PRCA - A Parallel Relational Concept Analysis Framework

Ines Moosdorf[1], Adrian Paschke[1], Alexandru Todor[1], Jens Dietrich[2], Hans W. Guesgen[2]

[1] Freie Universitaet Berlin, Germany
{moosdorf, paschke, todor} AT inf.fu-berlin.de
[2] Massey University, New Zealand
{j.b.dietrich, h.w.guesgen} AT massey.ac.nz

Abstract. Relational Concept Analysis (RCA) extends standard Formal Concept Analysis (FCA) by taking relations between objects into account. The scalability of RCA learning on top of huge amounts of sensor data is a challenge for applications such as smart home system monitoring in ambient assisted living environments. One possible approach to improve scalability is to exploit the capabilities of modern parallel computing architectures such as multi-core CPUs. In this paper, we propose PRCA (Parallel Relational Concept Analysis), a novel framework for parallel relational concept learning. [1]

1 Introduction

In the next few years, the world population will be ageing dramatically: the percentage of people over 65 will grow to more than 25% and average life expectancy will increase to 75. This will have a particular impact on the health care systems since there will not be enough health care workers to adequately attend to all elderly people. Especially elderly people who suffer from cognitive impairment are known to remain independent for longer when living in their own home. Despite their cognitive shortfalls these people are still able to perform everyday activities like washing, grooming and eating. These activities are called Activities of Daily Living (ADLs) and it has been demonstrated that they will be retained for a longer period if the elderly people remain in their familiar environment. [1] The application scenario of the research described in this paper is in the field of smart home systems that support elderly cognitive impaired people to stay independently in their own houses as long as possible with just minimal support from health care services. A smart home system monitors inhabitants with unobtrusive sensors, identifies particular behaviors and notifies health workers if an abnormal behavior, such as taking medication in the middle of the night, occurs. Abnormal behaviour detection is a core feature of smart home systems.

[1] This colloborative work between the Massey University Smart Environments (MUSE) group and der Corporate Semantic Web (CSW) group at Freie Universitaet Berlin has been partially supported by the Royal Society of New Zealand and the BMBF in the User Guided Semantic Content Enrichment project and the "InnoProfile-Transfer Corporate Smart Content" project

Concepts of normal behaviour are learned from positive and negative training data. New behaviours are classified using the concepts of normal behaviours.

Formal concept analysis (FCA) is a simple yet powerful and elegant representational concept learning mechanism introduced in [2]. The explicit separation between intention and extension makes FCA an ideal platform for symbolic machine learning: training data represents concept extensions from which concept intentions are being inferred that can later be used as classifiers.

Relational Concept Analysis (RCA) was first proposed in [3]. It extends standard FCA by taking relations between objects into account.

Given the amount of data that has to be processed by modern applications, the scalability of learning is of particular concern. One approach to tackle this problem is to take advantage of parallel computing platforms (multicore CPUs, GPUs, cloud computing), and to parallelise learning algorithms. In this paper, we present PRCA (Parallel Relational Concept Analysis), a novel framework for parallel concept learning. PRCA is based on RCA [3, 6, 4] in order to improve the expressiveness of pure FCA, and uses multicore CPUs to improve scalability. We evaluate the accuracy of the learning algorithm and the performance gains on a set of benchmark data sets widely used in description logic learning, and compare results with existing description logic learners (DLLearner [2], PARCEL [3]). The results indicate that on most data sets, PRCA outperforms DL-based learners. PRCA also provides a wide range of configuration options that can be used to implement project specific heuristics.

2 Related Work

Our research is mainly based on the mathematical foundations of FCA as described in [2]. Standard FCA is restricted to data sets thata are either already represented as binary relations or that can be easily transformed into such a representation using method such as conceptual scaling [2]. We are not interested in "pure" FCA-based learning, but in learning from data sets that also contain binary relations between objects. These data sets cannot be transformed via conceptual scaling and hence cannot be processed by standard FCA algorithms.

