TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies
Department of Software Sciences

Kadri Jaagura 166761IVSM

IMPROVEMENTS ON FCA ALGORITHM TO INCREASE ITS PERFORMANCE

Master Thesis

Supervisor
Sadok Ben Yahia
PhD

Tallinn 2020

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Tarkvarateaduse instituut

Kadri Jaagura 166761IVSM

FCA ALGORITMI TÄIENDUSED SELLE JÕUDLUSE PARANDAMISEKS

Magistritöö

Juhendaja
Sadok Ben Yahia
PhD

Tallinn 2020

Author's declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Kadri Jaagura .....
(signature)

Date: May 12, 2020

Annotatsioon

Formaalsete kontseptioonide analüüs on meetod, mille abil saab andmeid mõtestada ja klassifitseerida. Selle abil töödeldakse andmehulkasid, mis koosnevad kindlast hulgast objektidest ja nende objektide tunnustest.

QualityCover on ahne algoritm, edestades paljusid varasemaid algoritme oma tulemuste kvaliteedinäitajate poolest erineva suuruse ja keerukusega andmestikel. Siiani on seda algoritmi testitud ainult keskmise suurusega andmekogumitega ja võib arvata, et algoritmi kasutamisel muutub probleemiks pikk aeg, mis kulub suuremate andmehulkade töötlemiseks.

Selle magistritöö eesmärk on leida viise, kuidas parandada QualityCover algoritmi, et seda saaks kasutada varasemast suurematel andmehulkadel ja kiiremini, kaotamata lõpptulemuse kvaliteeti. Algoritmi analüüsiti ja katsetati erinevaid arvutuste vähendamise viise. Tulemusena suudab algoritmi uus versioon töödelda suuremaid andmehulkasid ning selle arvutuskeerukus ja andmehulkade töötlusaeg vähenesid võrreldes vana versiooniga. Algoritmi lõpptulemus ja kvaliteet jäi samaks (kogu andmestikku katvate konteptioonide arv ja sisu jäid samaks) kui eelmisel versioonil. Kontseptsioonide leidmise järjekord muutus mõnigal määral. Nüüd saab QualityCover algoritmi abil töödelda andmehulkasid, mis sisaldavad miljoneid andmepaare minutite kuni tundide jooksul.

Töö üks osa tööst oli suunatud Spark-klastri arvutuste proovimisele selle algoritmiga. Kuna Spark raamistik ei võimaldanud kasutada soovitud andestruktuuri, mis võimaldaks kiireid arvutusi, jäeti algoritmi Sparkis testimine ja parandamine pooleli.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 45 leheküljel, 6 peatükki, 1 joonist, 12 tabelit.

Abstract

QualityCover is a greedy algorithm that outperforms many previous algorithms in quality measures on given benchmark data sets with different size and complexity. Until now, it has been only tested with average sized data sets and it is expected that the run time on bigger data sets will become an issue, when considering using this algorithm.

The aim of this thesis was to find ways to run the algorithm on bigger datasets faster, but without losing quality in the final output. For that, the algorithm was analyzed and multiple ways to decrease computations were tested out. As a result, the new version of the algorithm is able to run on bigger data sets and have less computational complexity, but has the same quality (number of concepts that cover the data) as the previous version.

One part of the work was aimed to be running the calculations on a Spark cluster, but this was omitted, since the framework did not allow the usage of desired data structure.

The thesis is in English and contains 45 pages of text, 6 chapters, 1 figures, 12 tables.

List of abbreviations and terms

FCA	Formal Concept Analysis
FC	Formal Concept
QC	QualityCover
PC	Pseudo Concept
MC	Mandatory Concept
RA	Relational Algebra
BR	Binary Relation
TFS	Textual Features Selection
DM	Discernibility Matrix
BDD	Binary Decision Diagram
CR	Clarification and Reduction
$\{i\}^{\uparrow}$	Extent
$\{i\}^{\uparrow\downarrow}$	Intent

Table of Contents

List of Figures

List of Tables

1. Introduction

Formal concept analysis (FCA) is a method to structure and analyze data. It analyzes data that consists of a set of objects and a set of attributes and a relationship that is connecting the objects and attributes in a binary relationship [1].

In FCA data is represented in data type named formal contcept. A formal concept is defined as the following: "A concept is considered to be a unit of thought constituted of two parts: its extent and its intent." (international standard ISO 704). It includes objects, their attributes and the relationship between them [1]. There, the extent means objects that belong to the one concept and intent is made up of all the attributes that belong to that concepts (all the attributes of members of intent). So - a concept is a pair made up of an extent, that consists of a closed set of objects, and an intent that consists of a closed set of attributes.

FCA increases the performance of data mining algorithms and improves the visualization of the results, creating concept hierarchies, and to do further data analysis and represent results visually. However there remains a huge room for further use of FCA in data mining and knowledge discovery, because for many FCA algorithms the computational complexity doesn't fit for large datasets even when using greedy algorithms [2].

Pertinent conceptual coverage is a set of formal concepts, extracted from the dataset, that cover all the related couples in binary relationship with the number of formal concepts as low as possible to cover all given data [1]. This is needed for efficiency reasons - sets of concepts that cover all objects and attributes with a minimal number of concepts should be found, since finding each concept is computationally costly. In addition, for finding the concepts, the most computation efficient way possible should be used.

FCA can be used in real life for finding patterns from data. For example, it can help to connect medical patients and all symptoms and each concept itself represents a diagnosis the patients might have. Connect software users and their access permissions. Or it can be used to to connect countries and the socio-economic indicators, where the indicators have numerical values that can be binarized (for example - less than average, greater than average).

Decreasing computational complexity and decreasing the number of formal concepts extracted to cover all given data is a challenge that FCA is facing when using it on real data. At the moment, even average size datasets can produce contexts that include very large number of formal concepts [3], making results not practical for use in real life for finding a pertinent coverage of formal concepts. Furthermore, the algorithms tend to have high complexity. This is holding back using FCA on big datasets [3], [2]. Better performing algorithms that efficiently provide pertinent coverage of data while producing lower number of formal concepts are needed if FCA is going to be used in big data.

QualityCover algorithm outperforms previous algorithms, e.g GreCond, GreEss, GenCoverage [2], in term of quality measures on given benchmark data sets with different size and complexity from the UC Irvine Machine Learning Database Repository [4], [5]. It's a greedy algorithm that chooses concepts based on their quality measure, called Size. Each next concept is chosen considering what maximizes the size measure.

1.1 Aim of the thesis and Methods

A previous version of QualityCover algorithms had great results regarding of the quality of the output [2]. While it did not outperform all the other algorithms in every quality aspect, it showed good overall results. To achieve the same good results, but with bigger data it has to be rewritten considering computational efficiency in every step possible for finding minimal coverage while using the same QualityCover greedy algorithm core that is based on each couples' quality measures. The goal is to gain as much knowledge about each object as possible with the least computational effort.

QualityQover algorithm has not been tested on bigger data sets to this date. The aim of the masters thesis is rewriting the QualityCover algorithm to be more efficient and less computationally costly and compare its results to the previous version of the algorithm on different sized data sets. The aim of the recreation of the algorithm is to achieve better performance (shorter runtime) than the previous version of the algorithm without losing quality of the generated contexts.

