Scalable Data Mining Systems*

Nicholas J. Radcliffe[a,b] and Ian W. Flockhart[a]
{njr,iwf} @ quadstone.co.uk
[a]Quadstone Ltd, 16 Chester Street, Edinburgh EH3 7RA, UK
[b]Department of Mathematics and Statistics, University of Edinburgh,
King's Buildings, EH9 3JZ, UK

Keywords: *scalability*, *data mining*, *decision support*

Abstract

The growing number of large data warehouse installations is leading to an increasing emphasis on *scalable* decision support systems. These are usually thought to be characterised primarily by their use of linear-time algorithms and their ability to exploit parallel processing. This paper argues that other considerations, including integration, data access patterns and "full-processing-cycle" speed, are at least as important in constructing scalable decision support systems capable of delivering real business benefits.

*This work was partially funded by a research grant from the UK Engineering and Physical Science Research Council's AIKMS programme.

Abstract

The growing number of large data warehouse installations is leading to an increasing emphasis on *scalable* decision support systems. These are usually thought to be characterised primarily by their use of linear-time algorithms and their ability to exploit parallel processing. This paper argues that other considerations, including integration, data access patterns and "full-processing-cycle" speed, are at least as important in constructing scalable decision support systems capable of delivering real business benefits.

1 Orientation

If data mining is the process of turning data into information, then decision support is that of turning data into actionable, profit-increasing insights. As ever larger data warehouses are constructed and populated the focus inevitably turns to *scalable* decision support systems—those that can help large organisations to extract maximum value from extremely large (multi-terabyte) datasets. This paper is primarily concerned with the practicalities of building high-value decision support systems in such environments.

The reality of most "data mining" or "knowledge discovery" systems today is that they consist essentially of repackaged versions of traditional machine-learning algorithms for classification and clustering, usually augmented with some interface packaging to allow back-end exploitation of databases and front-end exploitation of spreadsheets and graphics packages. The standard classification methods in use include (additive) scorecarding (e.g. Lewis, 1992), feed-forward neural networks (Rumelhart *et al.,* 1986) decision trees (Breiman *et al.,* 1984; Quinlan, 1986, Quinlan, 1993) (usually constructed with a greedy algorithm such as ID3), nearest neighbour methods (e.g. Gates, 1972) and classifier systems (Holland, 1986); the clustering methods typically concentrate on the various recursive splitting algorithms or unsupervised neural networks, particularly Kohonen nets (Kohonen, 1989). One of the major contentions of this paper is that the heavily *algorithmic* focus of most data mining packages overlooks the real dynamics of practical knowledge discovery and leads researchers to ignore some critical impediments to genuine scalability present in current systems.

The main sections of this paper discuss, in turn, the "linear process" myth, which views knowledge discovery as a well-defined, straightforward but expensive process, the "linear time-complexity myth", which suggests that the main problem for research into data mining research is to discover appropriate linear-time algorithms, and the "accuracy myth", which concentrates attention, arguably too narrowly, on the mathematical accuracy of pattern-detection algorithms.

2 The Linear Process Myth

If we were to base our view of data mining strongly on machine learning, we might expect the process to look something like that illustrated in figure 1. The user comes with a well-defined goal (preferably a classification task), selects the appropriate (previously cleansed) data from the database, dividing it into appropriate training and validation sets, builds a model from the data (the time-consuming "mining" part of the process), validates the model and finally hands it over for operationalisation (whereby data is actually turned money). We call this the "linear process" view of data mining (or the "linear process myth").

Well-defined goal → Data selection/ sampling → Model building → Model Validation → Model use

Figure 1: A machine-learning-inspired view of data mining.

Experience suggests that a rather more accurate view is portrayed in figure 2. Here the user begins with a fuzzy goal, selects some seemingly relevant data from the database (probably mostly transactional data), aggregates and otherwise coerces that data into a form appropriate for the chosen mining algorithm and then inspects the resulting data visually. At this stage, if not before, the user typically discovers that (1) the ranges of most of the variables are inappropriate, (2) the "cleansed" data is still far from "clean", and (3) such patterns as can immediately be seen appear to be artifacts rather than genuine insights into the underlying data. This typically causes some or all of the stages already completed to be revisited. When eventually the data for the model building looks vaguely plausible, the user typically builds a trial model and then carries out some sanity checks by inspecting the model and the data visually. The sanity checks usually prove the model to be inadequate, resulting in further iteration of the earlier processes. The pattern, in our experience, usually continues for some while before converging on anything that might reasonably be described as "discovered knowledge".

