Symposium; proceedings/PKDD '97, J. Komorowski and J. Zytkow (eds.),
pp. 78 -- 87, Springer Verlag, Berlin, New York, 1997.

An algorithm for multi-relational discovery of subgroups

Stefan Wrobel

GMD, FIT.KI, Schloß Birlinghoven
53754 Sankt Augustin, Germany
stefan.wrobel@gmd.de

Abstract. We consider the problem of finding statistically unusual subgroups in a multi-relation database, and extend previous work on single-relation subgroup discovery. We give a precise definition of the multi-relation subgroup discovery task, propose a specific form of declarative bias based on foreign links as a means of specifying the hypothesis space, and show how propositional evaluation functions can be adapted to the multi-relation setting. We then describe an algorithm for this problem setting that uses optimistic estimate and minimal support pruning, an optimal refinement operator and sampling to ensure efficiency and can easily be parallelized.

1 Introduction

Data Mining or Knowledge discovery in databases (KDD) is concerned with the computer-aided extraction of novel, useful and interesting knowledge from (large) databases. A particularly important subclass of knowledge discovery tasks is the discovery of interesting subgroups in populations, where interestingness is defined as distributional unusualness with respect to a certain property of interest. As typical example, in a medical application we are looking at, an interesting subgroup that we would like to discover could be "the subgroup of patients who once had a treatment in a small hospital are significantly more likely to suffer from complications than the reference population".

Previous work on subgroup discovery, e.g. the system EXPLORA [7], has treated this learning task as a single-relation problem, i.e., all data are assumed to be available in a single universal relation. Many existing databases, however, are not stored as single relations, but as several, perhaps dozens of relations for reasons of nonredundancy or access efficiency. While theoretically, any multi-relational database can be transformed into a single universal relation, practically this can lead to universal relations of unmanageable sizes.

In this paper, we therefore consider the task of discovering statistically unusual subgroups in multi-relational databases. The main contribution of the paper is an algorithm for **m**ulti-relational **d**iscovery **o**f **s**ubgroups named MIDOS that uses optimistic estimate and minimal support pruning, an optimal refinement operator and sampling to ensure efficiency and can easily be parallelized. The paper is organized as follows. In section 2, we precisely define the

multi-relation subgroup discovery problem. In section 3, we describe the MIDOS algorithm and its pruning techniques. This is followed by a description of the use of sampling in section 4. In section 5, we discuss related work. Section 6 concludes.

2 Multi-relational subgroup discovery task

In subgroup discovery, the user typically is not interested in finding all subgroups that could be statistically interesting, but in finding the best, i.e., most interesting or most unusual subgroups. We can thus address the multi-relation subgroup discovery task more precisely as follows. Given

- a relational database D with relations $R = \{r_1, ..., r_m\}$
- a hypothesis language L_H (language of group descriptions)
- an evaluation function $d : h \in L_H, D \rightarrow [0, 1]$
- an integer $k > 0$

Find:

- a set $H \subseteq L_H$ of hypotheses of size at most k,
- such that for each $h \in H$, $d(h, D) > 0$
- and for any $h' \in L_H \backslash H$, $d(h', D) \leq min_{h \in H} d(h, D)$.

To further instantiate this discovery problem, we need to precisely identify the language L_H in which we want to express subgroup descriptions, and the evaluation function d that rates the statistical interestingness of such group descriptions.

2.1 Hypothesis language

In the propositional case, subgroups are typically defined as a conjunction of restrictions on the attributes of the single given relation. When extending this to the multi-relational case, in addition to restrictions on individual relations, the hypothesis language must also be capable of joining additional relations to the group description and restrict their attributes in turn. If we represent each relation (of arity a) by a corresponding first-order predicate (of arity a), such group descriptions simply correspond to a conjunction of first-order literals. As an example, for a database with relations/arity $r0/3, r1/2, r/2, r3/2$, a possible group description would be

r0(X,Y,Z) & r1(Y,U) & r2(Z,R) & r3(U,R) & X = x0 & R $\geq$ medium.

As shown in the example, in addition to the literals that introduce and join relations, additional literals can be used to introduce restrictions on individual attributes. For nominal attributes, equality comparisons to values from a (tree structured) value range are allowed, for ordered or continuous attributes comparisons are also allowed.