Huchard et al. have proposed Relational Concept Analysis (RCA) [3], a method that extends FCA for the purpose of taking relations between objects into account. PRCA is based on ideas from the relational data model, relational scaling and iterative relational property generation. RCA aims to generate complete lattices of data sets. This leads to scalability issues since the size of the concept lattices grows rapidly with the number of relationships between contexts. Our idea differs from RCA in that we do not focus on complete lattice creation but on building lattices of selected properties that are good for dividing positive from negative examples. Furthermore, in PRCA we try to address the scalability problem by using concurrent computing. In [5], Kuznetsov proposes the symbolic

[2] http://aksw.org/Projects/DLLearner.html
[3] http://code.google.com/p/parcel-2013/

machine learning method JSM in terms of FCA learning from positive and negative examples. This method consists of two parts, learning hypotheses from positive and negative examples and a classification of undetermined examples by the learned hypotheses. This method is adapted and employed in the PRCA framework. Our hypotheses generation algorithm differs from the approach presented in [5], in that it is not based on two separate formal contexts for positive and negative examples, but on a combined formal context which is processed in parallel. When a concept extension contains only positive examples, then its intention is regarded as positive hypothesis. When a concept extension contains only negative examples, then its intention is regarded as negative hypothesis. The difference between the RCA and PRCA approach is that after creating the formal context PRCA generates all possible combinations in the relational scaling step, instead of only relations to concepts as in RCA. In PRCA the relational properties are combinations of relational and basic information of the relational context. As a result PRCA creates more relational properties then RCA, because the concepts that already pre-group the data are not used. Although this seems to be a disadvantage on the first glance, it is necessary for the parallelization of the algorithm. Otherwise, there would be step dependency as in RCA.

3 Parallel Relational Concept Analysis Framework

The main approach of the Parallel Relational Concept Analysis (PRCA) framework is the parallelization of the scaling step of object-object relations to relational properties and the integration of basic and relational properties into one concept lattice. The aim is to learn positive and negative hypotheses. PRCA does not generate a complete lattice with all relational properties, but only finds suitable relational properties that are good for dividing the positive from the negative examples. Therefore, the PRCA framework iteratively generates new relational properties from the relational information given in the data set and combines them with the basic properties in one lattice until sufficient hypotheses are found.

3.1 Basic Steps

Figure 1 depicts the basic steps of the PRCA framework. The input of the framework are *relational contexts*. We define a relational context C as a pair (K, R) consisting of a set of formal contexts $K = \{K_i\}$, whereby each context $K_i = (O_i, P_i, I_i)$ has objects O_i, properties P_i and a relationship I_i between these objects and properties; and object-object relations $R = \{R_i\}$, with $R_i \subseteq O_1^i \times O_2^i$, associating objects from two contexts.

Each basic relation R_j has a source (relational) context C_i and a target (relational) context C_k, both source and target can be identical. The main (relational) context is a learning problem with multiple contexts. One context is the main context. This is the context that contains the positive, negative (and unknown) labelled objects of the learning problem.

The learning algorithm consists of several steps.

In the *Generation of Relations* step, relations are generated based on the basic relations and properties of the relational contexts. The generator yields basic relations as well as new *composite relations*. A composite relation is the result of composing two relations or one relation and an additional post condition. Different generation operators like joins, intersections and conditional joins exist. For instance, the relation join is defined as $R_{j.k} := R_j.R_k$. Applying a postcondition creates a new relation by applying filters based on properties in the target context.

In the following *Relational Scaling* step, these relations are then scaled to *relational properties*. Different Scaling operators exist: existential, universal and cardinality restricted. There are also different scaling directions: left and right direction (has/is).

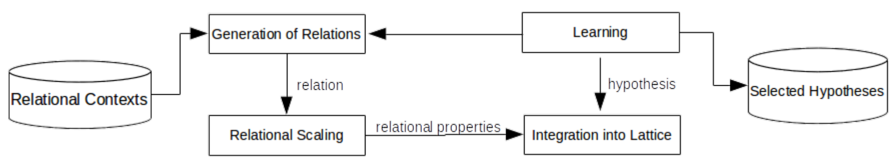

Fig. 1. The basic steps of the Parallel Relational Concept Analysis (PRCA) framework

In the *Integration into Lattice* step, the relational properties are integrated with the basic properties into one lattice to check for new positive hypotheses, i.e. intentions of concepts that contain only positive examples)=, All new formal concept intentions being hypotheses are selected and stored.

