



# *Query Optimization in Resource Constrained Databases*

**Aditee Badge** (04305002)  
**Prajakta Kalekar** (04329008)

under the guidance of  
**Prof. Krithi Ramamritham**  
**Prof. S. Sudarshan**

July 29th, 2005







## *Handhelds*

### *Features*

- Low cost, light weight, small sized
- Anytime, anywhere convenience

### *Applications*

Personal information management, educational and enterprise applications etc.

### *Example handheld - Simputer*

e.g. Simputer [Sim] has 32-bit 200MHz Strong Arm SA-1100 RISC CPU, 32 MB of DRAM, 24 MB Flash for Permanent Storage and Monochrome LCD Display Panel.





## *DBMS in handhelds*

### *Handheld applications*

- Have high volume of data
- Execute simple and complex queries on the device - select, project, complex join and aggregate queries
- Require maintaining ACID properties, Data Privacy, Synchronization, Mobility, Security

### *To meet these requirements efficiently,*

- DBMS should be lightweight
- DBMS techniques should be small memory and computation power cognizant







## *Overview of DELite*

DELite is an Open Source Light Weight Database Management System developed by IITB. Some features:

- Flash memory as read buffer and RAM as write buffer.
- Modified two phase optimizer generates only fully pipelined, left deep tree query execution plans.
- Pointer based storage model like ID-based and Domain based are supported.
- Index structures are generated in memory during query execution.
- Select, Project, Join, update and aggregate queries are supported.
- Read intensive environment is assumed.





## *Motivation*

- There is a mismatch between the kind of resources hand-helds are equipped with and the kind of queries they are expected to service. Though these devices are required to service queries that go well beyond simple Select-Project-Join queries, they have limited memory, computing power and communication band-width.
- The communication costs, coupled with the unreliable connectivity motivates the execution of queries at the client-end itself (without any delegation of work to the server).





## *Problem Definition*

*What are the factors utilizing resources (memory and energy) during query processing?*

*How to divide resources among these factors to get optimal plan for the query?*





## *Parameters affecting energy consumption*

### *Hardware*

processor, cache, memory and buses

### *Software*

compression, query evaluation plan used (The join algorithms selected, use of pipelining/materialization etc)

### *Query related parameters*

Type of query, result set size, number of attributes in the result, number and type of attributes participating in the WHERE clause etc.





## *Energy Consumption Models*

### *Model 1*

$E_{QEP} = p_{cpu} * t_{cpu} + p_{flash} * t_{flash} + p_{display} * t_{display} + p_{RAM} * t_{RAM}$   
 where  $p_{component}$  is the power consumed by *component* and  
 $t_{component}$  is the time for which it is used.

### *Model 2*

$E_{QEP} = n_{W_f} * E_{W_f} + n_{R_f} * E_{R_f} + n_{W_m} * E_{W_m} + n_{R_m} * E_{R_m} + n_C * E_C$   
 where  $E_W$  is the energy required for a write to flash/RAM  
 $E_R$  is the energy required for a read from flash/RAM  
 $E_C$  is the energy required for a unit of computation  
 $n_W$  is the number of writes to flash/RAM  
 $n_R$  is the number of reads from flash/RAM  
 $n_C$  is the number of units of computation





## *Experimental Setup*

- Hospital Schema used for experiments

Relation	Size
DOCTOR(DOCID int, NAME char[20])	91
PRES(VISITID int, DRUGID int)	2155
VISIT(VISITID int, DOCID int, DATE int)	830
DRUG(DRUGID int, TYPE char[20])	77

- Simputer Platform [Sim]
- 2 AA sized NiCd Eveready Rechargeable Batteries (1.2V, 700mAh)





## *Effect of number of attributes in result*

- Q1 : SELECT DOCTOR.NAME FROM DOCTOR, PRES, DRUG, VISIT WHERE VISIT.DATE = 1998 AND DOCTOR.DOCID = VISIT.DOCID AND DRUG.DRUGID = PRES.DRUGID AND PRES.VISITID = VISIT.VISITID AND DRUG.TYPE = 'Antibiotic'
- Q2 : SELECT \* FROM DOCTOR, PRES, DRUG, VISIT WHERE VISIT.DATE = 1998 AND DOCTOR.DOCID = VISIT.VISITID AND DRUG.DRUGID = PRES.DRUGID AND PRES.VISITID = VISIT.VISITID AND DRUG.TYPE = 'Antibiotic'

**Note:** The number of attributes in the result set for Q2 is more than that for Q1.

**Inference:** Changing number of attributes selected keeping other conditions same affects power consumption to a very small extent.

