Context-Sensitive Mobile Database Summarisation

Darin Chan and John F. Roddick

School of Informatics and Engineering
Flinders University of South Australia
P.O.Box 2100, South Australia
{darin.chan, roddick}@infoeng.flinders.edu.au

Abstract

In mobile computing environments, as a result of the reduced capacity of local storage, it is commonly not feasible to replicate entire datasets on each mobile unit. In addition, reliable, secure and economical access to central servers is not always possible. Moreover, since mobile computers are designed to be portable, they are also physically small and thus often unable to hold or process the large amounts of data held in centralised databases. As many systems are only as useful as the data they can process, the support provided by database and system management middleware for applications in mobile environments is an important driver for the uptake of this technology by application providers and thus also for the wider use of the technology.

One of the approaches to maximize the available storage is through the use of database summarisation. To date, most strategies for reducing data volumes have used compression techniques that ignore the semantics of the data. Those that do not use data compression techniques adopt structural (i.e. data and use-independent) methods. In this paper, we outline the special constraints imposed on storing information in mobile databases and provide a flexible data summarisation policy. The method works by assigning a level of priority to each data item through the setting of a number of parameters. The paper discusses some policies for setting these parameters and some implementation strategies.

1 Introduction

Mobility and the remote access to information is quickly becoming a common requirement for working in many areas. This has been encouraged by a new generation of mobile, laptop and palmtop computers. These computers, paired with the developments in wireless networking technologies, provide users with the ability to access information (almost) anywhere and anytime.

However, database-dependent information systems are only as useful as the data they have available to process. Since mobile computers are designed to be portable, they are also physically small and thus, with the present technology at least, are often unable to hold the large amounts of data usually held in centralised databases. It should be noted that improvements in mobile storage capacity are being matched by user requirements for capacity and thus the ratio of mobile systems capacity to server database size is not expected to change significantly in the near future.

In addition, as many of the wireless networking technologies required for mobile computing are limited in bandwidth (either for technological or economic reasons) (Imielinski & Badrinath 1994, Lubinski 2000, Pitoura & Bhargava 1993), there arises a need to carefully manage the bandwidth utilised. We argue that through the effective summarisation of databases it may be possible to use locally stored data to decrease the bandwidth usage.

This paper is structured as follows. The remainder of this section gives a working example that will be used to illustrate the discussions. Section 2 discusses issues affecting databases in a mobile environment, while Section 3 examines architectural models for databases operating in a mobile environment. Section 4 introduces data summarisation and examines the properties required of a good summarisation technique. Sections 5 and 6 outline a two-stage policy that employs priorities and heuristic techniques, respectively, to provide the summarisation of data, and discusses its use within the context of the mobile database. Section 7 discusses a bitmap view representations to describe the summary database, and the methods required for query processing. Section 8 discusses implementation issues. Section 9 describes some open research issues and concludes the paper.

1.1 Working Example

In order to better illustrate this work, we describe an application focussing on the health/medical records management area, which will be used to create examples to explain our arguments within this paper.

Medical data, including electronic medical records, represents one of the most complex and diverse forms of database available. They may contain generic information on diseases, treatments, pharmaceuticals, and so on, through to particular episodes of treatment and consultations with particular medical practitioners recorded on the medical records of individual patients. The types of data they may hold can range from bit-length codes and numerical values, to large and complex image files. An example E-R schema for a typical health-care information system (taken from (Golfarelli, Maio & Rizzi 1998)) is shown in Figure 1.

As the current technology trends toward mobile systems that have the potential to be more flexible and portable, the use of mobile devices by medical practitioners and others can be expected to increase[1],

Copyright ©2003, Australian Computer Society, Inc. This paper appeared at Twenty-Sixth Australasian Computer Science Conference (ACSC2003), Adelaide, Australia. Conferences in Research and Practice in Information Technology, Vol. 16. Michael Oudshoorn, Ed. Reproduction for academic, not-for profit purposes permitted provided this text is included.

This research was partially supported by Australian Research Council SPIRT Grant C00107533, held in collaboration with the New HP (formerly Compaq Computers).

[1] Indeed, a number of preliminary projects are underway, some

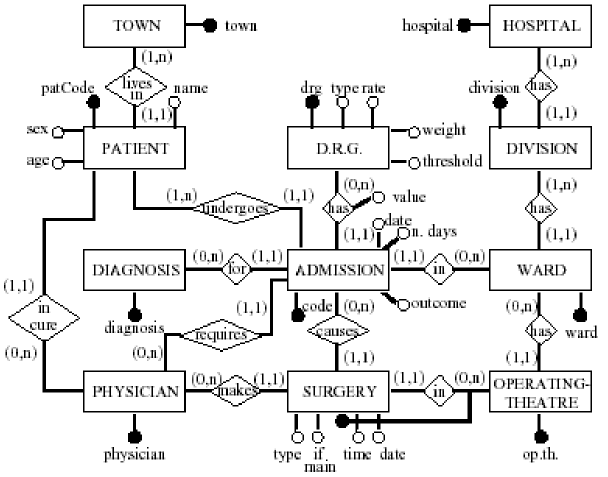


Figure 1: A medical E-R scheme from (Golfarelli et al. 1998)

as they provide a fast and reliable access point from which medical information may be extracted in a location independent manner. For example, a visiting consultant would be able to access up-to-date information to aid in diagnosis and the prescription of treatment, while other hospital staff may make use of a mobile device to check and update the condition of patients.

2 Mobile Database Issues

There are many issues concerning the effective use of mobile systems, both in respect of the current technology and those likely to become available in the near future. For mobile databases the most important of these issues are:

- the relative unreliability of connections (and the variability of bandwidth when connected),
- the limitations on storage capacity, and
- the security and privacy issues created when a computer is in a mobile environment.

This paper will focus only on the first two of these issues. Issues concerning security and privacy may cause significant problems in regards to databases in mobile environments and may include the identification and authentication of users and the theft of data during transmission (Heuer & Lubinski 1996, Zaslavsky & Tari 1998). However, these issues are outside the scope of this paper.

2.1 Connectivity and Disconnection

An important limiting characteristic of mobile computers is their finite battery capacity and the low communication bandwidth available and/or affordable (Imielinski & Badrinath 1994, Lubinski 2000, Pitoura & Bhargava 1993). These limits may lead to frequent disconnection due to both the need to reduce connection costs and because of technical limitations and failure (Heuer & Lubinski 1996, Lubinski 2000).

Advancements in battery technology have allowed longer intervals between recharge[2]. However, even with the longer life, there are higher energy costs involved while a mobile computer transmits or receives data (Pitoura & Bhargava 1993, Zaslavsky & Tari 1998), and this may rapidly decrease the life of the battery. Thus, intentional disconnection may be required through power management software to allow higher priority operations to continue.

