Misconception elicitation from the logs of an educational system

Master Thesis

Saan Rashid
3701034
Utrecht University
10-05-2019

First Supervisor: Dr. Sergey Sosnovsky, Utrecht University

Second Supervisor: Dr. Matthieu Brinkhuis, Utrecht University

INFOMMBI1

Master Business informatics

Institute of Information and Computing Sciences

Utrecht University

Acknowledgments

In the last year or so I have been investigating misconceptions of children who have solved arithmetic exercises. During this process various people have contributed who I would like to thank. First, my main supervisor Sergey Sosnovsky. He has enabled my development in an academic and personal manner. I would like to thank him for his valuable insights, patience and guidance. In various ways he has resolved some of my own misconceptions. Second, my second supervisor Matthieu Brinkhuis, for sparking my interest in the field of educational data mining and overseeing this research project as a second supervisor. Third, my fellow graduate students Laurens Müter, Marc Valkenier and Bertine van Deijzen, for our interesting discussions during our bi-weekly meetings. Fourth, Oefenweb, for allowing me to investigate their response logs. Fifth, my family and friends, for their ongoing support during this project. Finally, the children who have solved the arithmetic exercises. Without their creativity and ingenuity this research would not be possible.

Contents

List of Figures

List of Tables

List of Abbreviations

OLE	Online learning environment
ARM	Association rule mining
CRISP-DM	CRoss Industry Standard Process for Data Mining
SEMMA	Sample, Explore, Modify, Model and Assess
ECD-model	Evidence-centred design model
RQ	Research question
M	Modularity
X	Decrease in support per iteration
Y	Decrease in confidence per iteration
TP	True positive
TN	True negative
FP	False positive
FN	False negative

1 Introduction

Increasingly large amounts of educational data are collected in online learning environments (OLEs). This trend is leveraged by a relatively new field of study, namely educational datamining and learning analytics. In this area, knowledge discovery and data mining techniques are used to gain insights from big datasets that are not easily interpreted by humans (Baker & Inventado, 2014). Examples are research on disengagement (e.g. attempts to game the system), knowledge assessment through cognitive tutors, and the development of distributed networked learning systems (Baker & Inventado, 2014). One challenge is to alleviate the work of domain experts or cognitive scientists who manually investigate response trajectories to gain insight into learning behaviour (Barnes, 2005).

According to West (1971), the most fruitful diagnosis in teaching – specifically mathematics teaching – is to investigate error patterns in four steps. First, systematic errors are distinguished from random and careless mistakes. Second, the nature of systematic errors is identified; third, the causes of those errors are identified. Finally, the appropriate intervention is applied. As will become apparent below, this research focusses on the automation of steps 1, 2, and to some extent 3.

Systematic errors could arise because of a lack of knowledge regarding a concept or because a learner has established a false understanding of a concept (i.e. has developed a misconception). Research shows that arithmetic misconceptions occur when a learner confronts a difficult or unknown step in a procedure and replaces it with an erroneous step (Woodward & Howard, 1994; Pellegrino & Goldman, 1987; Resnick & Ford, 1981; Brown & Burton, 1978; Van Lehn, 1982, 1988). The identification of misconceptions is important because of the implications for giving instructions (Radatz, 1979; Yetkin, 2003). If the identified misconception is disregarded, more practice related to the same concept might be suggested or the difficulty level might be adjusted. However, the learner can benefit more by using the newly gained knowledge in an appropriate context (Nesher, 1987).

Early work towards automated misconception identification was done by Tatsuoka, who created the rule-space method. In this approach, the process of mapping learners' knowledge states through their responses is automated (Tatsuoka, 1983). Tatsuoka noted that binary scoring (1 for the correct answer and 0 for any wrong answer) did not consider the erroneous rules that could produce an incorrect (or correct) answer. In addition to Tatsuoka, several studies that modelled typical misconceptions, focused on procedural errors (i.e. errors in 'how to' knowledge), were fruitful (Brown & Burton, 1978; Brown & Van Lehn, 1980). In this context, 'bugs' are defined as erroneous rules that the system can identify. The bugs are stored in bug libraries and notify the system when there is an indication of a misconception and the nature of this misconception. However, the way that OLEs are set up introduces inevitable information gaps about a learner's knowledge state when assessing error responses. The systems that are designed to model procedural errors are different from typical OLEs, which makes the identification of misconceptions a costly and timely endeavour that requires specific domain expertise (Self, 1990; Guzman et al., 2010).

Identifying not only misconceptions but also their causes gives a more complete indication of a learner's knowledge state and allows for appropriate intervention. For example, an erroneous transfer of knowledge that is applicable in one context to another context where such knowledge is not applicable directly can be the cause. Additionally, it is interesting to understand what concepts the learners are dealing with. Misconceptions can occur between several concepts, and where several concepts are applied, a single misconception can occur. Investigating the relationship between questions can help the expert to understand the questions that occur in the error responses.

In summary, the identification of misconceptions in OLEs is a costly and timely endeavour that requires specific domain expertise. This research proposes a method for the semi-automatic identification of possible typical misconceptions and underlying causes, through analysing learning behaviour in an OLE. First, the research approach is outlined. Second, the experiment and its results are described. The thesis is concluded with a discussion of main outcomes, limitations and possible steps for future research.

1.1 Problem Statement

Frequent patterns of learning behaviour can help the system to understand a learner’s knowledge state. In turn this enables it to balance the level of difficulty, provide relevant hints or simply suggest the learner to continue practicing the same exercises (Guerra et al., 2014). In other words, the more information is extracted about the specific behaviour of a learner, the more effective interventions an adaptive system can produce to facilitate the learner’s learning needs.

One aspect of learner modelling is the identification of typical misconceptions. In this regard overlay models and perturbation models are distinguished (Stansfield et al., 1976; Mayo, 2001). In overlay models, learner behaviour is compared to expert behaviour and the learner’s knowledge is thus modelled as a subset of expert knowledge (Figure 1.1). Differences are assumed to be gaps but are not specifically modelled. Perturbation models try to overcome this problem by acknowledging the difference in terms of quality and quantity of a learner’s knowledge pertaining to expert knowledge (Figure 1.2).

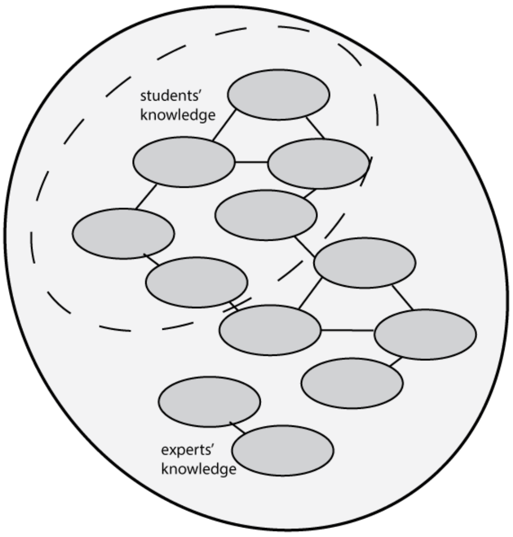


FIGURE 0.1 OVERLAY MODEL (STANSFIELD ET AL., 1976)

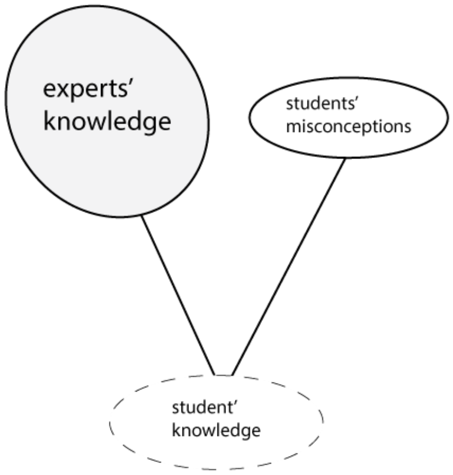


FIGURE 0.2 PERTURBATION MODEL (MAYO, 2001)

These models show that to better understand a learner’s knowledge state, one must investigate what a learner does or does not know, that is, the concepts; but also what a learner has learned incorrectly (i.e. misconceptions). Considering the latter, in environments where there is incomplete information about a learner’s learning behaviour, learner modelling is still a costly and timely endeavour (Self, 1990; Guzman et al., 2010). The method proposed in this research attempts to apply knowledge discovery and data mining techniques to model typical misconceptions and their causes. Doing so can alleviate part of the costly and timely work associated with modelling learners in OLEs.

2 Research approach

2.1 Research Objective

The objective of this thesis is to develop a method for the automatic identification of misconceptions and their possible underlying causes through the analysis of logged learning behaviour in an OLE. As in West's (1971) diagnostic teaching method, first systematic erroneous learning behaviour is identified; second, the nature of systematic erroneous learning behaviour is investigated; and finally, the cause of systematic erroneous learning behaviour is investigated. This research focuses on response trajectories that are the result of answering arithmetic exercises. Since the researcher was interested in patterns, a data mining technique called *association rule mining* (ARM) was applied to find meaningful patterns in these response trajectories and a *network analysis* was done to enable the interpretation of these patterns.

2.2 Research Questions

The research objective led to the following main research question:

[MRQ] How can we effectively identify misconceptions that cause frequent patterns of erroneous learning behaviour?

First, systematic errors are investigated because these might be an indication of typical misconceptions. Therefore, the following research questions are proposed.

[RQ1] Can systematic errors be distinguished from random and careless errors?

[RQ2] What are the systematic errors caused by erroneous learning behaviour?

Second, the underlying causes might be explained through investigation of the error patterns they relate to. Therefore, the following research questions were proposed.

[RQ3] Can we effectively apply knowledge about systematic errors to identify underlying causes?

Finally, to understand the systematic error patterns, an expert analysis was conducted. This expert analysis was expected to answer the following research question.

[RQ4] Can typical misconceptions be identified based on an expert analysis?

2.3 CRISP-DM

Feelders et al. (2000, p.272) noted that data mining – in this paper, association rule mining – is not only concerned about '*the extraction of knowledge from really large datasets'.* The authors emphasized the importance of study design for data mining processes. Following a proper data mining methodology for implementation of the proposed data mining techniques can help to identify important steps and the required expertise and tools, and can improve the quality and controllability of the process. Doing so is arguably vital to the ultimate success of the project.

In Azevedo and Santo (2008), two of the most commonly used datamining methodologies are investigated: SEMMA and CRISP-DM. The overview shows that for a knowledge discovery process, CRISP-DM is the most elaborate data mining methodology. For this reason, it was selected as a framework for this research. CRISP-DM stands for CRoss Industry Standard Process for Data Mining (Wirth & Hipp, 2000). This section describes the phases of CRISP-DM (shown in Figure 2.1). The CRISP-DM methodology was applied to structure this research, as discussed under Experiment Design. Specifically, it allowed the contextualization of the proposed approach within a standardized method (see section 4.6, Process Overview).

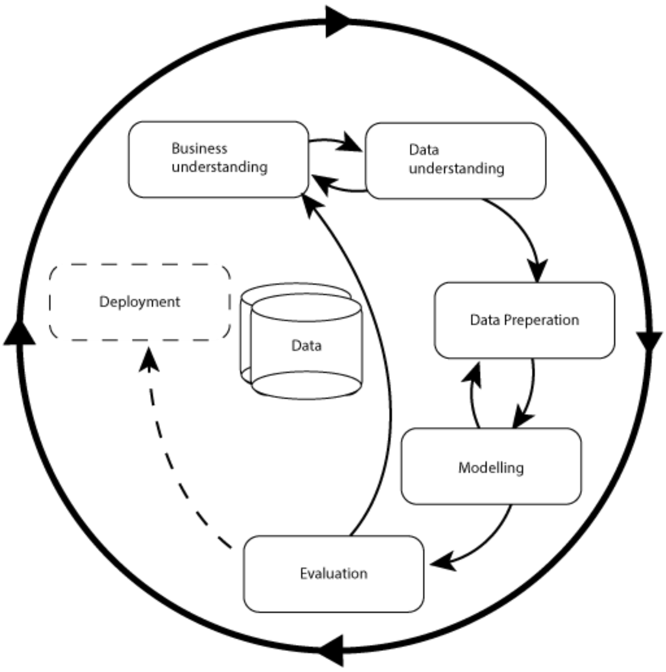


FIGURE 2.1 CRISP-DM (WIRTH & HIPP, 2000)

Business understanding: The initial phase focuses on understanding the objectives, requirements, and a plan to achieve those objectives. Since this methodology is used within a scientific context, domain understanding is more appropriate for the wording than business understanding.

Data understanding: The second phase focuses on collecting, describing and exploring data and verifying data quality.

Data preparation: The third phase focuses on the selection, cleaning, and formatting of the data.

Modelling: The fourth phase focuses on the selection of the modelling technique, building the model and assessing the model.

Evaluation: The fifth phase focuses on the evaluation of the results, a review of the process and determining what steps to take next.

Deployment: The last phase focuses on the deployment in the real world, which is out of the scope of this project and is therefore not elaborated on.

3 Background

3.1 Error analysis in Mathematics Education

Error analysis in mathematics education refers to analysing learner errors. The purpose is to diagnose individual learning difficulties and improve understanding about the mathematical educational process (i.e. learning and teaching students in the mathematics domain). In a literature survey by Radatz (1979), errors in mathematics education were found to be mostly the result or product of previous experience in the mathematics classroom. The systematic, persistent and analysable nature of some of these errors are helpful in automating the diagnosis of such errors. Specifically, student errors:

- are causally determined and often systematic
- are persistent and last for several school years unless there is appropriate intervention
- can be analysed and described as error techniques
- have causes that can be derived through the evidence that learners provide by interacting in the educational environment

Learners are thought to learn mathematics *procedurally* and *conceptually*. Procedural learning focusses on skills and step-by-step instructions, whereas conceptual learning focusses on ideas and generalizations that connect ideas. If procedures are taught without the learner appreciating the mathematical concepts, there is a risk these procedures will interfere with later meaningful learning (Ashlock, 2006). Some learners invent their own computational procedures that can result in erroneous answers; at other times the answers might be correct . Because these procedures sometimes are correct, students and teachers assume that a correct procedure has been implemented. These systematic procedures, which are often erroneous, create *error patterns* that reveal the *misconceptions* that were learned (Ashlock, 2006).

Anderson (1989) distinguished three causes of errors: *slips*, *importations of prior misconceptions into a new domain* and *within-domain misconceptions*. *Slips* can be described as errors that are not reliably made (i.e. random errors) and errors that can be corrected when pointed out (i.e. careless errors). This can be the result of little practice and memory overload when implementing exercises (West, 1971; Anderson & Jeffries 1985; Anderson, 1989; Gowda et al., 2011; Fisher & Frey, 2012). *Prior misconceptions into a new domain* are errors caused by transferring erroneous knowledge from another domain (e.g. physics) to the domain that is investigated (e.g. mathematics). Anderson argued that if learners lack an abundance of conceptions, investigating such errors is redundant. For that reason, *prior misconceptions into a new domain* were disregarded in this research. Finally, *within-domain misconceptions* are misconceptions that do not arise through prior belief but through the learning that takes place in a domain. This research was focussed on within-domain misconceptions that occur in the multiplication domain.

Each individual erroneous procedure can be interesting because of the many reasons learners tend to learn patterns of error. However, literature exists about how misconceptions are learned. Ashlock (2006) defined o*vergeneralization* and *overspecializing* as possible approaches for generating misconceptions. Overgeneralization can be described as jumping to a conclusion before having accurate data. A learner might falsely transfer the knowledge from the concept of multiplying by 1 when multiplying by 0, resulting in a misconception. In this case, it would be 'multiplying by 0 is the same as multiplying by 1'. The latter disregards the actual mathematical concepts of multiplication by 1 and multiplication by 0.

Overspecialization, on the other hand, occurs when learners restrict the resulting procedures inappropriately (e.g. the answer must always end with two decimals). Additionally, Brown and Skow (2016) defined misconception of place value as a conceptual error, where the learner does not understand place value and answers in the wrong place-value position. Finding out the cause of erroneous knowledge transfer can help to construct a correct learner model, and affects the implications for instruction.

For completeness sake, gaming the system should be mentioned as a possibility that yields an error response. Gaming the system refers to the exploitation of a system's properties for help and feedback, instead of attempting to learn the proposed material (Baker et al., 2008).

3.2 Elicitation, diagnosis and the cause of systematic errors

As noted by Guzman et al. (2010, p.245), '*Most authors have focused on misconception diagnosis rather than on its elicitation, leaving this issue to the domain experts'.* Distinguishing between misconception *elicitation* and *diagnosis* is important because of the difference between process and goal. In misconception *diagnosis*, the goal is to identify how learners learn procedural skills through subskills. Snapshots are taken of learners who are solving a problem, to acquire a relatively complete learner model. This process is illustrated in the cornerstone paper 'Repair Theory: A Generative Theory of Bugs in Procedural Skills' by Brown and van Lehn (1980). It goes as follows:

generative theory of bugs → bugs → systematic errors

In the above notation, the arrow (→) means 'explains'. The *generative theory* (i.e. a set of formal principles) is set up to understand what the cause of bugs is, based on bug 'stories' – which are informal explanations of bugs that occur when performing tasks. Bug stories could lead to a generative theory of bugs (e.g. bug stories → generative theory of bugs). After the generative theory is set up, bugs are generated in a bug library. Finally, the generated bugs are validated through systematic errors in the data. Doing so enables insight into what and why certain bugs occur and not others. In a similar fashion, an illustration can be made to explain the process of misconception *elicitation:*

systematic errors → bugs → bug stories

As can be seen, in the process of misconception elicitation, the goal is not to automate the identification of causes for misconception, but rather to automate a part of the process that domain experts need to perform to create informal bug stories. These bug stories can then be validated through misconception diagnosis (which was not part of this research). Although the goal of misconception elicitation is not to find the cause of misconceptions, one hypothesis in this paper is that the thorough investigation of systematic errors can give some explanation of the cause of systematic errors.

The *repair theory* that was discussed by Brown and van Lehn (1980) and van Lehn (1983) is important for understanding how systematic errors occur. These authors argued that learners in the midst of solving a problem reach an *impasse*, a situation where they do not have the right knowledge to solve the problem correctly. At this moment, learners become inventive and try to *repair* the procedure instead of applying the correct one. The authors stated that bugs can thus often be best explained as 'patches'.

In this research, the sub-steps learners take to reach an erroneous answer are not available; therefore, the process of misconception elicitation can at most give an informal explanation about the underlying cause. However, as Ben-Zeev (1998) noted, not all errors can be accredited to learners encountering an impasse. Instead, methods like the repair method are overly focused on errors that occur during execution and do not focus enough on the learning acquisition phase. For example, learners sometimes transfer knowledge erroneously through real-life analogies (Graeber, 1993). This research did not include data on the learning acquisition phase and was focussed only on within-domain misconceptions.

3.3 Misconception elicitation in the Educational Data mining domain

As mentioned in the problem statement, considering overlay models alone limits the possibility of revealing learners' misconceptions. The bug libraries that were introduced by Burton and Van Lehn (Brown & Burton, 1978; Brown & Van Lehn, 1980) were a leap forward in terms of automated misconception diagnosis. However, bug libraries remain difficult to set up by hand and are exhaustive; that is, unanticipated erroneous learning behaviour cannot be included in bug libraries. As a response to this problem, several attempts were made to extend bug libraries. The most notable attempt was by Sleeman et al. (1990), where two rule-based algorithms INFER* and MALGEN were implemented to elicit new rules, with and without student intervention, respectively. The latter algorithms both needed active participation of experts to decide whether the inferred rule was appropriate for the bug library.

The first algorithm that was less reliant on experts was proposed by Baffes and Mooney (1996), using examples of students' behaviour as input. Rules were modified until the behaviour was explained. More recently, Guzman et al. (2010) proposed a semi-automatic misconception discovery method by *association rule mining* erroneous response trajectories. Although their technique helps teachers to discover misconceptions, the authors did not elaborate on how to deal with the abundance of rules or how to interpret the discovered rules effectively.

Two research projects that do not fall under misconception elicitation but are interesting to mention are Buwalda et al. (2016) and Savi et al. (2018). Buwalda et al. (2016) developed a cognitive model that gives a comprehensible account of the errors made in single-digit multiplication problems, using the same dataset that was used in this thesis. Although the model elaborates common mistakes made in individual questions, it is an interesting account of explaining multiplication errors that should be considered. Second, Savi et al. (2018) proposed an approach to investigate the automation of diagnosing misconceptions in single-digit multiplication. The proposed method exploited known theoretical relations between misconceptions and errors as an indication of the probability of a misconception.

4 Experimental Design

4.1 Math garden: web-based platform for adaptive training of elementary school subject

This research is based on response trajectories collected in Math Garden, an OLE where children can practise arithmetic exercises (see Figure 4.1). Math Garden was initiated in 2007 and was designed to freely capture long and dense time-series data for cognitive development studies and mathematical development. Since its initiation, it has been extended to other domains, such as language learning, statistics and typing, using various games (Brinkhuis et al., 2018). This research was interested in the response logs collected in the games where multiplication is exercised.

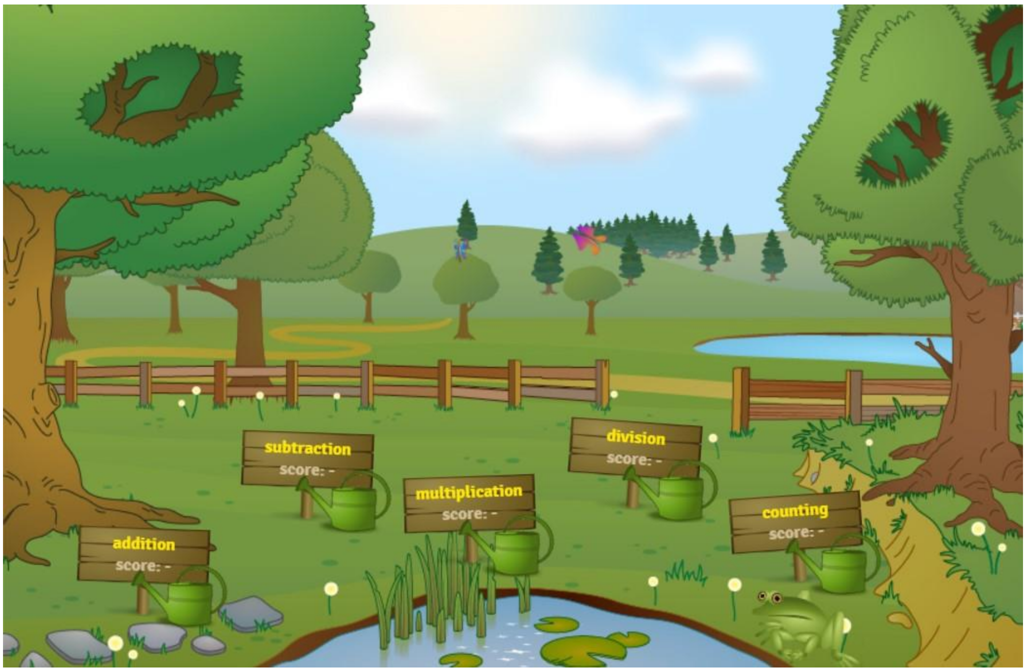


FIGURE 4.1 MATH GARDEN DOMAINS (MATH GARDEN, 2018)

As shown in Figure 4.2, children are presented with a question or item. The wellbeing of the plant depends on the frequency of exercise in a domain, with each domain having its own plant. Children respond in an open format. Math Garden tries to match children with items at an appropriate level of difficulty. Hence, children are shown easier items if they cannot answer the question within a certain timeframe, or answer incorrectly but slowly, and they are served more difficult items if they answer the question quickly and correctly. Math Garden's adaptive item selection focuses on facilitating learning and motivation rather than improving measurement precision, by considering the preferred difficulty level and response history (Brinkhuis et al., 2018).

