A high-performance data structure for mobile information systems.

John N. Wilson

Department of Computer and Information Sciences, University of Strathclyde, 26, Richmond St, Glasgow, G1 1XH, UK
jnw@cis.strath.ac.uk

Abstract. Mobile information systems can now be provided on small form-factor computers. Dictionary-based data compression extends the capabilities of systems with limited processing and memory to enable data intensive applications to be supported in such environments. The nature of judicial sentencing decisions requires that a support system provides accurate and up-to-date data and is compatible with the professional working experience of a judge. The difficulties caused by mobility and the data dependence of the decision-making process are addressed by an Internet-based architecture for collecting and distributing system data. We describe an approach to dictionary-based data compression and the structure of an information system that makes use of this technology.

1 Introduction

Over the last decade, information systems (IS) have developed to encompass Internet technology and the advantages that this paradigm provides for co-operation between decision-makers. More recent developments have resulted in progressive reduction of the form-factor of the kinds of hardware needed to underpin IS. This has coincided with the spread of IS technology to mobile users. The professional activities of such users require them to operate in sites away from the typical centralised provision of technical support that has underpinned conventional concepts of IS. Laptop computers have helped to provide technological support in this kind of working environment. The next stage in the miniaturisation of computer systems is the personal digital assistant (PDA). Such systems represent many benefits for mobile users in business environments however miniaturisation necessitates the use of small and relatively slow processors as well as limited random access memory. Whilst the increasing density of integrated circuits will undoubtedly result in improvements in processor and memory performance, the dependence of miniaturised systems on battery power is likely to remain a constraint on small form factor technology. Data compression can be used to mitigate some of these problems if it can be arranged that the subsequent processing of such data does not require extensive on-line use of decompression. This paper describes the development of a mobile information system to support judicial decision making and the compression strategy needed to ensure that operation of the system is compatible with the working environment of the target user group.

A taxonomy of mobile working environments provides a useful framework for understanding mobile information systems. We classify working environments as either static or peripatetic following the work of Dahibom and Ljungberg [2]. The peripatetic subtype can be further classified as deterministic, non-deterministic or dynamic. Deterministic mobility is characterised as taking place where there are fixed working locations and the worker has to travel between these locations. Information technology support is typically provided by desktop systems in the various locations. Non-deterministic mobility is defined by working activities in which the worker is expected to visit different locations but these locations can not be predicted. The location would typically provide some support for the working environment and a laptop computer would be an appropriate platform for meeting information technology needs. In dynamic mobility, there is no prior determination of the location in which work takes place and no provision for a conventional working environment. In an environment such as this, a PDA represents a useful platform for information technology support. Many professional activities involve different classes of mobility. Whilst formal interactions may take place in static or deterministic modes, the nature of professional work suggests that the process is often carried out in non-deterministic or dynamic modes. This is particularly true of some of the processes that contribute to judicial decisions. Whilst some of the work is carried out in static or deterministic modes, non-deterministic and dynamic work is also carried out within and outside the locations where justice is formally dispensed. Given this pattern, a laptop computer is appropriate where the judge is working in a defined location and a PDA provides the best platform for supporting dynamically mobile aspects of professional activity. A further factor that may influence the choice of platforms is the social acceptability of computer usage. A laptop computer is acceptable in a formal working environment. By contrast, a PDA can be used without intrusion in business meetings and other locations. As a consequence of our analysis of the mobility of our user group, the initial target platform chosen was a laptop computer. Subsequent developments of PDA technology have opened the possibility of providing support for dynamic mobility.

1.1 Judicial sentencing decisions

Judges tend to enjoy very wide discression in the manner in which individual cases are dealt with. There are many factors that may be taken into account by a judge during the sentencing process. Most jurisdictions provide a defined maximum penalty for any offence, but beyond that there is little statutory guidance. Similarly, the overall aim of the sentencing process is ill-defined. Deterrence, rehabilitation, protection of the public and retribution are all identifiable as valid aims but in reality, a particular sentence may well be informed by a combination of these factors. Given the emphasis on the individualised nature of the sentencing process, it is not surprising that disparity in sentencing practice emerges from time-to-time as an issue for public concern. Public expectations of consistency in sentencing are reflected in media response to sentences that appear to be unusually lenient or severe although there is often no clear concensus on what could be termed 'normal' in any particular set of circumstances. The focus of this paper is on the technical support that can be used to provide a judge with an overview of sentencing patterns for cognate offences.