In addition, in most situations there are likely to be many mobile users connected to their centralised or distributed database servers and in some locations, wireless coverage may be patchy. The available communication bandwidth in a mobile environment must therefore be shared between each user, thereby restricting the bandwidth available. As a result, networks may experience outages that may cause the mobile device to be subjected to frequent disconnection. Thus, in terms of mobile databases, access between central servers and the mobile system may be unreliable at times (Madria, Mohania & Roddick 1998). The disconnections that may occur can disrupt a query being performed and as a result, prevent or slow query response.

2.2 Storage Capacity

Another of the more important resource limitations of mobile devices is their limited information storage capacity (Lubinski 2000, Pitoura & Bhargava 1993). Large and multiple hard drive systems, such as those found in common desktop and server computers, cannot be accommodated into a mobile computer, given that they are required to be portable.

Improvements in hard drive technology have made modern hard drives smaller and with a greater capacity. Devices holding 30Gb are now commonly available for laptops and, while comparable to single drive desktop systems, commonly have a much smaller capacity compared to the centralised servers that support corporate databases. However, even with the reduced physical size of the modern hard drive, most are still as large as a PDA. Consequently, the storage capacities of even laptop-based hard drives are about 500 times more than that utilised by a flash disk emulator[3]. These flash disk emulators may provide attractive energy consumption and performance, but are still relatively expensive and have limited capacity (Douglis, Kaashoek, Li, Cáeres, Marsh & Tauber 1994).

Due to the storage limitation of mobile computers, it is thus difficult to create replicas of large databases on such devices, particularly on devices such as PDAs, which are the focus of this work.

3 Mobile Database Architecture

A typical model of a mobile computing environment is shown in Figure 2 (Barbará 1999, Dunham & Helal 1995, Madria et al. 1998, Pitoura & Bhargava 1993, Zukunft 1997). This model relies on a cellular architecture, such as that which is currently being employed by GSM network technology. It consists of mobile support stations (MSS) or base stations and mobile units (MU) or hosts. The MSS is a stationary component in the model and is responsible for a small geographic area called a cell. They are connected to each other through fixed networks. The MU is the mobile component of the model and may move from one cell to another. These MUs communicate with the MSS through wireless networks. Note that

of which have been reported in the literature (Dunham & Helal 1995, Pitoura & Bhargava 1993, Rakotonirainy 1999).

[2] In particular, PDAs (such as the iPaqs produced by HP) provide 600-1400 mAH Li-Ion batteries that may last for up to 10-14 hours of continuous usage, while the Lithium-Ion battery packs for laptops may provide continuous usage for 2-10 hours depending on the battery configurations of the laptop.

[3] Currently, a typical capacity for laptop is around 30Gb while flash disks may provide around 64Mb.

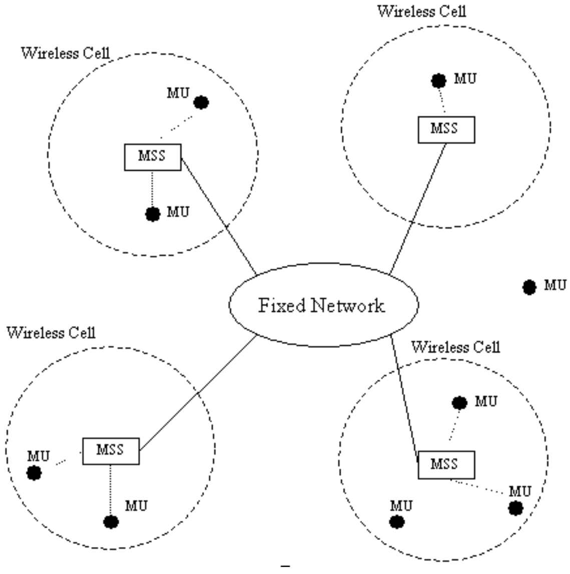


Figure 2: Architecture for a Mobile Environment

there are areas, even inside cells, where the reception may be poor or non-existent.

There are several modes of operation that an MU may experience and special protocols are be required to handle each mode (Pitoura & Bhargava 1993). The three modes of operation which are of interest to us are:

- **Full connection mode.** In fully connected mode, the MU is continually connected to the MSS and the only protocol required are those for hand-over, which are required when a MU moves from one cell to another or from one wireless form to another (for example, WLAN to G3). The hand-over protocol involves a new communication link between the MU and the new MSS, and the saving and transferring of states from the old to the new MSS (Madria et al. 1998). The communication hand-over to a new cell should be transparent to both users and applications not specifically involved with the hand-over process.

- **Disconnected mode.** In this mode, the model would be required to address the frequent disconnection that occurs to the MU as described in Section 2.1. One possible solution to deal with the disconnection would be the use of a proxy for the MU (Stanoi, Agrawal, Abbadi, Phatak & Badrinath 1999). This would ensure the query continues to run even when the MU is disconnected, and the MU may request an update from its proxy when it does reconnect. In addition, MUs may voluntarily move into this disconnected mode when idle or low on battery, to free up bandwidth resources and extend battery life (Pitoura & Bhargava 1993). While operating in disconnected mode, any applications that had used the communication link before the disconnection would be required to save their current communication state, and where possible continue with its other processes. Upon reconnection and depending on the saved communication states, applications may resume transmission or reception, or retransmit a request to begin the communication anew.

- **Partial or weak connection mode.** In this mode, the MU is connected to the rest of the network through low or intermittent bandwidth. Such a mode may occur when the MUs are in areas within or on the edge of a cell, where reception is poor. The partial-connection protocol would then be required to allow the MU to limit its communications to the network. Applications that use the communication link may then experience longer transmission time and may be required to extend timeouts to compensate for lengthy response time.

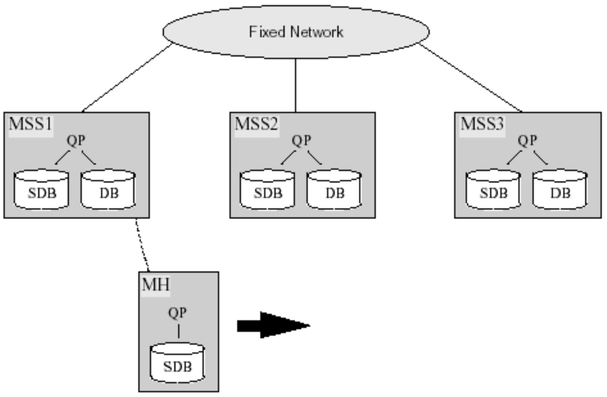


Figure 3: Architecture for a Mobile Database (Madria et al. 1998)

The latter two modes may also be deliberately invoked by the MU to allow conservation of its limited resources.