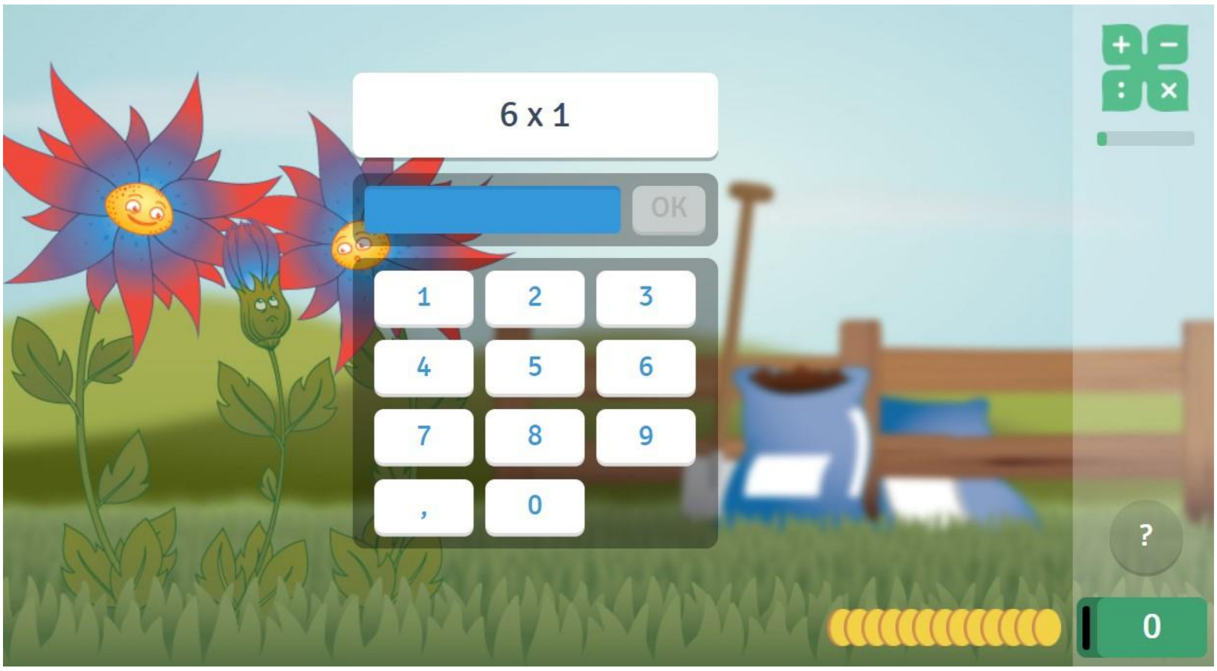


Figure 4.2 Math garden multiplication (Math Garden, 2018)

Math Garden applies two psychometric innovations, namely *scoring rules* and *adaptive item selection*. *Scoring rules* are applied to track and control the progress of learners. The *scoring rules* in Math Garden incorporate the correctness of the response and the time taken to respond. These features discourage guessing and enforce a speed–accuracy trade-off (Wickelgren, 1977; Maris & van der Maas, 2012). Each second that the game lasts, a coin is lost. When a learner answers correctly, they gain the remaining coins, which can be used to buy virtual prizes. Incorrect answers result in coins being subtracted. Children can answer with a question mark when they do not know the answer, and for a question-mark response they do not lose coins. However, to prevent gaming of the system, the use of question marks is restricted (Brinkhuis et al., 2018).

Adaptive item selection estimates the multiplication performance of the learner and selects an item based on that assessment. This method is based on Elo ratings, which originated in the chess community. Elo ratings where vital to dynamically assess a chess player's performance (Elo, 1978). In maths learning, it is used to predict the performance rating of a learner while considering the ratings of all participating learners. Because the Elo rating method allows many contestants to be compared in a dynamic environment, it is suitable for an OLE. Also, in OLEs, children's abilities and item difficulties change dynamically over time. Using the Elo method means that items and a learner's ability can be assessed and enables items to be adaptively selected (Brinkhuis et al., 2018).

4.2 Evidence model

Guzmán et al. (2010) proposed a technique for misconception inference, in which concepts, tasks and misconceptions were included in an evidence-centred design (ECD) model. This ECD model is based on the learner modelling framework built for assessment tasks by Mislevy et al. (2003). The ECD model consists of three layers: *the misconception layer*, *the task model* and *the concept layer* (Figure 4.3). According to Guzmán et al. (2010, p. 249), *'ECD models incorporate representations of what a learner knows and does not know, in terms of the results of his/her interaction performance (evidence) with assessment tasks'*. In other words, a learner answering tasks creates an opportunity to investigate the learner's knowledge state about performing a specific task. This model was used to define the relevant constructs and was extended with a *relationship component*.

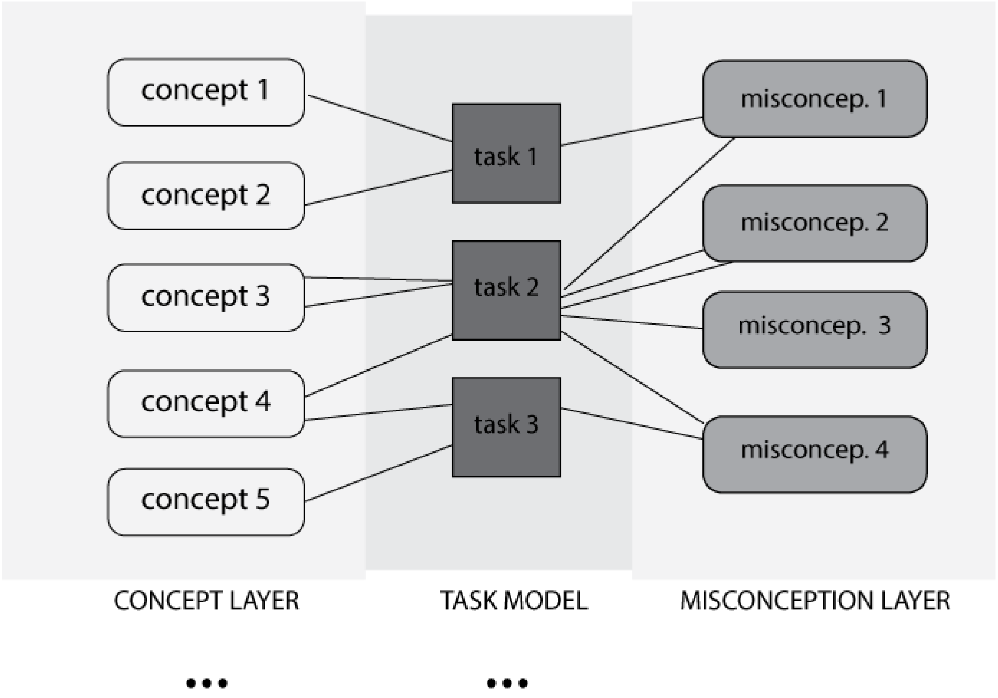


FIGURE 4.3 ECD MODEL (GUZMAN ET AL., 2010; MISLEVY, 2003)

The *misconception layer* consists of misconceptions that are made apparent through patterns of *systematic errors* which can be defined as ‘*a repeatedly occurring incorrect response that is evident in a specific algorithmic computation*’ (Cox, 1974, p.3). An example of a systematic error caused by a misconception in arithmetic multiplication exercises by children is that they do not multiply and place one of the multiplicands in the answer (Attisha & Yazdani; Cox, 1974). Additionally, literature state that three to five errors on a particular type of problem (e.g. multi-digit multiplication) indicate an error pattern (Brown et al., 2016; Howell, Fox, & Morehead, 1993; Radatz, 1979). An effective way to identify systematic error patterns is *association rule mining,* which is described in the next section.

The *task model* represents any task that could expose a learner’s knowledge state. Here concepts are linked to tasks (i.e. questions) and possible typical misconceptions, and their underlying concepts. The links are illustrated by black lines. The main hypothesis is that typical misconceptions responsible for erroneous learning behaviour can be found through the automated investigation of response logs. These logs contain the results of tasks the children have performed.

In addition to examining typical misconceptions, this research considers concepts to assess the knowledge state of a learner. This refers to *the concept layer*. Concepts could indicate the knowledge a learner possesses or the lack of knowledge. Identifying what the causes are, and concepts related to these causes, can give insight into what has sparked the erroneous knowledge transfer. The concept layer was derived through manual investigation and was not part of the approach.

Finally, the relationship components are shown by the dashed lines in Figure 4.3. These illustrate the possible erroneous knowledge derived from a concept that can cause a misconception to arise.

4.3 Association Rule Mining

An effective datamining technique for finding frequent patterns of erroneous learning behaviour, and the possible associations between these patterns, is association rule mining (Dogan & Camurcu, 2008). Association rule mining reveals how many times item X occurs with item Y in an item set (i.e. the support of an item set). It also determines how likely it is that item X occurs with item Y in an item set (i.e. the confidence of an item set).[1] This is formally written as

$$\text{Support} = \frac{freq(X,Y)}{N} \qquad Confidence = \frac{freq(X,Y)}{freq(X)}$$

where ***freq*** refers to frequency. Items are created by concatenating the question with an answer. For example, '5x2' and '5' become '5x2=5'. Table 4.1 shows an example of a simplified dataset with response trajectories from four learners on four items.

TABLE 4.1 EXAMPLE DATASET

Response trajectories	**5x2 =5**	**2x3=2**	**4x2=4**	**4x5=4**
1	0	1	1	1
2	1	1	1	0
3	1	1	1	0
4	0	1	0	0

Item 5x2=5 occurs twice with 2x3=2 and 4x2=4, so the support is 2/4 = 50%. Item 4x5=4 occurs once with 2x3=2 and 4x2=4, so the confidence is 0.25/0.75=33.3%. These thresholds allow for the identification of meaningful association rules – or more specifically, frequent patterns of erroneous learning behaviour that might indicate misconceptions (Table 4.2).

TABLE 4.2 EXAMPLE ASSOCIATION RULES

Association Rules
Item 1 → Item 2, item 3
Item 3 → Item 4, Item 5, Item 6
Item X → Item Y

In addition to support and confidence, another measure that might be important is lift. The lift is the *confidence* of a rule divided by the probability that a response trajectory contains X, the expected confidence. More formally,

$$Lift = \frac{Confidence}{Expected\ Confidence}$$

This is an indication of the probability that a response trajectory that contains X also contains Y, while considering the independence between X and Y. In other words, lift is a performance measure that incorporates how likely it is that the item will occur in the dataset. For example, if an item occurs only a few times in the dataset, it has a relatively high lift because it has a low expected confidence. A lift of 1 implies no association between X and Y; a lift greater than 1 implies a likely association between X and Y; and a lift smaller than 1 implies an unlikely association between X and Y.

[1] From now on, items refer to 'question + answer' instead of only 'question'.

The found association rules give an indication of the relationship between the investigated items. They can help to distinguish systematic errors from random and careless errors, and can indicate patterns of erroneous learning behaviour from which typical misconceptions might be derived. The support and confidence enforce significance and interestingness of the patterns of erroneous learning behaviour.

4.3.1 Association rule mining algorithm

The most commonly known algorithm is the Apriori algorithm. This algorithm is relatively simple and can be described in two main steps. First, all frequent item sets must be found with a minimum support threshold. This is done iteratively in a so-called 'level-wise search'. Here a frequent item set is any set that occurs *n* times. First, 1-item frequent item sets are found; second, 2-item sets are found, and so on. In the second step the frequent item sets are used to generate association rules. Association rules are generated if they meet the set support and confidence threshold. Because association rule mining regularly generates many – that is, too many – results, it is important to set the right confidence and support threshold to find interesting patterns (Agrawal & Srikant, 1994). In this research, the *Pymining* library was applied (Dagenais, 2015).

4.4 Network analysis

A collection of rules can be seen as a *network*. Here items can be considered as *nodes* and the connections between items as *edges*. This is helpful because it allows insight into the vast amount of rules that association mining typically produces. The following rules serve as an example:

1) A,B → C
2) C,D → E
3) B,C → D
4) H,I → J

Rules 1, 2 and 3 are connected as a network through items C and E. Rule 4 does not contain any items that rule 1, 2 and 3 possess, and is thus not connected (Figure 4.4). Visualizing rules as a graph can help to identify communality between the items and separate items that are not connected in any way. However, simply connecting rules that are syntactically similar does not mean network visualizations will provide insight. First, syntactically similar items might not be semantically similar. For example, item 0 x 1 = 1 could be the result of learners applying multiplication-by-1 to multiplication-by-0 questions. It could also be the result of learners adding 0 to 1, or could be the result of transferring the multiplicand 1. When semantically different rules are connected based on syntactically similar items, the explicability of a group can become clouded quickly.

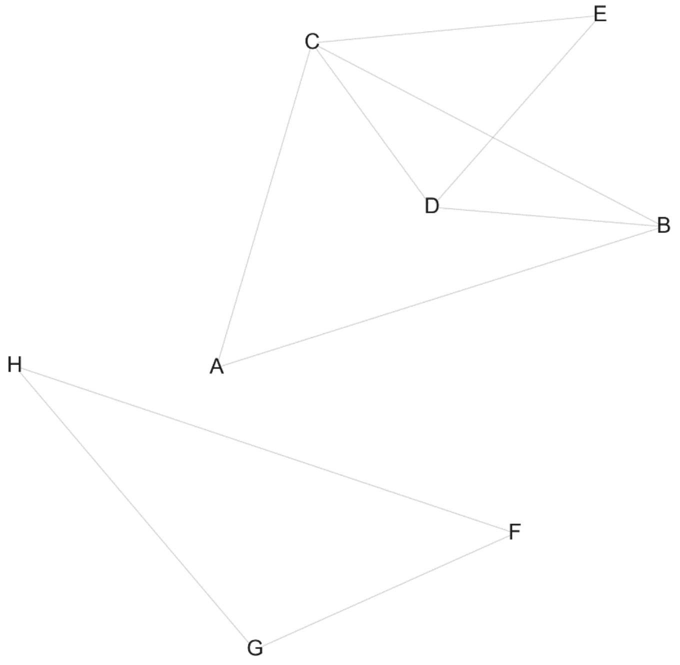


FIGURE 4.4 EXAMPLE NETWORK OF RULES

Second, rules do not guarantee that items occur together. Any redundant item in a faulty rule can be the cause for smaller meaningful groups to be merged into larger inexplicable groups. The confidence and lift thresholds can be set to prevent this. However, as explained in the following sections, doing so does not completely solve these challenges.

Finally, it is important to set some definitions to be able to discuss the network analysis in a specific manner. A *subgraph* is a subset of nodes in a network, with the edges linking these nodes. Any group of nodes can form a subgraph. A *component subgraph* is a portion of the network that is disconnected from another portion in the network (Tsvetovat & Kouznetsov, 2011). For example, Figure 4.4 shows two subgraphs, more specifically two component subgraphs.

In the next section, community detection is discussed. This approach counters some of the challenges by enabling the discovery of relevant groups within a larger network.

4.4.1 Community detection

To understand how and why items relate to each other in a network of rules, *community detection* is applied. *Community detection* is the process of discovering strongly connected groups within a larger network. In real-world networks, many examples of *communities* can be found around shared interests or themes; for example, communities may form around discussed topics in social networks, or voter communities around election candidates. The easy interpretability of these communities can be advantageous for misconception elicitation. Items that seem to be more connected with some items than others form communities of items, which allows the expert to interpret items and rules in a grouped manner.

Many community detection algorithms are set up to optimize for *modularity.* This is a metric that can be used to measure the quality of the cluster in terms of interconnectedness (Blondel et al. 2008; Clauset et al. 2004; Newman,2004;2006). It is formally written as

$$M_c = \sum_{c=1}^{n_c} \left[\frac{L_c}{L} - \left(\frac{k_c}{2L} \right)^2 \right]$$

where $\boldsymbol{n_c}$ represents the number of communities, $\boldsymbol{L_c}$ the number of links within a community, $\boldsymbol{k_c}$ the total degree of nodes in a community, and $\boldsymbol{L}$ the total links in the network. The higher the modularity, the more optimal the clusters are. The modularity of a network is depicted in Figures 4.5 and 4.6.

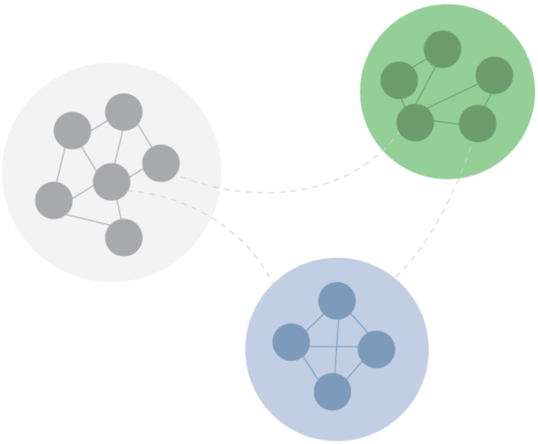


FIGURE 4.5 OPTIMAL CLUSTERS

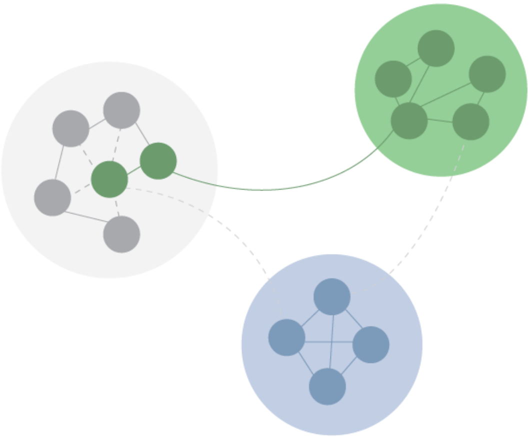


FIGURE 4.6 SUBOPTIMAL CLUSTERS

In Figure 4.5, the calculated $M_c = \left[\frac{9}{24} - \left(\frac{20}{48}\right)^2\right] + \left[\frac{6}{24} - \left(\frac{14}{48}\right)^2\right] + \left[\frac{6}{24} - \left(\frac{12}{48}\right)^2\right] \approx 0.55$. By contrast, in Figure 4.6, $M_c = \left[\frac{4}{24} - \left(\frac{11}{48}\right)^2\right] + \left[\frac{8}{24} - \left(\frac{21}{48}\right)^2\right] + \left[\frac{6}{24} - \left(\frac{12}{48}\right)^2\right] \approx 0.44$. In other words, the more disconnected the communities are, the higher the modularity.

In this research, the network of rules was treated as an undirected graph (such as in Figure 4.4). Hence, rules were treated as items that were likely to occur together. The *if A then B* relationship was disregarded to simplify the final visualizations for the expert. A greedy modularity-maximization algorithm, the Louvain algorithm implemented in NetworkX library, was used (Blondel et al., 2008; Aynaud, 2009). Legara (2016) presented a broader practical introduction of community structures in NetworkX. The next section discusses how association rule mining and community detection can further help the automation of the misconception elicitation process.

4.4.2 Approach towards Misconception Elicitation

The properties of a network have important implications for how the analysis is conducted and the results are interpreted. Examples are citation networks and product networks. In citation networks there clearly is a reason why a paper is cited, but in product networks people could be buying two unrelated products, causing many chance associations (Raeder & Chawla, 2011). In the case of children answering arithmetic exercises, some items are ambiguous but syntactically similar; in addition, redundant items may occur. The challenges in analysing these items must be targeted by an approach that takes these types of properties into account.

Apart from the challenges that come with analysis, interpretability of the results is also important. The main goal is to present results that are interpretable for an expert in two ways: 1) in *terms of numbers* (i.e. too many results decrease the effectiveness of the tool in terms of time spent analysing); and 2) in *terms of importance* (i.e. the approach must show error patterns that can indicate misconceptions). The following approach was set up to deal with these challenges and goals.

In an ideal situation, support and confidence thresholds are raised to a 'high' level. The precise meaning of 'high' is not examined here. This results in a limited set of rules with items that are highly likely to occur together. Applying community detection in this scenario is effective because of the disconnectedness of the rules. In such a scenario, the resulting communities are highly explicable. Unfortunately, this scenario will not result in many items that can be investigated, because some error patterns occur relatively often or are likely to occur together. These variations can be linked to several factors, such as the number of times a learner views a specific set of questions, the difficulty of the procedure, and the probability of the learner using an erroneous strategy. These variations have been shown to be advantageous in distinguishing the semantics of syntactically similar items.

To find new items in rules, the support threshold must be lowered from the 'high'-level ideal. Doing so leads to the following outcomes: 1) more *interesting* rules appear with *new items*; and 2) more *redundant* rules appear with *already discovered items.* Since redundant rules with already discovered items raise the interconnectedness of the network, they make it harder to find explicable communities. Therefore, it is better to prune these redundant rules altogether. Before lowering the thresholds, rules that contain syntactically similar items should be pruned to enable more interesting rules with new items and to prevent redundant rules with already discovered items. This process can be repeated until minimal support and confidence thresholds are met. To determine the measures by which the thresholds should be lowered, the quality of the cluster (i.e. the modularity) should be considered. These steps can be formalised in an algorithm which has the following steps:

1) Set minimum support and confidence thresholds.
2) Mine rules based on minimum support and confidence thresholds.
3) Set *number of steps* and *maximum number of allowed nodes in a cluster*.
4) Determine ***isup*** and ***icon*** (i.e. initial support and confidence thresholds).
5) Determine ***X*** and ***Y*** (i.e. the number by which the support and confidence thresholds decrease in each iteration).
6) Iterate.
    a. Lower support and confidence thresholds with ***X*** and ***Y*** based on the ***step*** number.
    b. Detect optimal communities.
    c. Return *modularity*, number of detected communities, rules, components and unique items.
    d. Prune rules that contain items from last iteration.
    e. If minimum set *support* and *confidence* threshold are met, go to (7). If not, go to (6).

