Introducing inductive methods in knowledge acquisition by divide-and-conquer

Maarten W. van Someren
Department of Social Science Informatics
University of Amsterdam
Roetersstraat 15
1018 WB Amsterdam
The Netherlands
maarten@swi.psy.uva.nl

Floor Verdenius
ATO-DLO
PO Box 17
6700 AA Wageningen
The Netherlands
F. Verdenius@ato.dlo.nl

Abstract

In this paper we present the outline of a method that combines a divide-and-conquer approach, that is common in knowledge acquisition and software engineering, with the use of inductive techniques. The idea is to guide decomposition of a knowledge acquisition problem into subpr oblems by the expected costs and benefits of applying elicitation or induction t o acquire the knowledge for part of the target knowledge. The method is illustra ted with a "rational reconstruction" of a knowledge acquisition process that involved inductive techniques.

Keywords: Knowledge acquisition, learning goals, decomposition

Introduction

Knowledge acquisition for knowledge-based systems involves the formalisation of human knowledge about a certain task to build a system that can perform this task. Knowledge can be acquired in different ways. The classical way is to elicit knowledge from a human expert and formalise this in an operational language, e.g. using an expert system shell. This is often extended to acquiring knowledge not only from a single human expert but also from documents and several different experts. Machine learning gives the prospect of (partially) automating this process. There are several ways to apply machine learning techniques to knowledge acquisition. The most direct approach is to collect a set of examples of problems with solutions that are provided or approved by a human expert and to apply an induction technique to automatically construct a knowledge based system. Other approaches focus on the task of refining or debugging knowledge that was acquired manually (e.g. (Shapiro 1982), (Ginsberg 1988), (Craw & Sleeman 1990)). Either approach can be improved by using domain specific prior knowledge, for example in the form of descriptions of rules that are to be learned.

Many knowledge acquisition problems are characterised by the availability of sources of relatively unstructured information that can be used to construct a knowledge-based system. Examples of sources are documents, human experts on related tasks, collections of observations or even existing knowledge in computational form. Another source is the *possibility* to acquire data on an aspect of a task. In addition to selecting appropriate sources, a choice has to be made for an acquisition approach: manual elicitation, collecting examples of solved problems and machine induction, or some combination of these. Once a choice of an approach is made, a particular method and a tool must be selected. Here we focus on the choice of the approach.

If a domain involves a complex relation between problem data and solutions then direct knowledge elicitation is not effective: it will lead to too global questions to a domain expert. For example, if only the possible problems and solutions are known, a knowledge engineer can only ask: "how do you find solutions like .. from problem data like ...?" which is hard to answer for a complex domain. An inductive approach also encounters problems if the knowledge acquisition problem is complex. One problem is that uncovering a complex structure will require many data. Another problem is that machine learning systems explore a particular hypothesis space. A hypothesis space is characterised by a language that involves an implicit bias on the possible hypotheses (that is: the mapping between problems and solutions), an ordering on the generation of hypothesis and usually a criterion for stopping the search. For many domains, a single uniform hypothesis space may not be adequate. This means that it is either necessary to search a space of "biases" or to search a very large space of hypotheses (in a space with little bias). Evidence for this is the current interest in multistrategy learning systems that construct hypotheses in "hybrid" languages, mixing decision trees, neural nets, linear threshold functions, etc.

Here we argue that solving real world knowledge acquisition problems requires a divide-and-conquer approach that aims at optimal use of available sources of knowledge. In knowledge acquisition and in software engineering in general, a "divide-and-conquer" approach based on top down development is a standard method. Many authors have defined languages that support top-down development (e.g. (Schreiber, Wielinga, & Breuker 1993)). However, these methods are not very specific about how to reduce a complex knowledge ac-

quisition problem to simpler subproblems. In this paper we describe a method for decomposing knowledge acquisition problems into subproblems that can be solved better than the original problem. There are two main differences with standard development methods: (1) our method includes the possibility of using induction to acquire a component and (2) decomposition is directed at the structure of the available knowledge or "sources" of knowledge and not at processes like data selection, data cleaning and data conversion (see for example (Engels, Lindner, & Studer 1997)). In knowledge acquisition, and software engineering in general, decomposition is directed at efficiency or modularity (e.g. (Schreiber, Wielinga, & Breuker 1993)) but here the decision to decompose a task is based on the possibility of acquiring an adequate component.