In addition to the architecture of the mobile computing environment, it is also necessary to consider the architecture of the database itself. The summary database is located within the MU and is derived from the main database, (see Figure 3). However, the way in which a query will be processed will not always be obvious. In previous work (Roddick 1997), there were three approaches identified.

- Firstly, to allow a query processor to determine the appropriate database to answer the query, whether it is the main or summary. This relies on in-built intelligence that can determine whether the local database can answer any arbitrary query.

- Secondly, to adopt a course grained approach where the query is sent to both databases and the first response to be received will be used. This option can be expensive but is guaranteed to produce an answer quickly.

- Lastly, to adopt a fine grained approach in which the query is segmented and those segments are run either in parallel or in series on both databases. This has the advantage of utilising the quicker local database more often and, on occasions, may obviate the need for parts of the query to be processed. The parallel option is the quicker may result in redundant communication. The serial option uses the central database when the local database is unable to process a given sub-query *and* the results of that sub-query are deemed necessary.

Within a data-intensive mobile environment, the three modes of operations discussed in Section 3, must be considered when deciding which approach to take. As there is a high probability that an MU will spend much of its time in disconnected mode, our approach is concerned with operating across all three modes.

4 Data Summarisation

Database summarisation involves the reduction of the size and information capacity of a database while maximising the useability of the resultant (summarised) dataset. That is, the measure of a summarisation technique can be seen as the relationship between the physical reduction in dataset size and the loss of *useful* information as a result. Clearly, this relationship is non-linear as some items in a dataset will be more critical than others and this will vary with context. In simple terms the effectiveness of a summarisation technique can be given as:

$$W = \frac{\zeta_{DB}}{\zeta_{SDB}} . \frac{\Upsilon_{SDB}}{\Upsilon_{DB}} \tag{1}$$

where ζ_{SDB} and ζ_{DB} are the *storage requirements* and Υ_{SDB} and Υ_{DB} are the *useful information capacities* of the summarised and original databases respectively (Roddick, Mohania & Madria 1999). Given that ζ_{DB} and Υ_{DB} are fixed for any given database and ζ_{SDB} has an upper limit on any given platform, the task is to maximise Υ_{SDB} for a target system. To date, many summarisation techniques involve the use of structural reduction techniques that reduce the volume of data without considering the use (i.e. the importance to the user) of the data.

In the following section, we discuss the characteristics of a good summarisation technique. This will then lead to the proposition of a two-stage data summarisation process. Figure 4 provides an overview of the summarisation process. Section 5 discusses the first stage which involves the use of priorities to determine the candidates for inclusion into the summarisation process. This stage would allow the identification of the relevance of data with respect to the users. A second stage is then discussed in Sections 6 and 7, which employs heuristic techniques to construct the summary database, which balances the calculated information requirements of the user with an optimum description length of the summarisation.

4.1 Characteristics of a Good Summarisation Technique

In developing a summarisation process, there are a number of characteristics that should be exhibited as follows:

- **Responsiveness**
  In order for a technique to be considered acceptable, it must provide an acceptable response to the user. However, the tolerance to waiting for a response may differ between different users and applications. The responsiveness of a summarisation technique can be an important characteristic if the process makes it difficult to scale the process or is time consuming to invoke.

- **Accuracy**
  Accuracy in the context of database describes the correctness and completeness of the database in response to a query. Significantly, depending on the type of user and the requirements that a user may have, it is possible to provide different levels of accuracy for queries. A good summarisation policy should be able to identify the needs of the user and provide at least the minimum level of accuracy. In some cases overcomplete (generalised) answers may suffice.

  For example, a medical practitioner doing research on a particular disease may not require the exact details of each patient who contracted a given disease, but merely the aggregated statistics and the associated epidemiological data. On the other hand, for a medical practitioner who is examining a patient, a detailed inspection of the medical record may be required.

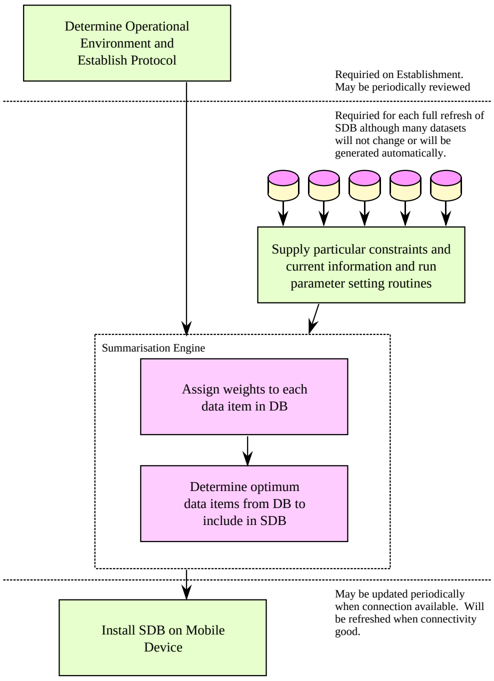

Determine Operational Environment and Establish Protocol
Requiried on Establishment. May be periodically reviewed
Requiried for each full refresh of SDB although many datasets will not change or will be generated automatically.
Supply particular constraints and current information and run parameter setting routines
Summarisation Engine
Assign weights to each data item in DB
Determine optimum data items from DB to include in SDB
Install SDB on Mobile Device
May be updated periodically when connection available. Will be refreshed when connectivity good.

Figure 4: Summarisation Process

- **Adaptability and Graceful Degradation**
  Adaptability is perhaps one of the more important but difficult aspects to define in a good summarisation technique. Regardless of the types of user or the environments that the summary database will be used in, it should be capable of adapting to changes in its usage and operating environment. This includes an ability for a technique to degrade sensibly as storage becomes more limited.

Criteria		ρ
Enumerated	e	100
Contextual	c	75
Previous Usage	u	65
Push-based	p	100
Model-based	m	50
Inductive	i	45
Time-based	t	20
Spatially-based	s	20

Table 1: Example of a protocol – the setting of ρ for each data inclusion criterion.

5 A Protocol for Priority-based Data Summarisation

Our priority-based data summarisation uses the notion of *priorities* (or *weights*) to select the appropriate data for the expected situation. To ensure an optimal accommodation of the most needed information, this needs to be done in a fine-grained manner. In our model these are calculated at the level of the data item. The priorities are numerical values either supplied or generated through observation and experimentation and are assigned through a multifaceted evaluation of different criteria. The relative use of each criterion is termed a *protocol*, (see for example, Table 1). The criteria that might contribute to the protocol are grouped into three categories, Primary, Secondary and Tertiary Criteria.