7) Calculate *average modularity*, total number of detected communities, rules, component subgraphs and unique items per step.
8) Repeat from (5) until average modularity no longer changes significantly.
9) Select optimal step based on *highest average modularity (k-core=3).*
10) If a community in an iteration has more than a *maximum number of allowed nodes in a cluster,* repeat from (4) but replace *isup* and *icon* with the support and confidence thresholds of that iteration. If not, stop.

The proposed algorithm implies that are four parameters that must be set: the *minimal* support and confidence thresholds, the *initial* support and confidence thresholds, the *number of steps*, and *the maximum number of nodes in a community*. There are various ways to approach setting the minimum support and confidence thresholds, and they depend on the preferences of the expert. If the expert is not specifically interested in how often a rule must be present in the data, an arbitrary number can be chosen and lowered until the number of rules becomes unmanageable. Alternatively, the expert can select a support threshold with a specific number based on other preferences (Raeder & Chawla, 2011).

Determining the *initial* support and confidence is slightly different from determining the *minimum* support and confidence thresholds. When deciding the ideal *icon* and *isup*, the goal is to find the most explicable communities in terms of likeliness and semantics, and to disregard the diversity of items. This can be done by raising the confidence to an interpretable number of rules, and selecting the accompanied support threshold that optimizes for modularity so that the rules are optimally clustered. The *isup and icon* values are determined by finding the ideal values for both support and confidence. Here more interesting rules with new items can be found by lowering the thresholds over two dimensions, *X* and *Y*. This is formally written as

$$X(isup, step) = \frac{isup - minsup}{step}$$

$$Y(icon, step) = \frac{icon - mincon}{step}$$

where ***isup*** and ***icon*** refer to the initial support and confidence, ***minsup*** and ***mincon*** refer to the minimum support and confidence thresholds, ***step*** is the step number, and ***X*** and ***Y*** are the decrease of the support threshold per iteration. The step number increases by 1 each step, starting at 1. The number of iterations per step is equal to the step number (e.g. step 1 has 1 iteration, step 2 has 2 iterations, step *n* has *n* iterations). In other words, in one step the algorithm iterates and decreases *X* and *Y* until *minsup* and *mincon* are met. In this approach, decreasing the *lift* is not interesting, since the lift is applied only to find the significance of the mined rule (i.e. the rule is likely not found coincidentally).

At the end of each iteration, the modularity of each iteration is summed with the modularity of the last step. At the end of each step, the total sum of modularity measures is divided by the number of iterations that were performed in that step (i.e. calculating the *average modularity*). This value indicates what step number obtained the highest quality of communities on average. The *number of steps* can be set to an arbitrary number that allows the algorithm to run until the average modularity no longer changes significantly.

The modularity increases with a higher step number, such that a higher step number may mean fewer rules, sparser communities, and higher modularity. This scenario can give a false indication of quality. However, if no rules are found in an iteration, the modularity for that iteration is 0, which lowers the average modularity score satisfactorily. This penalty ensures the quality of the overall metric.

Another counter-measure to prevent a false indication of quality is the implementation of *k-core networks* before calculating the modularity of the network. Implementing k-core networks prevents rules that are not connected to any other rules to be considered as communities, by generating a network for which all nodes have at least *k* edges. Although nodes with edges that are smaller than k can be interesting, this approach is set up to find the best set of groups of rules. Therefore, communities that are found with smaller than *k* edges are investigated but not considered when calculating the quality of groups. This makes communities of rules more explicable and minimizes the number of redundant rules.

To ensure that the found communities in each iteration are interpretable and are not too large, the number of nodes is limited a *maximum number of allowed nodes in a cluster*. If one of the communities seems larger than this number, the proposed algorithm is repeated with the support and confidence of that iteration replacing *isup* and *icon*. The latter step is also implemented to satisfy the goals in terms of numbers and importance.

In summary, the surplus of items, rules and clusters of rules are reduced to an interpretable amount, which is relevant to the proposed research questions. Once the communities are detected, they can be visualized in a similar fashion as that shown in Figure 4.4.

4.4.3 Ego networks

To add more granularity to the approach, an additional type of network visualization is proposed, the *ego network*. An ego network revolves around a single node (Figure 4.7). In social network analysis, the goal is to find the most important actor (Everett & Borgatti, 2005). Normally, egos are determined based on the *betweenness centrality*, the node with the shortest path to all other nodes. In this approach the ego was determined based on *degree centrality*, the number of edges that enter or leave a node. The expert can decide whether it is interesting to see the node with the highest or lowest degree centrality. An expert could find the largest hub if interested in the most interlinked node in the community (illustrated in Figure 4.7), or could find the node with the least nodes to enable insight in edge cases.

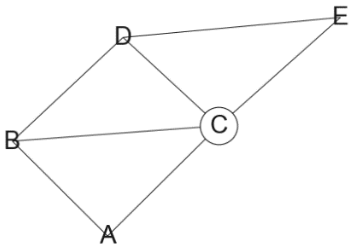


FIGURE 4.5 EXAMPLE OF EGO NETWORK

After an ego is selected, an expert can interactively change the support and confidence thresholds without pruning any rules. Doing so will give a more complete picture around one item. In the previously discussed method, rules below a certain support and confidence threshold were pruned and were thus not included in the final community visualizations.

4.5 Expert analysis

The final community visualizations were interpreted based on a literature study. In isolation, the error patterns did not answer RQ3 and RQ4 as proposed in this research. In a case study by Brown et al. (2016), manual instructions were given to conduct proper error analysis in the mathematics domain. The steps were similar to those mentioned in an introduction by West (1971):

1) Collect data – at least 3 to 5 errors to the same type of problem
2) Identify error patterns
3) Determine reasons for errors
4) Use the data to intervene

In the previous stages of this thesis, this approach was used with automation of steps 1 and 2. However, to understand what the reason is for an error pattern when learners solve mathematical problems, error types must first be understood. Brown et al. (2016) distinguished three error types:

1. Factual errors: caused by lack of factual information
2. Procedural errors: caused by incorrect performance steps
3. Conceptual errors: caused by faulty understanding of principles and ideas (i.e. misconceptions)

These error types are all related to lack of knowledge or misunderstanding (Brown et al., 2016; Fisher & Frey, 2012; Riccomini, 2014). Brown et al. (2016) illustrated the error types as follows in Table 4.3, 4.4 and 4.5.

TABLE 4.3 FACTUAL ERRORS

Factual errors	Examples
Has not mastered basic number facts	3 + 2 = 7, 2 x 3 = 7, 7 – 4 =, 8 / 4 = 3.
Misidentifies signs	2 x 3 = 5 (sees a multiplication sign as an addition sign), 8 / 4 = 4 (sees a division sign as a minus sign).
Misidentifies digits	The student identifies a 5 as a 2.
Makes counting errors	1,2,3,5 (the student skips 4).
Does not know mathematical terms	The student does not know what multiplication means.
Does not know mathematical formulas	The student does not know mathematical formulas (e.g. 2πr).

TABLE 4.4 PROCEDURAL ERRORS

Procedural errors	Examples	
Regrouping errors		
Forgetting to regroup (carry): the student forgets to regroup (carry) when multiplying	56 x 2 102	After multiplying 2 x 6, the student fails to regroup one group of 10 from the tens column.
Regrouping across a 0: When a problem contains one or more 0s in the minuend (top number), the student is unsure what to do	304 - 21 323	After multiplying 2 x 6, the student fails to regroup one group of 10 from the tens column.
Performing an incorrect operation: Although students understand operators, they often apply them incorrectly (e.g. multiplying instead of adding)	3 x 2 5	The student added instead of multiplied.
Fraction errors		
Failure to find common denominator when adding and subtracting fractions	3/4 + 1/3 = 1/7	The student added the nominators and then the denominators, without finding the common denominator.
Failure to invert and then multiply when dividing fractions	½ / 2 = ½ * 2	The student did not invert the 2.
Failure to change the denominator in multiplying fractions	2/8 * 5/8 = 10/8	The student did not multiply the denominators.
Incorrectly converting a mixed number to an improper fraction	2 *1½ = 4/2	To find the numerator, the student added 2 + 1 + 1 to get 4.

Decimal errors		
Not aligning decimal points when adding or subtracting	120.4 +63.21 75.25	The student did not align decimal points.
Not placing decimal in appropriate place when multiplying or dividing *Note: This could also be a conceptual error related to place value, as described in Table 4.5*	3.4 x .2 6.8	The student aligns the decimal point in the product with the decimal point factors.

Table 4.5 Conceptual errors

Conceptual errors	**Examples**	
Misunderstanding of place value: The student does not understand place value, and records the answer so that numbers have inappropriate place values	67 + 4 17	The student added all the numbers together.
	10 + 9 91	The student recorded the answer with the number reversed.
	Write the following as a number a) Seventy-six b) Nine hundred seventy-four c) Six-thousand, six hundred, twenty-four Student answer: a) 76 b) 90074 c) 600060024	The student does not have the conceptual understanding of place value position for values beyond two digits.
Overgeneralization: Because of a lack of understanding, the student incorrectly applies rules or knowledge to novel situations	321 + 245 124	The student subtracts the number that is less than the greater number.
	Place in the right order from smallest to largest 77/486, 1/351, 12/200	The student does not understand the relation between the numerator and denominator.
Overspecialization: Because of a lack of understanding, the student develops an overly narrow definition of a given concept, or of when to apply a rule or algorithm	Which of the shapes are triangles? a) ◢ b) ◥ c) both Answer a)	The student chooses 'a' just because of the orientation.

To determine why a student makes a particular error, Brown et al. (2016) proposed the following strategies:

1) Interview the student
2) Observe the student
3) Look for an exception to the error pattern

Interviewing is not applicable to large-scale OLEs, and observing the student is automated in this approach. Looking for an exception to the error pattern can be highly informative, since it can indicate whether a student has partial or basic understanding of the concept in question. When interpreting the final plots, exceptions are kept in mind.

It must be noted that procedural and conceptual knowledge often overlap and are thus hard to distinguish (Brown et al., 2016; Rittle-Johnson et al., 2001; Riccomini, 2014). Also, procedural errors are most common and are therefore the most likely to be found (Brown et al., 2016; Riccomini, 2014).

Accuracy

To find out how accurate the proposed approach was, the *precision* and *recall* were calculated. The precision allows calculating how many of the found cases are relevant (i.e. factual, procedural or conceptual errors). Formally this is written as

$$Precision = \frac{TP}{TP + TN}$$

where ***TP*** stands for *true positives* the correct found cases (i.e. are explicable misconceptions)*,* and ***TN*** stands for *true negatives* the incorrectly predicted cases (i.e. procedural, factual errors).

Recall allows to calculate how complete the results are, more formally written as

$$Recall = \frac{TP}{TP + FN}$$

where ***FN*** stands for *false negatives* for the incorrect cases that this approach (correctly) disregards (Powers, 2011). Because the rules that are pruned are redundant, they will not form sensible groups (i.e. communities that indicate misconceptions, factual or procedural errors). Therefore, calculating the recall is more appropriate by using the number of unique items in the discovered communities (***TP***) and the number of unique items that are pruned away (***FN***).

4.6 Process Overview

Based on the CRISP-DM model, a process overview is created of the steps that are needed to investigate the research questions in a rigorous manner (see Figure 4.8). First, the data are selected and described, and *popular errors* are identified to understand what systematic errors might arise. Popular errors are errors that are common but not necessarily part of a pattern. Second, data are cleaned and prepared for association rule mining. This results in rules of erroneous responses. Finally, network analysis is performed to enable interpretation of the rules. After the network analysis, the expert can – based on the insights gained – continue investigating in an interactive manner. Finally, the expert analysis is performed while considering the concepts that occur in the arithmetic exercises.

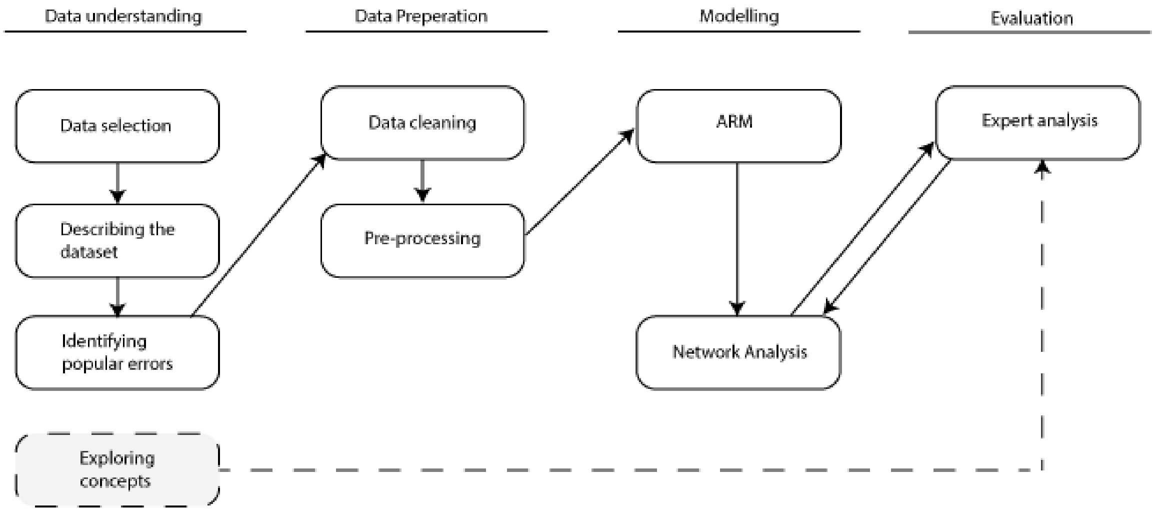


FIGURE 4.6 PROCESS OVERVIEW

5 Data understanding

5.1 Describing the dataset

The dataset consisted of six relevant columns, as shown and described in Table 5.1. The data set was obtained on 19 March 2018, and contained 10,440,954 responses, 115,044 user IDs and 1,232 questions. Assuming the data were aggregated in a similar fashion as in Buwalda et al. (2016), learners were school children between 5 and 13 years. Among all responses, 73% were correct and 27% incorrect. Some questions occurred more than others, as shown in Figure 5.1 – a point that should be considered. The occurrence of a question is shown on the y-axis and the specific questions are mapped on the x-axis. The labels on the x-axis removed as they were visually indistinguishable.

Apart from giving numerical answers, users have the option to respond with a question mark if they do not know how to answer the question. This is indicated by an inverted question mark (¿). If a user did not answer a question within the set time period, the answer response is indicated by an ellipsis (…). The top three error responses were ¿ (29%), … (17%), and 1 (1%).

TABLE 5.1 MATH GARDEN DATA DESCRIPTION

Column name	Description	Type	Example
user_id	Unique user	Continuous	1
correct_answered	Correct or erroneous response	Binary	0 or 1
difficulty	Difficulty of item	Continuous	0,1,2
created	Time relative from the previous item	Continuous	2980353
answer	Specific answer	Continuous	5, ¿, …
question	Specific question	Continuous	1 x 2

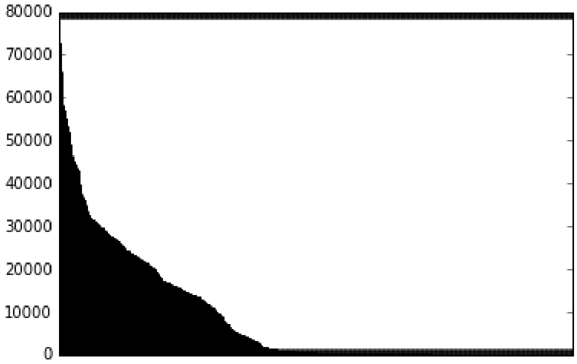


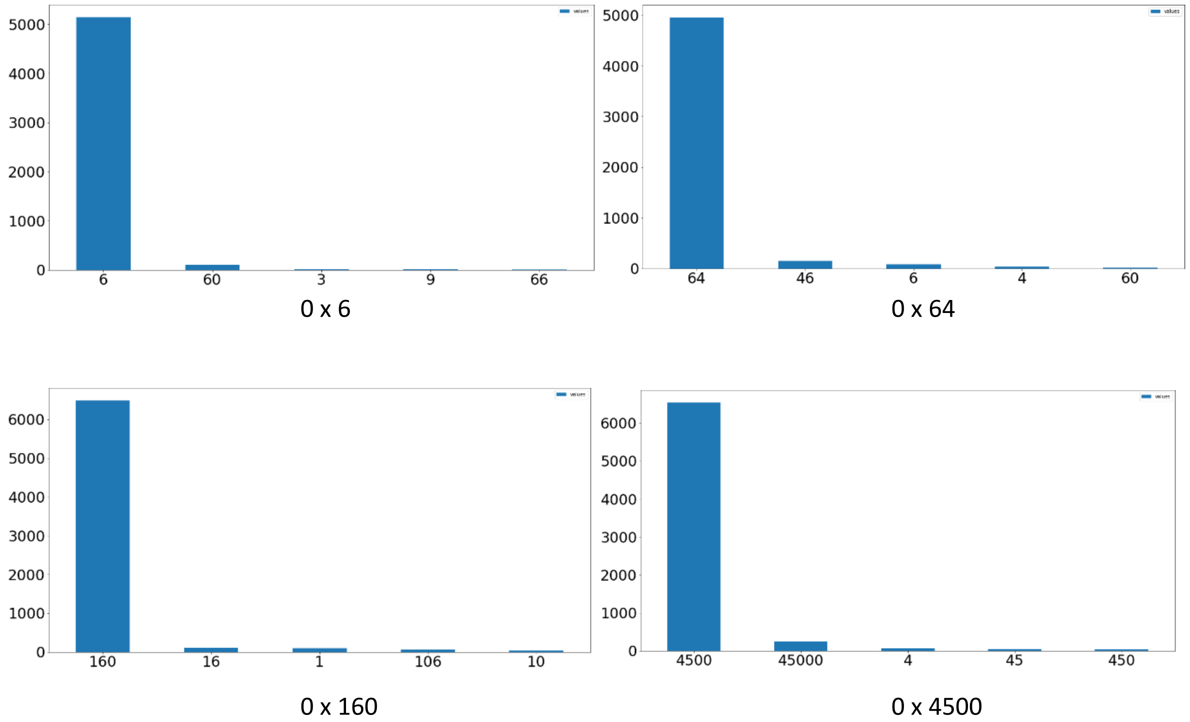
FIGURE 5.1 FREQUENCY OF UNIQUE QUESTIONS

5.2 Building the ground truth

In this section, popular errors are investigated. By investigating popular errors, hypotheses can be set up about what systematic error patterns might arise and thus which typical misconceptions might occur. The section that follows also explores the ground truth for the concepts that might occur. A data exploration is performed to enable hypothesizing about relationships between questions and item responses.

5.2.1 Popular Errors

To understand what systematic erroneous responses might arise, popular errors across many questions must be investigated. In this data exploration, popular errors are defined as the largest proportion of error responses per question. Specifically, this refers to responses that account for more than 30% of the number of error responses per question. For example, Figure 5.2 shows the top five erroneous responses to 0 x 6, 0 x 64, 0 x 160 and 0 x 4500. Responses that did not indicate anything about the nature of the error were excluded. These were answers that indicated either that a learner did not know the answer or that a learner had not answered the question within the allocated period.



0 x 6

0 x 64

0 x 160

0 x 4500

FIGURE 5.2 EXAMPLES OF TOP FIVE POPULAR ERROR RESPONSES

From the 378 questions, 151 questions had an error response that accounted for 30% of all error responses. By plotting the top five error responses when such a popular error arises, and by investigating those error responses, types of popular errors might be identified that indicate systematic errors. Table 5.2 describes the most popular error responses established through manual investigation.

TABLE 5.2 TYPES OF POPULAR ERRORS

ID	Description	Example	Occurrence
PE1	For questions that multiply by 0, the multiplicand that is not 0 is used as an answer	0 x 6 = 6	9
PE2	The decimal point is ignored and treated as a regular multiplicand, with or without decimal point	0.8 x 10 = 0.80, 0.13 x 2 = 26	8
PE3	The answer is incorrect by a power of 10	2 x 6000 = 120000	107
PE4	Addition is applied instead of multiplication	3 x 3 = 6	5
PE5	For questions that multiply by 1, 1 is inserted as an answer	500 x 1 = 1	16
PE6	For questions that multiply by 10, the multiplicand that is not 10 is used as an answer	10 x 8 = 8	10

PE7	For questions that multiply by 10, the first multiplicand is multiplied correctly, and the last number is concatenated	10 x 11 = 111	3
PE8	The multiplicand is used as an answer	2 x 2 = 2	1
Total			151

Popular errors differ from systematic errors in that a rule enforces systematic errors, whereas popular errors can arise because many people make a careless error or common guess. An error is not systematic unless a pattern can be found across many questions, as answered by the same person. In addition, many types of popular error can arise for one question. When looking for systematic errors, it is important to consider a question and answer together as an item.

It must also be noted that the most popular error does not have to be the most interesting one. If the second most popular was systematic across many questions, it was more interesting in this research than if it was common for one question.

At first glance, popular error types are reasonable and definitive. However, *almost all pattern types can be interpreted in multiple ways*. To illustrate this point, the proposed popular error types described in Table 5.2 were examined with the accompanying examples. Doing so helped to hypothesize how these errors might have occurred and what rules might be implemented. Not all plots are included, and the difficulty of items is disregarded for the sake of brevity.

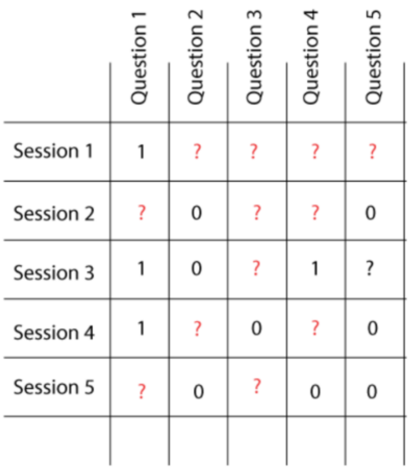
When looking at the examples in Figure 5.3, the above description of how popular errors occur seems logical. However, other rules like PE4 or PE8 (see Table 5.2) might also be applied. By looking at common errors individually, there is no way to know if children were adding or transferring the multiplicand. Similarly, there is no way to know if PE2, PE3 or PE6 were applied, or whether the errors were are all part of PE3. To investigate this point further, the systematic error patterns of learners must be investigated. This reasoning is the premise behind the proposed approach to automated misconception elicitation.