In the *Learning* step, it is checked if all positive and negative examples are covered by at least one positive, or negative hypothesis, respectively. If all positive examples of the main context are covered by at least one hypothesis, the best hypotheses are selected and returned as result of the learning process.

3.2 Components of the PRCA Framework

Figure 2 shows the realization of the basic steps of the PRCA framework. Input data are one or more *relational contexts*. In addition to formal contexts and the set of basic relations, each relational context contains a set for storing relational properties that are scaled during relational scaling step. *Uncovered positive examples* is an agenda containing all positive examples. When a new positive hypothesis is found , the examples covered by this hypothesis will be removed from the agenda. *Hypotheses* is a container shared by all parallel running workers to collect the learned hypotheses. The *global property pool* contains all basic properties and relational properties which are relevant for building the lattice. The parallel running workers scale new relational properties and add them to the global property pool. The *relation generator* generates new relations using the operations described above.

The steps *Relational Scaling* and *Integration into Lattice* are realized by parallel running *worker* threads. In each *working step*, the worker requests a new relation from the relation generator, scales new relational properties and integrates them into one lattice with the basic properties and previously scaled properties.

Step by step the lattice is extended by new formal concepts. When a new formal concept covers only positive examples the worker updates the agenda of *uncovered positive examples* and stores the intention of the concept as new hypothesis in its *local hypotheses pool*.

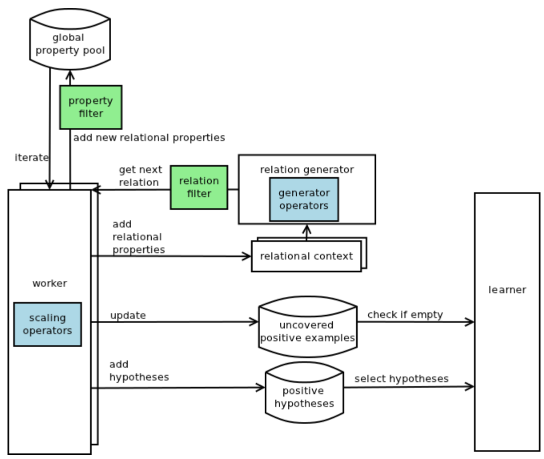

Fig. 2. Components and Configuration possibilities of the PRCA framework

The worker adds the new relational properties to the *global property pool* and to the *relational property pool* of the respective relational context.

The *Learning* step is done by the *learner*. The learning is finished when all examples have been removed from the agenda, i.e., all examples are covered. Then, each worker adds its found hypotheses to the global hypotheses pool. Then, the learner selects the best hypotheses to create the result of the learning process. A set cover algorithm is used for this purpose. By default, we use a simple greedy algorithm that selects hypotheses covering the most (not yet covered) examples. The use of other algorithms is possible as well.

3.3 Configuration

The purpose of the framework is to provide a tool for developing and evaluating different strategies of RCA learning. The framework offers several variability points and configuration options that can be combined. In particular, this includes operators to compose relations, filters and set coverage algorithms to select hypotheses.

Scaling operators define the type of the relational scaling. Multiple scaling operators can be defined at the same time. Each worker applies all the defined scaling operators to the given relation.

Property filter: The workers add their newly scaled relational properties to the global property pool. However, not all relational properties are relevant. To reduce the number of irrelevant properties, the configured *property filter* controls which properties are added to the global property pool.

A *relation filter* filters the generated relations. When the relation generator is requested for a next relation it will only return relations that pass the filter.

4 Evaluation

4.1 Methodology

The evaluation is conducted with a ten 10 fold cross validation. The evaluation metrics are:

Learning Time: duration from context to selected hypotheses.