- Primary Criteria
  - Enumeration, e
  - Contextual, c
  - Previous Usage u
  - Push-based, p
- Secondary Criteria
  - Model (ie schema) based, m
  - Induction, i
- Tertiary Criteria
  - Time-based inference, t
  - Spatially-based inference, s

In our work, the protocol and the priorities are combined through a single formula which gives a prioritisation $\mathcal{P}_d$ for each data item as shown in (3) and as shown in Figure 4. For each criterion, the calculation of a relative priority, ρ_x (where x represents a criterion), and an assigned priority, ϕ_x, are required. The first to determine which criteria are more important and the second to provide a priority as calculated by the criterion. Space precludes a full discussion of the possible algorithms for determining ρ_x and ϕ_x, however, we include in the discussions below some of the aspects that may need to be accommodated.

5.1 Primary Criteria

This category involves criteria that extract data from the main database to the summary database through an external mean. The following sections discuss the criterions that fall under this category.

5.1.1 Enumeration

Enumerated data are references to those items specifically indicated as useful by an agent external to the summarisation process and indicates information that is directly recommended to be represented in the summary database. It allows direct external input into the creation of the summary database and strongly encourages the summarisation process to include data which is of significance to users. Although potentially any specification can be made, the enumeration of data is commonly a horizontal specification of tuples within one or more relations.

> **Example:** A medical practitioner has a list of patients to be visited, and will thus enumerate a list which will be used to specify the patient data required. That is, specific tuples of a relation corresponding to the patients' details are given priorities that will encourage their inclusion in the summarisation database. Following the example given in Figure 1, these tuples may be from the relation corresponding to the entity PATIENT.

Since enumerated information are explicitly specified, perhaps in terms of a parameterised relational query, ρ_e is likely to be assigned, for most applications, a higher value in the protocol than data specified through the other criteria. Moreover, ϕ_e is likely to take either 0 or 1 depending on whether the data was enumerated or not (although a more involved method could be envisaged).

5.1.2 Contextual

Knowledge of the context of use can be useful in inferring the data that may be needed by users. Thus the *contextual* criterion can be used to include data that may be useful because they are related to the user's details or environment. As such this criterion differs from the enumerated criterion which is data-centric rather than user-centric. The contextual priorities are deduced through rules which relate aspects of a user, such as the user's medical specialisation, through to data that may be useful in that user's operation of the database.

> **Example:** A medical specialist may, by virtue of their specialisation, have a greater interest in particular aspects of a patient's case history. In addition, records of patients who have been seen recently by the medical practitioner and who are still under active treatment by others, might also be included into the summary database to ensure that the information is available if consultation is required.

In order to determine contextual priority, ϕ_c, there must be a set of rules concerning the user (specialisation, practice, projects, etc.). The assignment of ϕ_c due to the different contextual criteria may then vary as a result of the importance placed on the data. For instance, data relating to the specialisation of the practitioner may have higher importance than the inclusion of records relating to recently seen patients.

5.1.3 Previous Usage

This criterion allows the user's previous activity to act as a guide as to future activity. This is done through an inspection of the previous queries invoked and assigns a priority based on the frequency of access of either the data item explicitly or the likely access through some heuristic. For example, association rule mining might be used to associate the use of one attribute with another or between the characteristics of one query and the next.

> **Example:** A particular user might have a preference for referring to similar cases before deciding treatment. Thus, for that user, the presence of a patient record with a particular set of conditions may also cause the inclusion of other similar cases.

The manner in which the values for ϕ_u can be calculated is wide. At this stage we are experimenting with the use of association rule mining as described above.

5.1.4 Push-based Approaches

The push-based criterion is somewhat different. It indicates that the data has been determined important by the server and should be communicated to the clients. In this respect it can be used as a mechanism for partially updating a summary database without full synchronisation. Such data may include updates that the server has received (such as recent pathology laboratory results). There has been much research done on data dissemination and many were surveyed by Barbará *et al.* (Barbará 1999). It can also be used to manually override decisions made by the summarisation engine.

5.2 Secondary Criteria

This category involves the criteria using data that has been extracted by the Primary Criteria in order to determine any new data for the summary database. The following sections discuss the criterions that fall under this category.

5.2.1 Model or Schema-based

The model or schema-based criterion is based on the relationships implied through the data model. That is, data flagged for inclusion through this criterion is related to other data through the structure described in the database schema. As mentioned earlier, their inclusion into the summary database is dependent on data extracted by any Primary Criteria.

The model-based priority, ρ_m, determines its importance relative to the other criteria (if no model is available then this can be set to zero). It is important to note that as the modelled relationship to the data generated by the Primary Criteria becomes more distant, the importance of the data is assumed to decrease. Thus, the weight of ϕ_m would decrease as the modelled distance increases. An example formula for ϕ_m may be given by:

$$\phi_m = k^{-a} \tag{2}$$

where k is a constant and a is the length of the shortest path in the model of the data from Primary Criteria data that it is associated to. Additionally, ϕ_m is always zero when none of the Primary Criteria are present. That is, $\phi_m = 0$ when phi_e, phi_c, phi_u and phi_p are all equal to zero.

> **Example:** Following the example in Section 5.1.1, and the associated Entity-Relationship diagram for the example in Figure 1, we can see that there are two important entities - PATIENT and ADMISSION that provide information on the patient's details and the details of when a patient is admitted into the hospital, respectively. By observation, tuples from ADMISSION are related to the data in PATIENT through the relationship undergoes. These are annotated as useful through the model-based criterion and are given weights that indicate certain usefulness.
>
> Additionally, the data that resides in the entities WARD, DIVISION and HOSPITAL, will be given incrementally lower weightings depending on the number of relations they are from the entity PATIENT.

5.2.2 Induction

This criterion allows the use of inductive rules to specify the inclusion of data. By inspection of the data contained within the main database, it may be possible to derive rules to indicate data that are associated with each other. Data mining techniques, such as association rule mining (ARM) as typified in the Apriori algorithm (Agrawal, Imielinski & Swami 1993) and its many successors, may be useful for the derivation of inductive rules. ARM involves the identification of relationships between items that occur frequently together and may then be used to imply the storage of other items within the database.

> **Example:** Within the medical records of a patient, there may be an association between a disease and the pathology tests that would be used to diagnose and monitor this disease. Depending on the strength of this association, this then would imply that the appropriate pathology results of the patient suspected with this disease would be flagged for storage in the summary database.

Similar to the model-based criterion, data may relate to any Primary Criteria and are thus dependent on those data. Again, the condition, ($\phi_i = 0$) applies when no Primary Criteria are present. Note that the importance of each rule used may be specified, which could be used to determine the correct values for ϕ_i.

5.3 Tertiary Criteria

This category of criteria involves only data that were specified by the above mentioned criteria. Each criterion is restricted to only increase the priority of those data through their individual calculations.