5.2.2 Exploring Concepts

Two approaches are explored to find out what concepts underlie each question; that is, what knowledge is needed to answer the question. This helped to interpret the results. First, an attempt was made to automate the process through the clustering of concepts. This proved to be a challenge outside the scope of the research and is thus only briefly discussed. Second, the concepts were established by investigating the existing literature and scrutinising the questions.

Attempt at automated concept elicitation

Since the ability of a learner changes over time, a list of items per session was created to enable a snapshot of the learner's knowledge state. Looking at a restricted period of 15 minutes allows for a snapshot of a user's knowledge state and prevents the assessment of a user who has learned. These sessions were then transposed to allow for iteration through a matrix per data vector. After the data were cleaned, the input data were prepared to cluster the questions and their binary responses, with 1 denoting a correct response and 0 an incorrect response. However, this transposition caused many missing values, as shown in Figure 5.4.



	Question 1	Question 2	Question 3	Question 4	Question 5
Session 1	1	?	?	?	?
Session 2	?	0	?	?	0
Session 3	1	0	?	1	?
Session 4	1	?	0	?	0
Session 5	?	0	?	0	0

FIGURE 5.3 EXAMPLE OF TRANSPOSED MATRIX OF ITEM RESPONSES

To deal with missing values, several imputation methods were tested on a subset of the matrix (30 questions by 400 responses). The subset was less sparse than the bigger matrix and was imputed with a k-nearest neighbour algorithm. By creating a filled matrix that is close to reality, values can be removed randomly and imputed afterwards. The mean squared error of the imputed matrix and the filled matrix (close to reality) yields the performance of the imputation method on this dataset. Although the filled matrix is a subset of the larger matrix and therefore an approximation of what would happen to the full matrix, the 30 carefully selected questions were representative of the 378 questions in the full matrix, in terms of difficulty.

The following imputation methods were used: matrix completion by iterative soft thresholding of SVD decompositions (softImpute), matrix completion by iterative low-rank SVD decomposition (iterativeSVD), multivariate imputation by chained equations (mice), and matrix factorization (Mazumder et al., 2010; Troyanskaya et al., 2001; Buuren & Groothuis-Oudshoorn, 2010; Rubinsteyn et al., 2017). Gradually 10% of values were removed randomly and imputed with the imputation methods. For every 10%, the mean squared error was calculated. As shown in Figure 5.5, matrix factorization performed the best with the lowest MSE at 90% missing values. Therefore, matrix factorization was used to impute the entire transposed matrix.

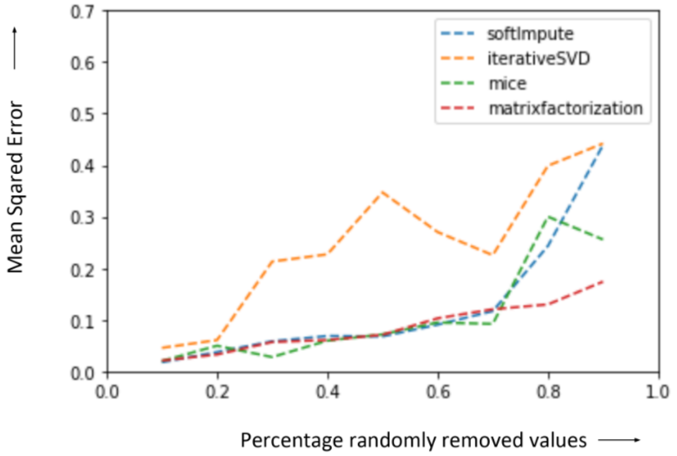


FIGURE 5.4 COMPARISON OF IMPUTATION METHODS BY MEAN SQUARED ERROR

After dealing with the missing values, two problems arose that created difficulties. First, transposing the questions caused the dataset to have around 1 million sessions for each question. There are currently no (standardized) unsupervised clustering methods that can deal with this type of high-dimensional data. Applying unsupervised clustering on a smaller subset of the data seemed to affect the accuracy of concepts that clustered negatively, causing the tested clustering methods to return insignificant results. Although a distinction could be made between 'hard' and 'easy' questions, the results were not significant enough for a fully interpretable concept per question. Second, transposing the matrix caused a dataset in which 97% of the values were missing; that is, most people did not answer most questions. Even if the imputation method that was applied was highly accurate, the input data for the clustering method would inevitably be noisy.

Since the main goal of this thesis was to create an approach for automated misconception elicitation, and not concept elicitation, a manual approach to concept elicitation was considered, leaving the automated concept elicitation for future work. Manual concept elicitation will be discussed in the next section.

Manual concept elicitation

All 1,238 questions were investigated manually. Concepts were assigned based on the rule that the user performed to answer the question and the difficulty of the question. For example, ‘5 x 1’ can be performed with rules ´multiplying by 5´ and ´multiplying by 1´. Since ´multiplying by 1´ is less difficult, the question was assigned with the rule ´multiplying by 1´. The difficulty of items was determined by the *difficulty* column and the guidelines by the Netherlands Institute for curriculum development (Noteboom et al., 2011; Noteboom et al., 2017). It must be noted that these concepts are a simplified version of reality, and are merely an indication that was used as a ground truth. The identified concepts are shown in Table 5.3 in ascending order of difficulty.

TABLE 5.3 GROUND TRUTH CONCEPTS

ID	Concept
C1	'Multiplying by 1'
C2	'Multiplying by 10'
C3	'Multiplying by 0'
C4	'Multiplying by 2'
C5	'Multiplying by 3'
C6	'Multiplying by 4'
C7	'Multiplying by 5'
C8	'Multiplying by 6'
C9	'Multiplying by 7'
C10	'Multiplying by 8'
C11	'Multiplying by 9'
C12	'Multiplying by 11'
C13	'Multiplying by 12'
C14	‘Decimal Multiplication’
C15	'Multiplying by 100 or 1000'
C16	'Multiplying by 25'
C17	'Multiplying by 50'
C18	'Multiplying by 75'

6 Data preparation

6.1 Data cleaning

The data were cleaned by removing questions that garnered fewer than 10,000 responses (Figure 6.1). This applied to most of the questions. As in Figure 5.1, the question labels on the x-axis were removed as they were visually indistinguishable. Also, responses of users that were suspected to have gamed the system were identified and removed. Sosnovsky et al. (2018) provided an elaborate description on anomaly detection and the criteria for establishing these anomalies. This step resulted in a cleaned dataset of 378 questions, with 9,518,226 responses in total.

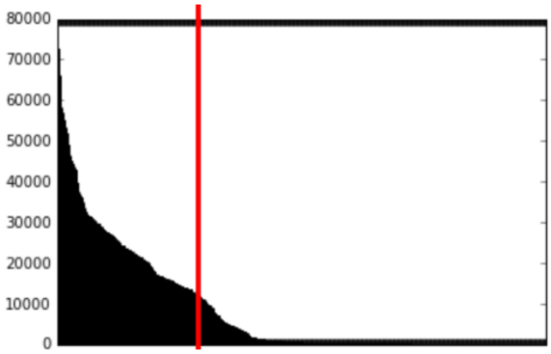


FIGURE 6.1 REMOVED QUESTIONS THAT HAD UNDER 10,000 RESPONSES

6.2 Pre-processing

This section discusses what pre-processing steps were taken for the association rule mining and network analysis. Generally, it is essential that item questions and answers are concatenated (e.g. ‘5x2’ and ‘5’ become ‘5x2=5’). This enables direct insight into the types of errors the learners might be making. Responses that do not reveal anything about the nature of the error were excluded. These were answers that indicated that learners did not know the answer (¿) and responses indicating that a learner had not answered a question in the allocated period (…). Each response trajectory considered the entire user history.

This approach is focused on the elicitation of misconceptions through systematic errors and therefore the correct answers are removed. Although a misconception can lead to a correct answer, the vast number of results hinders any interesting insights. Because errors are unlikely to occur (Buwalda et al., 2016), they have more explanatory power than correct answers, especially when they are systematic. After the first pre-processing step, the dataset consisted of 183,811 response trajectories (user histories) with 1,374,900 erroneous responses.

7 Experiment Results

7.1 Association rule mining

This section discusses how the minimum support and confidence thresholds (i.e. *minsup and mincon*) were determined. It also illustrates why raising the support and confidence threshold was not an effective strategy to deal with the explosion of rules that result from association rule mining.

As noted earlier, there are several ways to determine the *minsup* and *mincon* thresholds. It is important to understand what the goals are, to allow the set of thresholds to fulfil these goals. When thresholds are set high, interesting results can be missed; if they are set low, meaningful relationships are lost (Raeder, 2011). The goal for association rule mining in this research was to find systematic error patterns. Hence, reducing the explosion of rules was not the main priority at this stage. How to deal with the abundance of rules is discussed in the next section.

The rules were mined with a *minsup* of 40 (0.0005%) and a minimum confidence threshold of 40%. This resulted in 183,811 rules with 587 unique items. This seems a substantial number of rules to test the approach proposed in this research, for the following reasons. Raising the support would possibly leave out interesting cases, and cases with support lower than 40 were unlikely to be interesting (i.e. at least 40 users must display the systematic erroneous learning behaviour). The *mincon* was set at 40%, which generated enough rules and a diverse enough set of items to test the proposed approach.

The discovered rules were plotted along the dimensions of support, confidence and lift, as shown in Figure 7.1. As expected, the support of the rules increased when the confidence decreased. That is, if the constraints for the likeliness of items to occur together drops, item sets will occur together more often. Also, the lift generally decreased as the confidence decreased, which was also expected. That is, if the constraints for the likeliness of items to occur together drops, a smaller expected confidence is necessary to lower the lift.

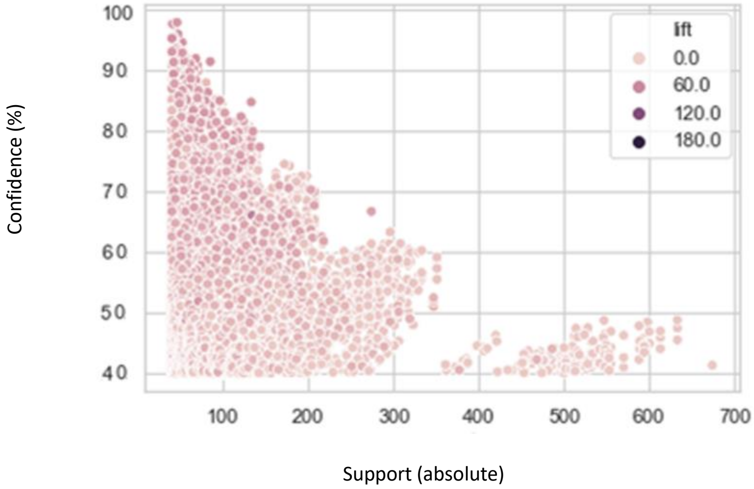


FIGURE 7.1 DISCOVERED ASSOCIATION RULES WITH MINSUP = 40 AND MINCON = 40%

The explicability of the rules having the lowest interestingness thresholds were investigated. The top 10 for lowest support, confidence and lift (in that order) are shown in Table 7.1. The rules in Table 7.1 seem to indicate systematic erroneous patterns that are interpretable. For example, there are several rules with indicative items that give hints regarding the erroneous learning process. First, items 2 x 4 = 4 and 2 x 2 = 2 indicate that learners are possibly transferring the multiplicand. Second, items 2 x 4 = 6, 10 x 10 = 20, 3 x 3 = 6 and 1 x 1 = 2 indicate that learners are possibly adding the multiplicands instead of multiplying them. Finally, items 0 x 6 = 6, 10 x 0 = 10, 0 x 64 = 64, 0 x 1 = 1, 700 x 0 = 700, and 4 x 0 = 4 can indicate both errors. The lift value seems to be bigger than 1 for rules with *minsup* = 40 and *mincon* = 0.4.

Table 7.1 Top 10 rules with the lowest support, confidence and lift

Antecedent (X)	Consequent (Y)	Sup	Con	Lift
0 x 6 = 6, 2 x 4 = 6, 10 x 0 = 10	700 x 0 = 700, 15 x 0 = 15	40	0.4	3.21
2 x 2 = 2, 0 x 6 = 6, 10 x 0 = 10, 0 x 64 = 64, 0 x 1 = 1	0 x 160 = 160, 15 x 0 = 15	40	0.4	3.3
0 x 64 = 64, 4 x 0 = 4, 0 x 1 = 1, 10 x 10 = 20	700 x 0 = 700, 0 x 6 = 6	40	0.4	3.4
0 x 6 = 6, 10 x 0 = 10, 2 x 4 = 6	700 x 0 = 700, 0 x 1 = 1	40	0.4	3.43
2 x 2 = 2, 10 x 10 = 10, 10 x 0 = 10, 0 x 1 = 1	0 x 160 = 160, 0 x 6 = 6	40	0.4	3.5
0 x 64 = 64, 4 x 0 = 4, 0 x 1 = 1, 10 x 10 = 20	0 x 6 = 6, 15 x 0 = 15	40	0.4	3.71
1 x 1 = 2, 700 x 0 = 700, 4 x 0 = 4	0 x 6 = 6, 15 x 0 = 15	40	0.4	3.71
10 x 10 = 20, 3 x 3 = 6, 0 x 64 = 64	4 x 0 = 4, 10 x 0 = 10	40	0.4	3.76
1 x 1 = 2, 700 x 0 = 700, 4 x 0 = 4	15 x 0 = 15, 0 x 64 = 64	40	0.4	3.77
2 x 4 = 4, 4 x 0 = 4, 0 x 64 = 64	0 x 6 = 6, 10 x 0 = 10	40	0.4	3.91

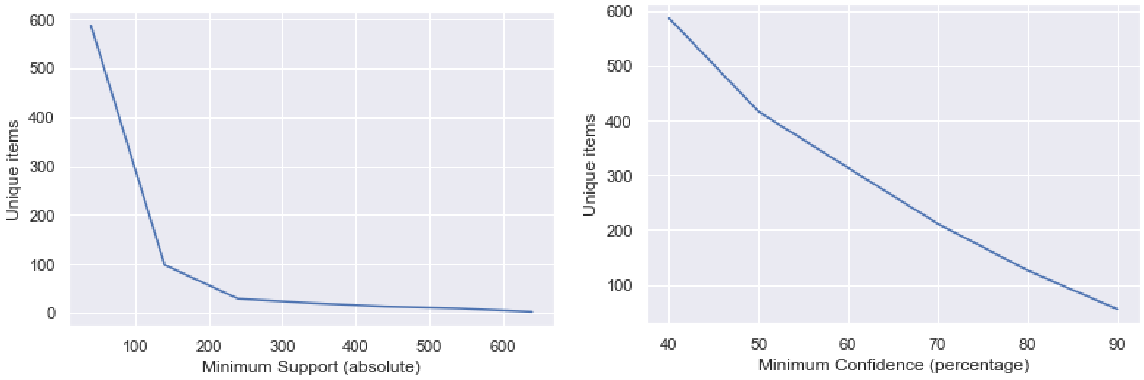
To investigate the rules with the lowest lift, a top-10 list of the rules with the lowest lift values was compiled (Table 7.2). Here, the minimum lift again appeared larger than 1. Learners were transferring the multiplicand, adding instead of multiplying, or simply applying multiplication-by-1 to multiplication-by-0 questions. Although these rules indicate systematic erroneous behaviour, individually they could mean several things.

Table 7.2 Top 10 rules with the lowest lift

Antecedent (X)	Consequent (Y)	Sup	Con	Lift
0 x 4500 = 4500, 0 x 6 = 6, 0 x 64 = 64, 0 x 1 = 1, 15 x 0 = 15	0 x 160 = 160, 700 x 0 = 700, 10 x 0 = 10, 4 x 0 = 4	46	0.4	1.7
0 x 4500 = 4500, 10 x 0 = 10, 0 x 64 = 64, 0 x 1 = 1, 15 x 0 = 15	0 x 160 = 160, 0 x 6 = 6, 700 x 0 = 700, 4 x 0 = 4	46	0.41	1.72
0 x 4500 = 4500, 0 x 64 = 64, 4 x 0 = 4, 0 x 1 = 1, 15 x 0 = 15	0 x 160 = 160, 0 x 6 = 6, 700 x 0 = 700, 10 x 0 = 10	46	0.41	1.75
0 x 4500 = 4500, 10 x 0 = 10, 0 x 64 = 64, 0 x 160 = 160, 0 x 1 = 1	0 x 6 = 6, 700 x 0 = 700, 15 x 0 = 15, 4 x 0 = 4	46	0.41	1.77
0 x 4500 = 4500, 700 x 0 = 700, 0 x 64 = 64, 4 x 0 = 4, 0 x 1 = 1	0 x 160 = 160, 0 x 6 = 6, 10 x 0 = 10, 15 x 0 = 15	46	0.4	1.79
0 x 4500 = 4500, 0 x 6 = 6, 700 x 0 = 700, 0 x 64 = 64, 15 x 0 = 15	0 x 160 = 160, 4 x 0 = 4, 0 x 1 = 1, 10 x 0 = 10	46	0.4	1.84
0 x 4500 = 4500, 0 x 6 = 6, 10 x 0 = 10, 0 x 64 = 64, 15 x 0 = 15	0 x 160 = 160, 700 x 0 = 700, 0 x 1 = 1, 4 x 0 = 4	46	0.43	1.85
0 x 4500 = 4500, 700 x 0 = 700, 10 x 0 = 10, 0 x 64 = 64, 15 x 0 = 15	0 x 160 = 160, 0 x 6 = 6, 0 x 1 = 1, 4 x 0 = 4	46	0.4	1.85
0 x 4500 = 4500, 0 x 160 = 160, 4 x 0 = 4, 0 x 1 = 1, 15 x 0 = 15	0 x 6 = 6, 700 x 0 = 700, 10 x 0 = 10, 0 x 64 = 64	46	0.4	1.85
0 x 4500 = 4500, 0 x 6 = 6, 10 x 0 = 10, 0 x 160 = 160, 15 x 0 = 15	700 x 0 = 700, 0 x 1 = 1, 4 x 0 = 4, 0 x 64 = 64	46	0.41	1.85

At first glance, the lowest ranking rules might indicate that the rules that were mined are meaningful. However, after investigating more rules with low thresholds, various side cases were found that indicated redundant rules. An example was rule 2 x 5 = 6 → 0 x 1 = 1. This rule was probably not the result of a systematic approach but rather a guess (2 x 5 = 6) that occurs often with a highly occurring item (0 x 1 = 1). Another example is rule 3 x 3 = 12, 3 x 3 = 6 → 10 x 0 = 10. Here, the learner is likely implementing multiplication by 3 through recall instead of having a conceptual understanding of multiplying 3 by 3 (e.g. 3 * 3 = 3 + 3 + 3 = 9). A final example is 10 x 10 = 10, 1 x 19 = 1 → 3 x 3 = 6. Here, transferring ('10 x 10 = 10, 1 x 19') could be applied together with addition ('3 x 3 = 6'), or the learner simply does not have conceptual understanding of multiplication. If the mined rules are to be useful, redundant and ambiguous rules must be pruned.

Simply raising the support or confidence thresholds does not solve these problems optimally, as this reduces the variety of items substantially (see Figure 7.2), leaving the expert with only a few insights. The next section discusses how the support and confidence thresholds can be applied to improve the rules' interestingness, without losing possible interesting items.



(A) MINCON = 40%, VARYING SUPPORT (B) MINSUP = 40, VARYING CONFIDENCE

FIGURE 7.2 NUMBER OF UNIQUE ITEMS DISCOVERED AT VARYING LEVELS OF CONFIDENCE AND SUPPORT

7.2 Network Analysis

The main goal was to allow the expert to reduce the time spent analysing the rules. Since the association rule mining resulted into 183,811 rules that were potentially interesting, this would not be a good prospect for a domain expert who was trying to elicit these rules. Therefore, in the following section, the rules are reduced to an analysable quantity in terms of *number* and *importance*. Items alone are limited in terms of explicability; meaningful rules are more explanatory, but meaningful groups of rules can be interpreted in a manner that aligns with the research objective.

First, community detection was explored to understand why the quality of a cluster of rules (i.e. modularity) was important. Second, the data were investigated by iterating over support and confidence thresholds, and noting what relevant clusters were found through communities, based on the average modularity metric (k-core =3) and maximum number of nodes in a cluster. Finally, ego networks were plotted for a few items, allowing singular items to be investigated more granularly.

In this network, the direction of the rules was neglected (e.g. rule A,B→C leads to the same subgraph as rule A,C→B). Hence, the number of rules with a similar item set was reduced to 52,639, reducing the number of rules by almost 30%. This process also automatically pruned rules that were subsets (e.g. rule A,B→C does not give more information in terms of edges and nodes than A,B,C→D). Appendix A.1 and A.2 shows the number of rules and unique items varying over minimum support and confidence thresholds.

7.2.1 Community detection

Modularity explains the quality of a cluster. To illustrate why it is important to consider the quality of a cluster, rules were categorized based on a local maximum of 5 component graphs with *minsup* = 60 and *mincon* = 50% (Appendix B.1). The plot is shown in Figure 7.3. The global maximum of 6 was found between *minsup* 150 and 200 with *mincon* = 70%, but these thresholds were neglected because of the sparsity of rules accompanying them (Appendix B.1). As shown in Figure 7.3, varying the support and confidence thresholds to optimise the number of component subgraphs did not consider the quality of grouped rules. The first blue and green components still contain most of the rules, whereas the purple and yellow components represent only one rule. To find the optimal groups, the quality of the groups must be considered.