This paper is organised as follows: in section 2 we define what we mean by a "knowledge acquisition problem", in section 3 illustrate this with an example, in section 3 we give the arguments for decomposing a "knowledge acquisition problem" (recursively) into subproblems, in section 4 we give a method for decomposing "knowledge acquisition problems", also illustrated with an example (section 5) and in section 6 we discuss the implications and applicability of this method.

What is a knowledge acquisition problem?

How are knowledge acquisition problems represented? To describe reduction of knowledge acquisition problems to subproblems, we need a more precise notion of knowledge acquisition problem. The goal of knowledge acquisition is to obtain knowledge that can be used for solving a certain class of problems. A "knowledge acquisition problem" therefore includes a description of possible problems, a description of possible solutions and a description of the relation between problems and solutions. Possible problems and solutions can be described in a number of ways but for simplicity we assume here that a problem is a subset of possible problem data and a solution is an element of a set. The relations between problems and solutions is described by reference to certain "sources" of knowledge or is left unspecified, which means that any source can be used. The goal can be for example, to construct a system that behaves like a given human domain expert, that follows certain prescribed methods, that finds solutions as entailed by the knowledge in certain documents or that predicts empirical phenomena. Decomposition of a knowledge acquisition problem recursively divides a problem into subproblems that are connected in a dataflow structure. Each subproblem is itself a knowledge acquisition problem. Decomposition therefore introduces intermediate datatypes (or splits datatypes into subtypes). Each "interface" between two components is associated with a description of the data that pass through it. At the lowest level of decomposition, problems are identified in terms of primitive tasks. At this point details of the representation become important because these determine the particular acquisition or machine learning system that is to be applied.

Finally, we assume that information is available about the costs and benefits of acquisition operations. In particular, the methods described below uses estimates of *accuracy* and *acquisition costs* of acquisition operations.

Example

Consider the following example, taken from (Polderdijk *et al.* 1996). The acquisition goal is:

> *construct a knowledge system that takes as input information on a batch of fruits that arrives from abroad and that produces a recipe for storing the fruits.*

The solution of the planning procedure is a storage recipe. A recipe is a prescription of external conditions over a certain period of time. The time interval is divided in a number of fixed-duration time slices. For each time slice the recipe contains prescriptions for a set of storage conditions. Storage conditions may include temperature, relative humidity and ethylene concentration. The choice of a recipe depends on the following information:

- *administrative information*, e.g. identification of the grower, harvest date, country and field of origin, exporter, transport medium and duration, maturity at harvest, etc.
- a record of information on *external conditions*, i.e. data on the behaviour of the set of relevant external conditions (see above), registered for the postharvest life of the product on the basis of fixed-duration timeslices
- a record of *quality measurements*, containing for a some moments in the product life measurement results on a number of indicative quality parameters; a typical parameter set may include colour, firmness, sugar content and qualitative measures for diseases, mould appearance and injuries
- *planned destination*, storage period duration, or date and quality specification; in case of a storage period, the assumption is that certain quality standards are still met at the end of the storage period; the duration of the storage period may be indefinite, in which case maximisation of the period is the target.

Figure 1 shows the problem in diagrammatical form.

To solve this knowledge acquisition problem, that is, to construct a system that can find an adequate recipe, several sources of knowledge are available. In general, knowledge can be acquired from a human expert, a document, a set of data or an existing system. Table 1 lists some sources that can be used for this knowledge acquisition problem. These sources include both machine learning, knowledge elicitation from experts and extraction from documents. The sources include information