The *coverage* of a such a group description is defined with respect to a designated *object relation* intended to be the master relation of the population of interest. In our medical application, depending on what our goals are, the master relation could be the patients relation (if we are interested in unusual subgroups of patients) or treatment (if we are interested in unusual subgroups of treatments). The precise definition is the following. Assume the object relation r_o has key attributes K and is part of a database D to be examined. Then define the coverage of a subgroup description $h \in L_H$ as

$$c(h) := \pi_K(\{\sigma \mid h\sigma \in D\}).$$

2.2 Foreign link declarative bias

To avoid a combinatorial explosion of this hypothesis space, we exploit additional information inspired by the *foreign key* links available in many database systems. A foreign key link is a link from an attribute in one relation to a key attribute in another relation. For example, in a medical database about treatments and patients, the patient-id attribute of the treatment relation would be a foreign key linking to the patient-id attribute of the patient relation. We interpret these links as designated paths along which different relations can be joined together. Since we are also interested in getting to non-key attributes, we will simply be speaking of *foreign links*, waiving the requirement that one of the linked attributes must be a key. Given a set of such foreign links, starting from the designated object relation, new relations may be joined to a group description only along such a link. In logical terms, variables may share only between linked argument places in literals, and there must be at least one variable shared with previous literals in the conjunction (linked literal restriction).

In our example

r0(X,Y,Z) & r1(Y,U) & r2(Z,R) & r3(U,R) & X = x0 & R $\geq$ medium.

would be a legal group description, assuming the object relation is $r0$, X is a nominal or tree structured attribute, R is an ordered attribute, and the set of foreign links is $F = \{r0[2] \rightarrow r1[1], r0[3] \rightarrow r2[1], r1[2] \rightarrow r3[1], r2[2] \rightarrow r3[2]\}$. On the other hand,

(a) r0(X,Y,Z) & r3(U,R)
(b) r0(X,Y,Z) & r3(X,R)
(c) r1(Y,U) & r2(Z,R) & r3(U,R)

would not be legal subgroup descriptions because they are not linked (a), do not respect the foreign links (b), or do not start with the designated object relation (c).

If we regard the foreign key declarations as edges in a graph with relations as nodes, each path through the graph corresponds to a set of possible clauses: whenever we reach a relation that has been reached n times before, we can continue with a new variable in our "target" attribute or with one occuring in

the target attribute of the n previous literals. Thus, if the foreign key graph contains cycles, the length of clauses (and thus the number of joins) is not limited. As in most ILP systems, we therefore introduce a depth limit i that bounds the length of the longest possible path. A reasonable default for i would be the maximum length of any shortest path from the object relation to another relation.

2.3 An optimal refinement operator

We precisely define the hypothesis space based on the following refinement operator. Let $F = \{f_1, ..., f_n\}$ be the set of foreign (key) links of the database D with relations $R = \{r_1, ..., r_m\}$. If there are several links starting at relation r, we assume they are assigned order numbers o_r.

Definition1 Compatibility of variables. Let $C = L_1, \ldots, L_n$ be a set of literals, V a variable first occuring in L_i with functor r at position p_i, U a different variable first occuring in L_j with functor s at position p_j. V and U are said to be *compatible* with respect to C and a link set F ($V \bowtie_{C,F} U$) iff $r[p_i] \rightarrow s[p_j] \in F$. For two sets of variables $\mathbf{V}$ and $\mathbf{U}$, we further define $\mathbf{V} \bowtie_{C,F} \mathbf{U} := \{(V, U) \in \mathbf{V} \times \mathbf{U} | V \bowtie_{C,F} U\}$.

We will usually omit the subscripts if they are clear from the context.

A refinement operator for L_H can be defined as follows. Let $h := L_1, ..., L_n$ with variables $\mathbf{V} := vars(h)$ be the hypothesis to be specialized. (According to the definition of L_H, $L_1, ..., L_n$ are positive literals.) Define the refinement operator ρ as follows.