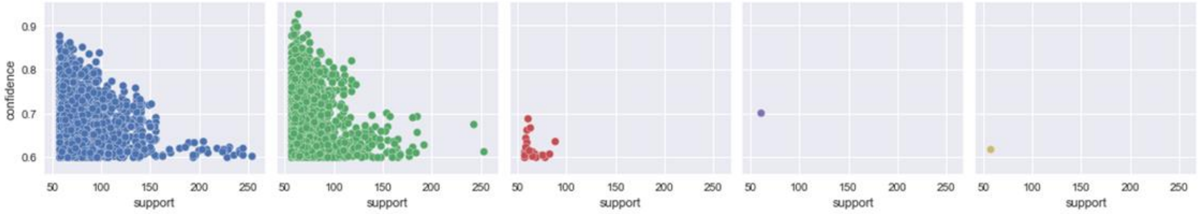
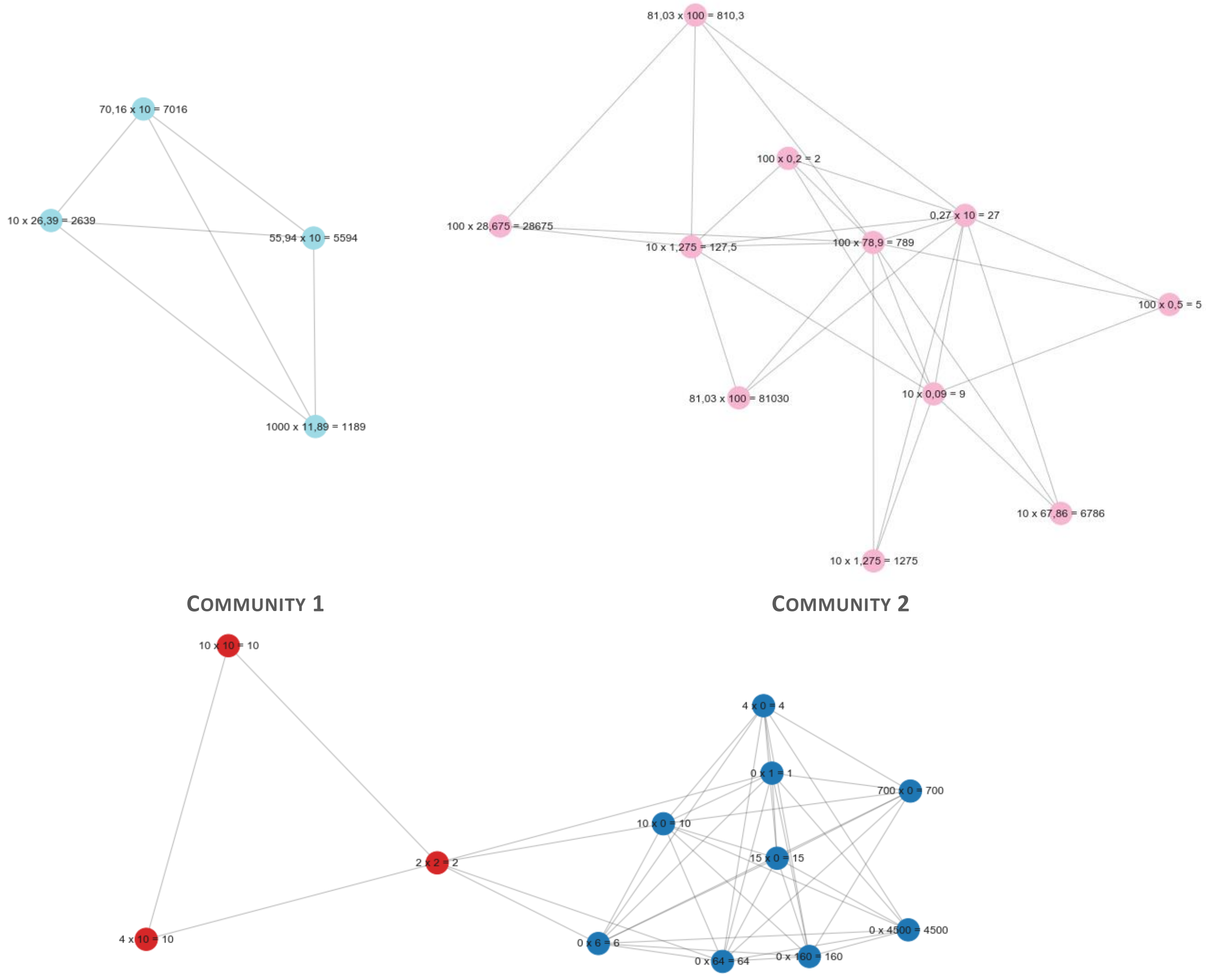


FIGURE 7.3 GROUPED RULES BASED ON LOCAL MAXIMUM NUMBER OF COMPONENT GRAPHS

However, trying to optimise for modularity has certain pitfalls. One is *the resolution limit,* which biases algorithms against finding small communities (Fortunato & Barthelemy, 2007). Instead of stopping at the optimal smaller communities, the algorithm accepts larger communities. Additionally, using syntactically similar items but different semantics and redundant rules does not help to find meaningful groups, but clouds the explicability of communities through connections that should not exist. The result can be seen in Appendix B.2, where the global maximum was increased only to seven communities; similarly, on investigation it was noted that the rules were not optimally clustered, but rather two large groups of rules contained most of the rules. To find more meaningful communities, the network must have more meaningful connections; that is, fewer items should become incorrectly connected. The next section shows that these challenges can be overcome.

7.2.2 Approach to misconception elicitation

The minimum confidence and support thresholds, *minsup* and *mincon,* were determined in section 7.1. In this section, the optimal initial support and confidence threshold, *isup* and *icon,* are determined. It is necessary that the items are highly likely to occur together; therefore, *mincon* is approximated as at least 80% (i.e. items are at least 80% likely to occur together). Setting this threshold limits the *isup* to between 40 and 130. To determine the support value, the support of 90 with the highest modularity ($M_c = 0.56$, Appendix B.1) was chosen (*isup* = 90, *icon* = 80%). This resulted in four communities, as depicted in Figure 7.4. As shown in community 3 and 4 of Figure 7.4, item '2 x 2 = 2' was linked to item '0 x 6 = 6' , '0 x 64 = 64', '10 x 0 = 10' and '0 x 1 = 1'. The rest of the items in community 4 were more highly interconnected, allowing community 3 to be separated.



Community 1

Community 2

Community 3 (red) and 4 (blue)

Figure 7.4 Communities (*isup* = 90, *icon* = 80%)

The communities found at (*isup* = 90, *icon* = 80%) only contained 27 unique items of the total 587 unique items found at (*mincon* = 40, *mincon* = 40%). To find more interesting rules with new items, the support and confidence thresholds must be decreased with *X* and *Y*. As shown in Figure 7.5, the optimal *X* and *Y* were found at step 2 for both the average modularity of the *k-core = 2* and *k-core = 3* networks (Appendix C.1, Table C.1). This meant that in terms of the quality of communities with nodes having at least a degree of 2 or nodes having a degree of 3, both were optimal at step 2.

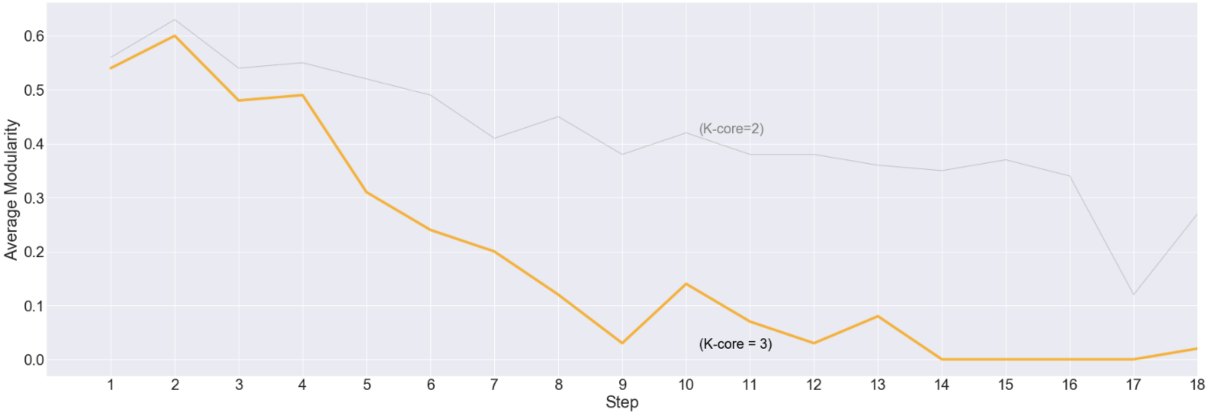


Figure 7.5 Determining optimal *X* and *Y* for finding communities (*isup* = 90, *icon* = 80%)

Both the first and second iteration of step 2 meet the limit of maximum nodes in a community and would result in the communities shown in Appendix C.2, Figures C.1 and C.2. The last iteration of step 2, however, was clouded with communities that had over 30 nodes.

To improve the explicability of the communities discovered in the last iteration of step 2, the approach was repeated with *isup* = 65, *icon* = 0.60, *minsup* = 40 and *mincon* = 0.40. Step 2 of Figure 7.6 shows that both the average modularity of the k-core = 2 and k-core = 3 networks could be improved by decreasing the support below 65 and confidence below 60% in a more granular manner (Appendix C.1, Table C.2). The discovered communities in Appendix C.2, Figure C.3 and C.4 were far more explicable in terms of number and importance.

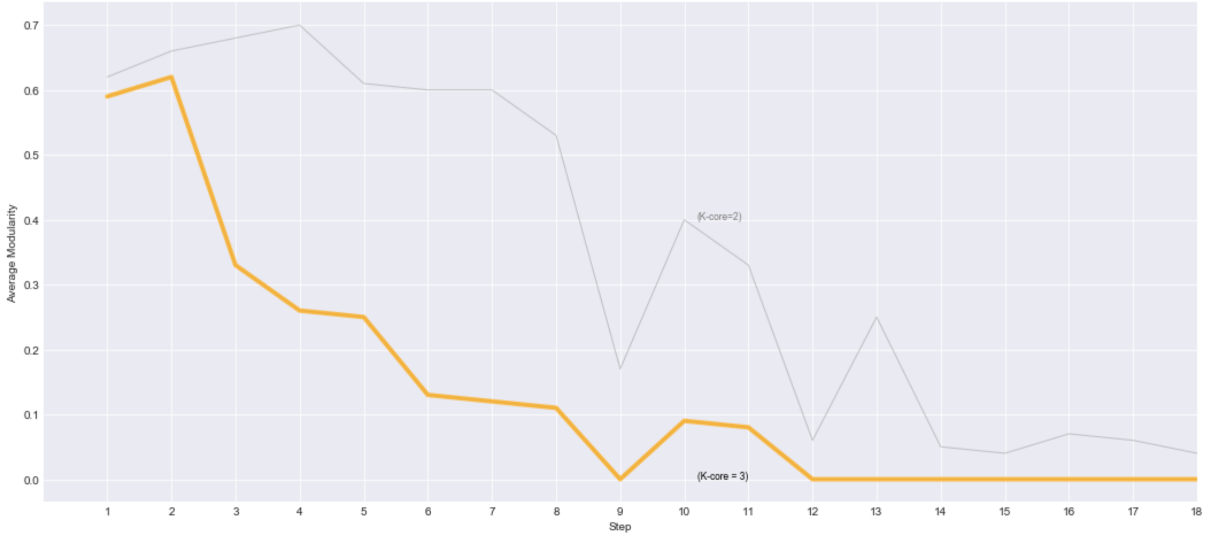


Figure 7.6 Determining optimal *X* and *Y* to find communities (*isup* = 65, *icon* = 60%)

After each iteration was optimized for the maximum allowed nodes in a cluster, the final *X* and *Y* are depicted in Table 7.3. The discovered communities are shown in Appendix C.2. There were 26 communities found in total.

TABLE 7.3 OPTIMAL X AND Y

Support	Confidence	*X* (absolute)	*Y* (percentage)	Communities	Figure
90	80	25	20	4	C.1
65	60	12.5	10	6	C.2
52.5	50	12.5	10	8	C.3
40	40	-	-	8	C.4

7.2.3 Ego networks

Ego networks allow the expert to inspect the found communities in a more granular way by selecting an individual item (node) and decreasing the support and/or confidence thresholds around that item. In section 7.2.2, at each iteration of a step, items were pruned and decreased by a fixed threshold (*X* and/or *Y*). Implementing ego networks enables the expert to have more freedom so that no items are pruned when decreasing the thresholds, and the granularity of decreasing can be determined interactively. For example, an expert could manually decrease the confidence threshold by 1% or 50% and see how it affects the relationship with other nodes.

To illustrate how an ego network analysis can be performed, the items with the highest degree of centrality were investigated. Items 0 x 6 = 6, 10 x 0 = 10, 0 x 64 = 64 and 0 x 1 = 1 had a similar high-degree centrality of 8. Since any of these items would suffice, 0 x 6 = 6 was selected as an ego. The results are plotted in Figures 7.5, 7.6 and 7.7.

In Figure 7.5, learners seem to apply an erroneous strategy to no items other than multiplication-by-0 items. When lowering the support by 5 and the confidence by 5%, cases of transferring the multiplicand emerged (e.g. 2 x 2 = 2 and 10 x 10 = 10). Finally, lowering the support by 10 and confidence by 10% more showed that addition was also implemented instead of multiplication (e.g. 3 x 3 = 6, 10 x 10 = 20, 1 x 1 = 2) together with node 0 x 6 = 6. What this means is that, when considering node 0 x 6 = 6, the most likely and most apparent nature of the systematic error is an erroneous strategy applied to multiplication-by-0 questions. Less likely and less apparent are cases of transferring the multiplicand, and finally, least likely and apparent are cases of addition instead of multiplying.

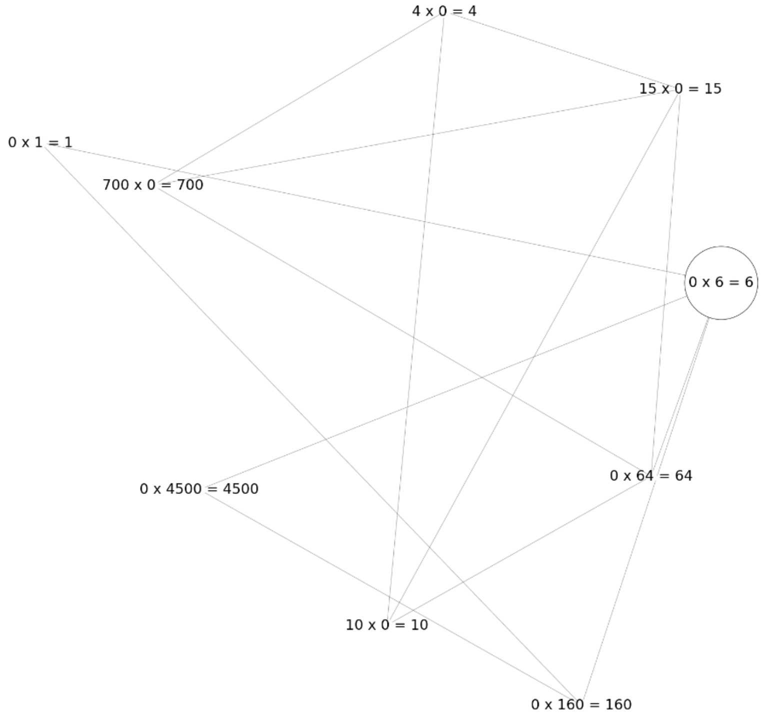


FIGURE 7.5 EGO NETWORK '0 X 6 = 6' (SUPPORT = 80, CONFIDENCE = 80%)

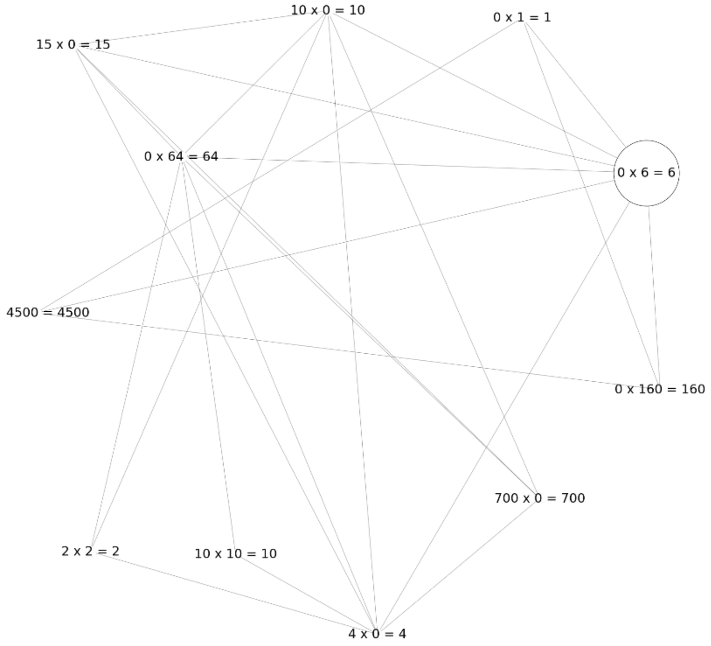


FIGURE 7.6 EGO NETWORK '0 X 6 = 6' (SUPPORT = 75, CONFIDENCE= 75%)

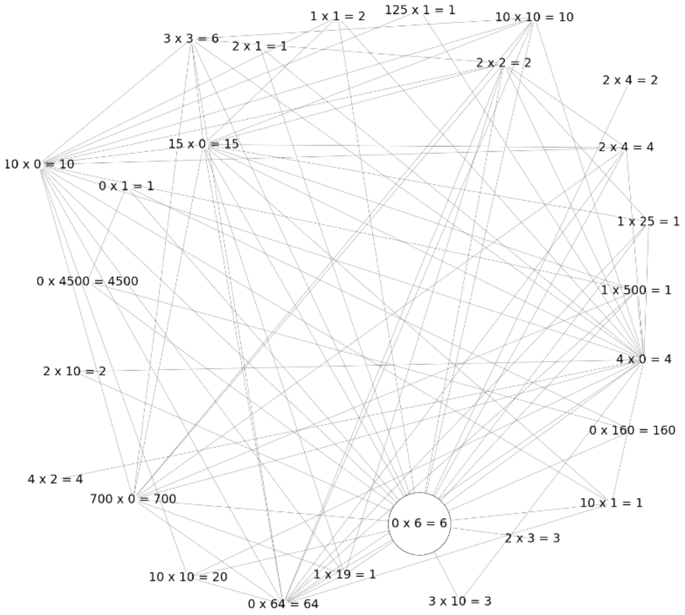
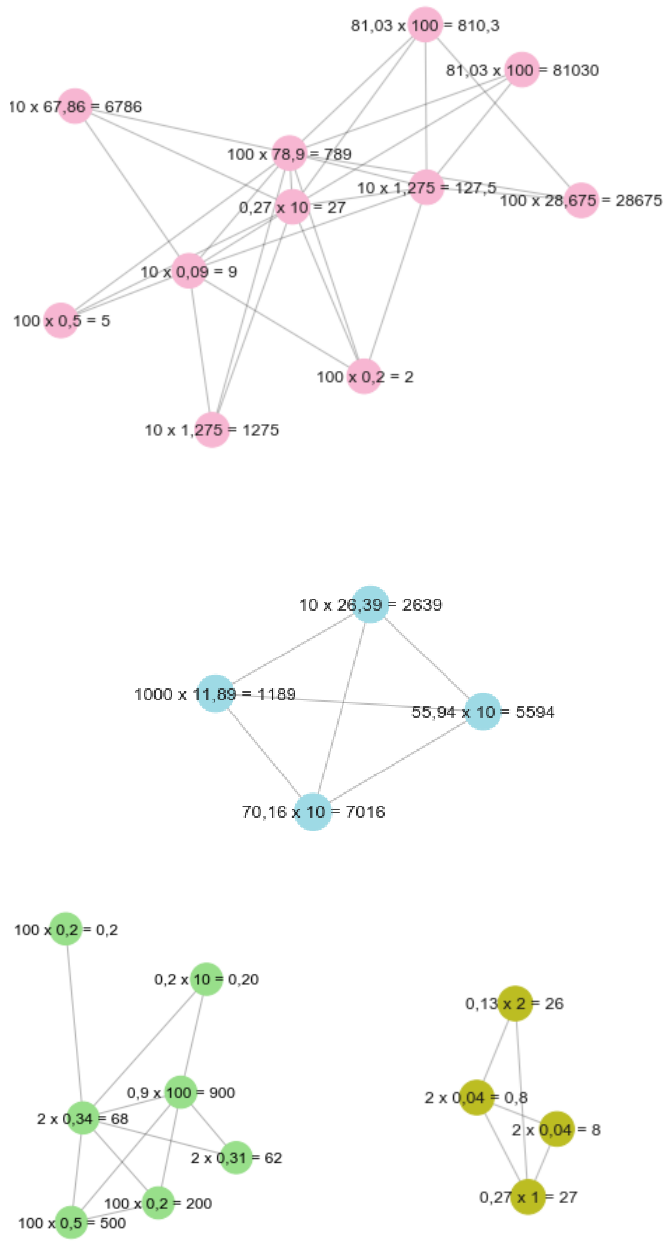
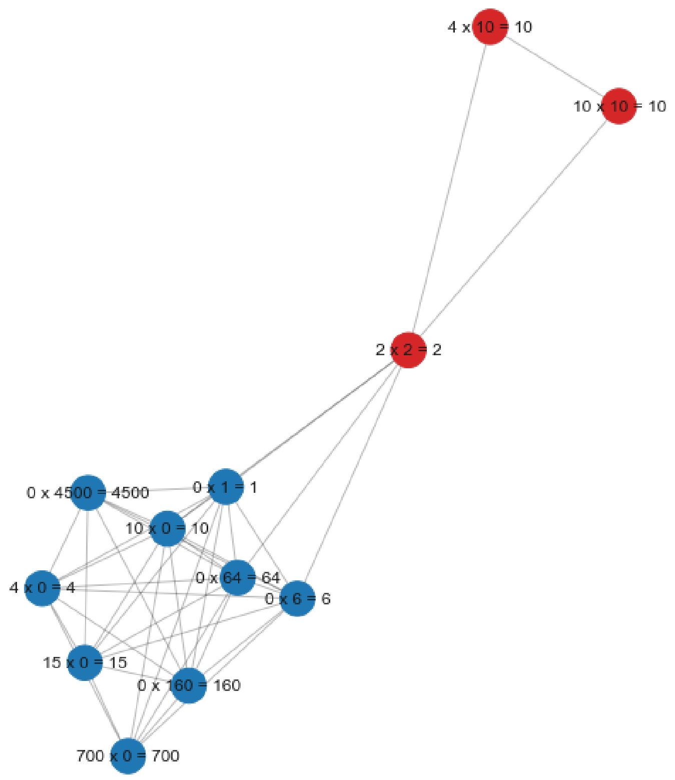