Decision support for sentencing has been investigated in a number of jurisdictions. The courts in New South Wales provide access to first instance sentences as well as to sentencing law and appeal court judgements [6]. The system operates by allowing the user to specify the act and section that relate to the offence. Characteristics of the offender may be specified by the user and sentencing patterns are displayed to show the distribution of particular types of sentence. The data is centralised and can be accessed over a wide area network from remote sites. An alternative approach to providing computer support for sentencing is to use a knowledge-based system to guide sentencing decisions. Murbach and Nonn [5] describe a prototype sentencing DSS limited to fraud cases. Discussion with the judiciary was used to establish a sentencing model and subsequently to construct a knowledge-based system to incorporate the reasoning process.

The requirement for our system is to provide a means whereby a judge may retrieve previous examples of sentencing behaviour. The judge is able to control the level of abstraction used to retrieve data. Given the mobile nature of a judge's working experience, it is necessary to provide this support in a flexible way using different platforms. Off-line compression of data presents a useful way of enabling rapid transfer of the database to a PDA-based system therby supporting the dynamically mobile working environment of our user group. The initial system design was based on a model of deterministic mobility. We provide support for this by the use of laptop computers. The additional implementation effort of data compression is not essential in the latter environment.

2 Architecture

Data compression is useful for underpinning the operation of the PDA-based platform since it helps to conserve memory and battery power. Conventional main-memory databases represent data items as strings [3] [4] although work is reported on the use of fixed length pointers to replace domain values [7]. Our strategy is to mimimise the size of the tokens used to represent data by basing token size on the entropy of the domain to be represented. Our method of compression enables us to compress the data off-line and resolve queries by de-compressing only the output data [1].

2.1 Dictionary-based data compression

The minimum entropy approach to data compression requires that each domain in a database is represented by a code with a minimum number of bits. The

NAME	START YEAR	SERVICE INTERVAL
Corolla	1980	6000
Carina	1984	10000
Camry	1983	10000
Celica	1985	10000
Supra	1980	6000
Space Cruiser	1985	10000

Fig. 1. MODEL relation

NAME	13
START YEAR	4
SERVICE INTERVAL	5
Tuple size	22

Table 1. Field lengths and tuple sizes for the MODEL relation

relation MODEL (Figure 1) represents data as it would be stored in an uncompressed database.

This data would typically be stored in tables with fixed length fields in which case the field lengths and associated tuple size in bytes would be as shown in Table 1 Since the relation MODEL contains six tuples, the size of the relation would be 6 * 22 * 8 = 1056 bits Inspection of the table suggests that it would be possible to represent the information content in a more compact form by using codes to represent the domain values rather than using the domain values themselves in the relation. The attribute NAME contains six different values therefore in its most compact form, it could be represented as a three bit integer. Since there are only two values in the SERVICE INTERVAL attribute, it could be represented as a one bit integer. The compressed integer representation of the relation MODEL is shown in Table 2.

Raw			Compressed
Corolla	1980	6000	000 00 0
Carina	1984	10000	001 10 1
Camry	1983	10000	010 01 1
Celica	1985	10000	011 11 1
Supra	1980	6000	100 00 0
Space Cruiser	1985	10000	101 11 1

Table 2. Compression of MODEL

The effect of representing data in this entropy-encoded format is to reduce the space occupied by MODEL to 36 bits. To this it is necessary to add the dictionary that allows the tokens to be converted to their string equivalents and for the reverse process to take place (Figure 2). Dictionaries can be represented as lists of domain values. In the case of all non-key attributes the number of entries in the dictionary will be less than the number of tuples in the relation.

NAME
Corolla
Carina
Camry
Celica
Supra
Space Cruiser

START YEAR
1980
1983
1984
1985

SERVICE INTERVAL
6000
10000

Fig. 2. Dictionaries for MODEL

NAME	ENGINE SIZE	ENGINE STYLE
Corolla	1290	OHV
Corolla	1587	OHV
Carina	1587	OHC
Carina	1974	OHC
Camry	1839	OHC
Camry	1974	OHC

Fig. 3. ENGINE relation

Suppose that we have a second relation, ENGINE, that specifies the engines fitted to various models (Figure 3). The domain from which the values in the NAME attribute of ENGINE are drawn is a sub-domain of that used in the MODEL relation. Since there are 3 models listed, first order compression of the column would require the use of 2 bit integers. This however would entail the establishment of a new dictionary to handle the values in the NAME attribute for the MODEL relation. In the case of a primary key, the size of such a dictionary is almost certain to be greater than the total number of bits required to encode the column.