**Possible reason:** results not displayed on screen. Only query plan is displayed. Energy spent in flushing results on screen is zero in all the cases. i.e. display consumes greater energy comparatively.





## *Effect of number of relations participating in the join*

- Q3 : SELECT DRUG.TYPE FROM DRUG,PRES WHERE DRUG.DRUGID = PRES.DRUGID;
- Q4 : SELECT DOCTOR.NAME FROM DOCTOR, PRES, DRUG, VISIT WHERE DOCTOR.DOCID = VISIT.DOCID AND DRUG.DRUGID = PRES.DRUGID AND PRES.VISITID = VISIT.VISITID ;

**Note:** The number of relations participating in the join are more in Q4 than in Q3

**Inference:** Q3, containing fewer number of relations participating in the join as compared with Q4, has executed more number of times than Q4. From this we can conclude that the more the number of relations participating in the join, the more the energy consumption.

**Possible reason:** This result is intuitive, since more number of relations means more I/O and more computations.





## *Effect of condition evaluation*

- Q5 : `SELECT DOCTOR.NAME FROM DOCTOR,PRES,DRUG,VISIT WHERE DOCTOR.DOCID = VISIT.DOCID AND DRUG.DRUGID = PRES.DRUGID AND PRES.VISITID = VISIT.VISITID AND DRUG.TYPE = 'Antibiotic'`
- Q6(Q1) : `SELECT DOCTOR.NAME FROM DOCTOR, PRES, DRUG, VISIT WHERE VISIT.DATE = 1998 AND DOCTOR.DOCID = VISIT.DOCID AND DRUG.DRUGID = PRES.DRUGID AND PRES.VISITID = VISIT.VISITID AND DRUG.TYPE = 'Antibiotic'`

**Note:** Q6 has an additional condition (test for date).

**Inference:** Size of result set (number of computations) affects energy consumption.

**Possible reason:** computation of extra conditions may be consuming additional energy.





## Results

Query	Number of executions	Time(min)
idle	none	10
Q1	75	6
Q2	74	6
Q3	1321	6
Q4	34	5
Q5	106	5
Q6	75	6

*Table:* Experimental Results





## *Issues in Memory Constrained Environments*

- Storage Management – a critical issue in handhelds.
- Due to small size, handhelds can use Systems on Chip (SoC), where RAM competes with other components on the same silicium die. More RAM implies less stable storage, and hence the less embedded data. This trade-off arises every time the silicium die size needs to be reduced to match physical constraints such as thinness, energy consumption.





## *Issues related to compression in database systems*

- Compression and decompression are CPU intensive operations. The cost is especially important at low-end clients.
- Database applications often need random access to data in small pieces (tuples, attributes, etc). The popular compression methods only work well for large chunks of results, and a large chunk of results must be decompressed even if only a small piece is needed.
- A DBMS holds domain specific knowledge of the query results, which can be used to enhance the effect of compression. For instance, instead of using a single compression method for all attributes, a combination of compression methods that compress each attribute separately based on the domain knowledge can achieve better effect.





## *Query performance aware compression optimizer*

As per Heuristics, Compression is always beneficial for query execution, however

- String compression is useful when there are no decompressions or small number of decompressions
- Attribute level integer compression code overhead nullifies the positive compression benefit of integer data.
- Hence, Heuristic approach is not sufficient. A more cost based approach is required - a query performance aware compression optimizer would be useful





## Multi-Query Optimization

**Scenario:** Multiple related, but slightly different queries

**Goal:** Save power and communication

**Challenge:** Combining multiple queries, finding common query parts

*Two approaches:*

- Materialization
- Pipelining

In DELite, writes to secondary storage (flash) should be minimized. So there is minimum materialization of intermediate results. Left-deep trees most suited for pipelined evaluation are used.







## *Future Work*

Future work includes

- Study effect of compression on energy
- To come up with a model for Estimation of Energy Consumption of a query evaluation plan.
- To come up with an optimizer for an energy constrained environment that makes use of the above mentioned model for determining the costs of the query evaluation plans. To start with, extend [Sen04] optimizer to compute cost in terms of energy consumed.
- To study the various query parameters that affect energy consumption. These parameters would be used for estimation of energy consumption of a query given the energy consumption of a similar query.