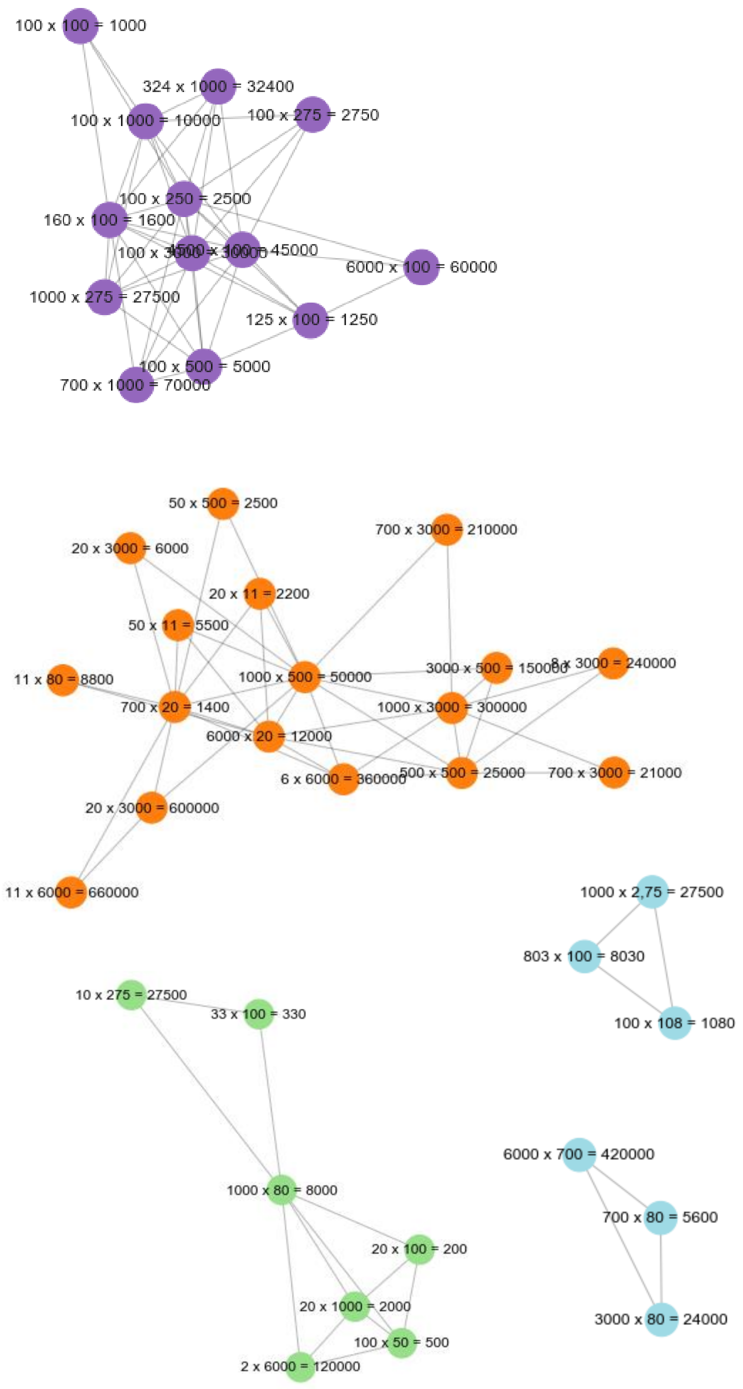
Figure 7.7 Ego network ‘0 x 6 = 6’ (support = 65, confidence = 65%)

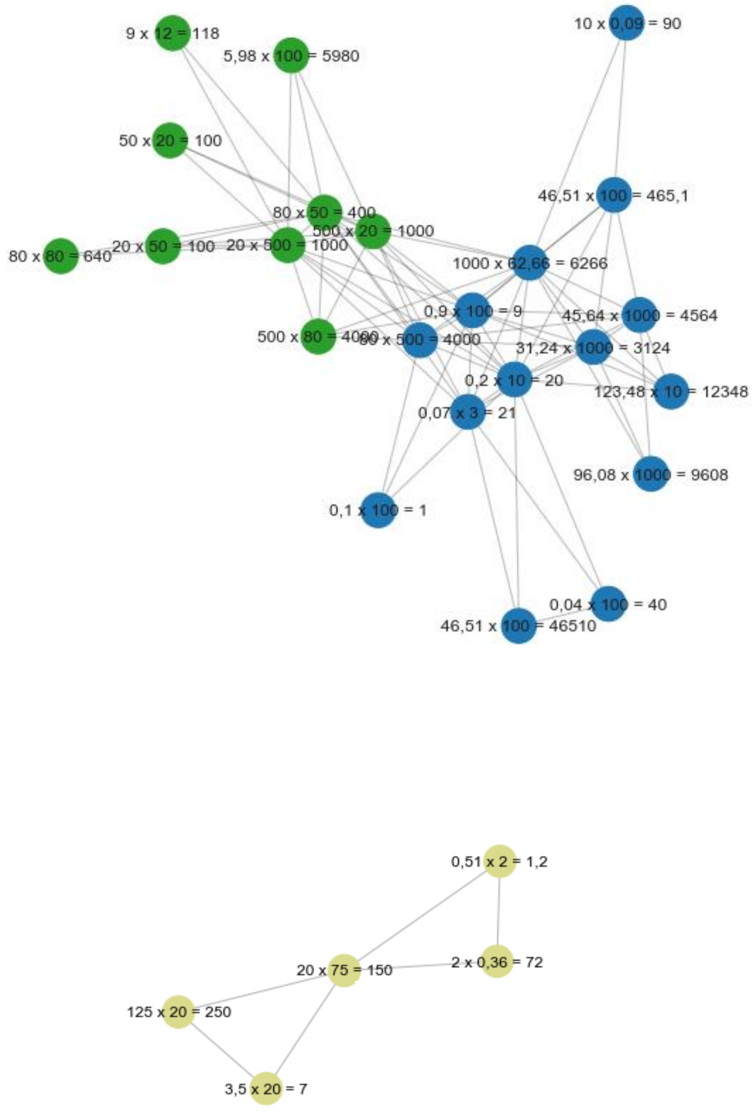
7.3 Expert analysis

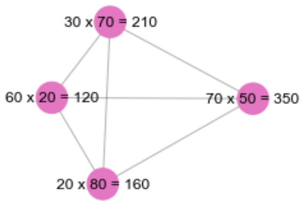
The following tables describe the interpretation of the discovered communities (Appendix C.2), the involved concepts and possible error types. Communities that are similar or connected are discussed in the same bug story.

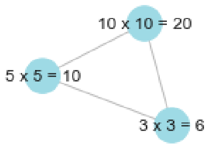
<table>
<tr><th colspan="2">Communities</th><th>Bug story</th></tr>
<tr><td><b>Involved concepts</b></td><td><b>Error type</b></td><td rowspan="3">Learners have trouble counting the correct number of decimal places in the product or placing values in the right places. It could be a procedural decimal error or a misconception of place value.
The respondents do not seem to ignore the decimal point, but rather miss the right answer by a factor of 10. This makes the error seem procedural rather than being a misconception.</td></tr>
<tr><td>Decimal multiplication</td><td>Procedural decimal error/ Misunderstanding of place value</td></tr>
<tr><td colspan="2">
</td></tr>
</table>

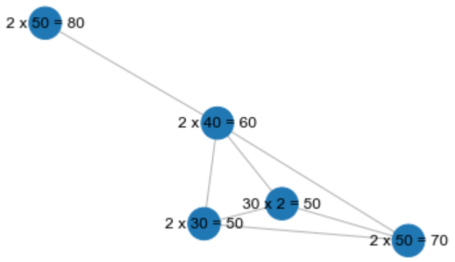
Communities		**Bug story**
Involved concepts	**Error type**	**Blue** Learners likely have trouble with times-zero multiplication, since the entire community comprises items linked to times-zero multiplication with a very high confidence. For questions that multiply by 0, the multiplicand that is not 0 is used as an answer. Multiplication by 0 is likely a new concept and the learner thus applies rules incorrectly to novel situations (i.e. overgeneralizes). **Red** Notably across the different concepts of multiplication by 0, 10 and 2, the multiplicand is transferred. These items are highly likely to occur together, but exist across various concepts. Hence, it seems that when transferring, the main concept of multiplication is disregarded.
Blue		
Multiplying by 0	Overgeneralization	
Red		
Multiplying by 0, Multiplying by 10, Multiplying by 2	Transferring the number	
 4 x 10 = 10 10 x 10 = 10 2 x 2 = 2 0 x 4500 = 4500 0 x 1 = 1 10 x 0 = 10 4 x 0 = 4 0 x 64 = 64 0 x 6 = 6 15 x 0 = 15 0 x 160 = 160 700 x 0 = 700		

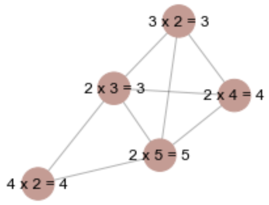
Communities		Bug story
Involved concepts	**Error type**	Learners seem to multiply by 10 instead of 100, and by 100 instead of 1000. The errors seem to occur with times -10, -100 and -1000 multiplication only. (This includes items 6000 x 700, 700 x 80 and 3000 x 80 = 24000 as times-100 and times-1000 for brevity purposes). Multiplication seems to be done correctly but appears out by a power of 10. Therefore, these errors also seem procedural in nature.
Multiplication by 10, Multiplication by 100, Multiplication by 1000	Procedural error: incorrect by a power of 10	
		

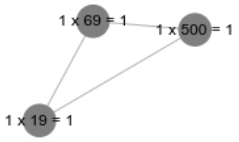
Communities		**Bug story**
Involved concepts	**Error type**	These communities seem to be the most clouded ones. The result shows a mesh of procedural errors in decimal multiplication and multidigit multiplication, by 10 or 100, and a redundant item (9 x 12 = 118) that does not seem to indicate any type of systematic erroneous behaviour. The communities separate the rules into a red and green cluster, but the semantic reason is unclear. Both clusters contain similar error types. The yellow cluster is added to this bug story as it also contains both procedural errors in decimal multiplication and multidigit multiplication by 10 or 100.
Decimal multiplication, Multiplication by 10, Multiplication by 100	Procedural decimal error/ Misunderstanding of place value / Procedural error: out by a power of 10	
 9 x 12 = 118 5,98 x 100 = 5980 10 x 0,09 = 90 50 x 20 = 100 46,51 x 100 = 465,1 80 x 50 = 400 500 x 20 = 1000 80 x 80 = 640 20 x 50 = 100 20 x 500 = 1000 1000 x 62,66 = 6266 0,9 x 100 = 9 45,64 x 1000 = 4564 500 x 80 = 4000 80 x 500 = 4000 31,24 x 1000 = 3124 0,2 x 10 = 20 123,48 x 10 = 12348 0,07 x 3 = 21 96,08 x 1000 = 9608 0,1 x 100 = 1 0,04 x 100 = 40 46,51 x 100 = 46510 0,51 x 2 = 1,2 20 x 75 = 150 2 x 0,36 = 72 125 x 20 = 250 3,5 x 20 = 7		

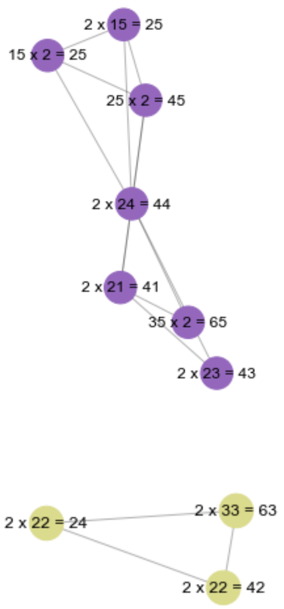
Communities		Bug story
Involved concepts	**Error type**	Learners seem to misunderstand place value where the last 0 is missed. Since this approach is specifically targeted at a single concept, the data likely reflect a misconception.
Multidigit multiplication	Misunderstanding of place value	
 30 x 70 = 210 60 x 20 = 120 70 x 50 = 350 20 x 80 = 160		

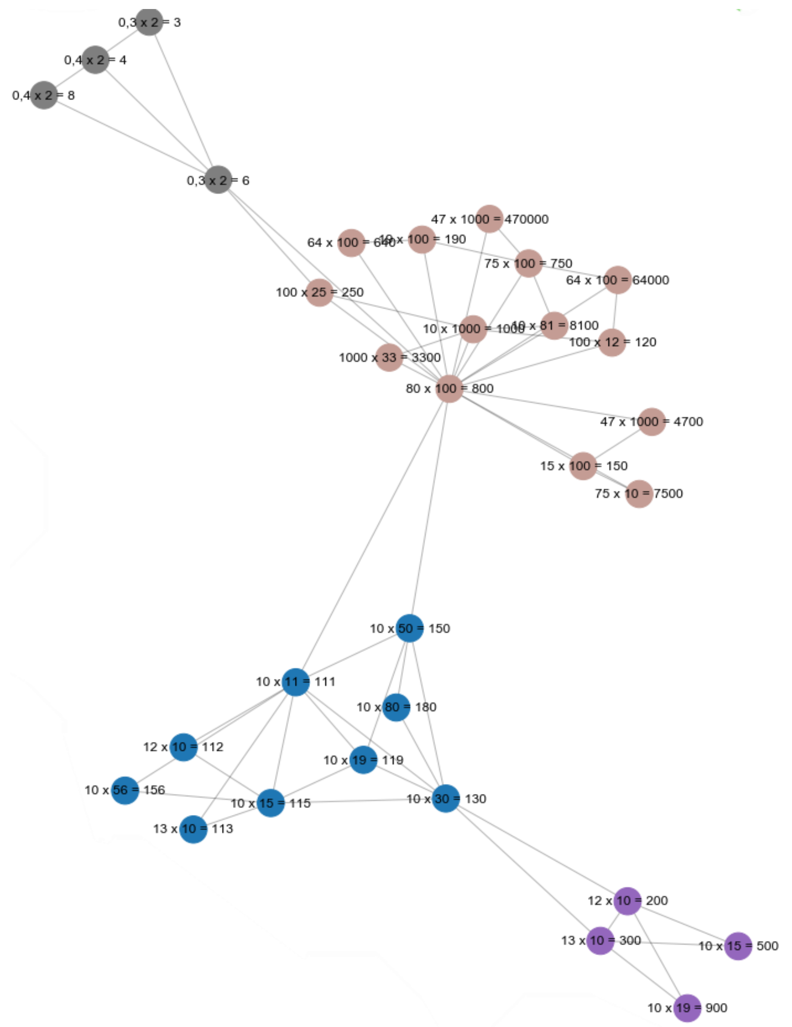
Communities		Bug story
Involved concepts	**Error type**	Learners seem to add instead of multiplying. The error occurs throughout several concepts (e.g. multiplication by 5, 10 and 3) and the main task is multiplication. Therefore, it can be assumed that the skill level of these learners is low and they do not understand what multiplication means. However, since it is done systematically, this is likely to be a strategy for learners derived from 2 x 2 = 4, where addition comes to fruition when applied to equal multiplicands.
Multiplication by 5, 10 and 3, Multiplication, Addition	Overgeneralization: addition	
		

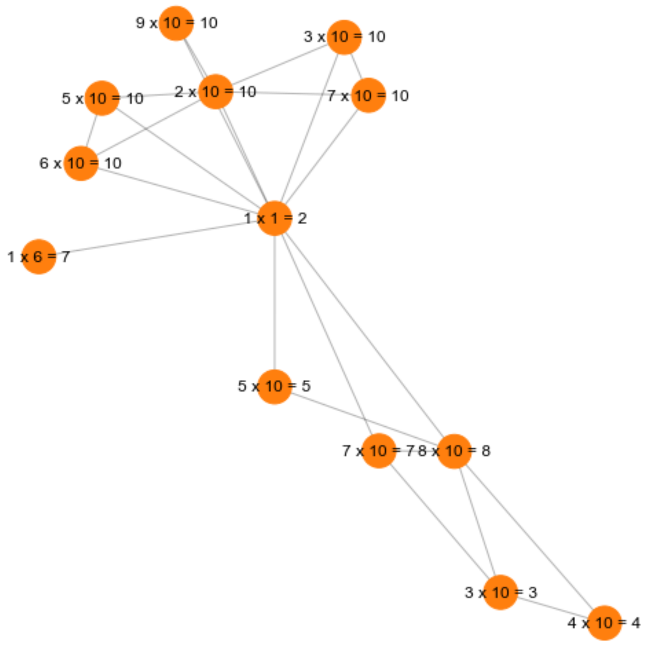
<table>
<tr><th colspan="2">Communities</th><th>Bug story</th></tr>
<tr><th>Involved concepts</th><th>Error type</th><td rowspan="3">Except for the redundant item 2 x 50 = 80, learners seem to add the single-digit multiplicand to the first number of the multidigit multiplicand, and concatenate the zero. The redundant item indicates answering by recall (2 x 50 = 80 is likely inspired by 2 x 40 = 80) and occurs often with item 2 x 40 = 60 (which could be inspired by 2 x 30 = 60). However, the other items are all interconnected, whereas the redundant item is not. Hence these items all tend to co-occur with each other. Therefore, it is likely that an erroneous strategy is applied instead of the error merely being an act of erroneous recall. Since this approach is specifically targeted at a single concept, the error is likely a misconception.</td></tr>
<tr><td>Multidigit multiplication</td><td>Adding the first multiplicand and concatenating the zero</td></tr>
<tr><td colspan="2"> </td></tr>
</table>

Communities		Bug story
Involved concepts	**Error type**	Learners seem to transfer a multiplicand that does not equal 2 as a strategy to answer multiplication-by-2 questions. Since this approach is specifically targeted at a single concept, it likely reflects a misconception.
Multiplication by 2	Transferring the number	
 3 x 2 = 3 2 x 3 = 3 2 x 4 = 4 2 x 5 = 5 4 x 2 = 4		

Communities		Bug story
Involved concepts	**Error type**	Learners seem to transfer the multiplicand (1) as a strategy to answer multiplication-by-1 questions. Since this approach is specifically targeted at a single concept, it is likely a misconception.
Multiplication by 1	Transferring 1	
		

Communities		**Bug story**
Involved concepts	**Error type**	**Purple** Learners seem to multiply with the first multiplicand only, and concatenate the second multiplicand. Although there is no question with 25 as a multiplicand, the concept 'multiplication by 25' is regarded as similar in terms of conceptual understanding. The learner understands times-2 multiplication, but is probably inspired by multiplication-by-10. In multiplication-by-10, only the first number of the multiplicand is used for an answer; the learner tries to apply this principle to times-25 multiplication. This therefore seems a case of overspecialization. **Yellow** Learners seem to multiply by the first multiplicand and concatenate the second multiplicand, or multiply by the second multiplicand and concatenate the first multiplicand (e.g. 2 x 22 = 24). This exception indicates that it differs from the previously identified community.
Multiplying by 10, Multiplying by 25	Overspecialization: Multiply with the first multiplicand and concatenate the second multiplicand	
 2 x 15 = 25 15 x 2 = 25 25 x 2 = 45 2 x 24 = 44 2 x 21 = 41 35 x 2 = 65 2 x 23 = 43 2 x 22 = 24 2 x 33 = 63 2 x 22 = 42		

Communities		Bug story
Involved concepts	**Error type**	
Gray		**Gray** Equivalent to previous bug story on procedural decimal error and misunderstanding of place value.
Decimal multiplication	Procedural decimal error/ Misunderstanding of place value	
Brown		**Brown** Equivalent to previous bug story on procedural error: out by power of 10.
Multiplication by 10, Multiplication by 100, Multiplication by 1000	Procedural error: out by power of 10	
Blue		**Blue** Learners seem to be concatenating 1 in front of the multiplicand that is not 10. Although the cause is hard to determine, this approach is specifically targeted at a single concept, and therefore is likely a misconception.
Multidigit multiplication by 10	Concatenating 1 in front of the multiplicand that is not 10	
Purple		**Purple** Learners seem to separate the multiplicand and multiply by 10 first, and then multiply by the remaining number of the multiplicand (e.g. 12 x 10 = 10 x 10 x 2 = 200). This approach is specifically targeted at a single concept, and therefore is likely a misconception.
Multidigit multiplication by 10	Multiplying by 10 first and then multiplying with the remaining number	
		

Communities		**Bug story**
Involved concepts	**Error type**	This community seems to indicate two types of systematic errors. Namely, transferring (e.g. 6 x 10 = 10 and 3 x 10 = 3) and adding (e.g. 1 x 1 = 2, 1 x 6 = 7). This could be because this community was discovered in the last iteration with the lowest support and confidence thresholds, making it harder to ensure likeliness between items. Multiplication-by-10 questions are the first type of questions presented to learners. These communities likely indicate learners who are struggling with the main concept of multiplication. Taking that into account, it must be noted that in the 'top' group, learners transfer the multiplicand 10; learners in the 'bottom' group transfer the multiplicand that is not 10. Since these two subgroups are connected only via 1 x 1 = 2, which is a redundant item, the two subgroups can be seen as two separate erroneous strategies.
Multiplication by 10, Multiplication by 1, Multiplication	Transferring the multiplicand that is 10, transferring the multiplicand that is not 10	
 9 x 10 = 10 3 x 10 = 10 5 x 10 = 10 2 x 10 = 10 7 x 10 = 10 6 x 10 = 10 1 x 1 = 2 1 x 6 = 7 5 x 10 = 5 7 x 10 = 7 8 x 10 = 8 3 x 10 = 3 4 x 10 = 4		

7.3.1 Accuracy

Precision

Precision is determined by the discovered cases that are misconceptions (***TP***) and procedural or factual errors (***TN***). Where doubt exists whether a community indicates misconceptions or procedural errors, the community is counted as procedural error.

$$Precision = \frac{TP}{TP + TN} = \frac{10}{10 + 16} \approx 39\%$$

Recall

Recall is determined by the number of unique items in the discovered communities that indicate misconceptions (***TP***) and the number of unique items that are pruned away (***FN***). Similarly, where there is doubt if communities are the cause of misconceptions or procedural errors, communities are counted as procedural errors.

$$Recall = \frac{TP}{TP + FN} = \frac{42}{42 + (587 - 185 - 42)} \approx 10\%$$

8 Conclusion

This section discusses the proposed research questions in section 2.2, and provides a conclusion for each question.

[RQ1] Can systematic errors be distinguished from random and careless errors?

Through investigating erroneous responses that resulted from arithmetic exercises, this research has shown that systematic errors can be distinguished from random and careless errors. By means of association rule mining and the accompanying minimum support and confidence thresholds, this research has shown that systematic error patterns arise.

[RQ2] What are the systematic errors caused by erroneous learning behaviour?

The expert analysis presented in section 7.3 describes the systematic error patterns that were caused by erroneous learning behaviour.

[RQ3] Can we effectively apply knowledge about systematic errors to identify underlying causes?

This approach has allowed for the nature of systematic errors to be explained and partly interpreted in terms of their likely causes. Specifically, the proposed approach in this research allowed misconceptions to be identified and distinguished from factual and procedural errors. In cases where the cause of the systematic error was unclear, the nature of the systematic error was clear, which allowed the expert to continue the investigation in a focused manner.

[RQ4] Can typical misconceptions be identified based on an expert analysis?

This approach allowed systematic error patterns to be interpreted informally, through the identified communities and the expert's expertise. The error patterns could then be described in a bug story. Based on this bug story, a generative theory of bugs can be set up to implement a bug diagnosis that can verify the informal analysis in a formal manner.

[MRQ] How can we effectively identify misconceptions that cause frequent patterns of erroneous learning behaviour?

This approach effectively identified misconceptions that caused frequent patterns of erroneous learning behaviour. It did so through identifying and grouping systematic error patterns, while also considering redundant items, ambiguous items, the number of interpretable results, and the relevance of the results that are presented to an expert. The expert can thus investigate an adequate series of misconceptions, provide context through items and their links, understand whether systematic error patterns occur in one or many concepts, and identify exceptions that would not have been considered when looking at a single systematic error pattern. Hence, this approach has achieved its goal of helping to alleviate part of the costly and time-consuming work associated with modelling learners in OLEs.