5.3.1 Time-based Inference

This criterion makes the assumption that the recency of events make information more likely to be of interest. In modelling time-based data there is a number of characteristics that must be considered (Roddick & Patrick 1992, Snodgrass 1987). In particular, two temporal dimensions are identified as important when an event occurs, namely *transaction* and *valid* time. The former refers to the time that the event is recorded into the database allows a user to *rollback* the database to an earlier view. In the context of this paper, transaction time is less relevant since we are attempting to create a summary database on the basis that the data stored is important to the user, and therefore when there are corrections within a database it is more likely that only the latest copy of the event would be of interest. The latter, valid time, refers to the time that events has occurred in reality, and facilitates the post or predating of changes to the database. The *valid time* of an event is likely to be of interest. Thus, the time-based criterion focuses an event's valid time reference.

> **Example:** A patient had an episode relating to a fractured arm some time ago. More recently, the patient made an appointment for a consultation with the medical practitioner for a cough. Apart from the relative association between a fracture and a cough (dealt with through the inductive priority (ie. the value of ρ_i)), the time since the first event will mean that it is less likely to be of relevance and would be given a lower priority. On the other hand, a sore back that the patient mentioned during a more recent consultation would be given a higher priority (and thus may be included in the summary database) as it may be mentioned again during this consultation.

ϕ_t indicates the relative importance of the data item due to its temporal information. ρ_t is a priority specified as part of the protocol and indicates the importance of the time-based criterion. Since time is continuous, ϕ_t may be calculated by either one of the following functions:

- **Stepwise**. This assumes that data has the same level of interest (and priority) until changed.
- **Linear or Continuous Change Functions**. These functions assume some degradation of interest over time.

5.3.2 Spatially-based Inference

In the same way that the time-based criterion assumes that recent events are more important that those in the past, spatially-based inference assumes that physically or geographical closer events are of more interest that more spatially distant events. Thus, by using any predetermined knowledge of the location, it is possible to assert a higher priority for data that corresponds to that location as there is an assumed higher probability that data relating to that area will be accessed.

> **Example:** A travelling medical practitioner is visiting patients in a remote location. In addition to the details relating to the (enumerated) patients, other information such as other remote patient details may also be stored.

5.4 Priority Formula

As a policy decision, any criterion that is not used is set to zero. That is, the relative priority of that criterion ρ_x is equal to zero. An experimental formula, (3), combines the priorities calculated by each criterion into a single formula $\mathcal{P}_d$.

$$\mathcal{P}_d = \left[\sum^{x} \rho_x . \phi_x \right] / ln(len_d + 1) \qquad (3)$$

where

x is one of the criteria in Table 1,

ρ_x is the relative priority of criterion x in the protocol,

ϕ_x is the assigned priority for that data item as calculated for that criterion, and

len_d is the size of the data item (in bits). This value is necessary to discourage the storage of overly large data items.

$\mathcal{P}_d$ provides an indication as to the importance of the data item and is thus proposed as the basis for the first stage of database summarisation. Initial analysis of the formula through a spreadsheet, using the settings in Table 1, shows that the results calculated with a range of different criteria is enough to allow different data items to be included or excluded.

6 Heuristic Summarisation Techniques

Having determined the priorities of each of the data items, the task is to construct a summary database such that the organisation of the summary is not overly complicated. It must be possible to calculate rapidly whether a query to the summarised database is able to return the same response as the same query to the original. That is, whether, for a given query Q, is

$$Q(DB) \equiv Q(SDB) \qquad (4)$$

where $SDB = \Psi(DB)$ and Ψ is the summarisation function.

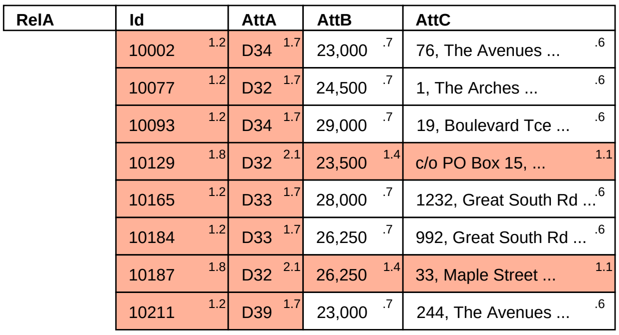
If it could be arranged that Ψ is able to be stored, for example, as (the equivalent of) a relational algebraic expression (ie. a view), then query rewriting techniques (together with a few extensions) could be used to evaluate the equivalence. For example, given the priorities shown shown in Figure 5, the optimum summarisation might be

$$\sigma_{Id \in \{10127,10187\}}(RelA) \oplus \pi_{\{ID,AttA\}}(RelA) \qquad (5)$$

where $\oplus$ is a combination function.

Before considering whether queries to the summary and main databases may be equivalent, it is necessary to understand the responses that may be returned. In (Madria et al. 1998), four types of query answers were identified as follows:

- Complete[4] and sound[5],
- Potentially understated (sound but may be incomplete),
- Potentially overstated (complete but may not be sound),



RelA	Id	AttA	AttB	AttC
	10002$^{1.2}$	D34$^{1.7}$	23,000$^{.7}$	76, The Avenues ...$^{.6}$
	10077$^{1.2}$	D32$^{1.7}$	24,500$^{.7}$	1, The Arches ...$^{.6}$
	10093$^{1.2}$	D34$^{1.7}$	29,000$^{.7}$	19, Boulevard Tce ...$^{.6}$
	10129$^{1.8}$	D32$^{2.1}$	23,500$^{1.4}$	c/o PO Box 15, ...$^{1.1}$
	10165$^{1.2}$	D33$^{1.7}$	28,000$^{.7}$	1232, Great South Rd ...$^{.6}$
	10184$^{1.2}$	D33$^{1.7}$	26,250$^{.7}$	992, Great South Rd ...$^{.6}$
	10187$^{1.8}$	D32$^{2.1}$	26,250$^{1.4}$	33, Maple Street ...$^{1.1}$
	10211$^{1.2}$	D39$^{1.7}$	23,000$^{.7}$	244, The Avenues ...$^{.6}$

Figure 5: Example relation with summarisation priorities

- Wrong.

The difference between 1 and 3 lies in the relaxation of the closed world assumption (Reiter 1978) that dictates that anything not recorded within the database may be assumed false, and thus potentially overstated responses are correct if there is no evidence to prove otherwise. This can be useful as part of a fine-grained query decomposition process (Roddick 1997).

With traditional databases, the objective is to provide query answers that are logically complete and sound. However, summary databases are by their nature associated with a loss in data and are thus incomplete but sound. Consequently, it is not always possible to provide both sound and complete answers. However, for these cases, it is sometimes possible to provide *knowingly overstated* responses (for example, through generalised responses), which are correct as long as there are no data to prove it otherwise. Thus, we can redefine our evaluation for equivalence to include not only responses that are the same as those given by the original database, but also, *where allowable*, responses that may be potentially overstated.

