Performance Comparison of Static vs. Dynamic Two Phase Locking Protocols

Sheung-lun Hung
City Polytechnic of Hong Kong

Kam-yiu Lam
City Polytechnic of Hong Kong

Queueing network models have been used extensively in the study of the performance of different concurrency control protocols in database systems. However, the results obtained are mostly qualitative in nature with few quantitative design rules which address performance issues such as data and resource contentions. In this paper, an attempt has been made to carry out comprehensive simulation studies of two well known locking protocols, namely the Static Two Phase Locking and the Dynamic Two Phase Locking, for database systems with the aim of obtaining practical design rules for such systems.

Various concurrency control algorithms for database systems have been proposed (Berstein & Goodman, 1987; Dasgupta & Kedem, 1990; Ibaraki, Kameda & Katoh, 1990; Yu & Dias, 1990). Two Phase Locking is the mechanism most commonly adopted in the design of commercial database systems. The impact of the different locking protocols on performance have been the theme of many studies (Huang & Chin, 1988; Pun & Belford, 1987; Ryu & Thomasian, 1990; Tay, 1987). Of these protocols, the Static Two Phase Locking and the Dynamic Two Phase Locking are two of the better known designs for locking schedulers (Berstein, Hodzilacos & Goodman, 1987; Date, 1985; Papadimitrious, 1986).

The Basic Two Phase Locking Protocol has two parts (Gray, 1978; Eswaran, Gray & Lorie, 1976). In the growing phase, each transaction has to set lock to a data item before accessing it. In the shrinking phase, a transaction releases its locks. Once a lock has been released, a transaction may not subsequently obtain any more locks. Some of the advantages of this protocol are that it is both recoverable and free from cascading aborts (Berstein, Hodzilacos & Goodman, 1987). However, it is susceptible to the problem of deadlocks. In Static Two Phase Locking, transactions are required to declare their lock set before accessing any data items in the database. Locks will be set for the transaction only when all the locks in its lock set are not in conflicting mode set by other transactions. The processing of transaction starts after the locks are granted. As cyclic-wait for locks is not possible, this protocol is completely free from deadlocks. All the acquired locks are released on commitment of the transaction (Gray, 1978). In Dynamic Two Phase Locking, each lock request must be satisfied before its transaction can access a data item and all locks are released on commitment of the transaction. Dynamic Two Phase Locking is expected to have fewer lock conflicts as a result of the smaller average lock holding time. However, Dynamic Two Phase Locking has the disadvantage of an increase in the probability of deadlocks and some deadlock resolution methods (Agrawal, Carey & McVoy, 1987; Raynal, 1988) have to be implemented.

Although there are many studies on the performance of different locking protocols, the results are far from conclusive and a number of issues have still to be resolved. In
certain cases, the results are even contradictory. Agarwal, Carey & Livny (1987) showed the importance of different model assumptions. Contradictions can be the result of a combination of the effects of different model assumptions and the use of different parameters and measures (Franiazek & Robinson, 1985; Hartman, 1989; Huang & Chin, 1990; Irani & Lin, 1978; Tay, Suri & Goodman, 1982; Thomasian & Ryu, 1989). In the work of Pun and Belford (1987), they have applied analytic techniques to study the performance of three variants of the Two Phase Locking protocols, namely preclaim, claim-as-needed and claim-with-order, under different lock selection methods and granularities which is the basic unit of locking. The preclaim protocol is similar to Static Two Phase Locking and the claim-as-needed is similar to the Dynamic Two Phase Locking protocol. Some of the weaknesses in their models are the way they handle blocked transactions and the priorities of different transactions in the system. The range for different parameters used in their study is also a limiting factor. More work is needed in clarifying model inconsistencies and to arrive at more quantitative design rules for such database systems. The effect of including read operations in a transaction has received little attention with the exception of Kumar (1988). His study had paid special attention to the effect of different read:write ratio of operations using an open model. However, the effect of increasing read-only operations in an interactive computer system is still not clear.