To test this, the former and the new version of the algorithm will be run on data sets with various sizes. It is expected that the former version of the algorithm will have slow performance on large data sets and the new one to outperform the old one by finding formal concepts faster and managing to compute concepts on larger data sets.

The result of the thesis is expected to be new version of the algorithm and comparison of the new and old version of the algorithm on data sets of increasing sizes.

2. Background and related work

2.1 Basic settings and terminology

The mathematical principles for this domain were established by Birkhoff in 1960's and the current form of FCA framework was introduced by Wille in 1980's [6].

[FORMAL CONTEXT]
The structure that is used in FCA as a formal concept is a triplet $\mathcal{K}$ = ($\mathcal{O}$, $\mathcal{I}$, $\mathcal{R}$), where:
$\mathcal{O}$ represents a finite set of objects.
$\mathcal{I}$ is a finite set of attributes (or items).
$\mathcal{R}$ is a binary (incidence) relation between objects and attributes (*i.e.*, $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{I}$).
Each couple $(o, i) \in \mathcal{R}$ expresses that the object $o \in \mathcal{O}$ possesses or is described by the attribute i.

The formal context $\mathcal{K}$ can be depicted by Table 1 with $\mathcal{O} = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ and $\mathcal{I}$ = $\{a, b, c, d, e, f, g, h\}$.

Table 1. An example formal context as a matrix.

	a	b	c	d	e	f	g	h
1		×					×	
2	×		×				×	
3				×	×	×	×	×
4	×			×	×	×	×	×
5	×				×	×	×	×
6		×				×	×	
7	×					×	×	
8	×				×		×	
9	×	×	×	×	×	×	×	×

A formal context is better understood if it is depicted by a *cross table* as shown by Table 1. In the latter, × indicates that the corresponding object has the corresponding attribute while a blank indicates no connection between the object and the attribute. But for knowledge

representation the objects and their attributes can also be represented in a lattice diagram or other type of graph.

[GALOIS OPERATORS]
Let be a formal context. For $A \subseteq \mathcal{O}$ and $B \subseteq \mathcal{I}$, we define: For a formal context $\mathcal{K}$ = ($\mathcal{O}$, $\mathcal{I}$, $\mathcal{R}$), operators $^{\downarrow} : 2^O \Rightarrow 2^I$ and $^{\uparrow} : 2^I \Rightarrow 2^O$ are defined as follows:

$$A^{\downarrow} = \left\{ i \in I \mid (\forall o \in A), (o, i) \in \mathcal{R} \right\} \tag{2.1}$$

$$B^{\uparrow} = \left\{ o \in O \mid (\forall i \in B), (o, i) \in \mathcal{R} \right\} \tag{2.2}$$

The operators $^{\uparrow}$ and $^{\downarrow}$ are known as concept forming operators or derivator operators.

[FORMAL CONCEPT]
A formal concept is any pair $\langle A, B \rangle$ of sets where $A \subseteq \mathcal{O}$ (extent, set of objects to which the concept applies) and $B \subseteq \mathcal{I}$ (intent, set of attributes characterizing the concept) such that B is just the set of attributes shared by all objects from A, and A is the set of all objects sharing all attributes from B. In symbols, this can be written as $A^{\downarrow} = B$ and $B^{\uparrow} = A$. The extent of the concept $\langle A, B \rangle$ is A while its intent is B.

Less formally, it can be said that a formal concept is a set of objects together with the attributes. These objects and attributes are found together and it is impossible to add an additional attribute without removing an object or add an additional object without removing an attribute.

[PSEUDO-CONCEPT]
The pseudo-concept associated to the couple (a, b), denoted PC_{ab}, is a binary relation computed by getting the cartesian product of the maximal set of attributes fulfilling the object a and the maximal set of objects having the attribute b [7]. Formally,

$$PC_{ab} = \{(o, i) \mid (o, i) \in b\uparrow \times a\downarrow \subseteq \mathcal{R} \mid o \in b\uparrow \wedge i \in a\downarrow\}. \tag{2.3}$$

PC_{ab} is the union of all the formal concepts containing the couple (a, b). Also the size of a given pseudo-concept PC_{ab} is defined as follows:

$$Size(PC_{ab}) = \frac{|PC_{ab}|}{|a\downarrow| \times |b\uparrow|} \tag{2.4}$$

Table 2 represents all information that needs to be derived from central matrix - extent, intent, and support of each object. This information can also be represented in a more simple format, like in Table 3

Table 2. The formal context with extent, intent, support, given by Table1

	a	b	c	d	e	f	g	h	$\lvert\{o\}^{\downarrow}\rvert$
1		×					×		2
2	×		×				×		3
3				×	×	×	×	×	5
4	×			×	×	×	×	×	6
5	×				×	×	×	×	5
6		×				×	×		3
7	×					×	×		3
8	×				×		×		3
9	×	×	×	×	×	×	×	×	8
$\{i\}^{\uparrow}$	245789	169	29	349	34589	345679	123456789	3459	
$\{i\}^{\uparrow\downarrow}$	ag	bg	acg	$defgh$	eg	fg	g	$efgh$	

Table 3. An example formal context.

i	a	b	c	d	e	f	g	h
$\{i\}^{\uparrow}$	245789	169	29	349	34589	345679	123456789	3459
$\{i\}^{\uparrow\downarrow}$	ag	bg	acg	$defgh$	eg	fg	g	$efgh$

Table 4. Extent and Intent given by Table 1

Attribute	**Extent**	**Intent**
a	245789	ag
b	169	bg
c	29	acg
d	349	$defgh$
e	34589	eg
f	345679	fg
g	123456789	g
h	3459	$efgh$

Table 5. List of objects and their supports given by Table 1

Object	**Support**
1	2
2	3
3	5
4	6
5	5
6	3
7	3
8	3
9	8

[INTRODUCER CONCEPT]

Let $\mathcal{K} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$ be a formal context. A formal concept $\langle A, B\rangle$ is said to *introduce* an

attribute i if $B = \{i\}\uparrow\downarrow$. It is said to *introduce* an object o if $A = \{o\}\uparrow\downarrow$.

2.2 Isolated points

[ISOLATED POINT]
Given a formal context $\mathcal{K} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$. A couple $(o, i) \in \mathcal{R}$ is called an *isolated point iif* it belongs to only one formal concept.

A formal concept is called a *mandatory concept* if it contains at least one isolated point. Khcherif et al. worked on the concept of isolated point - a point or a pair of object and attribute that can only belong to one formal concept. Because of that, to cover the data, coverage will have to include the mandatory concept,because there is no other way to cover the isolated point [8].

Both having the number of items in extent be equal to 1: ($|i\uparrow| = 1$) or number of items in intent be equal to 1 ($|i\uparrow\downarrow| = 1$) can mean that it is an isolated point. For example, in Table 6, $(1, a)$, $(1, b)$, $(1, c)$ can be considered isolated points because length of extent is 1.

Table 6. An example formal context with mandatory concepts.

	a	b	c	d	e
1	×	×	×		
2				×	×
3				×	×

Ferjani et al. proposed an approach to cover binary relation data based on isolated points that belong to single concepts first [9]. Starting from this approach means that the points where only single solution, for a conceptual coverage, is possible, will be first ones to be crossed out from later computations, together with other points that belong to the same concept. It can help reducing the number of concepts that need to be calculated early on and help to reduce algorithm execution time and number of concepts in final pertinent coverage.