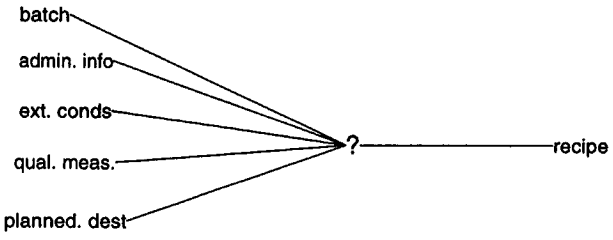

Figure 1: Overall learning problem

that is not part of the original problem. For example, the source Measure Quality 1 refers to "Quality Data" and "Sampling Instructions" that were not mentioned in the original problem statement. This means that there is a way to acquire knowledge to find Quality Data from Sampling Instructions and Batch. Including this in the decomposition means that this source will be used to acquire this knowledge. A consequence is that the relation between Quality Data, Sampling Instructions and the other types of information must be acquired. These become new subgoals. Table 1 gives the name, the conclusions and the data about which a source contains knowledge, the expected accuracy of the resulting knowledge and the costs of acquiring it. The expected accuracy and costs must be estimated.

In this table, Measure Quality is not done by inference but by observation of the fruit, which is not automated. There are two alternative versions of Measure Quality, one that uses Sampling Instructions and one that does not. Note that Revise Recipe takes a Detailed Recipe as input and also produces one as output. The table lists elementary sources but knowledge can also be acquired from compound sources. For example, knowledge can be acquired that directly relates External Conditions and Planning Destination to a Detailed Recipe. Table 1 lists the costs of these sources and the expected accuracy of the resulting knowledge. Some sources have no costs because the knowledge already exists. These costs and accuracies must be estimated with heuristics. If no good estimate is possible then it is necessary to empirically evaluate these. In the example, this was the case with Measure Quality 2. A system was built with decision tree learning to evaluate if the accuracy and costs of this. The test showed that costs were relatively low and accuracy was significantly better than that of Measure Quality 1. The knowledge acquisition problem is now to find sources and acquisition operators that will produce the target knowledge from available sources, at minimal cost and with maximal accuracy.

Solving a knowledge acquisition problem

In general a knowledge acquisition problem can be solved in three ways: (a) by direct elicitation of the knowledge from a source such as a human expert or a document, (b) by induction from observations and (c) by further decomposition into subproblems. Each of these options involves further choices. The main criterion for these choices is **acquisition economy** the expected costs of implementing the option and the expected accuracy of the resulting knowledge (see (O'Hara & Shadbolt 1996)).

In the case of inductive methods the accuracy of the result will depend to quite some extent on the availability of reliable data, the complexity of the actual relations and on knowledge about the type of relation that is to be induced. If many reliable data are available, the underlying relation is not very complex and the type of function is known then induction is likely to be successful but otherwise there is a serious risk of constructing knowledge that is incorrect. In the case of manual acquisition methods, the accuracy depends on the quality of the sources (e.g. human experts) and of the communication process.

More specifically, to decide if decomposition is a good idea, we compare the expected costs and benefits of acquisition with and without decomposition. The benefit of an acquisition operation (elicitation based or induction based) is the accuracy of the resulting knowledge[1]. The costs of the acquisition process depend on the acquisition process. In case of elicitation, this involves time of the expert and of the knowledge engineer, equipment, etc. In case of an inductive approach this involves the costs of collecting and cleaning data and of applying an induction system to the result. If we decompose the acquisition problem, the costs and benefits are simply the sum of those of the subproblems. So we get:

for elicitation:

expected Gain(elicitation) = (w1 * expected Accuracy(elicitation)) - expected Costs(elicitation)

for induction:

expected Gain(induction) = (w1 * expected Accuracy(induction)) - expected Costs(induction)

for decomposition:

expected Gain($operations_{1..i}$) = sum(expectedCosts($operation_i$)) + (w1 * minimum(expected Accucary($operation_i$))

Here the weight parameter w1 indicates the importance of accuracy relative to "real" costs of acquisition. The expected accuracy of a compound acquisition is derived from the minimal accuracy of its components, which is a pessimistic estimate. As we argued above, in some cases elicitation is almost impossible because the expert cannot answer very global questions. This means that the "costs" are very high and the accuracy of the knowledge is 0. In machine learning applications the costs of actually running a system are usually rather small compared to other costs, such as designing the target system, collecting data and tuning the induction tool, so this could be left out.

[1]Ultimately the benefit may be expressed in terms of financial benefits but these are often even more difficult to estimate than the accuracy and the acquisition costs.