Our objective is to minimise the size of the total database. It therefore makes sense to determine the columns in the database that can share dictionaries. These are the columns that contain values belonging to the same domain. An advantage of a tokenised architecture is that it can accelerate the evaluation of queries. If a query initially expressed in terms of string pattern matches, is compressed, its new expression is in terms of integer comparisons. These are inherently faster because the amount of data that has to be moved into the CPU and compared for each selection operation is significantly reduced. Clearly the final answer has to be converted from the abstract to the concrete representation. This involves a decompression phase in which the field codes are mapped to their associated keywords. The computational cost of this will be borne only on the tuples that are returned as a result. Since this phase is dominated by output delays in the screen handling software, the perceived cost of this decompression is likely to be minimal.

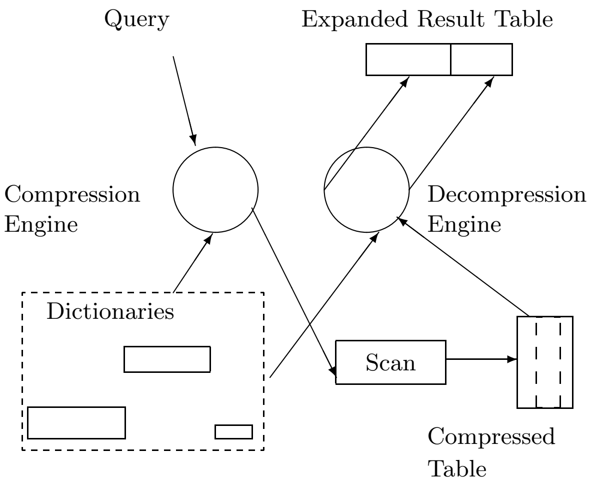


Fig. 4. Querying a database in the compressed form.

The purpose of the information system is to provide judges with a means of retrieving previous examples of sentencing behaviour. The compression strategy we have developed can be used to enable this function to be provided in small form-factor systems. The architecture is shown diagrammatically in Figure 5. This structure has evolved over a period of eight years since the start of the project. There are three main elements in the structure: a non-mobile element that is used by clerical staff to carry out the functions of data entry and data quality control, a server that acts as a means of collating and distributing updates and a number of mobile client systems. As far as the end users of the system are concerned, the data we provide is read-only. This means that the compression needed to support the PDA architecture can be carried out off-line. In general we are able to arrange our compression system so that it can handle updates but in the case of this application it is not necessary. Instead, updates are made to the conventional representation of the data and the compression is carried out off-line so as to generate a new distribution of the database.

Database inserts can be carried out from any convenient workstation in the High Court buildings distributed round the country. These buildings all provide wide area network facilities and clerks are familiar with the use of web browsers. Passwords are used to identify data entered with particular clerks (in case of consistent misunderstandings about the taxonomy of offence details). It is also simpler if data is entered directly into the consolidated database rather than stored locally for forwarding and consolidation at a later date. The advantages of a browser-based solution to this problem are considerable. As well as familiarity and controlled access, the code and the underlying data of the data entry system remains accessible for maintenance purposes. In this particular case, CGI scripts were used since this environment was familiar to development staff, however other server-side scripting technologies would be equally acceptable.

The server requires a number of database management systems to solve licensing problems in a multi-operating system environment. The data entry scripts and the main database are resident on a Unix system but the mobile clients run on versions of MS Windows. To resolve this problem, an ODBC link is used to replicate the consolidated database into the proprietary system that is appropriate for the laptop clients.

On each client system, a wrapper is used to contain the data retrieval system. The purpose of the wrapper is to update the database if a newer version is available on the server than that present on the client. In many cases it will be inappropriate to carry out the copying process that would be necessary to transfer the replica database from the server to the client. The wrapper contains a number of rules that enable it to determine whether it is appropriate to carry out the update.