It takes less computations to start by covering mandatory formal concepts with isolated points, and from there on, move to using other quality attributes for finding next pair and concept to cover [8]. In addition, Belohlavek and Trnecka have built their algorithms on finding isolated points and mandatory concepts from a binary matrix in the beginning, and then carrying on with finding other concepts to cover all the data that is still left uncovered [10]. Where they instead of going through all possible concepts, considered only columns

that maximized the value of the factor being constructed. This has helped them not to compute each and every concept for achieving pertinent coverage and achieving greater speed and memory efficiency.

In the GreEss algorithm, authors were trying to cover greedily as much uncovered objects and attributes as possible with each next concept added. This provides the most informative concepts in the early on and the last concepts considered will be the ones that possibly add the least value or do not need to be added at all because items in them are already covered [10].

2.3 Mandatory Formal Concepts

An isolated point belongs to a unique FC, called a mandatory concept (MC), that should exist in any conceptual coverage. Mandatory concepts have an important role in data mining since they allow to discover regular structures from data, based on FCA. They are qualified as mandatory because they belong to each and every conceptual coverage of a formal context (FCT) [11]. From Relational Algebra (RA) perspective, a MC contains at least one isolated point as proved by Riguet [12].

As a mathematical background, FCA and RA have been combined and used to discover regularities in data [8]. And a FC represents the atomic regular structure for decomposing a binary relation (BR). Riguet's difunctional relation [12], whose elements are defined as isolated points, describes invariant regular structures that could be used for database decomposition and Textual Features Selection (TFS) [13].

2.4 Frameworks

Many works from the past years are targeted to solve problems for better efficiency in big data analysis algorithms, minimizing storage in big data and that compare different algorithms by performance. For computing formal concepts using parallel algorithms have been proposed [14] with the con that parallel algorithms require several processors or processor cores for computations [8]. Another alternative, as for many big data tasks, MapReduce using Hadoop can be one of the solutions [15]. Also, Apache Spark has been proposed for FCA since Spark has more options for graph construction [5], [16]. The experimental analysis of the proposed work proves that the algorithm for concept generation using Spark is performing better than previous distributed approaches [5], [16].

2.5 Methods to make FCA more efficient

To solve scalability on large contexts, many have been creating more efficient algorithms [17], [15] for very computationally costly problem, since it has been proven that calculating optimal coverage of binary relation is a NP-hard problem [18].

The most popular methods for this purpose are nested line diagrams for zooming in and out of the data, conceptual scaling for transforming many-valued contexts into single-valued contexts, interestingness measures including concept stability and size of concept extent [19]. One of recent FCA strategies to reduce the computational complexity in obtaining concept lattices has been knowledge or attribute reduction [20], [21]. In case of high dimensionality and high number of objects, Binary Decision Diagrams (BDD) can be used [22]. In case of lattice reduction method, the goal of reduction is to find as small as possible set of objects and their attributes that contain the structure and information of original data, for example [20], or to simplify the concept lattice and keep the most important information [23], or choose concepts, attributes or objects considering their quality [16]. It is also has been proved how computation reduction in FCA can also be approached as a graph optimization task of finding a minimal vertex cover and how it has some performance advantages compared to FCA granular reduction algorithms [24]. Dias and Vieira proposed classification for concept lattice reduction and optimization methods mentioned above. They concluded that all methods do lead to some information loss [25].

At the same time, there is critique discerning that using only one of the methods mentioned above, will not be enough to provide good quality output. For example, discernibility matrix method (DM) for reduction method will not be enough and it is important to use a CR-method (clarification and reduction) as well, since it outperforms DM-methods [21]. In original CR-method [1], each two columns are compared if they are equal, and if they are, one of them is removed.

Kuznetsov and Obiedkov concluded that for different data with different properties, the optimal algorithm can be different [26]. It depends on how big is dataset, density of contexts, also depending on if there is a need to build a diagram graph or not. So, we can expect even a good and optimal algorithm not always perform with similarly good results and this is also in compliance with no free lunch theory where good performance of algorithm under one type of condition will mean worse performance in some other situation [27].

With increasing size of data and the number of attributes it has, the final concept lattice of the data can grow exponentially. It is a big downside when using FCA on bigger

data. One of the solutions for this problem can be not putting every concept into final pertinent coverage, but assuming that there are more informative and important concepts and finding those as long as all the data will be covered by at least one concept [28]. To select interesting concepts for the coverage stability measure can be used [29]. In addition to computational cost, there is practicality to consider - extremely large number of formal concepts in final output makes understandability of the output difficult for a human interpreter [30].

Kuznetsov and Makhalova proposed a hierarchy for concept mining algorithms that try to decrease the number of concepts that make up the final pertinent coverage [31]. They concluded that one of the best ways would be to create concepts using background information that we have about its attributes and objects. Furthermore, depending on the background information, concept will be calculated if we know in advance that the concept will be valuable. It is very similar to the approach where weight is calculated for each concept before adding it to final result [23].

Depending on how it is done, this method can have the downside of including computations to find the concept, computations to find the chosen quality measure about the concept, and then one more computation for deciding if to use the concept at all. Thus, it can leave to exponential time complexity [31].

3. Design

3.1 Description of initial algorithm

The starting point of the thesis was existing version of the QualityCover greedy FCA algorithm (hereinafter: old version) [2]. The goal of the work was to improve efficiency of the algorithm, so it could be applied on bigger data sets. In addition, the algorithm would work faster on small data, without losing quality in the conceptual coverage.

The initial idea for new improved algorithm includes the following steps (see Algorithm 1):

- Calculate mandatory concepts (Lines 4, 5);
- Calculate size for each couple not yet covered (Lines 6-10);
- Organize by size in descending manner (Line 11);
- Choose uncovered couple with highest size value (Line 13);
- Calculate the best concept for this couple by using bond value calculation (Line 15);
- Remove all couples that are covered by the newly chosen concept (Lines 17-18);
- Repeat until all the couples are covered.

Algorithm 1: QUALITYCOVER (initial idea)

Input: Formal Context $\mathcal{K} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$
Output: Conceptual Coverage $C_{\mathcal{K}}$

```
1  begin
2      C_K ← ∅;
3      COMPUTE_INTRODUCTORY_CLOSURE (K);
4      /* step 1 Computing Mandatory concepts                    */
5      C_K ← COMPUTE_MANDATORY_CONCEPTS (K);
6      if ( IS_COVERED (K)=False ) then
7          /* step 2 Computing the size of couples               */
8          forall i ∈ I do
9              forall o ∈ i^↑ do
10                 Size ← COMPUTE_SIZE ( PC_o i);
11         SORT_COUPLES (K, Size);
12         /* step 3 Covering the remaining concepts             */
13         while ( IS_COVERED (K)=False ) do
14             if ( IS_EXPLORED (o,i)=False ) then
15                 CALCULATE_BEST_FC: C_K ← C_K ∪ ⟨i^↑, i^↑↓⟩;
16                 /* remove all covered couples from the concept
                       ⟨i^↑, i^↑↓⟩                                */
17                 forall (x, y) ∈ ⟨i^↑, i^↑↓⟩ do
18                     K ← K − (x, y);
19 return C_K
```

3.1.1 Detecting mandatory concepts

Detecting the set of mandatory formal concepts (in case they exist) is given by COMPUTE_MANDATORY_CONCEPTS procedure in Algorithm 1. The descriptive pseudo-code is given by Algorithm 2 on Page 22.

