specialists, which is important in analytics systems. At the same time, the use of RIL allows you to centralize the sample of data from several sources in several queries, and to divide the results in accordance with the segments specified in the SLEI, which solves the problem of
loading analysts / data engineers with routine tasks. The ability to interpret SLEI in the WHERE clause of SQL allows you to easily add add-ons over the filtering capabilities provided by the DBMS, leaving the construction of filtering (segmentation) conditions clear to the end user.

Note that, in the general case, a segment can be considered as a piece of data within a certain subject area that meets a given set of restrictions. Taking into account this interpretation, the above approach can also be applied when analyzing data from scientific experiments or the results of monitoring procedures. An example is the processing of the results collected by an autonomous environmental laboratory located on an unmanned aerial vehicle. In this case, the target segment of the data upload can be a set of records in which an excess of the permissible level of a pollutant was recorded.

Acknowledgments
The reported study was funded by RFBR and the Government of the Irkutsk Region, project number 20-41-385001.

References
[1] Shen A 2014 Recommendations as personalized marketing: insights from customer experiences J. of Services Marketing 28 414-27
[2] Mirović M, Miličević M and Obradović I 2018 A framework for dynamic data-driven user interfaces 41st Int. Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO) pp. 1421–5
[3] Kurgansky V I 2014 Relational Interactive Logic. from Logical Equations and Inequalities to Answers to Difficult Questions (Saarbrücken: Lambert Academic Publishing) p 124
[4] Codd E F 1970 Relational completeness of data base sublanguages Communications of the ACM 13 377-7
[5] Dubois D and Prade H 2001 Possibility Theory, Probability Theory and Multiple-Valued Logics A Clarification Annals of Mathematics and Artificial Intelligence 32 35-66
[6] Zadeh, Lofti A 1978 Fuzzy Sets as a Basis for a Theory of Probability Fuzzy Sets and Systems 1 3-28