Currently, there are many techniques to reduce the volume of data in a database including those available to summarise a database, transform a query to receive less results and to reduce the size of a result set (Lubinski 2000). We discuss below some of the most common heuristic techniques, particularly those that relate to our goal of summarising data for mobile databases. We break these down into those that retain the basic schema but manipulate the data through a change to the domain, and those that modify the schema.

6.1 Domain Modification

6.1.1 Abstraction through Aggregation

Abstraction through Aggregation is the summarisation of selected attributes through the use of meta data by creating new data items such as the sums and averages of existing data (Heuer & Lubinski 1998, Lubinski 2000). In (Barbará, DuMouchel, Faloutsos, Haas, Hellerstein, Ioannidis, Jagadish, Johnson, Ng, Poosala, Ross & Sevcik 1997), there is an examination of some techniques for providing rapid but approximate answers that may be used for data warehouse applications.

This is an important heuristic technique since it allows fast answers to queries while decreasing the amount of data required for storage. The contribution that abstraction makes is that it generalises the data into a new collection of objects (see Table 2). Such a reduction results in information loss in that an approximate form is used to replace the exact details of the data.

[4] In that a query on the summary database will return at least the data that the same query would return on the main database.

[5] In that a query on the summary database will return no additional data that the same query would return on the main database.

6.1.2 Concept Hierarchies

Concept hierarchies organise data and concepts in a hierarchical form. The process provides a mapping or generalisation of the lower layer concepts to their corresponding higher level concepts (Han & Fu 1994). The central idea is that of concept ascension in which low level data items are elevated onto a higher level within the hierarchy and duplicates tuples are then removed. This hierarchy may be supplied externally or, as in the work done by Han and Fu (1994), the hierarchy may be dynamically generated or refined. In (Madria et al. 1998), a model is presented using concept hierarchies to facilitate data volume reduction for a summary database. This was proposed as a possible model that may be used by databases within mobile environment. However, current techniques for concept hierarchies are insensitive to the use of the data and do not take into account the priorities that had been set up by first stage of the database summarisation.

6.1.3 Layered Data Reduction

In (Lubinski 2000), Lubinski introduces the gradual data reduction technique as a way of reducing the data of the query results before it reaches the MU. This technique provides different domains of data precision and a layered reduction for those domains. The domains of data precision relate to the interestingness that the user has placed on the resultant data through domain boundaries. For example, *Domain 1* might contain data that is most relevant to the user within a specified domain boundary, and corresponds to *Layer 1*, where no reductions are performed. Furthermore, *Domain n*, where *n* is the final domain, consists of data that is of marginal interest. These data corresponds to *Layer n* that utilises the most extreme data reduction.

6.1.4 Surrogates

In many instances, complicated and large data types may lead to extra resources being required to access, transfer and present them (Lubinski 2000). Surrogates are the substitution of those complex data types with corresponding simpler data types. A popular example for the replacement technique is to substitute a large image file with its equivalent but simpler text description.

The surrogate technique is more effective when used with more complicated data types that have an equivalent but simpler substitute. Such a technique would modify and replace the data type of the attributes of the base relations (see Table 2).

6.2 Schema Modification

6.2.1 Projection

This method of reduction is one of the two most basic database reduction techniques. It involves the vertical reduction of the attributes in a relation within the database (Lubinski 2000, Roddick et al. 1999), and is one of the more commonly used techniques for reducing the data of a database. In addition, the method of projection may also be utilised in conjunction with other types of summarisation techniques, such as concept hierarchies.

Projection provides a modification to the structure of the database schema. That is, the technique is used to remove certain attributes within the base relations (see Table 2).

6.2.2 Selection

The other basic and commonly used database reduction technique which involves the horizontal reduction of the tuples within a database (Lubinski 2000, Roddick et al. 1999). Again, selection may also be found in conjunction with other summarisation techniques.

6.3 Data Compression

There are many methods of data compression, which are based mainly on character encoding and repetitive string matching (Graefe & Shapiro 1991). More details of the available compression techniques may be found in surveys such as (Bell, Witten & Cleary 1989, Lelewer & Hirschberg 1987, Severance 1983). Many techniques surveyed in those papers include compression methods that deal with only text data types rather than a diversity of data types. One particular method that does consider the different data types is the RAY algorithm as described by Cannane, Williams and Zobel (Cannane, Williams & Zobel 1999). However, data compression is a large area of research and it is orthogonal to the work outlined here.

Given the methods above, views may then be constructed to provide a mechanism for stage two of our process. Views may define a function from a subset of base tables to a derived table, and they may be *materialised* by physically storing the tuples of the view (Lauzac & Chrysanthis 1998*b*). The use of views in terms of mobile systems has largely been discussed within data warehouse applications to allow maintenance and updates of the MU's database (Lauzac & Chrysanthis 1998*a*, Lauzac & Chrysanthis 1998*b*, Stanoi et al. 1999). In (Lauzac & Chrysanthis 1998*a*, Lauzac & Chrysanthis 1998*b*), they introduce a *materialised view holder*. During network disconnection, the view holder acts as a proxy to provide the required updates and views to the MU upon its reconnection. Note that since views are predetermined queries, an equivalent query employing rewriting techniques may also be used to allow query optimisation (Halevy 2001).

7 View Representations

As well as the views which are constructed to provide a mechanism for Stage 2 of the summarisation process, it is also necessary to provide descriptive representation of the structure of the resultant summary database to allow the query processor to determine if a given query can be answered. There are many methods capable of representing a view. Our adopted format is through the use of bitmaps, which will be discussed in the following section.

7.1 Bitmaps

As part of Stage 1 of the process, each data item is assigned a particular priority which is used to determine the items selected for inclusion. In addition, as the summary database is built we develop a second parallel set of matrices that hold a boolean flag corresponding to each of the data items within the database. This second set of matrices, together with the schema itself, are kept after summarisation to enable query answering. The main advantages of using bitmaps to store the description are:

- a fixed, relatively short, description length. Since a bit is used to represent each value within the database, it is easy to determine the length of the description,

Heuristic Technique	Domain Modifications		Schema Modifications		
	Data Generalisation	Modification	Insertion	Deletion	Modification
Abstraction	•				
Concept Hierarchies	•	•			
Layered Reductions	•	•			
Surrogate		•			•
Projection				•	
Selection				•	