Steps of the algorithm are as follows

- Iterate over the set of attributes $\mathcal{I}$.
- If the cardinality of the extent part of attribute is equal to 1, then the formal concept it belongs to is considered as a mandatory one (Line 3), all the couples in the concept are marked as covered.
- Otherwise, iterate over the extent part, looking for an object whose support is equal to the cardinality of the intent part (Lines $9 - 12$), all the couples in the concept are marked as covered.

Considering Table 6 on Page 15, couple $(1, a)$, is an isolated point, since the attribute a has an extent with only one object in it (the length is equal to 1). Thus, the concept where the couple $(1, a)$ belongs to, $1abc$, is a mandatory concept.

When considering the couple $d2$, object 2, has a support of 2 $\{de\}$, and both of its' elements have intent that is also equal to 2. Thus, the concept $\{de; 23\}$ is a mandatory concept.

3.1.2 Size calculation

Size calculation of a pseudo-concepts include:

$$Size(o, i) = \sum_{i1 \in o'} \frac{|i \uparrow \cap i1 \uparrow|}{|i \uparrow| \cdot supp(o)} \tag{3.1}$$

Sum of lengths of intersects of extents of each member of intent and original extent of x in the couple. Divided by the length of intent multiplied by length of extent of item (len int*len ext).

For example, when considering data set from Table 1 on Page 12 and couple 2;b:
Intent of 1 is bg. Extent of b is 169.

1. To get denominator of size calculation, lengths of intent and extent are multiplied. Thus, denominator is $|bg| \times |169| = 2 \times 3 = 6$
2. For numerator, for every attribute of intent, the extent of that attribute is found. Intersection of this newly-found extent and original extent of the attribute in the couple is taken. The length of this intersection is added to the numerator.
   (a) Extent of b is 169. $169 \cap 169 = 169$. $|169| = 3$
   Thus, 3 is added to the numerator.
   (b) Extent of g is 123456789. $123456789 \cap 169 = 169$. $|169| = 3$
   Thus, 3 is added to the numerator.
   Value of the numerator will be $3 + 3 = 6$.
3. Value of Size(2,b) will be $6/6 = 1$

3.1.3 Bond calculation

Bond Calculation is used to choose best FC, that maximises bond value, for a given couple (Algorithm 3, Page 23).
This calculation includes the following steps

Algorithm 2: COMPUTE_MANDATORY_CONCEPTS

Input: Formal Context $\mathcal{K} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$
Output: $C_{\mathcal{K}}$

```
1  begin
2      forall i ∈ I do
3          if ( |i↑| = 1 ) then
4              C_K ← C_K ∪ ⟨i↑, i↑↓⟩
5              /* remove all covered couples from concept ⟨i↑, i↑↓⟩
                   */
6              forall (x, y) ∈ ⟨i↑, i↑↓⟩ do
7                  K ← K − (x, y)
8
9          else
10             if (∃ o ∈ i↑ s.t. |o↓| = |i↑| ) then
11                 C_K ← C_K ∪ ⟨i↑, i↑↓⟩
12                 /* remove all covered couples from concept
                       ⟨i↑, i↑↓⟩                                  */
13                 forall (x, y) ∈ ⟨i↑, i↑↓⟩ do
14                     K ← K − (x, y)
15
16 return C_K
```

- Loop over each member of intent and find the bond value and formal concept it introduces.
- Bond calculation itself includes
  - Length of extent of item divided by length of union of extents of each member of intent.

Algorithm 3: CALCULATE_BEST_FC

Input: The couple (o, i)
Output: Best FC

```
1  begin
2      maxbond ← 0
3      forall k ∈ o^↓ s.t. k ≠ i do
4          if ( k^↑↓ ⊂ i^↑↓ ) then
5              X ← i^↑
6              FC ← ⟨i^↑, i^↑↓⟩
7              bond ← |X| / max{sup(o)|o∈X}|
8          if (bond > maxBond) then
9              maxBond ← bond
10             bestFC ← FC
11 return bestFC
```

3.2 Starting points

Starting points for the work were:

- Emphasize finding maximum amount of isolated points in the kick-off of the algorithm to reduce workload in the latter parts of the algorithm;
- Approximate size calculation to make it less computationally costly;
- Organize data points into different data structures considering their size, so there would be less sorting;
- Find ways to make bond calculation more effective;
- Implement the algorithm in R with consideration that it could be run in Spark session using SparklyR.

4. Implementation

4.1 Size calculation

Original size calculation can be seen in Equation 3.1, Page 21.

To reduce it's complexity the following methods were tested out.

4.1.1 Approximate the size calculation

In case extent of object does not include all attributes of original object in the couple, approximate its' size to 1.

$$Size(o,i) = \frac{1}{supp(o)} + \sum_{i1 \neq i \in o'} \frac{|i\downarrow \cap i1\downarrow|}{|i\downarrow| \times supp(o)} \tag{4.1}$$

This approach was omitted because it did not give desired reduction in complexity and runtime but did increase the number of formal concepts in the final coverage.

4.1.2 Calculate size and the best concept together

This version (see Algorithm 4, Page 26) included the following changes:

- Replace extent with intent in calculation, considering that intent usually consists of a smaller number of elements.
- Size calculation also includes calculation of best concept for every couple and it replaces previous bond and best concept calculation.

This option was not used because size calculation together with best concept calculation increased run time. In original QC, best concept calculation was not done until the concept was needed for final coverage for a chosen couple. So, it was done the same number of times that there were number concepts in the coverage. Calculating size and best concept together meant, the best concept was calculated for each and every couple in the data set. This increased computational load considerably.

Algorithm 4: COMPUTE_BY_COUPLE

Input: The couple (o, i)

Output: Size= the size of the pseudo-concept induced by the couple (o, i)

begin
  $sizeNumerator \leftarrow |i^{\uparrow}|$
  $X \leftarrow \emptyset$
  $bond \leftarrow maxbond$
  $bestFC \leftarrow FC$
  /* $BestFC$ is meant to be the best formal concept containing (o, i) in terms of the Bond measure */
  forall $k \in o^{\downarrow}$ *s.t.* $k \neq i$ **do**
    sparkingConcept $\leftarrow false$
    $maxBond \leftarrow \frac{|Ext(i)|}{max\{sup(o)|o \in i'\}|}$
    $bestFC \leftarrow \langle i', i" \rangle$
    /* Computing the numerator of the size */
    if ($k^{\uparrow\downarrow} \subset i^{\uparrow\downarrow}$) **then**
      $X \leftarrow i^{\uparrow}$
      $FC \leftarrow \langle i^{\uparrow}, i^{\uparrow\downarrow} \rangle$
      sparkingConcept $\leftarrow true$
    else
      if ($i^{\uparrow\downarrow} \subset k^{\uparrow\downarrow}$) **then**
        $X \leftarrow k^{\uparrow}$
        $FC \leftarrow \langle k^{\uparrow}, k^{\uparrow\downarrow} \rangle$
      else
        $X \leftarrow i^{\uparrow} \cap k^{\uparrow}$
        /* $X^{\uparrow}$ fetches that intent part of a formal concept given an extent part equal to X. */
        $FC \leftarrow \langle X, X^{\uparrow} \rangle$
    $sizeNumerator \leftarrow sizeNumerator + |X|$
    if *(sparkingConcept=false)* **then**
      $bond \leftarrow \frac{|X|}{max\{sup(o)|o \in X\}|}$
    if $(bond > maxBond)$ **then**
      $maxBond \leftarrow bond$
      $bestFC \leftarrow FC$
  $Size \leftarrow \frac{sizeNumerator}{|o^{\downarrow}| \times |i^{\uparrow}|}$
return *Size, bestFC*

4.1.3 Omit most of the calculations

with only (len int*len ext) determining the size value.

$$Size(o, i) = |i \uparrow\downarrow| \times supp(o) \tag{4.2}$$

Unlike previous calculation where bigger size indicated more valuable couple, and maximum value possible was 1, the data would be sorted in an ascending value hereafter and there is no maximal possible value. Size of the dataset is the best predictor of how big the size value can be.