Name	Conclusions	Determined by	Cost	Acc
Measure Quality 1	Quality data	Batch (of fruit), Sampling Instructions	0.1	0.5
Measure Quality 2	Quality Data	Batch	0.3	0.9
Select Products	Sampling Iinstructions	Administrative Data, ExternalConditions	0.3	0.8
Make Global Recipe	Specification	Administrative Data, External Conditions, Quality Data, Planned Destination	0.5	0.8
Specify Recipe 1	Detailed recipe	Specification, Planned Destination	0.3	0.4
Specify Recipe 2	Detailed recipe	Specification, Discrepancy (required/predicted fruit quality), Planned Destination	0.3	0.7
Ripening Model	Predicted Output	Quality Data, Detailed Recipe	-	1
Predict Fruit Quality	Predicted Fruit Quality	Predicted Output	0.2	0.8
Analyse Recipe Effect	Discrepancy	Predicted Fruit Quality, Planned Destination	0.1	1
Revise Recipe	Detailed Recipe	Detailed Recipe, Discrepancy	0.1	0.9

Table 1: Some sources of knowledge for the example

In general, decomposition is useful if cheap and accurate sources of knowledge are available for subtasks of the overall knowledge acquisition task but not for the overall task. For example, there may be abundant data for a subproblem and an available and communicative expert for another subproblem but not for the problem as a whole. This is a reason to split the knowledge acquisition problem into subproblems that are then acquired separately. Another reason why a decomposition can be cheaper and give more accurate results than a single step inductive approach is when there is no prior knowledge to bias induction. Suppose that we know that some variables have a "linear threshold" relation with class membership and others have a "complex boolean" relation. Induction techniques are usually biased on one type of relation (e.g. (Langley 1997)) and perform badly if there is a mismatch between this bias and the actual relation. For example, decision tree learning will do badly if the underlying relation can be formulated as a simple linear function. The type of relation may vary in a knowledge acquisition problem. For example, some variables may have a linear relation with a variable that is to be predicted, where others have a non-linear relation. If this is known in advance then decomposing the problem into two subproblems may have a beneficial effect on the accuracy of the result, even when the effects of both types of variables on the predicted variable are not independent.

The method for decomposing a knowledge acquisition problem is based on the idea that the reasons for decomposing a knowledge acquisition problem that we gave in section 3 above are applied in the order given above. The method is a form of *best-first search* that uses expected costs and benefits to evaluate candidate decompositions.

If a step results in a decomposition then the method is applied recursively to the subproblems. The method is as follows (we do not give a detailed algorithm because some of the steps cannot be automated anyway):

KA(ka-problem, sources, w):

1. *IF a source is available for goal THEN use this ELSE*
2. *estimate expected gain of elicitation(ka-problem, sources, w)*:

(a) *estimate costs(elicitation, sources, w)*
(b) *estimate accuracy of resulting knowledge(elicitation, sources, w)*
(c) *expected gain(elicitation) = (w * estimated accuracy) - estimated costs*

3. *estimate expected gain of induction(ka-problem, sources, w)*:

(a) *estimate induction costs(elicitation, sources, w)*
(b) *estimate accuracy of resulting knowledge(elicitation, sources, w)*
(c) *expected gain(induction) = (w * accuracy) - estimated costs*

4. *REPEAT:*

(a) *use any source to decompose ka-problem into ka-subproblem*
(b) *estimate acquisition costs and accuracy and compute gain*

UNTIL *no more decompositions*

5. *select the option (elicitation, induction or a decomposition) that has the highest expected gain.*

6. *IF the result of the last step is a compound ka-problem THEN FOR EACH ka-subproblem DO KA(ka-subproblem, sources, w)*

A knowledge acquisition problem consists of a description of the problem data and the solutions of the task involved. A decomposition consists of knowledge acquisition problems that are connected in a dataflow structure with the original problem data and solutions as input and output. A decomposition is constructed by

- inserting a source description (cf. table 1) that is connected to a type of data in the current goal
- adding or deleting a connection in the dataflow structure

Estimating the cost and accuracy of elicitation and induction is of course the main problem here. There are two methods to obtain these. The first is by using general heuristics and prior knowledge (e.g. the number of variables, the number of data, the strength of bias that can be put used in induction and the expected complexity of the relation between input and output) can be used to predict the accuracy of induction and the costs can often be estimated. The second method is to perform a pilot study to assess the costs and expected accuracy. In the context of elicitation this amounts to performing elicitation on part of the task and evaluating the result. In the context of induction it amounts to cross validation.