The hypotheses learned by the prototypes are used to classify unknown examples. To determine the quality of the learned hypotheses their *correctness, completeness* and *accuracy* are measured. Therefore a set of positive and negative labelled examples is used. Each example of the data set is classified by the learned hypotheses. The results of the classification are compared to the original labels of the examples. *Correctness* determines the ability of the learned hypotheses to classify negative examples as negative. *Completeness* determines the ability of the learned hypotheses to classify positive examples as positive. *Accuracy* combines correctness and completeness. It determines the ability of the learned hypotheses to classify undetermined examples correctly.

$$correctness = \frac{|negative\ examples\ classified\ as\ negative|}{|all\ negative\ examples|}$$

$$completeness = \frac{|positive\ examples\ classified\ as\ positive|}{|all\ positive\ examples|}$$

$$accuracy =$$

$$\frac{|negative\ examples\ classified\ as\ negative|\ +\ |positive\ examples\ classified\ as\ positive|}{|all\ examples|}$$

Definition length: A further quality property of the learned hypotheses is their length. A shorter hypothesis is regarded as better than a longer one describing the same objects.

- property length: To compute the length of a property the containing relations, properties and scaling operators are counted, e.g.,
  - female = 1
  - exists has sibling (exists has child (female)) = 5
- hypotheses length: The hypothesis is the conjunction of all its properties. The *hypothesis length* is influenced by the *number of properties per hypothesis* and the *length of the properties*. It is the sum of the length of all its properties plus n-1 "`AND`s" between n properties. For example, the hypothesis {`female, old`} consists of two hypotheses with length one. Its *hypothesis length* is three (`female AND old`).

– definition length:The *definition length* is the sum of the length of all its hypotheses plus the n-1 `OR`s between n hypotheses. For example, in the learning problem Uncle of the Family data set the learned definition consists of two hypotheses: {`male, exists has sibling.child`} `OR`
{`male, exists has married.sibling.child`}.
It has a definition length of twelve (4 + 1 *(AND)* + 1 (OR) + 5 + 1 *(AND)*)

4.2 Data sets

Data sets used for evaluation are the family data set: machine learning data set from DL learner repository [4] and the straight data set: a (randomly) generated data set.

Data Sets	**Examples**	**Positive**	**Negative**	**Relational Contexts**	**Basic Properties**	**Relations**
Uncle (Family)	202	38	38	1	2	4
Cousin (Family)	202	71	71	1	2	4
Aunt (Family)	202	41	41	1	2	4
Grandson (Family)	202	30	30	1	2	4
Grand mother (Family)	202	17	16	1	2	4
Straight200	200	100	100	2	Deck 0, Card 17	Deck 1, Card 3
Straight800	800	400	400	2	Deck 0, Card 17	Deck 1, Card 3

Table 1. Summary of the data sets used for evaluation.

4.3 Evaluation Results

– First experiments (family benchmark): configuration PRCA I: minimal configuration to solve family problems: existential scaling (left), join, all-filter
– Second experiments (family benchmark): configuration PRCA II: more expressive configuration: join, both post conditional join, existential scaling (left), universal scaling (left)
– Third experiments (poker benchmark): configuration PRCA III: trade-off high accuracy and short definition length: all-filter, intersection, join, bloom relation filter, existential scaling (left)
– Fourth experiments (poker benchmark): configuration PRCA IV: trade-off learning time: 80% uncovered positive filter, intersection, join, bloom relation filter, existential scaling (left)

Family data sets:

– The learning time is faster with minimal configuration (PRCA I) than with more expressive configuration.