1. For any L_i in h, $h' := L_1, \ldots, L_{i-1}, \rho(L_i), L_{i+1}, \ldots, L_n$ is in $\rho(h)$. $[o = i.1]$.
2. For any $L_i = r(V_1, \ldots, V_{a(r)})$ in h such that $r[m] \rightarrow s[k] \in F$, let $\mathbf{U} = \{U_1, \ldots, U_{k-1}, U_{k+1}, \ldots, U_{a(s)}\}$ be a set of new variables, i.e., $vars(h) \cap \mathbf{U} = \emptyset$. Then

$$h' := \begin{array}{l} L_1, \ldots, L_n, s(U_1, \ldots, U_{k-1}, V_m, U_{k+1}, \ldots, U_{a(s)}), \\ \bigwedge_{U \in \mathbf{U}} any(U) \bigwedge_{(V,U) \in \mathbf{V} \bowtie \mathbf{U}} any(V, U) \end{array}$$

   is in $\rho(h)$. $[o = i.2.o_r(r[m] \rightarrow s[k])]$.

Now we need to define the refinement operator for an individual literal L.

1. $L = any(X)$, X is ordered with values $\{v_1, v_2, \ldots, v_{n-1}, v_n\}$. If $n = 2$, $X = v_1$ and $X = v_n$ are in $\rho(L)$, otherwise $X \geq v_2$ and $X \leq v_{n-1}$ are in $\rho(L)$.
2. $L = X \geq v$ ($L = X \leq v$), X is ordered with values $\{\ldots, u, v, w \ldots\}$. If $w = v_n$ ($u = v_1$), then $X = v$ and $X = w$ ($X = u$) are in $\rho(L)$, else $X = v$ and $X \geq w$ ($X \leq u$) are in $\rho(L)$.
3. $L = any(X)$ ($L = X = a$), X is tree-structured, and b_1 to b_n are the children of *any* (of a) in the value tree of X. Then $X = b_1$, ..., $X = b_n$ are in $\rho(L)$.
4. $L = any(X, Y)$. Then $X = Y$ is in $\rho(L)$.

Within square brackets, the ordering of refinements in ρ is specified. For any particular hypothesis h, two elements $h_1 \in \rho(h)$ and $h_2 \in \rho(h)$ can be ordered by a lexicographic comparison of $o(h_1)$ and $o(h_2)$, thus completely ordering $\rho(h)$. This in turn allows an optimal refinement operator ρ_{opt} to be defined as follows.

$$\rho_{opt}(h) := \{h' \in \rho(h) | o(h') > o(h)\}.$$

The most important property of ρ_{opt} is that it generates each hypothesis along exactly one refinement path, thus avoiding redundant regeneration of hypotheses. This in particular is an important prerequisite for parallelization.

Proof sketch. Consider a node in the search where two different paths p_1 and p_2 originate. If p_2 differs from p_1 by choosing to refine a literal further to the right, p_2 can never return to the literal refined by p_1, since its major index (i) is now higher. Since all new literals introduce new variables, literals introduced by following links from different existing literals will be different. If p_2 starts at the same literal, but differs from p_1 by choosing to introduce a new literal instead of refining the existing one, it can never go back to refining this existing literal since its secondary index will prevent this. Finally, note that each refinement of an *any*(...) literal can be generated in only one way (by definition of the interval comparisons for nominal and the tree nature of other attributes).

2.4 The evaluation function

The goal of MIDOS is to find subgroups of the object relation (defined by first-order conjunctions) that have unusual distributional characteristics with respect to the entire population. If we want to find the k best such subgroups, we need a way to measure the quality of candidate groups. To find such evaluation functions, we can build on the evaluation measures that have been defined for propositional algorithms — [7] provides a detailed discussion. Since the interestingness of a group depends both on its unusualness and size, an evaluation function needs to combine both factors.

Assume we are given a designated object relation r_o with key attributes K that is part of a database D to be examined. Let us first consider the simple case of a binary goal attribute A_g in r_o. We let $T := \{t \in r_o \mid r_o[A_g] = 1\}$ denote the set of target object tuples, define $g(h) := \frac{|c(h)|}{|r_o|}$, the generality of a hypothesis, and probabilities $p_0 := \frac{|T|}{|r_o|}$ and $p(h) := \frac{|c(h) \cap T|}{|c(h)|}$. The chosen evaluation function is defined as:

$$d(h) := \begin{cases} 0 if \mid c(h) \mid / \mid r_o \mid > s_{min} \\ g(h)(p(h) - p_0) \text{otherwise} \end{cases}$$

The function consists of two components, the first requiring that each hypothesis must cover at least a certain fraction s_{min} of the object relation tuples, the second is our adaptation of the standard evaluation functions discussed in [7], where generalizations of this component for categorical attributes with more than two values, ordered discrete attributes and continous attributes can also be found.