Example - continued

We illustrate the method with the example of planning fruit storage introduced above, a further development of the approach of (Verdenius 1996). This problem involved both knowledge acquisition and machine learning. The project was not run with this method in mind but our description can be viewed as a "design rationale" that was constructed afterward. Figure 1 shows the final decomposition and table 2 gives an overview of the sources and techniques that were actaully used to acquire knowledge for the various components. Below we reconstruct the process that lead to this decomposition and choice of acquisition methods.

The estimated costs of acquiring the complete system by elicitation are very high (because there is no expert) and the same is true for induction. There are about 30 input variables and between 20 and 60 output variables (depending on the duration of the storage). The relation is therefore likely to be very complex and it would take many data to find an initial model if it is possible at all. We estimate costs and accuracies of single step acquisition (elicitation or induction). The estimates are in table 3. The last column gives the expected gain using a weight of 1.

We now consider decomposition. The available sources and the causal and temporal structures define a number of possible decompositions of the initial knowledge acquisition problem. There are many possibilities and here we describe some possibilities with estimates of the expected accuracy and acquisition costs.

Method	Expected accuracy	Exp. costs	Gain
Elicitation	0.2	1	-0.8
Induction	0.7	1	-0.3

Table 3: Estimated cost and accuracy of single step acquisition

Component	Exp. accuracy	Exp. costs
C1	.8	.5
C2	.9	-
C3	.3	.9

Table 4: Expected costs and accuracies for decomposition 1

Decomposition 1 Quality Data are obtained by first finding Sampling Instructions and then applying these to sample fruits from the batch. The resulting quality measures are used directly to find a Detailed Recipe. This is summarised in the diagram in 3.

The expected costs and accuracies of acquiring the components is given in table .

The `Quality Data` are not suitable as input for the `Ripening Model` and therefore an additional step must be inserted. Recipes are high dimensional: a small recipe covers ca. 14 days, with 2-6 prescribed values a day. Learning this in one step will require huge data sets, with as an additional problem that identical batch characterisations may lead to different recipes (for a complex of reasons). However, all these recipes will share their global specification. With w equal to 1, this gives a total expected gain of $(1 * 0.3) - 1.4) = -1.1$. This is no improvement over single step elicitation or induction.

Decomposition 2 This decomposition does not construct sampling instructions but takes measures from a random sample. It uses this with other available data to first construct a Specification of a recipe (global recipe) that is then refined. This can be summarised in 4. The expected costs and accuracies are in table .

This gives a total expected gain of -0.4 for decomposition 2. This option is better than the previous one.

Decomposition 3 This is summarised in 5. The estimated costs of acquiring and accuracies of components are in table 6.