9 Discussion

9.1 Limitations

This section discusses the limitations of the proposed approach. First, errors are hard to find and therefore hard to examine systematically (Buwalda., 2016; Lebiere, 1999). This approach might not be as effective when a smaller dataset is used. The main limitation was thus related to using a single dataset in a single domain.

Second, working with a large dataset implies a complex set of teaching strategies that make a learner behave in a certain manner. Since there is no possibility to learn about the precise educational context in which every data entry has been produced, this approach lacks some of the context of the learning environment of individual students.

Third, this approach assumes that certain variations in the data can be leveraged to distinguish syntactically similar and semantically different items, by iteratively decreasing the support and confidence thresholds. These variations pertain to the number of times a learner is presented with a specific set of questions, the difficulty of the procedure, and the probability of the learner implementing an erroneous strategy. However, changing the thresholds does not account for syntactically similar error patterns that occur around similar support and confidence thresholds, which means syntactically similar items might occur with semantically different items.

Finally, the analysis could have been improved if experts had been interviewed. In other words, the analysis could have been based not only on a literature study but also on interview information from experts with several years of experience in error pattern analysis.

9.2 Future work

This section discusses possible directions for future work. First, association rule mining and community detection are the most commonly used algorithms in this field of study. There is room for improvement both in optimizing the results for rules with more interestingness measures (Tan et al., 2004) and in using different types of community detection algorithms (Yang, 2016). However, the approach is kept simple so that experts can explore more optimized solutions while using the proposed approach as a foundation.

Second, another improvement that could be explored is to add more interactivity to the tool than just filtering based on support and confidence. The tool could include a feature that allows the expert to test its intuition. The expert could filter based on a specific item or question, and explore what causes these rules might have. Similarly, there could be a filter based on concepts and the ability to select multiple items, to examine which items co-occur (e.g. above a minimum confidence threshold).

Third, the proposed approach can work for any response trajectory that has redundant and syntactically similar but semantically different items. It could be interesting to investigate misconception elicitation in other domains. For example, in the domain of division, learners could be answering questions with the dividend that is not 0 (e.g. 4 ÷ 0 = 4). There is no way of knowing whether the learner transferred digits, confused the ÷ symbol with a – symbol, or simply applied division-by-1 to a division-by-0 question. Either way, the learner failed to understand that division by zero is not possible. Although theoretically this approach is generalizable, it must be validated with several datasets from different types of OLEs.

Fourth, methods to increase the evidence from the tasks could be interesting. Insight into a learner's thinking when the learner is solving a problem can be a rich source of information about what the learner does and does not understand, and might change the implications for instruction.

Fifth, the causes of the found bugs must be tested through a bug diagnosis. As noted in section 3.2, finding the actual cause is only possible through investigation of the sub-steps a learner performs when solving a problem. Systematic errors are demonstrated in these sub-steps. It could be revealing to try the proposed approach on a dataset that includes not just final student responses but also their solution steps.

Finally, it would be interesting to study the optimal number of nodes for a community. Such knowledge would enable an expert to effectively investigate these communities.

References

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. In Proc. 20th int. conf. very large data bases, VLDB (Vol. 1215, pp. 487-499).

Anderson, J. R., & Jeffries, R. (1985). Novice LISP errors: Undetected losses of information from working memory. Human–Computer Interaction, 1(2), 107-131.

Ashlock, R. B. (2006). Error patterns in computation: Using error patterns to improve instruction. Prentice Hall.

Aynaud, T. (2009). Community detection for NetworkX.

Rubinsteyn, A., Feldman, S., O'Donnell, T., & Beaulieu-Jones, B. (2017). hammerlab/fancyimpute: Version 0.2.0. Zenodo. doi:10.5281/zenodo.886614

Attisha, M., & Yazdani, M. (1984). An expert system for diagnosing children's multiplication errors. Instructional Science, 13(1), 79-92.

Azevedo, A. I. R. L., & Santos, M. F. (2008). KDD, SEMMA and CRISP-DM: a parallel overview. IADS-DM.

Baffes, P., & Mooney, R. (1996). Refinement-based student modeling and automated bug library construction. Journal of Artificial Intelligence in Education, 7(1), 75-116.

Baker, R. S., & Inventado, P. S. (2014). Educational data mining and learning analytics. In *Learning analytics* (pp. 61-75). Springer New York.

Baker, R., Walonoski, J., Heffernan, N., Roll, I., Corbett, A., & Koedinger, K. (2008). Why students engage in' gaming the system' behaviour in interactive learning environments. Journal of Interactive Learning Research, 19(2), 185.

Barnes, T. (2005). The Q-matrix method: Mining student response data for knowledge. In *American Association for Artificial Intelligence 2005 Educational Data Mining Workshop* (pp. 1-8).

Barnes, T., Bitzer, D., & Vouk, M. (2005). Experimental analysis of the q-matrix method in knowledge discovery. In *International Symposium on Methodologies for Intelligent Systems* (pp. 603-611). Springer, Berlin, Heidelberg.

Blondel, V. D., Guillaume, J. L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. Journal of statistical mechanics: theory and experiment, 2008(10), P10008.

Dagenais, B. Pymining, a few data mining algorithms in pure python. https://github.com/bartdag/pymining, 2015.

Ben-Zeev, T. (1998). Rational errors and the mathematical mind. Review of General Psychology, 2(4), 366.

Brinkhuis, M., Savi, A., Hofman, A., Coomans, F., van der Maas, H., & Maris, G. (2018). Learning As It Happens: A Decade of Analyzing and Shaping a Large-Scale Online Learning System.

Brown, J. S., & Burton, R. R. (1978). Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive science*, 2(2), 155-192.

Brown, J. S., & Van Lehn, K. (1980). Repair theory: A generative theory of bugs in procedural skills. *Cognitive science*, 4(4), 379-426.

Brown J., Skow K., & the IRIS Center. (2016). Mathematics: Identifying and addressing student errors. Retrieved from http://iris.peabody.vanderbilt.edu/case_studies/ics_matherr.pdf

Buuren, S. V., & Groothuis-Oudshoorn, K. (2010). mice: Multivariate imputation by chained equations in R. Journal of statistical software, 1-68.

Buwalda, T. A., Borst, J. P., van der Maas, H., & Taatgen, N. A. (2016). Explaining mistakes in single digit multiplication: A cognitive model. In Proceedings of the 14th International Conference on Cognitive Modeling, University Park, PA, USA (pp. 11-18).

Cox, L. S. (1974). Analysis, Classification, and Frequency of Systematic Error Computational Patterns in the Addition, Subtraction, Multiplication, and Division Vertical Algorithms for Grades 2-6 and Special Education Classes.

Clauset, A., Newman, M. E., & Moore, C. (2004). Finding community structure in very large networks. Physical review E, 70(6), 066111.

Dogan, B., & Camurcu, A. Y. (2008). Association rule mining from an intelligent tutor. Journal of educational technology systems, 36(4), 433-447.

Elo, A. E. (1978). The rating of chess players, past and present. London: B. T. Batsford, Ltd.

Everett, M., & Borgatti, S. P. (2005). Ego network betweenness. Social networks, 27(1), 31-38.

Feelders, A., Daniels, H., & Holsheimer, M. (2000). Methodological and practical aspects of data mining. *Information & Management*, *37*(5), 271-281.

Fisher, D., & Frey, N. (2012). Making Time for Feedback. Educational leadership, 70(1), 42-47.

Fortunato, S., & Barthelemy, M. (2007). Resolution limit in community detection. Proceedings of the National Academy of Sciences, 104(1), 36-41.

Tsvetovat, M., & Kouznetsov, A. (2011). Social Network Analysis for Startups: Finding connections on the social web. ' O'Reilly Media, Inc.'.

Green, B. N., Johnson, C. D., & Adams, A. (2006). Writing narrative literature reviews for peer-reviewed journals: secrets of the trade. *Journal of chiropractic medicine*, 5(3), 101-117.

Guzmán, E., Conejo, R., & Gálvez, J. (2010). A data-driven technique for misconception elicitation. In *International Conference on User Modeling, Adaptation, and Personalization* (pp. 243-254). Springer, Berlin, Heidelberg.

Graeber, A. O. (1993). Research into Practice: Misconceptions about Multiplication and Division. Arithmetic Teacher, 40(7), 408-11.

Gowda, S. M., Rowe, J. P., de Baker, R. S. J., Chi, M., & Koedinger, K. R. (2011). Improving Models of Slipping, Guessing, and Moment-By-Moment Learning with Estimates of Skill Difficulty. EDM, 2011, 199-208.

Howell, K. W., Fox, S. L., & Morehead, M. K. (1993). Curriculum-based assessment: Teaching and decision making.

Lebiere, C. (1999). The dynamics of cognition: An ACT-R model of cognitive arithmetic. Kognitionswissenschaft, 8(1), 5-19.

Legara, E. (2016). Community Structures Retrieved from http://tinyurl.com/commdet2017

Maris, G. & van der Maas, H. L. J. (2012). Speed-accuracy response models: scoring rules based on response time and accuracy. Psychometrika, 77 (4), 615–633.

Math Garden (2019). Oefenweb. Retrieved from https://www.mathsgarden.com/

Mayo, M. J. (2001). Bayesian student modelling and decision-theoretic selection of tutorial actions in intelligent tutoring systems.

Mazumder, R., Hastie, T., & Tibshirani, R. (2010). Spectral regularization algorithms for learning large incomplete matrices. Journal of machine learning research, 11(Aug), 2287-2322.

Mislevy, R. J., Almond, R. G., & Lukas, J. F. (2003). A brief introduction to evidence-centered design. *ETS Research Report Series*, 2003(1).

Nesher, P. (1987). Towards an instructional theory: The role of student misconceptions. *For the learning of mathematics*, 7(3), 33-40.

Newman, M. E. (2004). Detecting community structure in networks. The European Physical Journal B, 38(2), 321-330.

Newman, M. E. (2006). Finding community structure in networks using the eigenvectors of matrices. Physical review E, 74(3), 036104.

Noteboom, A., Os, S. & Spek, W. (2011). Concretisering referentieniveaus rekenen 1F/1S. *SLO*

Noteboom, A., Aartsen, A. & Lit, S. (2011).Tussendoelen rekenenwiskunde voor het primair onderwijs1F/1S. *SLO*

Powers, D. M. (2011). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation.

Raeder, T., & Chawla, N. V. (2011). Market basket analysis with networks. Social network analysis and mining, 1(2), 97-113.

Radatz, H. (1979). Error analysis in mathematics education. Journal for Research in mathematics Education, 163-172.

Resnick, L. B., & Ford, W. W. (2012). Psychology of mathematics for instruction. Routledge.

Riccomini, P. J. (2014). Identifying and using error patterns to inform instruction for students struggling in mathematics.

Rittle-Johnson, B., Siegler, R. S., & Alibali, M. W. (2001). Developing conceptual understanding and procedural skill in mathematics: An iterative process. Journal of educational psychology, 93(2), 346.

Pellegrino, J. W., & Goldman, S. R. (1987). Information processing and elementary mathematics. Journal of Learning Disabilities, 20(1), 23-32.

Savi, A., Deonovic, B., Bolsinova, M., van der Maas, H., & Maris, G. (2018). Automated Diagnosis of Misconceptions in Single Digit Multiplication.

Sleeman, D., Hirsh, H., Ellery, I., & Kim, I. Y. (1990). Extending domain theories: Two case studies in student modeling. Machine Learning, 5(1), 11-37.

Self, J. A. (1990). Bypassing the intractable problem of student modelling. Intelligent tutoring systems: *At the crossroads of artificial intelligence and education*, 41, 1-26.

Sosnovsky, S., Müter, L, Valkenier, M., Brinkhuis, M., & Hofman, A. (2018). Detection of Student Modelling Anomalies. In V. Pammer-Schindler, M. Pérez-Sanagustín, H. Drachsler, R. Elferink, & M. Scheffel (Eds.) Proceedings of EC-TEL'2018: 13th European Conference on Technology Enhanced Learning (pp. 531-536). Berlin/Heidelberg, Germany: Springer.

Stansfield, J. L., Carr, B. P., & Goldstein, I. P. (1976). Wumpus advisor 1: A first implementation program that tutors logical and probabilistic reasoning skills.

Tatsuoka, K. K. (1983). Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of educational measurement*, *20*(4), 345-354.

Tan P, Kumar V, Srivastava J (2004) Selecting the right objective measure for association analysis. Inf Syst 29(4):293–313

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., ... & Altman, R. B. (2001). Missing value estimation methods for DNA microarrays. Bioinformatics, 17(6), 520-525.

Utrecht University (2018, 20 May). Freudenthal Institute. Retrieved from https://www.uu.nl/en/research/freudenthal-institute

Van Lehn, K. (1982). Intelligent tutoring systems. New York: Academic Press.

Van Lehn, K. (1988). Bugs are not enough: Empirical studies of bugs, impasses and repairs in procedural skills. Journal of Mathematical Behaviour, 3(2), 3-72.

West, T. A. (1971). Diagnosing pupil errors: Looking for patterns. The Arithmetic Teacher, 18(7), 467-469.

Wirth, R., & Hipp, J. (2000). CRISP-DM: Towards a standard process model for data mining. In Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining (pp. 29-39).

Woodward, J., & Howard, L. (1994). The misconceptions of youth: Errors and their mathematical meaning. Exceptional Children, 61(2), 126.

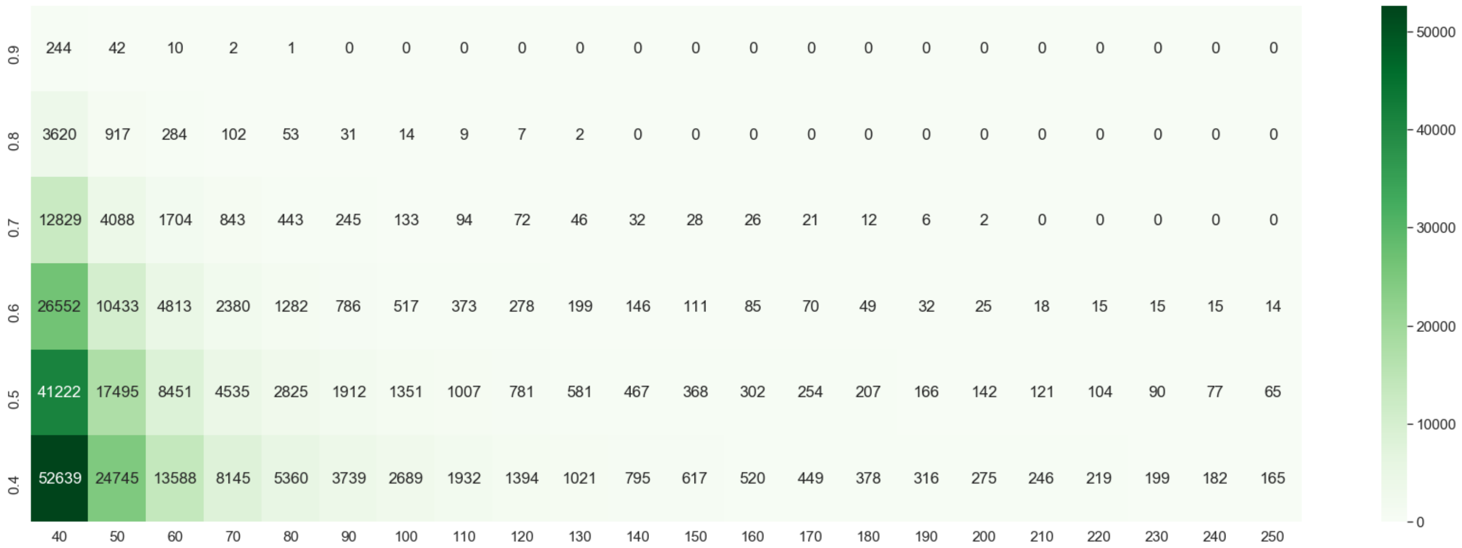
Wickelgren, W. A. (1977). Speed-accuracy tradeoff and information processing dynamics. Acta psychologica, 41(1), 67-85.

Yetkin, E. (2003). Student Difficulties in Learning Elementary Mathematics. ERIC Digest.

Yang, Z., Algesheimer, R., & Tessone, C. J. (2016). A comparative analysis of community detection algorithms on artificial networks. Scientific reports, 6, 30750.

Appendix A – Results ARM

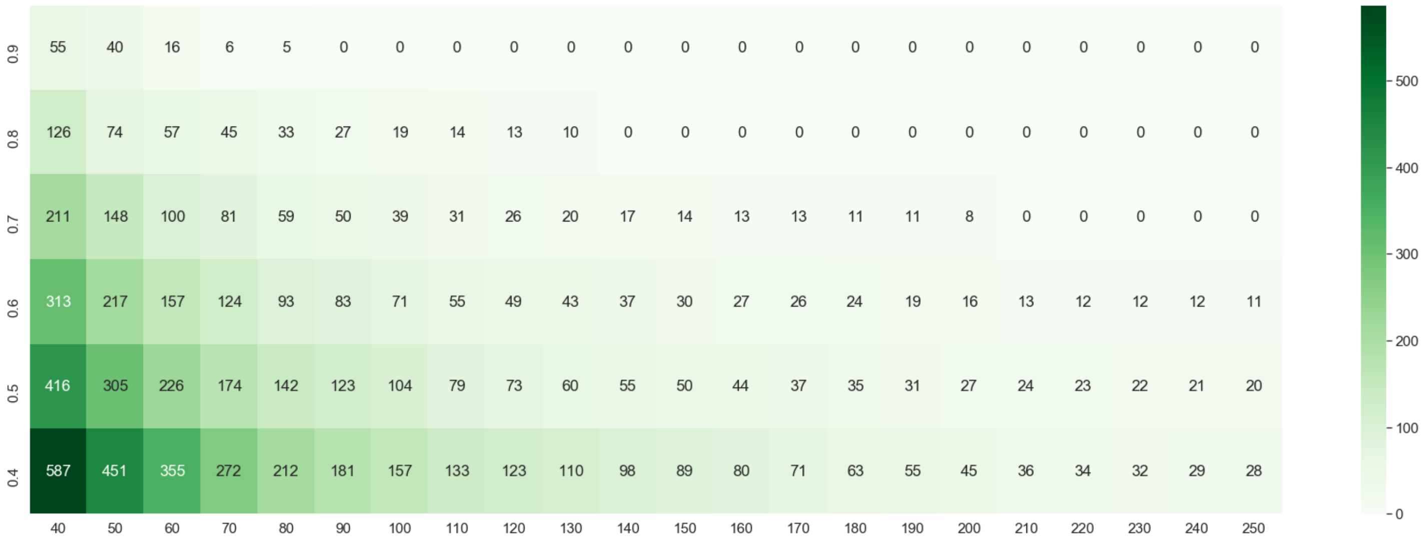
A.1 Number of rules variating over minimum support (x) and confidence thresholds (y)



	40	50	60	70	80	90	100	110	120	130	140	150	160	170	180	190	200	210	220	230	240	250
0.9	244	42	10	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.8	3620	917	284	102	53	31	14	9	7	2	0	0	0	0	0	0	0	0	0	0	0	0
0.7	12829	4088	1704	843	443	245	133	94	72	46	32	28	26	21	12	6	2	0	0	0	0	0
0.6	26552	10433	4813	2380	1282	786	517	373	278	199	146	111	85	70	49	32	25	18	15	15	15	14
0.5	41222	17495	8451	4535	2825	1912	1351	1007	781	581	467	368	302	254	207	166	142	121	104	90	77	65
0.4	52639	24745	13588	8145	5360	3739	2689	1932	1394	1021	795	617	520	449	378	316	275	246	219	199	182	165

FIGURE A.1 NUMBER OF RULES VARIATING OVER MINIMUM SUPPORT (X) AND CONFIDENCE THRESHOLDS (Y)

A.2 Number of unique items variating over minimum support (x) and confidence thresholds (y)

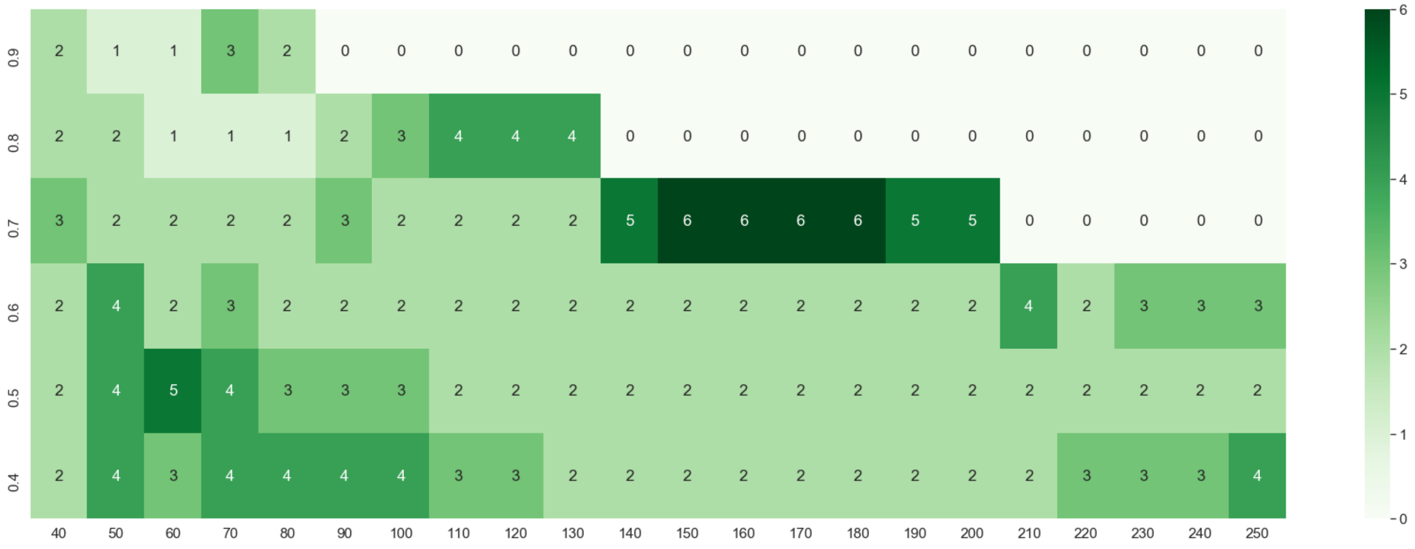


	40	50	60	70	80	90	100	110	120	130	140	150	160	170	180	190	200	210	220	230	240	250
0.9	55	40	16	6	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.8	126	74	57	45	33	27	19	14	13	10	0	0	0	0	0	0	0	0	0	0	0	0
0.7	211	148	100	81	59	50	39	31	26	20	17	14	13	13	11	11	8	0	0	0	0	0
0.6	313	217	157	124	93	83	71	55	49	43	37	30	27	26	24	19	16	13	12	12	12	11
0.5	416	305	226	174	142	123	104	79	73	60	55	50	44	37	35	31	27	24	23	22	21	20
0.4	587	451	355	272	212	181	157	133	123	110	98	89	80	71	63	55	45	36	34	32	29	28