3 The MIDOS algorithm

The basic MIDOS algorithm is a top-down, general-to-specific search that can use breadth-first, depth-first, or best-first or parallel control regimes. The algorithm is shown in table 1 and centers around the refinement operator ρ for L_H defined in section 2.3.

Let $Q := r_o(V_1, \ldots, V_{a(r_o)})$, $H := \emptyset$.
WHILE $Q \neq \emptyset$

- select a subset C from Q according to *search strategy*
- let $\rho(C) := \{\rho(h) \mid h \in C\}$
- test each $h \in \rho(C)$ on D (compute $d(h, D)$)
  - if $d(h, D) = 0$, prune.
  - else if $d_{max}(h, D) < min_{h' \in H} d(h', D)$, prune.
  - else
    - if $d(h, D) > min_{h' \in H} d(h', D)$, remove the bottommost element of H and add h
    - add h to Q.

Table 1. MIDOS top-down search algorithm

To avoid exploring uninteresting parts of the search space, the algorithm exploits the properties of the discovery task for two different kinds of pruning, minimal support pruning and optimistic estimate pruning.

3.1 Minimal support pruning

Since ρ is a specializing refinement operator, we know that any descendant of a hypothesis will cover no more objects than the hypothesis itself. Since we are not interested in solutions that cover too few members of the population, as soon as we reach a hypothesis that fails to cover that many elements, we can prune the entire subtree rooted as this hypothesis.

3.2 Optimistic estimate pruning

The second kind of pruning used by MIDOS is somewhat less common and exploits the fact that we are interested only in the k best solutions. Thus, if we knew that a hypothesis and its descendants cannot make it into the top k list, we could safely prune that branch. For the evaluation functions that are being considered here, exact estimates are possible only by actually expanding a subtree of the search space. However, it is possible to reason about the maximal quality that could potentially be reached within a subtree. Let d_{max} be an optimistic estimate of hypothesis quality in a subtree, i.e., for $S \subseteq L_H$, and a database D,

$d(h') \leq d_{max}(h)$ for all h' that are refinements of h.

For the evaluation function $d(h)$ defined above, it is easy to derive such an optimistic function d_{max} for a general-to-specific search order. When refining an existing hypothesis h, its specializations will cover at most as many object relation tuples as h itself, so g cannot increase when specializing. p, on the other hand, can increase if the proper specializations are made — but it can never be more than 1. The required optimistic estimate function is therefore:

$$d_{max}(h) := g(h)(1 - p_0).$$

3.3 Search strategy

The MIDOS algorithm can work with a number of different search strategies simply by changing the selection ordering among the hypotheses in Q. Besides the standard breadth-first (least recent element first) or depth-first (most recent element first) traversal orderings, best-first and parallel regimes are particularly important. For best-first search, the algorithm always chooses to further refine the hypothesis in Q with best quality rating according to function d. For a parallel search strategy, it is possible to test all descendants of a hypothesis in parallel.

Furthermore, different search branches can be run in parallel. First, since only the k best solutions are being required, each parallel search branch never needs to know more than the quality of the worst hypothesis accepted into the solution set so far. Second, since we use an optimal refinement operator, each local branch is guaranteed to not regenerate hypotheses that have been generated by branches running in parallel, and thus would be redundant[1].

4 Use of sampling

Checking the distributional unusualness according to function d requires joining together the relations involved in the subgroup description h, selecting according to the additional restrictions on attributes and projecting on the object relation key attributes to determine $c(h)$. While potentially feasible on small databases with low branching factors, for larger databases computing the full join should be avoided, suggesting the use of sampling.