Component	Exp. accuracy	Exp. costs
C1	.6	-
C2	.7	.9
C3	.9	.1

Table 5: Expected costs and accuracies decomposition 2

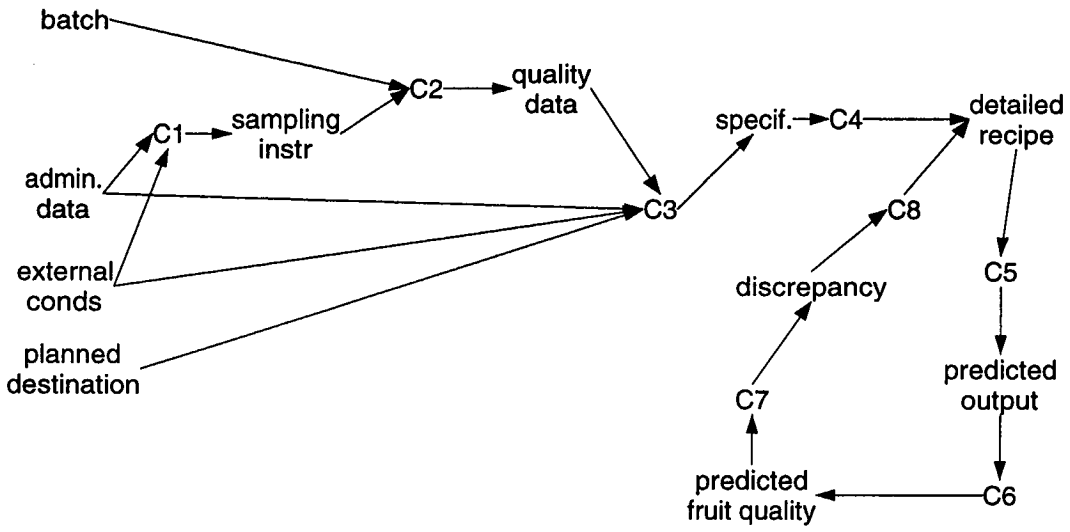

Figure 2: Decomposition of the learning problem

Component	Name	By
C1	select products	machine learning (Decision tree)
C3	specify recipe	machine learning (Case-Based Reasoning)
C4	detail recipe	elicitation
C5	evaluate recipe	- (given)
C6	assess recipe effect	common sense
C7	analyse recipe effect	common sense
C8	adapt recipe	common sense