FIGURE A.2 NUMBER OF UNIQUE ITEMS VARIATING OVER MINIMUM SUPPORT (X) AND CONFIDENCE THRESHOLDS (Y)

Appendix B– Results Community detection

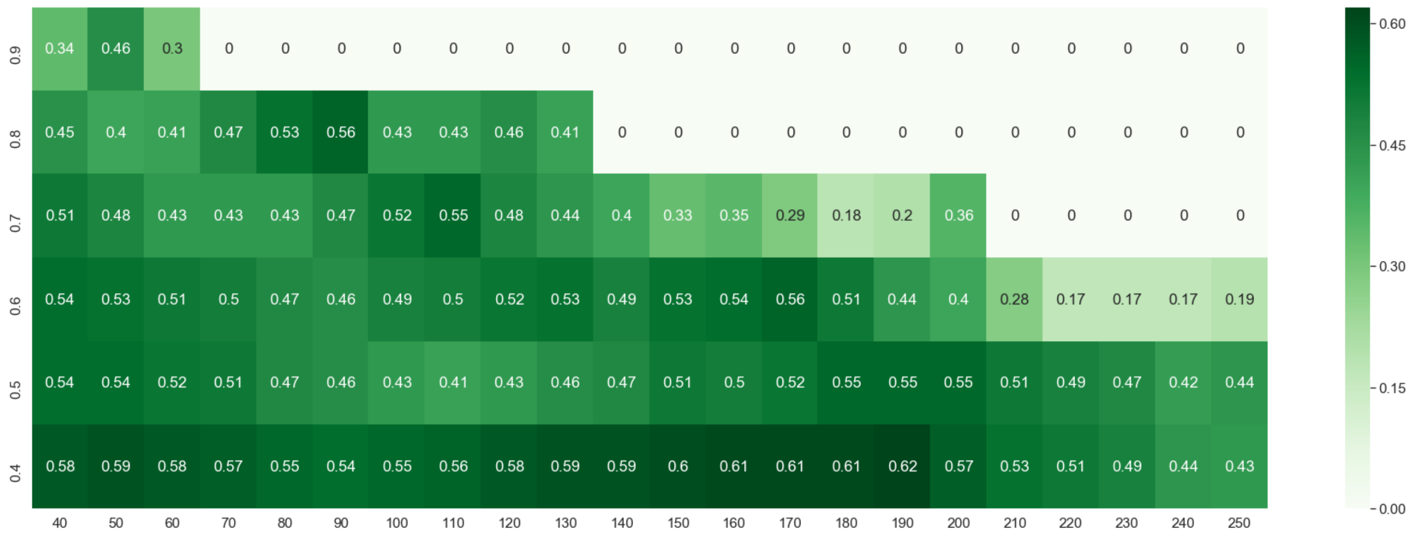
B.1 Number of component graphs variating over minimum support (x) and confidence thresholds (y)



	40	50	60	70	80	90	100	110	120	130	140	150	160	170	180	190	200	210	220	230	240	250
0.9	2	1	1	3	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.8	2	2	1	1	1	2	3	4	4	4	0	0	0	0	0	0	0	0	0	0	0	0
0.7	3	2	2	2	2	3	2	2	2	2	5	6	6	6	6	5	5	0	0	0	0	0
0.6	2	4	2	3	2	2	2	2	2	2	2	2	2	2	2	2	2	4	2	3	3	3
0.5	2	4	5	4	3	3	3	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
0.4	2	4	3	4	4	4	4	3	3	2	2	2	2	2	2	2	2	2	3	3	3	4

Figure B.1 Number of component graphs variating over minimum support (x) and confidence thresholds (y)

B.2 Modularity variating over minimum support (x) and confidence thresholds (y)



	40	50	60	70	80	90	100	110	120	130	140	150	160	170	180	190	200	210	220	230	240	250
0.9	0.34	0.46	0.3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.8	0.45	0.4	0.41	0.47	0.53	0.56	0.43	0.43	0.46	0.41	0	0	0	0	0	0	0	0	0	0	0	0
0.7	0.51	0.48	0.43	0.43	0.43	0.47	0.52	0.55	0.48	0.44	0.4	0.33	0.35	0.29	0.18	0.2	0.36	0	0	0	0	0
0.6	0.54	0.53	0.51	0.5	0.47	0.46	0.49	0.5	0.52	0.53	0.49	0.53	0.54	0.56	0.51	0.44	0.4	0.28	0.17	0.17	0.17	0.19
0.5	0.54	0.54	0.52	0.51	0.47	0.46	0.43	0.41	0.43	0.46	0.47	0.51	0.5	0.52	0.55	0.55	0.55	0.51	0.49	0.47	0.42	0.44
0.4	0.58	0.59	0.58	0.57	0.55	0.54	0.55	0.56	0.58	0.59	0.59	0.6	0.61	0.61	0.61	0.62	0.57	0.53	0.51	0.49	0.44	0.43

Figure B.2 Modularity variating over minimum support (x) and confidence thresholds (y)

B.3 Number of communities variating over minimum support (x) and confidence thresholds (y)

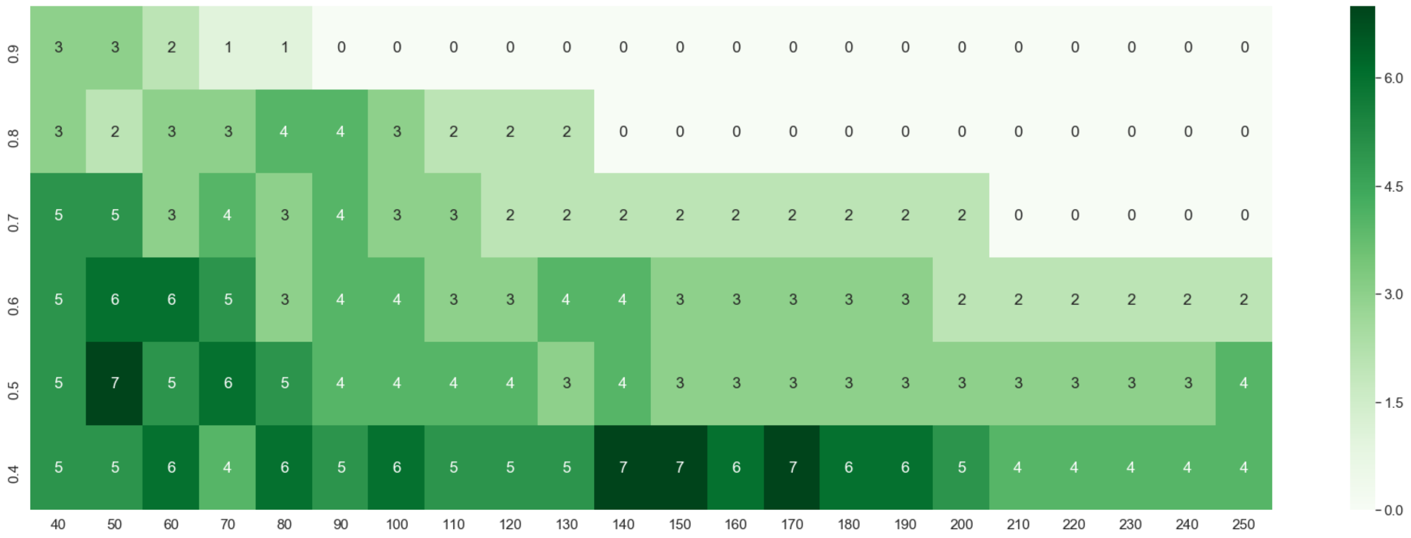


FIGURE B.3 NUMBER OF COMMUNITIES VARIATING OVER MINIMUM SUPPORT (X) AND CONFIDENCE THRESHOLDS (Y)

Appendix C – Results approach to misconception elicitation

C.1 Determining optimal *X* and *Y*

TABLE C.1 RESULTS DETERMINING OPTIMAL X AND Y (ISUP = 90, ICON = 0.80, MINSUP = 40, MINCON = 0.40)

Step	*Average Modularity (K-core = 2)*	*Average Modularity (K-core = 3)*	*Communities*	*Rules*	*Component graphs*	*Unique items*
1	0.56	0.54	10	6683	4	406
2	*0.63*	*0.60*	*18*	*668*	*13*	*271*
3	0.54	0.48	22	555	17	245
4	0.55	0.49	27	169	23	190
5	0.52	0.31	29	178	25	187
6	0.49	0.24	31	134	29	179
7	0.41	0.20	28	95	26	141
8	0.45	0.12	30	79	29	137
9	0.38	0.03	31	82	28	144
10	0.42	0.14	31	67	31	136
11	0.38	0.07	30	74	28	132
12	0.38	0.03	32	54	32	127
13	0.36	0.08	35	72	31	137
14	0.35	0.00	32	58	31	127
15	0.37	0.00	36	48	35	126
16	0.34	0.00	36	57	34	130
17	0.12	0.00	13	16	13	43
18	0.27	0.02	31	52	31	118

Table C.2 Results Determining optimal X and Y (isup = 65, icon = 0.60, minsup = 40, mincon = 0.40)

Step	*Average Modularity (K-core = 2)*	*Average Modularity (K-core = 3)*	*Communities*	*Rules*	*Component graphs*	*Unique items*
1	0.62	0.59	17	660	12	271
2	*0.66*	*0.62*	*26*	*183*	*20*	*185*
3	0.68	0.33	29	155	25	173
4	0.70	0.26	30	135	27	158
5	0.61	0.25	30	136	26	158
6	0.60	0.13	30	131	27	158
7	0.60	0.12	33	124	29	151
8	0.53	0.11	33	128	29	154
9	0.17	0.00	9	14	9	32
10	0.40	0.09	33	126	30	154
11	0.33	0.08	32	126	28	148
12	0.06	0.00	5	7	5	16
13	0.25	0.00	22	26	21	69
14	0.05	0.00	4	5	4	13
15	0.04	0.00	4	5	4	13
16	0.07	0.00	10	12	10	31
17	0.06	0.00	9	10	9	27
18	0.04	0.00	5	6	5	15

C.2 Communities

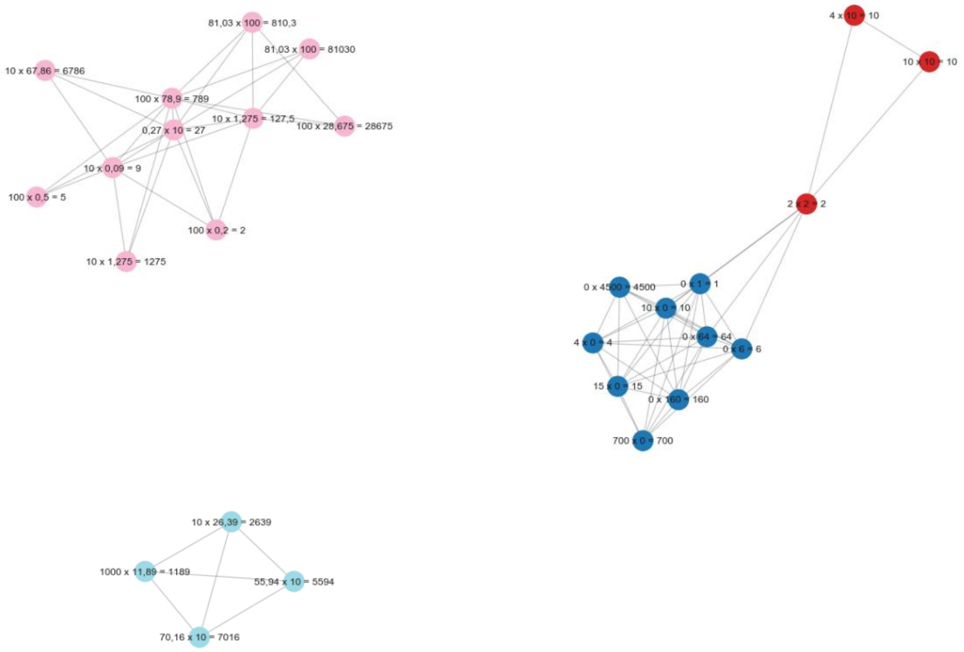


Figure C.1 Communities (support = 90, confidence = 80%)

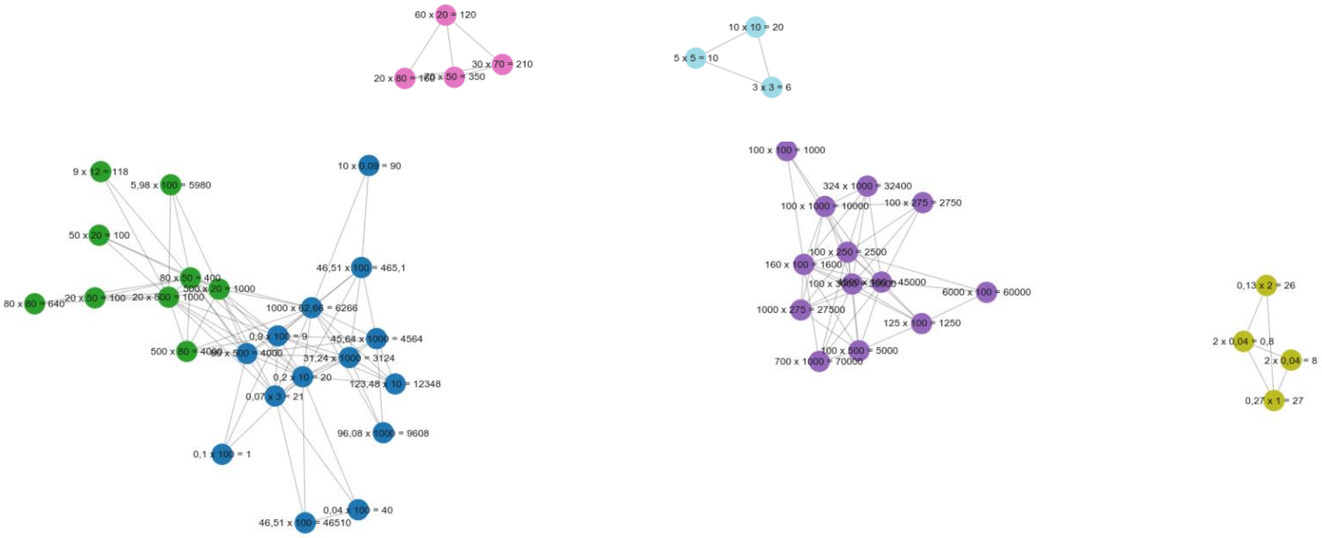


FIGURE C.2 COMMUNITIES (SUPPORT = 65, CONFIDENCE = 60%)

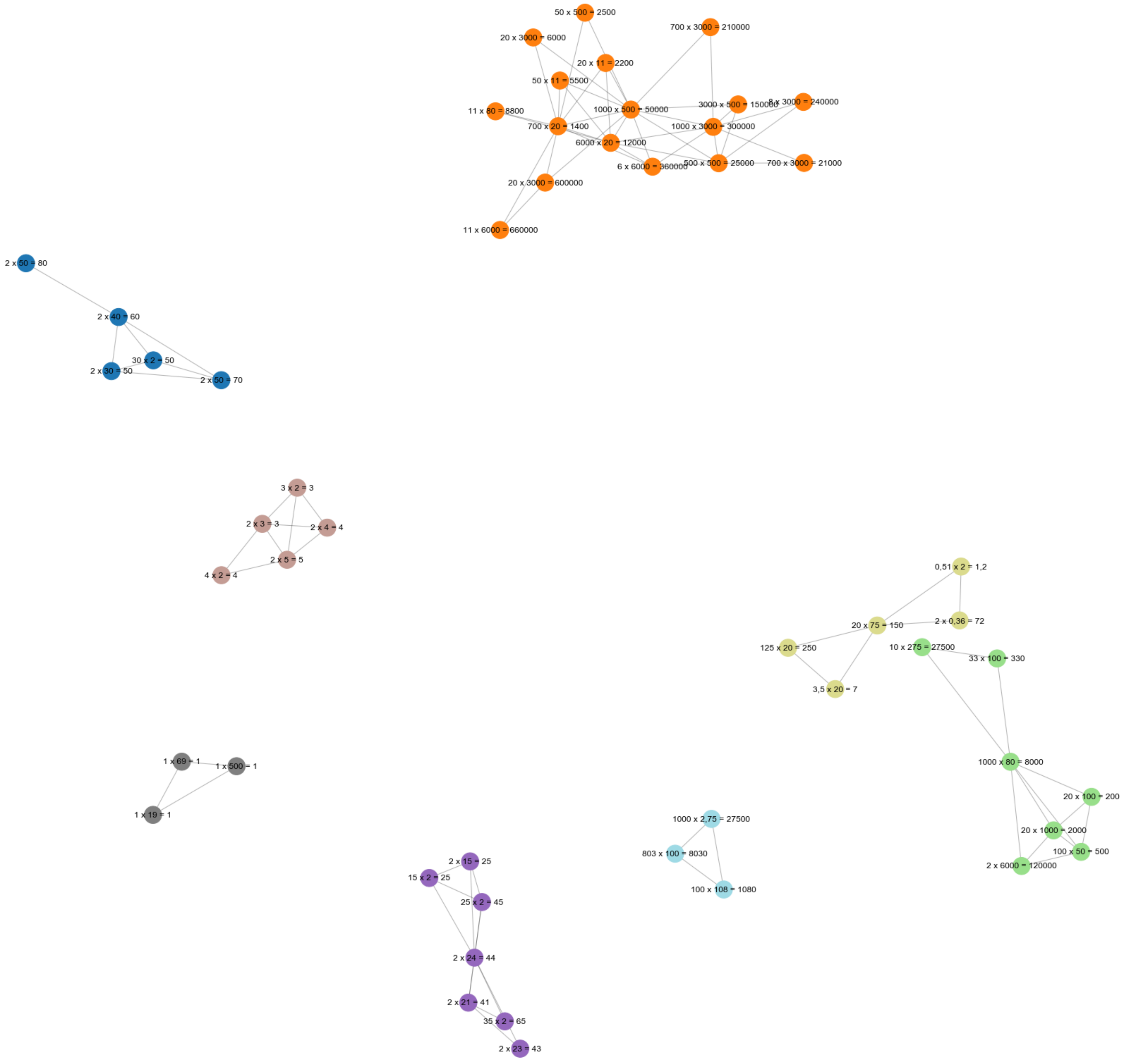


Figure C.3 Communities (support = 52.5, confidence = 50%)

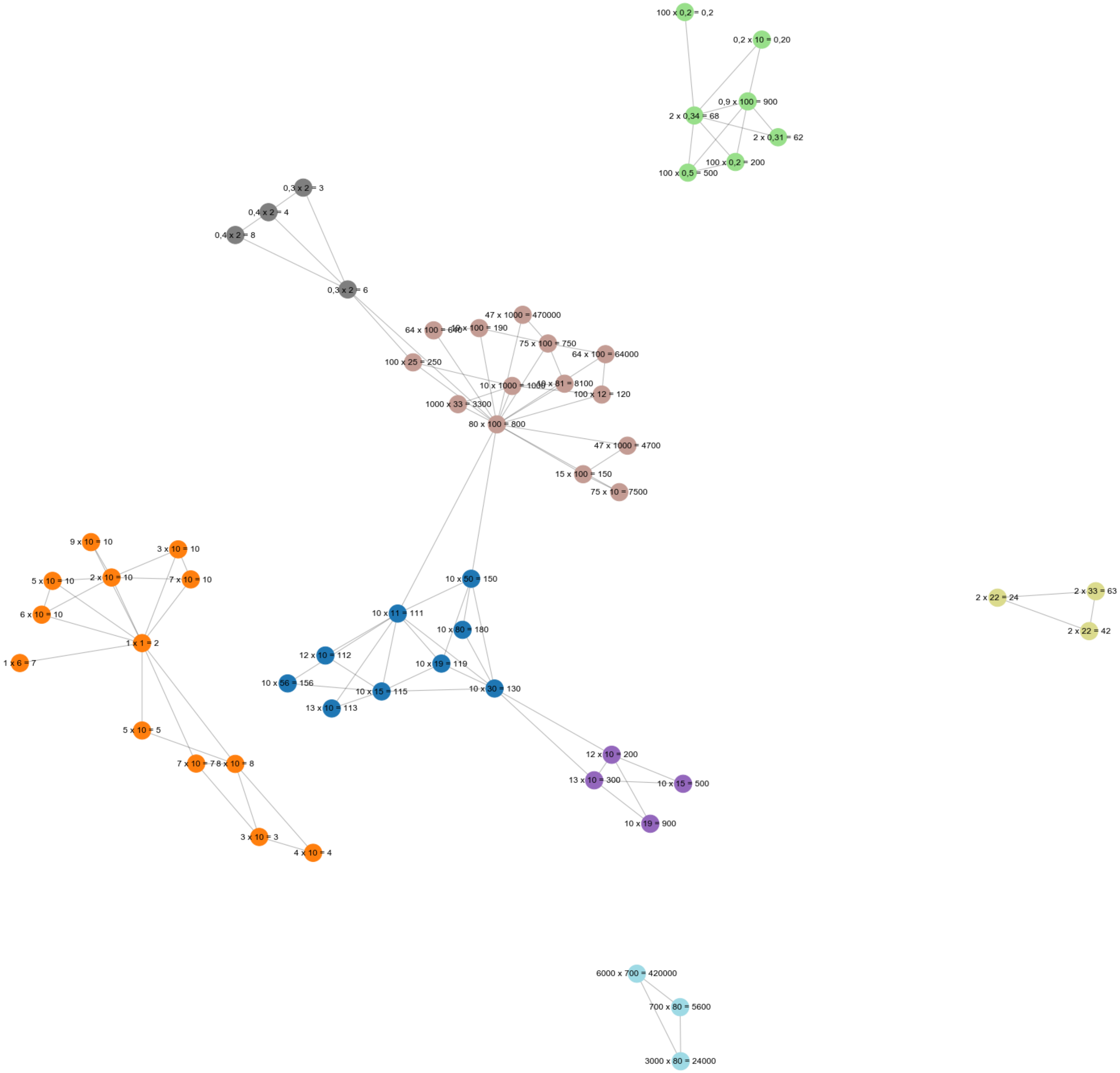


FIGURE C.4 COMMUNITIES (SUPPORT = 40, CONFIDENCE = 40%)