Sampling can be an efficient way of estimating the relative frequency of a group in a population. Assuming a truly random sample, the number x of occurences of a group with underlying probability p can approximately looked at as a binomial variable, i.e., distributed as follows (sample of size s):

$$P(x) = \binom{s}{x} p^x (1-p)^{s-x}$$

[1] This does not, however, avoid the problem of generating hypotheses that could be specializations of pruned hypotheses in other branches.

with mean $\mu_x = sp$ and standard deviation $\sigma_x = \sqrt{sp(1-p)}$. Since we estimate the true probability p by $p' := \frac{x}{s}$, what we need is a statement about the probability that $|p-p'|$ is smaller than a specified error threshold ϵ. From the statistics literature, we can use the so-called Chernoff bound ([2] foll. [1]) for this purpose. According to this bound, for any $a > 0$,

$$P(x > sp + a) < e^{-2a^2/s}$$

For the difference between estimated and actual probability, we obtain

$$P(p' > p + \epsilon) = P(x > sp + s\epsilon) < e^{-2s\epsilon^2}$$

Thus, for truly random samples, we can precisely determine which sample size is needed to stay below a desired error probability δ.

Applying this to the problem of determining d without computing the full underlying group, we see that if we could randomly sample $c(h)$, we can use the above bounds to limit the probability of missing one of the k best hypotheses to an arbitrarily small number.

Computing $d(h)$ requires computing $p(h)$ and $g(h)$. $p(h)$ amounts to determining the probability of T in $c(h)$, and thus requires random sampling $c(h)$. Computing $g(h)$ requires random sampling r_o, and determining the probability of $c(h)$ by checking h on each sampled tuple. Thus the primary problem is random sampling from $c(h)$, which is defined by a project-select-join query. Obviously, if the set $c(h)$ were available, we could easily draw random samples from it. Computing this set, however, is precisely what we would like to avoid. We would like to compute only those tuples that are in the chosen sample.

In a multi-relation join, this could be done by sampling after each step of the join: randomly select a tuple in the first relation, then select one of the matching tuples in the second relation. Repeat this until the desired sample size is reached. Clearly, such an algorithm does not result in equal selection probabilities for each tuple of the computed result, since probabilities depend on branching factors.

In database terminology, we need to push the sampling operation down the query tree that defines $c(h)$ resp. d. Following [11, ch. 6], the non-uniform inclusion probabilities of sampled joins can be compensated for by accept/reject sampling, i.e., accepting each sample with probability proportional to the branching factor of the join. For the project part of the query, a more complex algorithm is necessary, since compensating the non-uniform inclusion probabilities requires knowing the multiplicity of each projected tuple. This essentially entails following all join paths for each different projected sample [11, ch. 6].

For an in-memory implementation of this algorithm, the sampling approaches mentioned here can be implemented directly in the algorithm. For database-based implementation, sampling would need be implemented directly in the database system to be efficient.

5 Related Work

The discovery task defined in this paper is related to a number of other learning tasks that have been proposed in the past [3, 4, 13, 8], mostly in the field of

ILP. The primary differences are in the instantiations of the evaluation function d and the requirements on the solution. First, since we use distributional unusualness d as evaluation function, which does not monotonically decrease when specializing, we uses minimal support and optimistic estimate pruning instead of simply pruning according to d. Second, requiring only the k best solutions allows further pruning and optimizations especially for a parallel algorithm.

In terms of the algorithm, MIDOS is related to a number of other systems. MIDOS uses a declarative bias whose definition, besides its inspiration from the database area, owes a lot to the declarative bias languages that have been proposed in the field of ILP. In particular, foreign links are closely related to the type declarations in use in many ILP systems, such as e.g. FOIL [12], RDT [5], or PROGOL [9]. In contrast to type declarations, foreign links are not symmetric, and thus allow somewhat finer control of the introduction of new variables.

In terms of search and pruning, the idea of using an optimal refinement operator has been used by a number of learning systems, e.g. in [4]. Optimistic estimate pruning is closely related to general AI search procedures such as $A^\star$ and has already been used e.g. in the temporal pattern discovery system MSDD [10]. Minimal support pruning, of course, is a central element e.g. of any association rule algorithm.