This "bitter experience" view of data mining immediately suggests several lessons relevant to constructing scalable systems:

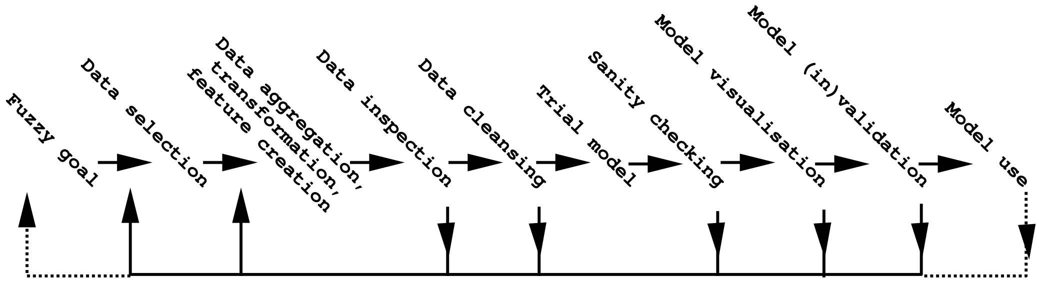

Figure 2: A "bitter experience" view of data mining.

- No matter how fast the "mining" algorithm, knowledge discovery will be slow if all the pre- and post-processing stages are not similarly fast, or if integrated visualisation tools are not provided.
- The raw data is rarely in a form suitable for mining. As much attention needs to be paid to feature creation as to the mining technology. In particular, scalability is a consideration for the pre-processing stages as well as for the mining components of knowledge-discovery systems.
- The user will be performing each stage many times during one complete analysis: it must be easy to move large datasets between the different stages quickly.

3 The Linear Time-Complexity Myth

A second worryingly pervasive myth about data mining is that the key to performance is ensuring that the time complexity of the algorithm be linear (or near-linear) "in everything". While linear time complexity is certainly a *sine qua non,* at least in the number of data elements, this seems to over-emphasize sheer *computational* considerations relative to another critical algorithm parameter—the data access pattern. It is a major contention of this paper that data access is the single most critical determinant of performance not only for the actual 'mining' component of decision support systems, but also for the other components and thus for the system as a whole.

Consider first the possible locations of a data element immediately before it is accessed. In decreasing order of speed of access, but (arguably) increasing order increasing speed →

remote database	local database	remote file	local file	main memory	secondary cache	on-chip cache

← increasing flexibility

Figure 3: Data accessibility for processing

of flexibility and likelihood, its location (figure 3) might be on-chip cache, secondary cache, main memory, virtual memory, local disk file, remote disk file, local database, remote database (or data warehouse).

Even in this simplified view, it should be emphasized that the differences between access times for the different data locations is large (some orders of magnitude), with a typical factor of at least one hundred between cache and a (non-memory-resident) database. Given that the datasets now mined, even after aggregation and preprocessing, are often significantly larger than the main memory available, this discussion brings into sharp focus the importance of:

- the number of passes through the data required;
- the locality of the data access pattern required by the algorithms (for both maintenance of cache-coherency and minimisation of disk-head movement);
- the performance of the cache and I/O subsystems on the target hardware platform.

Data access patterns, of course, become even more critical when parallel processing is considered as a route to scalable performance. Naïvely, this might be thought to be the case only for MPP (distributed-memory) architectures, where an off-processor fetch is typically significantly slower than a local fetch. In reality, however, the difference between the performance of modern cache compared with main memory is such that locality of access is also a highly significant issue on SMP (shared-memory) platforms.

One of the fundamental considerations in exploiting parallel processing as a route to enhanced performance is Amdahl's law (e.g. Morse, 1994), which states (loosely) that best wall-clock time performance available on a parallel platform is governed by the largest sequential task (figure 4). In the context of data mining on large datasets, data access typically dominates processing to such a degree that it is often possible to increase the complexity of processing on each data element quite