Table 2: Summary of Base Data Modifications

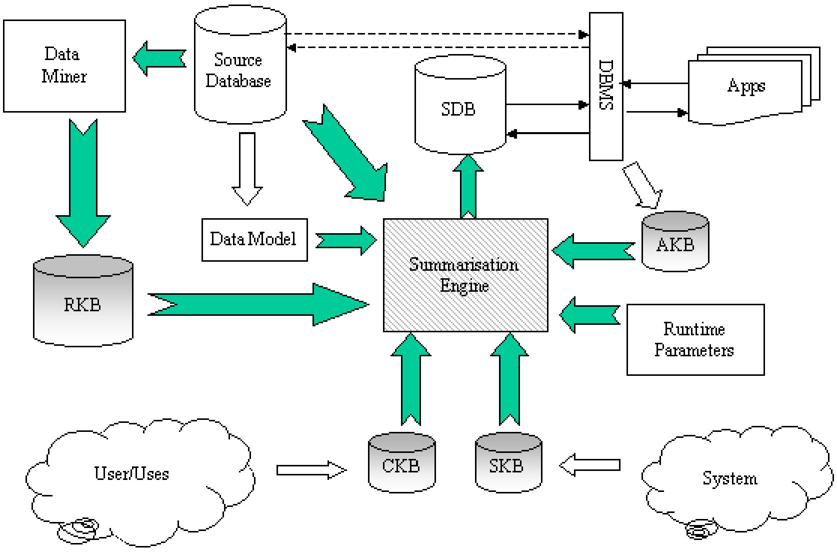


Figure 6: Context of the Summarisation Engine

- for many cases (although not always) a lower storage requirement than the equivalent algebraic expression,
- a fast way to determine if queries may be answered. Simple logical binary operators may be used to quickly determine if a query may be answered.

The main disadvantage of using a bitmap representation is that the technique may not provide the most optimum description of the summary database. That is, since bitmaps are of fixed length, it may not be possible to reduce the size of the description without using compression techniques. If all keys within a database are stored into the summary, a larger storage requirement will be required.

There are thus two cases that must be considered when creating a bitmap representation. The first is to store all the keys found in the main database. For large databases, this case may be expensive since they are likely to contain many keys. Additionally, new keys added to the main database must also be reflected on the summary database and thus frequent synchronisation may be required. Since all keys in the summary database reflect the main database, it is possible to provide a negative answer when queries refer to keys not present in the summary database. The second approach is to store only selected keys into the summary database. This method would resolve the large database problem and does not require frequent synchronisation. However, reference to key values not held requires reference to the centralised database.

7.2 Query Answering using Bitmaps

As bitmaps provide a method to describe the contents of the summary database, a transformation of a query to a corresponding bitmap is required for determining whether a query is answerable by the summary database.

Consider the bitmap description for the relation, Patient, in Figure 1 (see Table 3). In the figure the Patient Identifier, patCode, is included in full while other attributes are listed as boolean values indicating that the value is held in the SDB.

The easiest way to determine if a query may be answered would be to perform a bitwise AND NOT operation to the description. That is, if $Q \wedge \neg S = 0$ where Q and S are the bitmap representations of the query and the summary database respectively, then the query is answerable. In addition, a non-zero answer also indicates the particular items that are causing the inability to answer. Consider the following query, $Q1$ (Table 3),

```
SELECT  Name   FROM Patient
        WHERE  SEX = 'F'
        AND    patCode < 1003
```

As can be seen, $R1 \neq 0$ and thus the query is unanswerable by the summarised *Patient* relation. Moreover the missing values are shown and these may then be used as a request to the server to return the required information.

By storing all the keys present in the main database within the summary database, range queries (such as the one above) can be answered. However, if only selected keys are stored (saving space), range queries would be unreliable and thus only queries that enumerate the key values would be answerable.

8 Summary Database Construction

As mentioned earlier, our goal is to construct a summary database reflecting the priorities generated in Stage 1 such that the description length of the summary database is a small fraction of the total space available. There is thus a trade-off between including

Patient	*patCode*	**name**	**sex**	**age**	**town**	**physician**
	1000	0	1	0	1	0
	1001	1	0	0	0	1
	1002	1	0	0	1	1
	1003	0	0	0	0	1
	1004	0	1	1	0	1
	1005	1	1	0	0	0
	1006	1	1	1	1	1
	1007	1	1	0	0	1

Q1	*patCode*	**name**	**sex**	**age**	**town**	**physician**
	1000	1	1	0	0	0
	1001	1	1	0	0	0
	1002	1	1	0	0	0
	1003	0	0	0	0	0
	1004	0	0	0	0	0
	1005	0	0	0	0	0
	1006	0	0	0	0	0
	1007	0	0	0	0	0

R1	*patCode*	**name**	**sex**	**age**	**town**	**physician**
	1000	1	0	0	0	0
	1001	0	1	0	0	0
	1002	0	1	0	0	0
	1003	0	0	0	0	0
	1004	0	0	0	0	0
	1005	0	0	0	0	0
	1006	0	0	0	0	0
	1007	0	0	0	0	0

Table 3: Bitmaps for *Patient* Relation, Query, $Q1$ and Result $R1 = (Q1 \wedge \neg Patient)$

low priority data items and the costs of specifying the view structure.

At this stage we have taken advantage of the offline nature of summary database construction and have implemented a relatively simple policy using bitmaps as discussed in the previous section[6]. Figure 6 provides an overview of the *Summarisation Engine*. The two databases, *Source Database* and *SDB*, represent the main (source) database and the resulting summary database, located within the mobile unit. Through a DBMS, application programs will have access to the summary database and possibly (perhaps intermittent) access to the main database.

The *Summarisation Engine* has inputs from seven sources:

- **The Source Database**
- **Knowledge Bases**
  - **Context Knowledge Base (CKB)**
    This consists of user-supplied information and includes a list of enumerated data, and information concerning the users which can be used for contextual values. Either or both of these are optional.
  - **System Knowledge Base (SKB)**
    This is generated by the system and includes system information.
  - **Rule Knowledge Base (RKB)**
    This contains the rules generated by any external data mining engine.
  - **Application Knowledge Base (AKB)**
    This will provide a feedback mechanism when deciding which data is more important for storage. The AKB will be generated by DBMS to track data requested by applications. The information may then be used to include or delete data during future recreation of the summary database. For example, data that were not used or used the least may indicate that it is not very important and may be excluded in the next recreation.
- **Data Model**
  The schema/model of the database which, to date, we have assumed to be in a simplified EER format. However, by changing the implementation of the model-based criterion, it is possible to allow the use of other types of data models. The rest of the criteria are compatible with other data models since they do not deal with the data model but with individual data items.
- **Runtime Parameters**
  These include specification of the protocol as described in Table 1 and other information such as target location and summary database size.

The development of a *Summarisation Engine* prototype is nearing completion. Its modular, plug-compatible approach means that modules representing, for example, the determination of ϕ for each of the criteria described earlier can either be developed specifically for the application or a generic module adopted. The prototype also provides interfaces for accessing the main and summary databases, and to extract the schema definition of the main database.