In terms of the sampling ideas used here, the idea of using Chernoff bounds to determine error limits on sampled frequencies was e.g. proposed by [1] for use with association discovery algorithms. The novelty here is that this idea is combined with sampling from relational expressions [11], which has not been used in the context of a discovery algorithm yet. Finally, [6] have studied the use of sampling for discovering universal sentences.

6 Summary

In this paper, we have discussed the multi-relational variant of the KDD subgroup discovery problem. As we saw, the problem definition requires some adjustments with respect to the propositional case, in particular a different definition of coverage of a subgroup. Given these adjustments, the evaluation functions from the propositional case can be reused for the multi-relational case. We have then proposed a particular instantiation of the multirelational subgroup discovery problem employing a declarative bias based on the idea of foreign links and requiring only the k best solutions to be returned.

The corresponding algorithm MIDOS, which we regard as the primary contribution of the paper, is a typical top-down algorithm, but adds a number of features specifically designed for efficiency. First, the algorithm uses an optimal refinement operator, i.e., generates each hypothesis exactly once. This avoids the redundant regeneration of the same hypotheses and allows easy parallelization of the algorithm. Second, besides the minimal support pruning used in many algorithm, MIDOS also employs an optimistic quality estimate function, derived from the hypothesis quality function, for additional pruning. Finally, we have shown how sampling from relational expressions could be valuably employed in

speeding up the evaluation of hypotheses; precise proofs of speed-up or error bounds are outstanding, however.

In the future, an in-memory version of the algorithm will be used on a medical application involving 6 relations about hospitals, patients, diagnoses, therapies, and the relation between patients and therapies and patients and diagnoses.

Acknowledgements

Thanks to Willi Klösgen and Dietrich Wettschereck for comments on this paper. H. Mannila pointed me to [11]. This work was partially supported by ESPRIT (LTR project ILP2, P20237).

References

1. R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. Verkamo. Fast discovery of association rules. In U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, eds., *Advances in Knowledge Discovery and Data Mining*, ch. 12, pp. 307 – 328. AAAI/MIT Press, Cambridge, USA, 1996.
2. N. Alon and J. H. Spencer. *The Probabilistic Method*. Wiley, N.Y., 1992.
3. L. De Raedt and S. Džeroski. First order jk-clausal theories are pac-learnable. *Artificial Intelligence*, 70:375–392, 1994.
4. L. De Raedt and L. De Haspe. Clausal discovery. *Machine Learning*, 1997.
5. J.-U. Kietz and S. Wrobel. Controlling the complexity of learning in logic through syntactic and task-oriented models. In S. Muggleton, ed., *Inductive Logic Programming*, ch. 16, pp. 335 – 359. Academic Press, London, 1992.
6. J. Kivinen and H. Mannila. The power of sampling in knowledge discovery. In *Proc. 1994 ACM SIGACT-SIGMOD-SIGACT Symp. on Principles of Database Theory (PODS'94)*, pp. 77 – 85, Minneapolis, 1994.
7. W. Klösgen. Explora: A multipattern and multistrategy discovery assistant. In U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, eds., *Advances in Knowledge Discovery and Data Mining*, ch. 10, pp. 249 – 271. AAAI/MIT Press, Cambridge, USA, 1996.
8. H. Mannila and H. Toivonen. On an algorithm for finding all interesting sentences. In R. Trappl, ed., *Cybernetics and Systems '96*, pp. 973 – 978, 1996.
9. S. Muggleton. Inverse entailment and Progol. In K. Furukawa, D. Michie, and S. Muggleton, eds., *Machine Intelligence 14*, pp. 133 – 188. Oxford Univ. Press, Oxford, 1995.
10. T. Oates, M. Schmill, and P. Cohen. Parallel and distributed search for structure in multivariate time series. In M. van Someren and G. Widmer, eds, *Machine Learning: ECML-97*, Berlin, New York, 1997. Springer Verlag.
11. F. Olken. *Random Sampling From Databases*. PhD thesis, UC Berkeley, 1993.
12. J.R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5(3):239 – 266, 1990.
13. S. Wrobel and S. Dzeroski. The ILP description learning problem: Towards a general model-level definition of data mining in ILP. In K. Morik and J. Herrmann, editors, *Proc. Fachgruppentreffen Maschinelles Lernen (FGML-95)*, 44221 Dortmund, 1995. Univ. Dortmund.