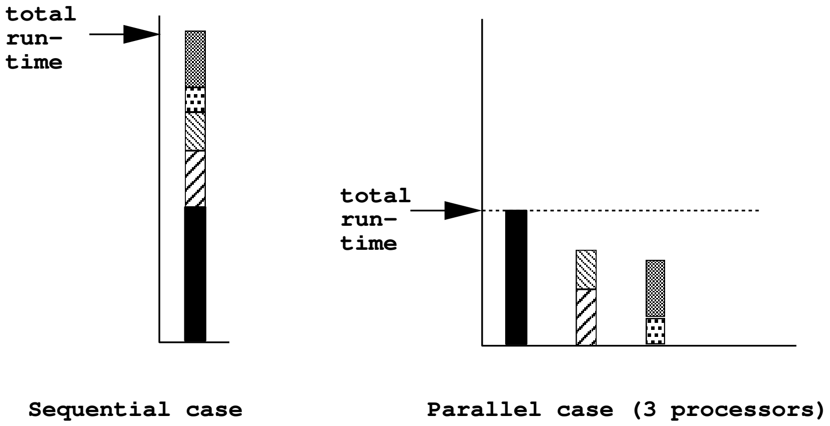

Figure 4: Amdahl's law states that no matter how many processors are used, the total run-time of an application can never be less than that of the largest sequential task it contains. The left-hand figure shows a stacked bar graph with the execution times of various irreducible sequential tasks. The right-hand figure shows these tasks allocated to different processors (on the x-axis). If the dominating 'task' is fetching data, and the fetch operations can be overlapped with computation, more computation can be carried out without a performance penalty being incurred.

significantly without severely degrading overall performance. This breaks down, providing that the data access pattern is not disturbed, only when the capacity of the data access system is such that it can deliver data for processing as fast as that processing can be carried out. This is rare today.

4 The Accuracy Myth

The constant goal in machine learning is to drive up the accuracy (and robustness) of models created from the data. Without in any way wishing to diminish the importance attached to accuracy, the authors feel that it is important to bear in mind that models are limited not only by the accuracy with which they model the *data,* but also by the accuracy with which they represent the actual *goal* of the model builder (the user). Experience suggests that in reality users typically have imprecise rather than sharply-defined goals, and multiple (often competing) objectives rather than one. Moreover, users often place at least as much emphasis on the comprehensibility, plausibility, adaptability and speed of construction of models as they do on their (strictly mathematical) accuracy against the objective *as formulated.* (Perhaps even more fundamentally, the models that are built are of use only in so far as they produce *actionable* insights, but this is peripheral to the main theme being pursued in this paper.)

While the considerations above in no way diminish the desirability of accurate models, they throw accuracy into somewhat different relief. Very often what data analysts need is a tool that allows them to follow a train of thought as they explore data, testing hypotheses, spot patterns and so forth. Even when complex models are built (for example, a decision tree to profile the response to a marketing campaign), it is often ultimately used only to extract one or two key insights, such as a pair of key variables, or a single interesting niche subpopulation from a customer base. For these reasons, providing "quick and dirty" models of limited accuracy can be every bit as valuable as making available more accurate models that (inevitably) usually take longer to construct. Even if the ultimate goal is a high-accuracy model, given the iterative nature of the overall knowledge discovery *process* discussed above, and illustrated in figure 2, it may well be desirable to use faster, lower accuracy methods in the early stages of analysis.

5 Conclusion

This paper has presented an informal consideration of some of the key features of scalable decision support systems, and has argued that the principal determinants of scalability are not necessarily those most commonly discussed. In particular we have argued that truly scalable decision support systems must embrace the entire knowledge discovery process, from data source to actionable output. Only then will decision-support systems generate profit-increasing insights from large datasets. Where traditionally the emphasis has been on accurate task specification, linear-time analysis algorithms and high-accuracy models, we have here argued that the reality of ill-specified tasks must be accepted and users must be supported in train-of-thought data exploration, aided by analysis algorithms tuned for low-cost data access patterns and producing comprehensible, tunable, actionable models of controllable accuracy.

References

L. Breiman, J. Friedman, R. Olshen, and C. Stone, 1984. *Classification and Regression Trees*. Wadsworth International Group.

G. W. Gates, 1972. The reduced nearest neighbour rule. *IEEE Transactions on Information Theory*, IT-18(431).

John H. Holland, 1986. Escaping brittleness: the possibilities of general-purpose learning algorithms applied to parallel rule-based systems. *Machine Learning, an artificial intelligence approach*, 2.

T Kohonen, 1989. *Self-Organisation and Associative Memory*. Springer-Verlag (Berlin) Springer Series in Information Sciences 8.

E.M. Lewis, 1992. *An Introduction to Credit Scoring*. The Athena Press (California).

Stephen H. Morse, 1994. *Practical Parallel Computing*. AP Professional (Boston).

J. R. Quinlan, 1986. Induction of decision trees. *Machine Learning*, 1(1):81–106.

J. R. Quinlan, 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann series in Machine Learning. Morgan Kaufmann, San Mateo, California.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams, 1986. Learning representations by back-propagating errors. *Nature*, 323.