University of Bristol

DEPARTMENT OF COMPUTER SCIENCE

Decompositions, dependencies, and Bayesian networks

Peter A. Flach Nada Lavrac Blaz Zupan

Also issued as ACRC-99:CS-001

February 1999 | CSTR-99-001

Decompositions, dependencies, and Bayesian networks

Peter A. Flach

Department of Computer Science, University of Bristol, United Kingdom

Nada Lavrac, Blaz Zupan

Department of Intelligent Systems, Jozef Stefan Institute, Ljubljana, Slovenia

Abstract. This brief note describes some of the relations between function and relation decompositions, functional and multivalued attribute dependencies, and Bayesian belief networks. It suggests possible research directions for the second year of the Royal Society Joint Project with Central/Eastern Europe.

1. Introduction

1.1 Conditional independence

Consider the joint probability distribution over the discrete random variables A,B,C,D. By the definition of conditional probability we have P(ABCD) = P(A | BCD) P(BCD), which leads to the chain rule of probability:

$$P(ABCD) = P(A \mid BCD)\, P(B \mid CD)\, P(C \mid D)\, P(D)$$

Notice that for each ordering of the variables we obtain such an equation.

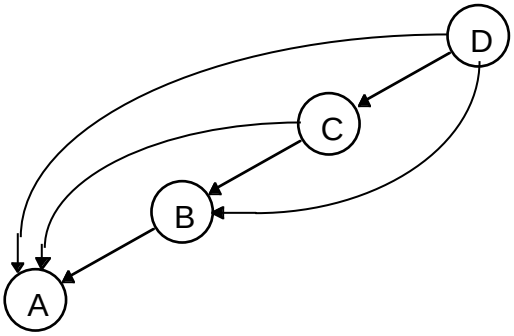
Suppose now that *A and B are (conditionally) independent given CD*, which is the case iff P(A | BCD) = P(A | CD); in that case we have

$$\begin{aligned} P(ABCD) &= P(A \mid BCD)\, P(B \mid CD)\, P(C \mid D)\, P(D) \\ &= P(ACD)\, P(BCD) / P(CD) \end{aligned}$$

This means that we can decompose the joint probability distribution P(ABCD) into P(ACD) and P(BCD) (NB. P(CD) is just a normalising term, which can be obtained from either P(ACD) or P(BCD) by projection).

1.2 Bayesian networks

For the ordering ABCD, the chain rule of independence can be depicted graphically as follows:



For every node X we have a term P(X | Y), where Y is the set of parents of X. This is the trivial Bayesian network, because it encodes no independence assumptions whatsoever.

Independence assumptions are encoded in Bayesian networks by the absence of links: *nodes are independent of their non-descendants given their parents*. In the above network, for any pair of nodes one is parent of the other, so there are no independence assumptions. In order to encode the conditional independence of A and B given CD, we cut the link between A and B. The resulting network is shown below (left), which can in this case be simplified by grouping C and D together (right).

That these two networks are equivalent can be seen as follows. The independence assumptions encoded in a Bayesian network ensure that the joint probability distribution over all variables is equal to the product of the conditional probabilities of each node given its parents. For the left network we obtain

$$P(ABCD) = P(A \mid CD)\, P(B \mid CD)\, P(C \mid D)\, P(D)$$
$$= P(A \mid CD)\, P(B \mid CD)\, P(CD)$$

which is expressed by the network on the right.

The direction of the arrows in a Bayesian network is not always meaningful. For instance, the direction of one of the arrows (but not both) in the right network above can be reversed without changing the independence assumption encoded in the network. This also implies that the same independence assumptions may be encoded by many different networks.

1.3 Database dependencies

Consider a database relation R over the attributes A,B,C,D. A functional dependency CD→B means that for every tuple in R, the value of B is uniquely determined by the value of CD. The function from CD to B can be recorded in a new relation R1(BCD); notice that CD is a key in this relation. Given R1 the attribute B has become redundant in R and can be removed, leading to a new relation R2(ACD). R can be reconstructed from R1 and R2 by a join over CD; therefore, the decomposition is said to be *lossless*. In Prolog, this join (and conversely the decomposition) can be expressed as follows:

```
r(A,B,C,D):-r1(B,C,D),r2(A,C,D).
```

The existence of a functional dependency is a sufficient but not necessary condition for lossless decomposition: the fact that CD is a key in R1 is not essential. This leads to the notion of a multivalued dependency CD→→B, which requires that every value of CD uniquely determines a *set* of possible values of B. The decomposition can proceed as before, but now CD is a key in neither R1 nor R2. In fact, the situation is now totally symmetric, and the dependencies CD→→B and CD→→A are equivalent — sometimes they are jointly written as CD→→A | B.