## *Future Work*

- To study the energy behavior of various Index and Data Organizations in DELite. This can be used for determining what organization should be used for storing data while downloading it to the handheld.
- The insights gained by the above study can also be used by the optimizer to determine what organization should be used for storing the query results.
- To study the relation between the execution time and the energy consumed.





## References

**[ACN00]** Sanjay Agrawal, Surajit Chaudhuri, and Vivek R. Narasayya. Automated selection of materialized views and indexes in SQL databases. In The VLDB Journal, 2000.

**[AG92]** R. Alonso and S. Ganguly. Energy efficient query optimization. Technical Report MITL-TR-33-92, Matsushita Infotech Lab, Princeton, NJ, 1992.

**[ASV+01]** Ning An, Anand Sivasubramaniam, Narayanan Vijaykrishnan, Mahmut T. Kandemir, Mary Jane Irwin, and Sudhanva Gurusurthi. Analyzing energy behavior of spatial access methods for memory-resident data. In The VLDB Journal, 2001.

**[BBPV00]** C. Bobineau, L. Bouganim, P. Pucheral, and P. Valduriez. Picodbms: Scaling down database techniques for the smartcard. Technical report, PriSM, 2000.





## References

**[DEL]** DELite. <http://www.cse.iitb.ac.in/delite>.

**[DSRS01]** Nilesh N. Dalvi, Sumit K. Sanghai, Prasan Roy, and S. Sudarshan. Pipelining in multi-query optimization. In PODS '01: Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, 2001.

**[GAS+03]** Sudhanva Gurumurthi, Ning An, Anand Sivasubramaniam, N. Vijaykrishnan, Mahmut Kandemir, and Mary Jane Irwin. Energy and performance considerations in work partitioning for mobile spatial queries. In IPDPS '03: Proceedings of the 17th International Symposium on Parallel and Distributed Processing, 2003.

**[HSS00]** Arvind Hulgeri, S. Seshadri, and S. Sudarshan. Memory cognizant query optimization. In COMAD, 2000.





## References

**[NA03]** P. Pucheral N. Ancaix, L. Bouganim. Memory requirements for query execution in highly constrained devices. In 29th International Conference on Very Large Data Bases, VLDB'03, Berlin, 2003.

**[PCK04]** Jayaprakash Pisharath, Alok Choudhary, and Mahmut Kandemir. Reducing energy consumption of queries in memory-resident database systems. In CASES '04: Proceedings of the 2004 international conference on Compilers, architecture, and synthesis for embedded systems, 2004.

**[Rao05]** Ashwini G Rao. Compression in Memory Constrained DBMSs. Masters Dissertation, Indian Institute of Technology - Bombay, 2005.





## *References*

**[MRSR01]** Hoshi Mistry, Prasan Roy, S. Sudarshan, and Krithi Ramamritham. Materialized view selection and maintenance using multi-query optimization. SIGMOD Rec., 2001.

**[Sen04]** Rajkumar Sen. An open source DBMS for handheld devices. Masters Dissertation, Indian Institute of Technology - Bombay, 2004.

**[Sim]** The Simputer. <http://www.simputer.org>.

**[SKS02]** A. Silberchatz, H. Korth, and S.Sudarshan. Database System Concepts. 2002.





## References

**[CGK01]** Zhiyuan Chen, Johannes Gehrke, and Flip Korn. Query optimization in compressed database systems. In SIGMOD: Proceedings of the 2001 ACM SIGMOD international conference on Management of data, 2001.

**[TX04]** Dimitri Theodoratos and Wugang Xu. Constructing search spaces for materialized view selection. In DOLAP '04: Proceedings of the 7th ACM international workshop on Data warehousing and OLAP, 2004.

**[RSSB00]** Prasan Roy, S. Seshadri, S. Sudarshan, and Siddhesh Bhohe. Efficient and extensible algorithms for multi query optimization. In ACM SIGMOD International Conference on Management of Data, 2000.





# THANK YOU



- 
- 
- 
- 
- 
- 

- 
- 
- 
- 
- 

- 
- 
- 



- 
- 
- 
- 
- 
- 

- 
- 
- 
- 
- 

- 
- 
- 



- 
- 
- 
- 
- 
- 

- 
- 
- 
- 
- 

- 
- 
- 