Table 2: Acquisition methods and sources for components

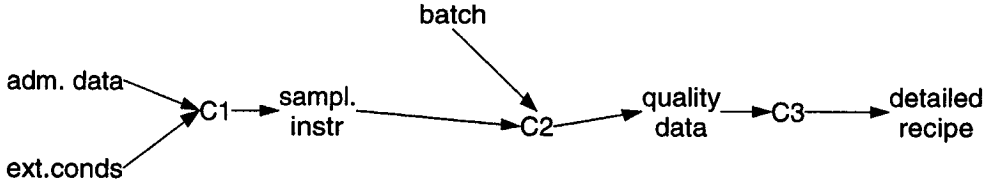

Figure 3: Decomposition 1

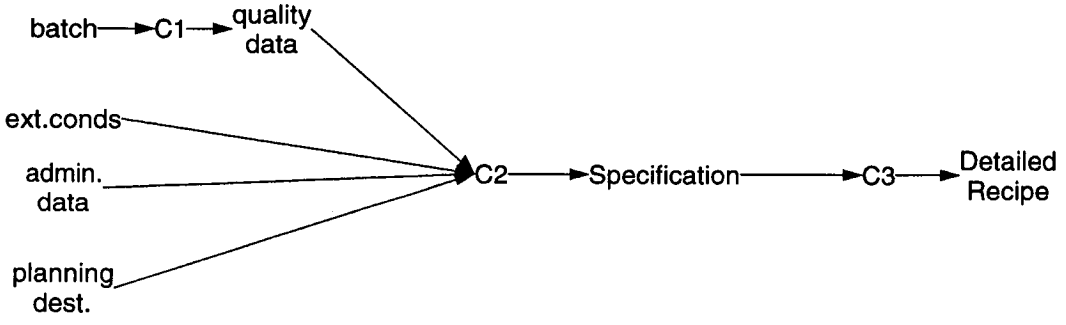

Figure 4: Decomposition 2

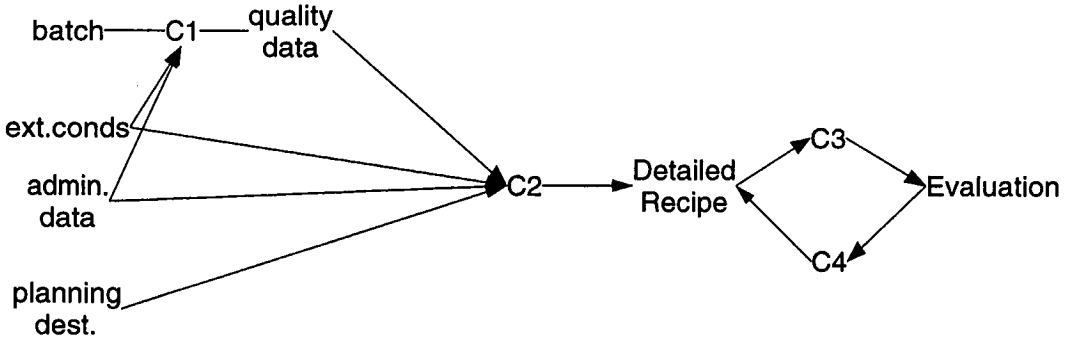

Figure 5: Decomposition 3

Component	Exp. accuracy	Exp. costs
C1	.5	-
C2	.5	.5
C3	.6	.9
C4	.8	.1

Table 6: Expected costs and accuracies decomposition 3

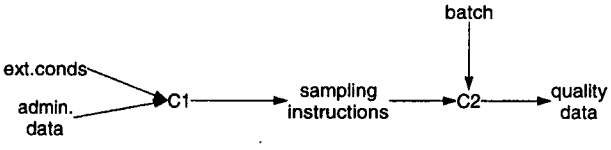

Figure 6: Decomposition C1.1

This gives a total expected gain of -1.0. This suggests that at this level the best decomposition is decomposition 2. The next step is to find the best way to acquire C1, C2 and C3. This can be done by elicitation, induction or further decomposition and we apply the approach recursively.

Acquiring component C1

First consider C1 in decomposition 2 above. This component means that fruits are taken from a batch and quality measurements are taken on the fruits. This does not involve inference because it is done outside the system by a human user, who enters the quality data into the system. For this component no knowledge needs to be acquired and therefore it involves no acquisition costs. We estimate the accuracy as only 0.3, because unsophisticated sampling gives less relatively poor quality data. This is also the gain. Using the sources from table 1 there are many decompositions for C1. Again, rather than reviewing them all we give two examples, see 6 and 7.

The first of these uses the source Measure Quality 1 and the others use only part of this. Table 7 gives the expected costs and accuracies of acquiring C1 in the form of C1.1 and C1.2 respectively. C1.1 can be

Figure 7: Decomposition C1.2

Component	Exp. accuracy	Exp. costs
C1.1	.5	-
C1.2	.5	.5

Table 7: Expected costs and accuracies of decompositions of C1

realised by using a source that brings no costs. The expected costs of acquiring C1.2 are about average as is the expected accuracy.

This gives gain 0.5 for C1.1 and 0.0 for C1.2. C1.1 is better than the original C1 so we replace C1 by C1.1. There is no source for further decomposition. This leaves only the choice between elicitation and induction. Because a source is available, C1.1 is not more expensive than C1.2 but it is more accurate.

Acquiring components C2 and C3

Component C2 of decomposition 2 should find a Specication from Quality Data, External Conditions, Administrative Data and Planned Destination. This corresponds to the source Specify Recipe and can be acquired from this source. Similarly, component C2, that should find a Detailed Recipe from a Specification, can be acquired from source Specify Recipe 1. The accuracies and costs involved can be found in table 1. We shall not pursue the example further here. The method finds that further decomposition of C3 is beneficial and finally constructs the model in figure 1. This is then used as the plan for knowledge acquisition for the original problem.

Discussion

We presented a rational reconstruction of decisions to use machine learning in a knowledge acquisition con-

text. Applications of machine learning to knowledge acquisition involve more than selecting and applying an appropriate induction tool. In general knowledge or data are not or only partially available and decisions must be taken on how to acquire them. Knowledge acquisition problem are often better solved using a "divide-and-conquer" approach that reduces the overall problem to subproblems that can be solved by machine learning or direct elicitation. The process is guided by estimations of costs of the acquisition process and of the expected accuracy of the result.

Several current knowledge acquisition methods rely heavily on the idea of decomposition (e.g. (Terpstra *et al.* 1993), (Schreiber, Wielinga, & Breuker 1993), (Marcus 1988)). However, these methods focus on "modelling" languages and do rarely make the underlying principles explicit that are needed for a rational application of the methods. These methods also do not cover the use of inductive techniques. Here we reconstruct the rationale behind these methods and use this to extend them to include the use of machine learning methods. We presented criteria and a method for decomposing knowledge acquisition problems into simpler subproblems and illustrated this with a reconstruction of a real world application. This method can be applied both to inductive methods and to, for example, knowledge elicitation or other manual acquisition methods.