When a couple has smaller intent and extent it can belong to a smaller number of possible concepts - so the smaller the size, the more valuable the couple is and should be considered among the first ones. Couples that have very big intent or extent values can possibly be covered by many different concepts and should be last to be considered when searching for optimal conceptual coverage.

This is the version of the size calculation that was chosen for the final version of the algorithm. It included the least calculations and proved to generate the same conceptual coverage of the data like the original QC algorithm with minor differences in what order couples are looped through and concepts are calculated.

4.2 Mandatory concepts

Separated calculation of mandatory concepts based on isolated points was abandoned. The purpose of calculating mandatory concepts first was to 'cross out' all the data covered by these concepts first, so the these point of data would be out of the way. While it is valid idea to do this, calculating mandatory concepts includes multiple loops (see Algorithm 2, Page 22) and in bigger or more dense data sets with no mandatory concepts, it will bring bigger losses in run-time than calculating the same concepts together with all other concepts. Mandatory concepts will become covered still and most likely will be among one of the first ones to do so, because of their characteristic structure that also affects their size value.

For example, if there is a concept where extent includes only one object, characteristic that makes this concept a mandatory one, size calculation will generate smaller size value, since size is only equal to the length of intent. And this will result the couple to be considered among one of the first ones.

Omitting this calculation did not increase run time in smaller data sets. On medium and bigger sized data sets omitting it did decrease run time. Compared to the original CQ algorithm, this change did not increase the number of concepts in the coverage, but created minor differences in what order couples are looped through and concepts are calculated.

4.3 Bond calculation

Bond calculation was used to choose between pseudo-concepts to add to final pertinent coverage. Bond calculation together with best concept calculation was omitted altogether. This was done because for each couple, the best possible concept is always the extent and intent of the attribute. In addition, as a result of previous bond calculation the same intent and extent were always chosen in the end, thus it proved to be a pointless calculation.

Bond calculation and best pseudo-concept choice were based on what attributes were shared between objects (Algorithm 3, Page 23). Chosen elements to the intent of concept needed to include all the same objects in their extents that were present in the original attributes extent. In that case, the Bond value was maximized and those attributes were chosen to intent of the concept.

Thus, it can be derived, that extent of any couple will we the extent of the attribute in the considered couple. And if there are other attributes whose extents include all the same objects, those attributes will be included in the intent of that couple. In case there are no other objects that include all the same attributes, the concept will include only original object from the couple, and all of its attributes.

For example, when we consider couple $(1, h)$ in Table 2, Page 14, pseudo-concepts it generates are {349;defgh} and {3459;efgh}. The latter would be chosen as a result of bond calculation because it maximises the extent in the concept. The same will be chosen because this pseudo concept includes all the objects that have the attribute in the considered couple, that introduced the concept, includes.

4.4 Data structure

Data structure considered in the beginning was a table that included all central information (each object, its extent and intent) like in Table 4, Page 14. And structure for attributes that support each object, like in Table 5, Page 14. This kind of setup enables running algorithm itself relatively fast, but requires a lot of pre-calculation and multiple data tables. Calculation of intent for each item has computational complexity of $O(n \times m)$ and this

calculation is not needed for each and every item later in the algorithm.

In final version, only the boolean matrix that represented data was used. In case initial data was transactional, it was still turned into this kind of matrix.

Formal context from Table 1, Page 12 turned into boolean matrix.

Table 7. An example formal context as a Boolean matrix.

	1	2	3	4	5	6	7	8
1	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE
2	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE
3	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE
4	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE
5	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE
6	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE
7	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE
8	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	TRUE	FALSE
9	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE

For size calculation the matrix was transposed to a table of couples, where also size value was added (see Table 8).

Table 8. Couples and their sizes given by Table 7

Index	**Object**	**Attribute**	**Size**	**Index**	**Object**	**Attribute**	**Size**
[1,]	1	2	6	[19,]	4	8	24
[2,]	2	3	6	[20,]	5	5	25
[3,]	6	2	9	[21,]	5	6	25
[4,]	3	4	12	[22,]	2	7	27
[5,]	8	5	15	[23,]	6	7	27
[6,]	6	6	15	[24,]	7	7	27
[7,]	7	6	15	[25,]	8	7	27
[8,]	9	3	16	[26,]	5	1	30
[9,]	3	8	16	[27,]	4	5	30
[10,]	2	1	18	[28,]	4	6	30
[11,]	7	1	18	[29,]	9	8	32
[12,]	8	1	18	[30,]	4	1	36
[13,]	4	4	18	[31,]	3	7	36
[14,]	1	7	18	[32,]	9	5	40
[15,]	3	5	20	[33,]	9	6	40
[16,]	5	8	20	[34,]	5	7	45
[17,]	9	2	24	[35,]	9	1	48
[18,]	9	4	24	[36,]	4	7	54
				[37,]	9	7	72

No intent or extent was calculated prior to size calculation, since they are easily derivable from the central matrix. Size table was ordered in an ascending manner.

Before starting calculating concepts additional matrix is created to keep track what is already covered of the data. The matrix has same dimensions like central data matrix, but includes no information initially.

For each iteration, the next best uncovered couple is chosen (considering the size value). Concept for the couple will be intent and extent of the item in the couple. After each iteration, discovered couples are added to the coverage matrix and algorithm stops when both matrices are identical.

Example data set introduced in Table 1, Page 12 will produce size table like Table 8, after the table has been sorted by Size value. In first iteration, the first couple is taken ($1; 2$ or $1; b$ if attributes are represented as letters). Extent and intent of the attribute are derived as the formal concept ($\{bg; 169\}$ or $\{27; 169\}$). This concept is added to the output of the algorithm and all the couples in this concept are marked as covered.

After first iteration, the next couple in the size table is considered, if the couple is not covered, the formal concept from it is added to the output. In this example, the next couple is ($2; 3$ or $2; c$) and it brings a formal concept ($\{acg; 29\}$ or $\{137; 29\}$) to the output. In the third iteration, couple ($6; 2$ or $6; b$) is considered. Since it is covered already, no concept is derived from it, and iteration heads to the next couple. The concepts produced by the algorithm are as depicted in Table 9, in the order they are derived.

Table 9. Output of the new version of the algorithm in the order concepts are derived

Index	**Couple**	**Concept**
1	$2; 1$	$\{27; 169\}$
2	$3; 2$	$\{137; 29\}$
3	$4; 3$	$\{4578; 349\}$
4	$5; 8$	$\{57; 34589\}$
5	$6; 6$	$\{67; 45679\}$
6	$1; 7$	$\{17; 245789\}$
7	$8; 5$	$\{578; 3459\}$

4.5 Framework

Chosen language for implementing the algorithm was R, developed in RStudio. For framework, SparklyR was chosen initially because it enables to use Spark for big data analysis in R. Downside for using it was, that SparklyR only accepts data frame data structure, that is one of the slowest and least efficient one in R.