9 Conclusion and Future Work

The area of context-sensitive is large and while we have made considerable progress, this work inevitably represents a work in progress. Current work includes the verification of the most appropriate algorithms for determining the values of ϕ_x, some of the details of which have not been discussed in detail here due to space constraints. Moreover, it is our contention that the values for ρ_x (the protocol values) will vary according to application context (indeed, this is the reason for the protocol) but this has not yet been verified in practice.

Another issue which has been discussed with us (but which, for the moment, is outside of the scope of this research) is the utility of this approach in determining legacy data within a centralised database system. That is, is it possible to use this approach to identify the least used parts of a main database by applying these techniques reflectively? This might provide a mechanism for fragmenting large databases over varying I/O-speed devices and thus improving centralised performance as well.

In this paper, we proposed a two-stage data summarisation policy that allows flexible information summarisation for mobile databases. The first stage involves priorities to determine the optimum data to include into the mobile database; the second stage applies heuristic techniques to effect the creation of the mobile database with regard to the data / description tradeoff. An architectural model of our summarisation engine was also provided.

We believe that issues, such as frequent disconnection and limited bandwidth that affect database usage in a mobile environment can be better accommodated as a result.

References

Agrawal, R., Imielinski, T. & Swami, A. (1993), Mining association rules between sets of items in large databases, *in* 'ACM SIGMOD International Conference on Management of Data', Vol. 22, ACM Press, Washington DC, USA, pp. 207–216.

[6] An alternative method of analysing the average and standard deviations of the priorities for each column and row has also been considered and can, on occasion, provide a shorter description length. In this method, if the standard deviation is low and the average priority is high then complete rows and columns are included. If the standard deviation is high then individual items may be included if their individual priorities are high.

Barbará, D. (1999), 'Mobile computing and databases - a survey', *Knowledge and Data Engineering* **11**(1), 108–117.

Barbará, D., DuMouchel, W., Faloutsos, C., Haas, P. J., Hellerstein, J. M., Ioannidis, Y. E., Jagadish, H. V., Johnson, T., Ng, R. T., Poosala, V., Ross, K. A. & Sevcik, K. C. (1997), 'The New Jersey data reduction report', *IEEE Data Engineering Bulletin: Special Issue on Data Reduction Techniques* **20**(4), 3–45.

Bell, T., Witten, I. H. & Cleary, J. G. (1989), 'Modeling for text compression', *ACM Computing Surveys (CSUR)* **21**(4), 557–591.

Cannane, A., Williams, H. E. & Zobel, J. (1999), A general-purpose compression scheme for databases, *in* 'Data Compression Conference', p. 519.

Douglis, F., Kaashoek, M. F., Li, K., Cáeres, R., Marsh, B. & Tauber, J. (1994), Storage alternatives for mobile computers, *in* 'First Symposium on Operating Systems Design and Implementation', Monterey, Californie, US, pp. 25–37.

Dunham, M. H. & Helal, A. (1995), 'Mobile computing and databases: Anything new?', *SIGMOD Record* **24**(4), 5–9.

Golfarelli, M., Maio, D. & Rizzi, S. (1998), Conceptual design of data warehouses from E/R schemes, *in* 'Proc. of the Hawaii International Conference On System Sciences', Kona, Hawaii.

Graefe, G. & Shapiro, L. D. (1991), Data compression and database performance, *in* 'Proc. ACM/IEEE-CS Symp. on Applied Computing', Kansas City, MO.

Halevy, A. Y. (2001), 'Answering queries using views: A survey', *The VLDB Journal* **10**(4), 270–294.

Han, J. & Fu, Y. (1994), Dynamic generation and refinement of concept hierarchies for knowledge discovery in databases, *in* 'KDD Workshop', pp. 157–168.

Heuer, A. & Lubinski, A. (1996), Database access in mobile environments, *in* 'Database and Expert Systems Applications', pp. 544–553.

Heuer, A. & Lubinski, A. (1998), Data reduction - an adaptation technique for mobile environments, *in* 'In Interactive Applications of mobile Computing (IMC'98)'.

Imielinski, T. & Badrinath, B. R. (1994), 'Mobile wireless computing: Challenges in data management', *Communications of the ACM* **37**(10), 18–28.

Lauzac, S. W. & Chrysanthis, P. K. (1998*a*), Programming views for mobile database clients, *in* 'DEXA Workshop', pp. 408–413.

Lauzac, S. W. & Chrysanthis, P. K. (1998*b*), Utilizing versions of views within a mobile environment, *in* 'the 9th ICCI'.

Lelewer, D. A. & Hirschberg, D. S. (1987), 'Data compression', *ACM Computing Surveys* **19**(3), 261–296.

Lubinski, A. (2000), Small database answers for small mobile resources, *in* 'International Conference on Intelligent Interactive Assistance and Mobile Multimedia Computing (IMC2000), Rostock, November, 9-10'.

Madria, S. K., Mohania, M. K. & Roddick, J. F. (1998), A query processing model for mobile computing using concept hierarchies and summary databases, *in* 'Foundations of Database Organisation', pp. 146–157.

Pitoura, E. & Bhargava, B. (1993), Dealing with mobility: Issues and research challenges, Technical report, Department of Computer Science, Purdue University, USA.

Rakotonirainy, A. (1999), Trends and future of mobile computing, *in* 'DEXA Workshop', pp. 136–140.

Reiter, R. (1978), On closed world databases, *in* H. Gallaire & J. Minker, eds, 'Logic and Databases', Plenum Press, New York, pp. 55–76. Reprinted in Artificial Intelligence and Databases. J. Mylopoulos and M.L. Brodie (eds.), Morgan Kaufmann, 248–258.

Roddick, J. F. (1997), The use of overcomplete logics in summary data management, *in* '8th Australasian Conference on Information Systems', Adelaide, Australia.

Roddick, J. F., Mohania, M. K. & Madria, S. K. (1999), Methods and interpretation of database summarisation, *in* 'Database and Expert Systems Applications', pp. 604–615.

Roddick, J. F. & Patrick, J. D. (1992), 'Temporal semantics in information systems - a survey', *Information Systems* **17**(3), 249–267.

Severance, D. G. (1983), 'A practitioner's guide to data base compression', *Information Systems* **8**(1), 51–62.

Snodgrass, R. T. (1987), 'The temporal query language tquel', *ACM Transactions on Database Systems* **12**(2), 247–298.

Stanoi, I., Agrawal, D., Abbadi, A. E., Phatak, S. H. & Badrinath, B. R. (1999), Data warehousing alternatives for mobile environments, *in* 'MobiDE', pp. 110–115.

Zaslavsky, A. B. & Tari, Z. (1998), 'Mobile computing: Overview and current status', *Australian Computer Journal* **30**(2), 42–52.

Zukunft, O. (1997), Rule based adaptation in mobile database systems, *in* 'Selected Areas in Cryptography', pp. 310–317.