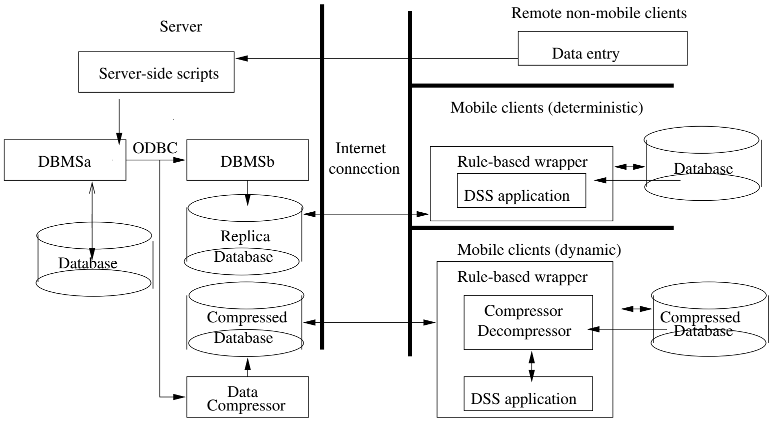


Fig. 5. Architecture for a mobile data-intensive information system.

The application operates by allowing the judge to select cases that match some of the criteria of the case under consideration. We directly involved the user community in the design of the interface and in the development of a taxonomy for the cases to be entered into the system. Early in this process it became clear that the focus of the system was the retrieval of cases appropriate to a particular decision. We considered and rejected a knowledge-based approach that could have been used to suggest an appropriate course of action in a given set of circumstances. One of the challenges of the design process was to ensure that data was aggregated in such a way as to provide a substantial number of cases that would help with any specific decision. There are about one thousand serious criminal convictions in Scotland every year. The bulk of these are assaults and robberies however our user group was able to identify fifteen different categories of criminal activity. Associated with each category are a variable number of attributes of both the offender and the offence.

The data can be considered as a multi-dimensional cube with each dimension being one of the indexed attributes. The cells within the matrix contain aggregations of the cases that correspond to the index values on each dimension. The sparsity of the matrix is the proportion of non-zero elements. The retrieval of data by specifying values along each dimension quickly reveals the curse of dimensionality. Judges retrieving cases from a small pool of data by specifying many dimensions are likely to find only small numbers of matches.

A detailed analysis of the judges needs to search the database on a relatively large number of attributes results in a data cube with a sparsity of 0.14%. The effect of this is that users are unlikely to be able to retrieve interesting cases if they specify values for all or most of the indexed attributes. We use two approaches to counteract this problem. Allowing the user to define the attributes used to index data permits aggregations to be constructed over fewer dimensions, thereby reducing the sparsity. During the user training, we emphasise that the most productive use of the system is achieved by specifying only the most important criteria in the first instance. The user is left to decide which attributes are of importance in a particular case. It is quite easy to gauge whether there is sufficient data to support queries with greater specificity. This approach implies that the system responds rapidly enough to be searched on an interactive basis. As an alternative approach to the problem, the users developed a second taxonomy that could be used to group cases on an holistic basis. This approach lost some of the detail from individual cases but the coarser grouping provided larger aggregations for each cell in the resulting matrix. Using this approach, the sparsity of the data structure was 11.25%.

We provided a facility that allows the user to drill down into the aggregations produced by the top-level query. Drill down provides progressively more information about each aggregate and at the lowest level, users are able to view the details of specific cases.