Using this data structure would mean changing central data structure the algorithm was built on (matrix of T/F values) and using a more complex data structure where a lot of values are pre-calculated before the start of the algorithm without knowing if they will be used or not. This approach slowed down the algorithm considerably and the option to parallelize the computations between clusters did not outweigh the additional complexity and increased runtime in the used example data sets. For these reasons, the final version of the algorithm was not created considering usability in Spark session.

4.6 Theoretical Complexity Analysis

In the old version, the size calculation required additional loops over each extent and intent. For adding concepts to output, bond calculation included double loop over intent for each concept. However, since this was done only once per every concept, in result this did not prove to be costly yet, with tested medium-sized datasets, where number of concepts was not that high.
The old version has complexity of
$O((1 \times \#Couples \times |i \uparrow\downarrow| \times |i \uparrow|)$ + $(1 \times \#Coverage \times |i \uparrow\downarrow|^2 \times))$.

The new version of the algorithm loops through the data twice - first time for calculating size value for each couple (it is to be assumed one size calculation is constant). Then, the couples are reordered and looped through second time. In case given couple is not covered yet, FC is calculated from it, and all the newly introduced couples are marked as covered. Thus, the new version has complexity of
$O((1 \times \#Couples)$ + $(1 \times \#Coverage))$.

5. Evaluation

The new solution is written in R, implemented and executed on a Core i5 PC, CPU 2.3 GHz, with 16 GB of RAM and Windows operation system. Quality of the new and old version of the algorithm was assessed by number of concepts and the time it took to calculate them. Both of the versions resulted the same number of concepts and were considered to be equal in that measure.
The previous version of the algorithm was implemented as a Python package with C++ extension, run on an Ubuntu virtual machine with 4096 MB RAM memory.

Table 10 introduces used benchmark datasets from UC Irvine Machine Learning Database Repository and Hypergraph Dualization Repository and the number of formal concepts in the pertinent coverage, calculated by the QualityCover algorithm.

Table 10. Formal concepts, size of coverage with QualityCover algorithm and the dimensions of the datasets

Dataset	$X \times Y$ **Dimensions**	**# Couples**	**# Coverage**
shuttle-landing-control	15 X 24	105	15
adult-stretch	20 X 10	100	9
lenses	24 X 12	120	12
zoo	101 X 28	862	26
hayes-roth	132 X 18	660	17
servo	167 X 19	668	19
post-operative	90 X 25	720	22
balance-scale	625 X 23	3125	23
flare2	1066 X 32	10 660	29
car	1728 X 21	10 368	21
breast-cancer-wisconsin	699 X 110	6990	92
house-votes-84	435 X 18	3856	18
SPECT-test	187 X 23	1644	23
audiology.standardized	26 X 110	1768	36
tic-tac-toe	958 X 29	9580	29
nursery	12 960 X 31	116 640	30
mushroom	8124 X 119	18 6852	109
soybean-large	307 X 133	10 744	102
dermatology	366 X 130	12 078	128
chess	3196 X 75	118 252	72
bms2 800	3339 X 62	206 936	52
bms2 400	3339 X 237	790 911	154
ac 90k	336 X 4322	1 411 815	37
dual-matching34	34 X 131 072	2 228 224	34
ac 70k	336 X 10 968	3 576 449	48
bms2 100	3339 X 2591	8 641 922	907
bms2 50	3339 X 6946	23 164 305	1545
bms2 30	3339 X 17 315	57 732 266	2040
ac 30k	442 X 135 439	58 291 353	85

Table 11 represents results of running the two new versions of the QualityCover algorithm and old version runtime. Runtime1 represents algorithm with central boolean matrix algorithm, Runtime2 algorithm with central data frame that is SparklyR friendly but requires more computations. Runtime3 is old version of the algorithm, implemented as a Python package with C++ extension.

Datasets were tested in the order of their size in kilobytes, starting from the smallest one. After dataset run time with an algorithm exceeded reasonable time, the algorithm was not run with bigger datasets. This is marked as $'-'$ in the Table 11.

It can be seen that the new version of the algorithm outperforms old version on most

datasets. The old version is faster on datasets that have small number of concepts in final coverage regardless of their size, and slower if coverage includes more concepts. Runtime of the new version of the algorithm increased more steadily together with increase in number of couples and was less influenced by the final coverage size.

Table 11. Formal concepts from UCI repository and runtimes of three versions of algorithm in seconds. '-' means algorithm was not run on the dataset.

Dataset	**Runtime1(s)**	**Runtime2(s)**	**Runtime3(s)**
shuttle-landing-control	0.086	2.1	**0.060**
adult-stretch	0.067	4.7	**0.050**
lenses	**0.060**	3.9	0.063
zoo	**0.108**	42.1	0.462
hayes-roth	**0.090**	19.3	0.243
servo	**0.085**	16.0	0.274
post-operative	**0.094**	37.4	0.297
balance-scale	**0.148**	100.3	0.861
flare2	**0.334**	597.2	2.187
car	**0.353**	439.5	1.990
breast-cancer-wisconsin	**0.474**	515.8	4.005
house-votes-84	**0.141**	227.2	0.883
SPECT-test	**0.100**	115.8	0.480
audiology.standardized	**0.167**	123.6	1.814
tic-tac-toe	**0.265**	600.8	2.146
nursery	7.559	6886.0	**1.509**
mushroom	10.880	24 297	**5.757**
soybean-large	**0.493**	2295.1	5.985
dermatology	**0.539**	2647.0	8.033
chess	**3.914**	26 865	84.861
bms2 800	**14.489**	-	11 116
bms2 400	**88.671**	-	11 268
ac 90k	**46.194**	-	139.592
dual-matching34	2669.6	-	**16.308**
ac 70k	**225.572**	-	913.259
bms2 100	**1427.0**	-	110 246
bms2 50	**4767.9**	-	-
bms2 30	**15 885**	-	-
ac 30k	**81 596**	-	-

The new version of the algorithm decreased complexity of size calculation considerably. Since this is the only calculation that has to be done for each and every couple in the data, it granted considerable decrease in complexity. At the same time, relatively more load was given to calculation of concepts. This was not anticipated, since computational complexity was reduced as well. Fortunately this part of the algorithm is not run for each couple, but only the number of times new concepts are added to the final coverage.

Table 12, Page 36 represents results of running the new version of the QualityCover algorithm and run times per part of the algorithm - calculation of size, calculation of final concepts, and total run time.

In addition of size and concepts, algorithm also spends some time on preparation (reading in the script, creating the central matrix). These were not studied separately, since they were not altered in this work and take relatively little time compared to other operations.

Table 12. Runtimes in seconds by stages of the new version of the algorithm.