[4] `http://sourceforge.net/p/dl-learner/code/HEAD/tree/trunk/examples/family/`

	Learning Time (ms)	Testing Accuracy (%)	Definition Length	No. of hypotheses
Family data set - Aunt				
PRCA I	24.5 ± 0.9	100 ± 0	12 ± 0	2 ± 0
PRCA II	59.9 ± 8.5	100 ± 0	15 ± 0	2 ± 0
DL Learner	134.3 ± 29.8	99.5 ± 0.6	21.9 ± 2	2 ± 0
Family data set - Grandgrandmother				
PRCA I	24.3 ± 1.3	100 ± 0	6 ± 0	1 ± 0
PRCA II	58.4 ± 6.2	99 ± 2.4	13.1 ± 2	1.6 ± 0.2
DL Learner	68 ± 4.4	82.3 ± 6.3	26 ± 3	1.7 ± 0.3
Family data set - Grandson				
PRCA I	19.1 ± 0.5	100 ± 0	5 ± 0	1 ± 0
PRCA II	19.8 ± 1.1	99.6 ± 0.8	6 ± 0.1	1 ± 0
DL Learner	23.8 ± 4.5	99.4 ± 0.7	10.1 ± 2.4	1.1 ± 0.2
Family data set - Uncle				
PRCA I	25.5 ± 1.1	99.3 ± 0.8	12 ± 0	2 ± 0
PRCA II	65.2 ± 11.3	99 ± 0.8	16.7 ± 0.3	2 ± 0
DL Learner	140.2 ± 13.9	97.9 ± 1.7	23.5 ± 3.7	2.1 ± 0.2
Family data set - Cousin				
PRCA I	37.8 ± 1.5	100 ± 0	10 ± 0	2 ± 0
PRCA II	1727.2 ± 489	99.7 ± 0.5	17.8 ± 3	2.1 ± 0.2
DL Learner	346 ± 24.7	99.3 ± 0.8	23.3 ± 7	2.1 ± 0.1
Poker data set - Straight 200				
PRCA III	2196.5 ± 131.8	100 ± 0	10 ± 0	1 ± 0
PRCA IV	1061.9 ± 55.5	99.1 ± 0.1	26.3 ± 1.5	1 ± 0
DL Learner	1596.6 ± 30.7	73.8 ± 2.5	466.2 ± 19.4	16.7 ± 0.6
Poker data set - Straight 800				
PRCA III	9504.3 ± 510.4	100± 0	10 ± 0	1 ± 0
PRCA IV	2609.5 ± 144.8	99.95 ± 06	33.8 ± 0.2	1 ± 0
DL Learner	runs out of memory	-	-	-

Table 2. Experiment result summary: PRCA and DL-Learner with ParCEL-Ex on several Family and Straight learning problems. The values are the averages and standard deviations of ten 10-fold cross-validations.

- The additional generator operators lead to the generation of more irrelevant relations, that need to be scaled and integrated into the lattice. Furthermore, the additional scaling operators and the less restrictive property filter lead to more properties that need to be integrated into the lattice as well. Hence bigger lattices are generated. The generation and scaling of more irrelevant relations and the generation of lattices with more concepts increases the learning time when PRCA is run with a more expressive configuration.
- PRCA achieves high testing accuracy on all learning problems, but not 100% because the data sets are small and the learned definitions are over fitted to the training data set.
- A general problem of FCA (for our purpose) is that it generates most specific descriptions instead of most general ones. According to the definition of*formal concept* a concept consists of **all** properties common to all objects in the concept extension (because FCA is based on closure operator). (this happens on small data sets, but may happen on noisy data sets as well)
- PRCA with minimal configuration outperforms DL Learner regarding learning time while achieving the same testing accuracy values.
- definition length: the definitions of the DL Learner tend to be a bit longer than those of PRCA because DL Learner combines partial definitions and counter partial definitions, e.g., one partial definition for the uncle learning problem is `not female and exists sibling.exists child.top`.

- DL Learner and PRCA find the same number of partial definitions/hypotheses (hypotheses in PRCA correspond to partial definitions in DL Learner).