3 Implementation

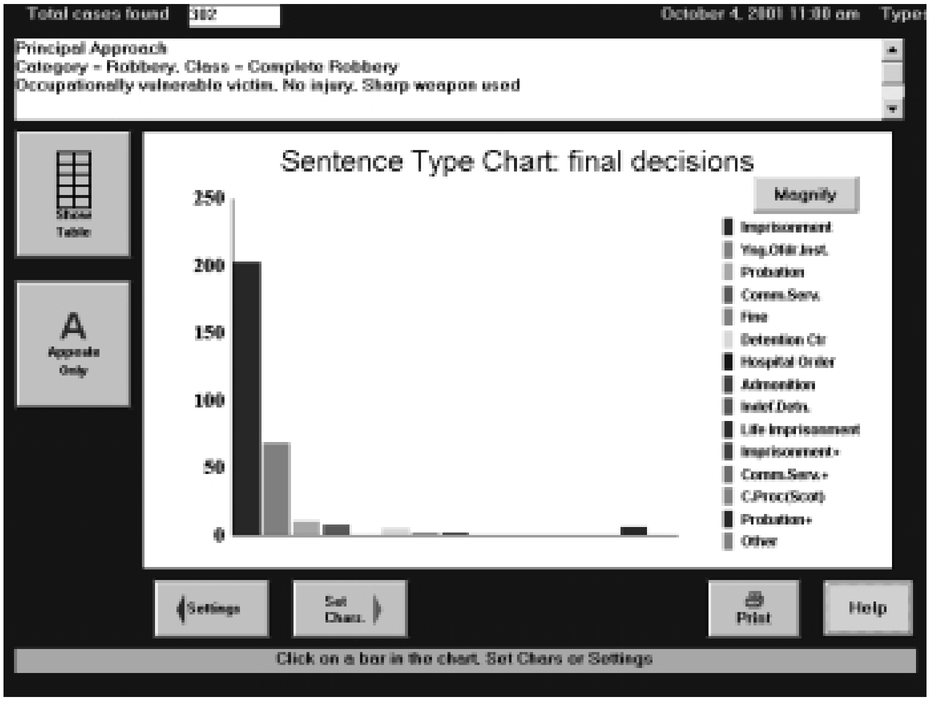
The implementation of the compression algorithm is based on the use of a string pool to store the lexemes (the uncompressed representation of attribute values). The tokens are represented by integers that give the indices of offsets of the lexemes in the string pool. This approach ensures that the tokens are of minimal length. Off-line data compression is carried out by hashing the tokens to discover the entry point in the offset table. If the hash value is pointing at a vacant entry in the table (after handling overflows) then the lexeme is inserted into the dictionary and the offset stored in the appropriate cell. The hash value is then stored in the data structure that represents the relation. On the retrieval side, tokens recovered from the database can be converted into lexemes by retrieving the offset and using that as an entry point into the string pool.

We used this strategy to compress the data collected for the information system and overall were able to reduce the volume of the data to 13% of the initial representation as stored in a conventional database system. This represents a useful benefit for the PDA version of the information system. The data can be stored in an optimally compacted form without prejudicing the performance of queries on the system. In addition, since the database file is now represented in minimal space, updated files can be transferred from the server to the mobile client with optimal efficiency.

	Conventional Storage S_p (Mb)	Compressed Storage S_c (Mb)	Compressed to Conventional $\frac{S_c}{S_p}$
relation1	1.9	0.23	12%
relation2	0.35	0.03	9%
relation3	0.34	0.05	15%
relation4	0.18	0.03	17%
relation5	0.05	0.02	40%
Cumulative	2.82	0.36	13%

Table 3. Comparison of conventional and compressed storage for sentencing data.

We used a direct manipulation interface to the system since our user group was initially unfamiliar with the concept of using computer support for sentencing decisions. Figure 6 shows the appearance of the laptop version of the interface. To arrive at this point, the user has first to set a number of characteristics of the offence. The chart presents the aggregations of cases in each category of sentence and these can be explored further by drilling down into the respective bars. The PDA interface uses dynamic queries to achieve the same result. The text entries at the foot of the screen (Figure 7) represent cascading menus that allow the user to make choices about the details of the cases to be displayed. Each time a choice is made, the distribution chart is updated to show the aggregation of cases that match the currently selected criteria. The bars in the chart can be drilled in a similar manner to the laptop based system. The direct manipulation interface was well received and judges found that they were rapidly able to construct meaningful queries and retrieve appropriate data despite having limited prior experience of computer use.



Total cases found 302

October 4, 2001 11:00 am Type

Principal Approach
Category = Robbery. Class = Complete Robbery
Occupationally vulnerable victim. No injury. Sharp weapon used

Click on a bar in the chart, Set Chars or Settings

Fig. 6. Laptop interface to the information system.

A preliminary evaluation of the laptop based system was carried out with two cohorts of users each consisting of four users. Users were assigned to cohorts in a random manner. Restrictions caused by administration of the project allowed monitoring to be carried out over twelve weeks for the first cohort and five weeks for the second cohort. The results for system usage are shown in Figure 8. There is a clear novelty aspect to the system usage pattern for the first cohort in the first week of the trial but thereafter, usage settled down at a much lower level. The second cohort did not show the same novelty peak but subsequent usage reflected a similar pattern to the first cohort.