The main limitation of the method is that the decisions for decomposition and tool selection require prior knowledge about possible decomposition and about expected benefits and costs. In knowledge acquisition these are part of the "experience" of knowledge engineers. In machine learning and in statistical data analysis, rules of thumb and experience are used to estimate the expected accuracy of the result of applying an induction system. For example, for many statistical techniques, rules of thumb relate the number of variables, the complexity of the function to be induced and the number of data to an estimate of accuracy. The main alternative, if the is no prior knowledge, is currently a "reactive" approach. The expected accuracy of applying an operator can be determined empirically by trying it out. For inductive techniques this is done by cross validation which produces an estimate of the accuracy and in knowledge elicitation this is done by simply asking an expert to provide the knowledge. If this fails it means that decomposition is necessary. See (Brodley 1995) for a method following this approach. A similar approach in the context of knowledge acquisition is followed by Graner and Sleeman (Graner 1993). Their model does not include search through possible decompositions or the use of estimated costs and accuracies.

The method outlined here can be extended to include the expected gain of having the resulting system. This would give a more comprehensive model including both the costs of acquisition and the costs of having and using the acquired knowledge. See (van Someren, Torres, & Verdenius 1997) for a model of induction methods that include costs of measurements and costs of errors, in the context of learning decision trees. These two models can be integrated into a single model, see for example (DesJardins 1995) for a similar model for robot exploration.

References

Brodley, C. 1995. Recursive bias selection for classifier construction. *Machine Learning* 20:63–94.

Craw, S., and Sleeman, D. 1990. Automating the refinement of knowledge-based systems. In Aiello, L. C., ed., *Proceedings ECAI-90*, 167–172. London: Pitman.

DesJardins, M. 1995. Goal-directed learning: a decision-theoretic model for deciding what to learn next. In *Goal-Driven Learning*. MIT Press.

Engels, R.; Lindner, G.; and Studer, R. 1997. A guided tout through the data mining jungle. In *Proceedings of the 3rd International Conference on Knowledge Doscovery in Databases (KDD-97)*.

Ginsberg, A. 1988. *Refinement of Expert System Knowledge Bases: A Metalinguistic FrameWork for Heuristic Analysis.* Pitman.

Graner, N. 1993. The muskrat system. In *Proceedings second workshop on multistrategy learning.* George Mason University.

Langley, P. 1997. *Elements of Machine Learning.* Morgan Kaufmann.

Marcus, S., ed. 1988. *Automatic knowledge acquisition for expert systems.* Boston: Kluwer.

O'Hara, K., and Shadbolt, N. 1996. The thin end of the wedge: Efficiency and the generalised directive model methodology. In Shadbolt, N.; O'Hara, K.; and Schreiber, G., eds., *Advances in Knowledge Acquisition.* Springer Verlag. 33–47.

Polderdijk, J.; Verdenius, F.; Janssen, L.; van Leusen, R.; den Uijl, A.; and de Naeyer, M. 1996. Quality measurement during the post-harvest distribution chain of tropical products. In *Proceedings of the Congress Global Commercialization of Tropical Fruits*, volume 2. 185–195.

Schreiber, A. T.; Wielinga, B. J.; and Breuker, J. A., eds. 1993. *KADS: A Principled Approach to Knowledge-Based System Development*, volume 11 of *Knowledge-Based Systems Book Series.* London: Academic Press. ISBN 0-12-629040-7.

Shapiro, E. Y. 1982. *Algorithmic Program Debugging.* ACM Distinguished Dissertations series. Cambridge, Massachussetts: MIT Press.

Terpstra, P.; van Heijst, G.; Wielinga, B.; and Shadtbolt, N. 1993. Knowledge acquisition support through generalised directive models. In David, J.-M.; Krivine, J.-P.; and Simmons, R., eds., *Second Generation Expert Systems.* Berlin Heidelberg, Germany: Springer-Verlag. 428–455.

van Someren, M.; Torres, C.; and Verdenius, F. 1997. A systematic description of greedy optimization algorithms for cost-sensitive generalization. In Cohen, P.,

and Lui, X., eds., *Proceedings Intelligent Data Analysis 1997 (IDA-97)*. Springer Verlag. 247–258.

Verdenius, F. 1996. Managing product inherent variance during treatment. *Computers and Electronics in Agriculture* 15:245–265.