Dataset	**Size(s)**	**Concepts(s)**	**Total Runtime(s)**
shuttle-landing-control	0.030	0.008	0.086
adult-stretch	0.048	0.003	0.067
lenses	0.016	0.003	0.060
zoo	0.053	0.016	0.108
hayes-roth	0.023	0.011	0.090
servo	0.025	0.006	0.080
post-operative	0.020	0.011	0.094
balance-scale	0.044	0.028	0.148
flare2	0.139	0.098	0.334
car	0.160	0.082	0.353
breast-cancer-wisconsin	0.113	0.239	0.474
house-votes-84	0.053	0.028	0.141
SPECT-test	0.023	0.017	0.100
audiology.standardized	0.043	0.081	0.167
tic-tac-toe	0.109	0.070	0.265
nursery	5.773	0.855	7.559
mushroom	6.662	2.860	10.880
soybean-large	0.101	0.277	0.493
dermatology	0.164	0.277	0.539
chess	2.099	1.072	3.914
bms2 800	10.562	3.825	14.489
bms2 400	62.093	26.422	88.671
ac 90k	36.968	9.061	46.194
dual-matching34	2624.3	44.891	2669.6
ac 70k	191.235	33.720	225.572
bms2 100	706.172	718.675	1427.0
bms2 50	2454.2	2310.7	4767.9
bms2 30	8859.5	7018.8	15 885
ac 30k	80 628	958.895	81 596

Because the old and the new version of the algorithms were implemented on different languages and ran on different systems, one on one comparison of each dataset run time will not be valid. Because of this reason, only increase of run times should be compared. The increase in run times as datasets become bigger and more dense is lowest in the new version of QualityCover, as presented on Figure 1, Page 37.

Figure 1 represents total runtimes of the two versions of the algorithm. In the new version of the algorithm grows with slower rate when number of couples increases when compared to the old version. With large datasets the old version runtime increased more and the old algorithm was not ran on the last few datasets there it exceeded reasonable runtime, so it is not possible to compare runtimes on last few larger datasets. Even though the old version was not able to run the bigger datasets, it can be derived from the rise angle of runtimes graph that the old versions runtime increases more with growing data size than the new

version. Thus it can be assumed, that the new version is better equipped to handle bigger datasets.

Logarithmic scale was used on both axis of the plot because the results were skewed towards a few large values.

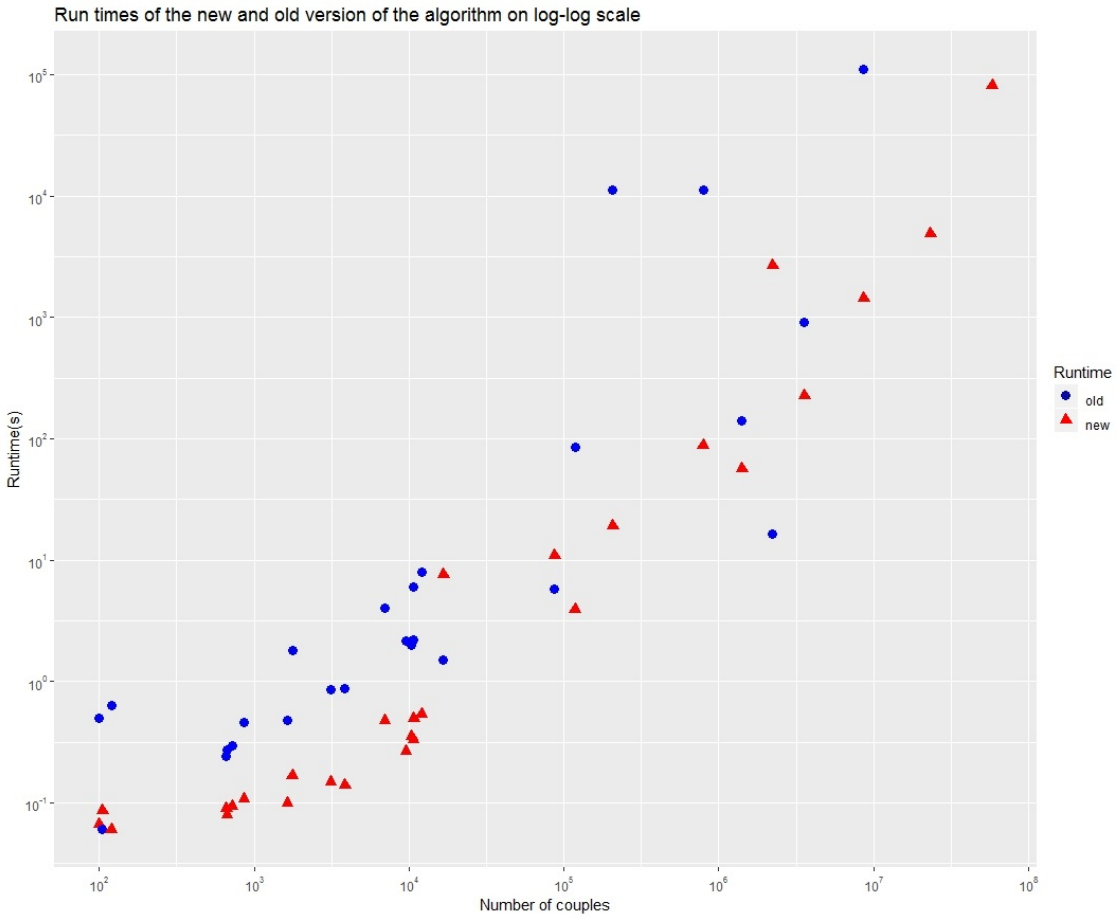

Figure 1. Runtimes of old and new algorithm compared on a log-log scale

6. Summary

For each algorithm, the challenge is to reduce computational complexity from exponential to linear and to use data structures that support fast creation, filtering, and transformations, and at the same time enable to do the calculations with low computational complexity. It can be that some of the data structures are easy to create, but slow to transform. Fast in computations but at the same time do not enable the type of calculations that are the most rational. In these cases, the decision what combination of data structure and calculation types to use has to depend on the intention and type of the algorithm, and type and size of data.

In case of moderate sized data sets, the computation time of $O(n^2)$ complexity might not rise so high yet, and it is possible to choose between computational complexity and preferred data structures without having any right or wrong answers. When data sets are bigger, it can be anticipated there will be also considerable increase in the calculation time. In this case, using the data structure that enables the fastest computations can be a wise choice, if the data structure will only be created once but calculations with it will be multiple.

With rising data size, there can also be a need to distribute memory and calculations between different nodes, and this is also case where the situation - framework chosen - dictates what type of data structure and calculations to use, to enable parallelization and distribution. In this thesis, the algorithm was modified to enable running it in Spark session using SparklyR. This attempt was not a successful one, since approved data structure (data frame) did not allow the type of calculations that enabled to have fast algorithm computations. As a result of modifying the algorithm for this computational complexity and run time increased considerably and this trade-of made it not reasonable to continue with this attempt further. Thus, parallelization was omitted, because the increase in complexity of calculations for the purpose of parallelizing them between clusters did not prove to be beneficial.

In addition, the density and disposition of data can hugely change effectiveness of an algorithm. In this thesis, calculation of isolated points and mandatory concepts was omitted, because the data sets with average or high density did not provide any practical

example of usefulness of these calculations. But it is very likely, that with low density data, where many isolated points are present, the calculation of mandatory concepts first can be a good solution. And as a result, the size calculation could be omitted instead with very few concepts left after finding mandatory ones.

It became evident that the two versions of the algorithm performed differently on different types of datasets. The old version outperformed the new one on a big dataset where final coverage is a small number of concepts. The new version was better when final coverage includes more concepts. So there are advantages of using the old version, if it is known there will not be many concepts in the final coverage. Otherwise the new version, where runtime has stable dependency on number of couples in the data, can be used.