Straight data set:

- The DL-Learner has troubles with learning these problems. On Straight200 its testing accuracy is only 73.9% which is useless for practical application and on Straight800 it runs out of memory. The definition length value and analysis of the result files reveal that ParCEL-Ex learns specific partial definitions whereas PRCA learns one generic hypothesis and achieves high accuracy (99-100%) in all test runs.
- The configurations for the straight data sets are trade-offs between learning time.
- With PRCA III the hypothesis with minimal length is learned and 100% testing accuracy is gained, e.g.,
  `exists has [[card+[card+[card.nextRank+card].nextRank].nextRank].nextRank+card]`
- However, learning times are large on both data sets: more than 2 seconds on Straight200 and more than 9 seconds on Straight800.
- With the weaker *80% uncovered positives filter* learning time is reduced on both data sets: the learning duration of the Staight200 learning problem becomes two times faster and the learning duration of Straight800 becomes 3.6 times faster. The trade-offs are that the testing accuracy gets less (but is still more than 99%) and the definition length becomes 2.6 times longer on Straight200 and 3.4 times longer on Straight800.
- The definition length values and result file analysis reveal that when the filter gets weaker the hypotheses consist of more properties. These properties are shorter than in the PRCA III hypothesis, but describe the straight only partially. For instance, the hypothesis describes a sequence of four cards of sequential rank and three cards with two of sequential rank and a third of the "next-next-next" rank. This leads to worse testing accuracy. The reason is the small training data set: the hypothesis covers all positive examples and not any negative example.

In summary,

- the benchmark shows that PRCA outperforms DL Learner on the used data sets (when run with appropriate configuration). It is faster with always higher accuracy.
- the experiments revealed the general problem of FCA. It generates most specific descriptions instead of most general ones. We tried to reduce the irrelevant properties by property filters. However, hypotheses still contain irrelevant properties. Further work may investigate property reduction during the final hypotheses selection in the learner.
- PRCA find short definitions because DL Learner (ParCEL-Ex) combines counter partial definitions.
- for generalizing the results evaluations on bigger data sets (more examples, more relations, more properties, more complex definition, noisy data) need to be conducted

– the slowing down on more expressive configurations and on learning problems with long hypotheses indicates that the relation generator needs to be improved, e.g., by atm breadth first search → heuristic based search, relation filter mechanism for filtering duplicate relations
– for improving lattice creation (the more properties are in the pool the bigger the lattice the more time is needed to create the lattice) we apply distributed lattice creation.

5 Discussion

Our application scenario is in learning positive concepts of normality for detecting abnormal (i.e. negative) behaviours in the context of smart monitoring environments in ambient assisted living. Our event data sets don't only contain object-property relations but also more complex information relating objects to other objects which have properties. We therefore extended standard FCA-based learning on the basis of RCA for parallel learning on top of data with object-object relations. Due to the amount of data, high scalability of the learning method is relevant and the proposed parallel learner addresses this problem. The proposed approach is configurable and extensible which allows us to further study and evaluate relational concept analysis in different parallel configurations. We conducted experiments that have shown that we can handle data sets with one or multiple relational contexts and that PRCA outperforms DL Learner on the used data sets: PRCA finds a solution on straight data sets, where DL Learner doesn't find a correct solution. On data sets where both find solutions PRCA learns the definitions faster and achieves similar results for testing accuracy.

References

1. R.S. Bucks and J. Haworth. Bristol activities of daily living scale: a critical evaluation. *Expert Rev Neurother*, 2(5):669–76, 2002.
2. B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1st edition, 1997.
3. M. Huchard, A. Napoli, M.R. Hacene, and P. Valtchev. Mining Description Logics Concepts With Relational Concept Analysis. In P. Bertrand P. Brito, G. Cucumel and F. de Carvalho, editors, *Selected Contributions in Data Analysis and Classification*, Studies in Classification, Data Analysis, and Knowledge Organization, pages 259–270. Springer Berlin Heidelberg, August 2007.
4. M. Huchard, A. Napoli, and P. Valtchev M.R. Hacene. Relational concept analysis - a gentle introduction. http://hal.archives-ouvertes.fr/docs/00/61/62/75/PDF/rca_icfca2011.pdf, May 2011.
5. S.O. Kuznetsov. Machine learning on the basis of formal concept analysis. *Automation and Remote Control*, 62(10):15431564, 2001.
6. Mohamed Rouane-Hacene, Marianne Huchard, Amedeo Napoli, and Petko Valtchev. Relational concept analysis: mining concept lattices from multi-relational data. *Annals of Mathematics and Artificial Intelligence*, 67(1):81–108, 2013.