In other words, multivalued dependencies are not dependencies at all: they are (conditional) *independencies*, stating that A and B are independent given CD. Consequently, a functional dependency CD→B says two things:

1. B is independent of A given CD;
2. B is functionally dependent on CD.

2. Research directions

2.1 From function decomposition to relation decomposition

Suppose we have a table describing a class attribute Y as a function f of X1, X2 and X3. The aim of function decomposition is to introduce a new attribute C, such that C is a function of some of the Xs and Y is a function of the remaining Xs and C, for instance (the last variable is the determined one):

```
f(X1,X2,X3,Y):-f1(X2,X3,C),f2(X1,C,Y).
```

Since such a decomposition is always trivially possible by introducing a distinct value of C for each distinct pair of values of X2X3, we are interested in decompositions which result in a minimum number of values of C. The number of C-values for a given partition of the Xs is determined by constructing the partition matrix: e.g. for the above partition this is a matrix with rows for each value of X1, and columns for each value of X2X3, and cells containing the corresponding values of Y if known, and don't-cares otherwise. The don't-cares are used to group similar columns together, and each group gets labelled with a distinct value of C. Thus, in general, the decomposed function generalises the original one.

Notice the similarity with relation decomposition: if we construct a relation R as follows:

```
r(X1,X2,X3,C,Y):-r1(X2,X3,C),r2(X1,C,Y).
```

then we see that R satisfies the multivalued dependency $C \rightarrow\rightarrow X2X3 \mid X1Y$. The task is then to discover a C-attribute such that this multivalued dependency holds. Since there is no class attribute the partition matrix is two-valued, indicating which tuples are known to be in the relation. There is now a trivial solution even when minimising the number of C-values, since we can overgeneralise by grouping all columns together. One way to avoid this is to assume complete knowledge, avoiding any generalisation. We can then simply minimise the number of C-values. Effectively this means that we are clustering tuples, assigning each tuple a distinct C-value. Since we don't generalise experimental evaluation is problematic, but we can look into the clustering literature for ideas (e.g. distance between clusters, homogeneity within clusters).

This approach should be easily realisable using HINT, since we're back to function decomposition (the function to be decomposed is a mapping of all possible tuples into {true,false}).

HINT works in two modes, one being categorical function decomposition as described above, and the other applying probabilistic techniques to deal with noisy data. It is interesting to investigate whether these techniques can also be meaningfully applied to relation decomposition, using the same approach as sketched above.

Alternatively, if we want to generalise we need a heuristic that trades off the number of C-values and the degree of generalisation. This requires a distance measure between columns (e.g. edit distance), grouping those columns together that are sufficiently similar. This again suggests a clustering approach, but now on the level of columns in the partition matrix, and with a conventional distance measure.

2.2 From database relations to probability distributions

Any database relation trivially defines a joint probability distribution over its attributes: any tuple in the relation is assigned probability $1/n$, where n is the size of the relation, and any other tuple is assigned probability 0. Given this underlying *qualitative* probability distribution, the equivalence of multivalued dependencies and statements of conditional probabilistic independence can be formally established: decomposing the database relation is equivalent to decomposing the qualitative probability distribution.

This suggests that Bayesian networks can be used as a graphical representation of sets of attribute dependencies. This could be incorporated into the work with Iztok Savnik on induction of attribute dependencies.

2.3 From probability distributions to database relations

If we can find a sensible way of converting an arbitrary probability distribution into a qualitative one, and thus into a database relation, learning a Bayesian network would become at least partly identical to finding sets of attribute dependencies in that relation. The added advantage is that attribute dependencies can be learned in a modular way.

2.4 Decomposing probability distributions

Alternatively, we could apply the function deocmposition approach to decompose a joint probability distribution. This would give us a concept hierarchy akin to a Bayesian network. The analogy is not complete, as the concept hierarchy will contain invented attributes, but the conditions under which this decomposition can be mapped to a Bayesian network should be investigated.

3. Conclusion

We have investigated a number of relations between function and relation decomposition, attribute dependencies, and Bayesian networks, and suggested possible research directions for the second year of the project.

Acknowledgements

Writing of this report was supported by a Royal Society Joint Project with Central/Eastern Europe.