The new version of QualityCover algorithm includes considerably simplified Size calculation and has removed Bond and best concept calculation. As a result of this, the run time increases in linear manner together with dataset size (number of couples the data set includes). The new version of the algorithm is able to run data sets with the size of millions of couples in matter of minutes to hours with the same quality as the previous version.

The most complex and time consuming part of the calculation remains to be the Size calculation. To furthermore improve the speed and complexity, it should be considered removing the size calculation in total, and accepting some increase in number of concepts in the final coverage, but being able to process bigger data sets.

Bibliography

[1] Bernhard Ganter, Rudolf Wille, and C. Franzke. *Formal Concept Analysis: Mathematical Foundations*. 1st. Berlin, Heidelberg: Springer-Verlag, 1997. ISBN: 3540627715.

[2] Amira Mouakher and Sadok [Ben Yahia]. "QualityCover: Efficient binary relation coverage guided by induced knowledge quality". In: *Information Sciences* 355-356 (2016), pp. 58–73.

[3] Alexandre Albano and Bogdan Chornomaz. "Why concept lattices are large: extremal theory for generators, concepts, and VC-dimension". In: *International Journal of General Systems* 46.5 (2017), pp. 440–457.

[4] Radim Belohlavek and Martin Trnecka. "Basic Level of Concepts in Formal Concept Analysis". In: vol. 7278. May 2012, pp. 28–44.

[5] Raghavendra K Chunduri and Aswani Kumar Cherukuri. *Scalable Formal Concept Analysis algorithm for large datasets using Spark*. 2018.

[6] Rudolf Wille. "Restructuring Lattice Theory: An Approach Based on Hierarchies of Concepts". In: *Ordered Sets*. Ed. by Ivan Rival. Dordrecht: Springer Netherlands, 1982, pp. 445–470. ISBN: 978-94-009-7798-3.

[7] Slim Chaabane, Chokri Elhechmi, and Mohamed Jaoua. "Error estimates in smoothing noisy data using cubic B-splines". In: *Comptes Rendus Mathematique - C R MATH* 346 (Feb. 2008), pp. 107–112.

[8] Raoudha Khchérif, Mohamed Mohsen Gammoudi, and Ali Jaoua. "Using difunctional relations in information organization". In: *Information Sciences* 125.1 (2000), pp. 153–166.

[9] Fethi Ferjani et al. "Formal context coverage based on isolated labels: An efficient solution for text feature extraction". In: *Information Sciences* 188 (2012), pp. 198–214.

[10] Radim Belohlavek and Martin Trnecka. "From-below approximations in Boolean matrix factorization: Geometry and new algorithm". In: *Journal of Computer and System Sciences* 81.8 (2015), pp. 1678–1697.

[11] R. Belohlavek and V. Vychodil. “Discovery of optimal factors in binary data via a novel method of matrix decomposition”. In: *Journal of Computer and System Sciences* 76 (1) (2010), pp. 3–10.

[12] J. Riguet. “Relations binaires, fermetures et correspondances de Galois”. In: *Bull. Soc. Math. France 78* (1948), pp. 114–155.

[13] S. Elloumi, F. Ferjani, and A. Jaoua. “Using minimal generators for composite isolated point extraction and conceptual binary relation coverage: Application for extracting relevant textual features”. In: *Information Sciences* 336 (6 2016), pp. 129–144.

[14] Petr Krajca, Jan Outrata, and Vilem Vychodil. “V.: Parallel Recursive Algorithm for FCA”. In: *Palacky University, Olomouc*. 2008, pp. 71–82.

[15] Petr Krajca and Vilem Vychodil. “Distributed Algorithm for Computing Formal Concepts Using Map-Reduce Framework”. In: vol. 5772. Aug. 2009, pp. 333–344.

[16] Gerd Stumme. “Efficient Data Mining Based on Formal Concept Analysis”. In: July 2002.

[17] Simon Andrews. “In-Close, a fast algorithm for computing formal concepts”. In: 483 (Jan. 2009).

[18] Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. USA: W. H. Freeman Co., 1990. ISBN: 0716710455.

[19] Jonas Poelmans et al. “Formal Concept Analysis in knowledge processing: A survey on models and techniques”. In: *Expert Systems with Applications* 40.16 (2013), pp. 6601–6623.

[20] Jesús Medina. “Relating attribute reduction in formal, object-oriented and property-oriented concept lattices”. In: *Computers Mathematics with Applications* 64.6 (2012), pp. 1992–2002.

[21] Jan Konecny and Petr Krajča. “On attribute reduction in concept lattices: Experimental evaluation shows discernibility matrix based methods inefficient”. In: *Information Sciences* 467 (2018), pp. 431–445.

[22] Sebastiao M. Neto, Luis E. Zárate, and Mark A.J. Song. “Handling high dimensionality contexts in formal concept analysis via binary decision diagrams”. In: *Information Sciences* 429 (2018), pp. 361–376.

[23] Radim Belohlávek and Juraj Macko. “Selecting Important Concepts Using Weights”. In: May 2011, pp. 65–80.

[24] Jinkun Chen, Jusheng Mi, and Yaojin Lin. “A graph approach for knowledge reduction in formal contexts”. In: *Knowledge-Based Systems* 148 (2018), pp. 177–188.

[25] Sérgio M. Dias and Newton J. Vieira. “Concept lattices reduction: Definition, analysis and classification”. In: *Expert Systems with Applications* 42.20 (2015), pp. 7084–7097.

[26] Sergei O. Kuznetsov and Sergei A. Obiedkov. “Comparing performance of algorithms for generating concept lattices”. In: *Journal of Experimental & Theoretical Artificial Intelligence* 14.2-3 (2002), pp. 189–216.

[27] D. H. Wolpert and W. G. Macready. “No free lunch theorems for optimization”. In: *IEEE Transactions on Evolutionary Computation* 1.1 (1997), pp. 67–82.

[28] H. Gmati and A. Mouakher. “Fast and Compact Cover Extraction from Big Formal Contexts”. In: *2018 IEEE 27th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*. 2018, pp. 209–212.

[29] Amira Mouakher, Oumaima Ktayfi, and Sadok Ben Yahia. “Scalable computation of the extensional and intensional stability of formal concepts”. In: *International Journal of General Systems* 48.1 (2019), pp. 1–32.

[30] Michael D. Rice and Michael Siff. “Clusters, Concepts, and Pseudometrics”. In: *Electronic Notes in Theoretical Computer Science* 40 (2001). MFCSIT2000, The First Irish Conference on the Mathematical Foundations of Computer Science and Information Technology, pp. 323–346.

[31] S.O. Kuznetsov and T. Makhalova. “On interestingness measures of formal concepts”. In: *Information Sciences* 442-443 (2018), pp. 202–219.

Appendices

Appendix 1 - Code

two new versions of the algorithm can be found:
https://github.com/kadmok/QC4

Appendix 2 - Previous version of the algorithm

previous version of QualityCover algorithm can be found here:
https://pypi.org/project/quality-covers/

Appendix 3 - Datasets

The data sets used in this thesis can be found in Hypergraph Dualization Repository:
$http://research.nii.ac.jp/\sim uno/dualization.html$

and UC Irvine Machine Learning Repository:
https://archive.ics.uci.edu/ml/datasets.php