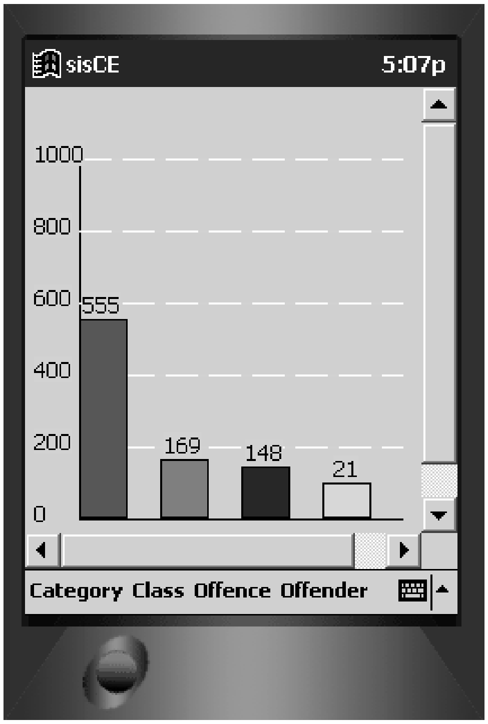


Fig. 7. PDA interface to the information system.

4 Conclusion

Judges claim, with justification, that each sentencing decision is made on the merits of the specific case. Within this constraint, it is recognised by all the participants in our project that sentencing decisions can profit from contextualisation within the patterns set by precedent. The challenge is how to reconcile the need to make each decision unique with the benefits of contextualisation. Our solution to this conflict is to build a system that supports the decision makers and is compatible with their mobile working environment. Data compression makes a significant contribution to the provision of this support by facilitating the use of non-intrusive systems. The small form factor of the PDA allows quick and easy access to sentencing patterns. Data compression presents an opportunity to reduce response times as well as the on-line time needed to update the system. We intend to use wireless technology to support updates. Data compression will accelerate wireless updates and reduce the power consumption of the process.

The laptop-based system we have developed has been in use by some of the judges in Scotland for three years and has proved robust and useful. We are currently extending the provision to all the judges in the jurisdiction and installing the support architecture. The PDA-based system we are developing

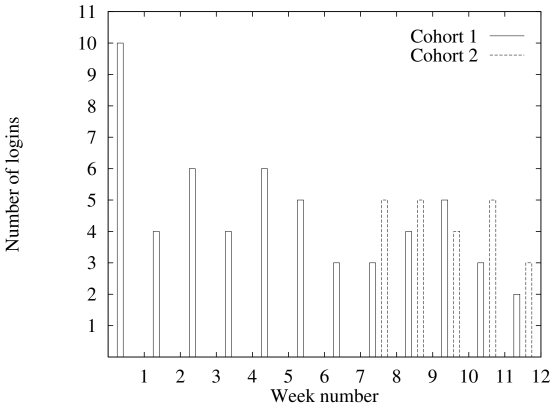


Fig. 8. System usage in the pilot deployment.

provides a step in the direction of support for dynamic mobility and opens up a number of technical issues. Current PDA technology is at a relatively early stage of development. Memory sizes and processor speeds need to be increased in order to make even medium-scale data intensive applications feasible on such platforms. The approach that we have developed of using data compression to optimise the storage of data on the PDA makes the best possible use of the memory and processing capability that is available in these small form factor devices.

The support for sentencing that we are providing is limited to statistical information on sentencing patterns. Our users are interested in increasing the textual content of the system both to enable the storage and retrieval of narrative descriptions of particular cases and also to enable them to search databases of case law. The provision of mobile access to this kind of Intranet data remains a challenge for database systems.

References

1. W.P. Cockshott, D.R. McGregor, and J.N. Wilson. High performance operations using a compressed database architecture. *Computer Journal*, 41(5):283–296, 1998.
2. B. Dahlbom and F. Ljungberg. Mobile informatics. *Scandinavian Journal of Information Systems*, 10(1):227–234, 1999.
3. H. Garcia-Molina and K. Salem. Main memory database systems: an overview. *IEEE Transactions on Knowledge and Data Engineering*, 4:6:509516, 1992.
4. T. Lehman, E. Shekita, and L. Cabrera. An evaluation of starburst's memory resident storage component. *IEEE Transactions Knowledge and Data Engineering*, 4:6, 1992.
5. R. Murbach and E. Nonn. Similarity in harder cases. *Proceedings of the 4th International Conference on Artificial Intelligence and Law Amsterdam, June,*, pages 236–244, 1993.

6. I. Potas, D. David Ash, M. Sagi, S. Cumines, and N. Marsic. Informing the discretion: The sentencing information system of the Judicial Commission of New South Wales. *International Journal of Law and Information Technology*, 6:99, 1998.
7. P. Pucheral, J. Thevenin, and P. Valduriez. Efficient main memory data management using dbgraph storage model. *Proceedings of the 16th VLDB Conference,Brisbane*, pages 683–695, 1990.