In this paper, an in-depth study of different factors such as granularity, multiprogramming level, lock selection methods, read/write ratio of transactions on the two locking protocols has been carried out. In the case of Dynamic Two Phase locking, the time-out deadlock resolution method is used and the effect of different time-out intervals under different granularity of database has been examined. One of the very important factor which has a great impact on performance is the degree of resource and data contention (Franiazek, Robinson & Thomasian, 1990; Ryu & Thomasian, 1990; Tay, Suri & Goodman, 1982; Thomasian & Ryu, 1989). An attempt has been made to study the performance of the database systems under different degree of resource and data contention. The aim of this investigation is to derive rules that may exist for such systems and to explore the theoretical underpinning of some of the results. The remainder of this paper is organized as follows. Section 2 describes the queuing network models used for the two locking protocols. A summary and description of the system parameters used in the simulation is given in section 3. Section 4 presents simulation results and their interpretations. Last, the conclusion of this paper is drawn in section 5.

**Queuing Network Models**

In the following, two queuing network models are presented for database systems with the two locking protocols. The systems hardware comprised of one CPU with a number of disk units and terminals where transactions are submitted after a think time which is the average interarrival time of transactions. Each operation of a transaction may access a number of data items. The data items in the database are grouped into granules which are the basic unit for locking. Fine granularity is the term used when the number of data items in a granule is small whereas coarse granularity is used when there are a large number of items in each granule. Transaction size is the number of data items accessed by a given transaction. It is considered to be large if its size is greater than 1/100 of the total number of data items in the database.

The following diagram is the queuing network model for the database system using the Static Two Phase Locking protocol:

![Figure 1: Queuing Network Model for Static 2 Phase Locking](image)

The service centres in the model are the central processing unit (CPU), which is responsible for the processing of computations and locking overheads; disk units which are responsible for data items accesses in the database; and a block queue to handle transactions failing to obtain locks.

The transactions in the model are classified into 5 with different priorities in CPU access:

<table>
<thead>
<tr>
<th>Class</th>
<th>Transaction</th>
<th>**Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TerminalOriginated</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Lock Access*</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>Object Access*</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>Lock Release*</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>BlockedTransactions*</td>
<td>3</td>
</tr>
</tbody>
</table>

* This includes the checking and setting of locks
** Priorities: 1<2<3<4

**Table 1**
Each Class-1 transaction enters the ready queue and is admitted to the system as long as the total number of active transactions does not exceed the MPL. In our model, if a transaction has passed the MPL allocate switch and admitted into the system for processing, it is considered to be active. The transaction is then changes to Class-2 for lock setting. Providing none of the lock requests are in conflict with those set by other transactions already in the system, all the locks will be set immediately. Completion of setting locks changes the transaction to Class-3 which then starts to retrieve data items from the database and join the CPU queue for processing. If any of the lock requests are denied as a result of lock conflicts with other transactions, none of the locks will be set and the transaction will be changed to Class-5 and placed in the Block Queue until any locks have been released by a Class-4 transaction. Then it queues for CPU to try to set lock again. Class-4 signifies the completion of processing and ready to release all its locks. After releasing its locks, it is changed to Class-1 and returned to terminals.

The major differences between our model for Static Two Phase Locking and Pun’s (1987) Pre-Claim model is in the handling of blocked transactions. In Pun’s work (1987), it is assumed that transactions in the block queue are delayed by an exponentially distributed holding time and locks would be granted unconditionally for released transactions (an unrealistic assumption). In the present model, a blocked transaction can only leave the block queue whenever its waiting lock is released by the seized transaction. Further blocking may occur on the released transaction if some of its locks are again seized by other transactions. Another matter which has been overlooked in Pun’s model is the importance of scheduling priorities. Different classes of transactions in the model has different priorities in resources access as tabulated in Table 1.

The queuing network model for the Dynamic Two Phase Locking database system is more complex in the sense that one has to cater for the deadlock situation. The following is a diagram of the model.

![Figure 2: Queuing Network Model for Dynamic 2 Phase Locking](image)

For Dynamic Two Phase Locking, the following modifications to the old model is required. One more service centre, Delay Server, is added to delay any aborted transactions for a period of time before it is restarted and the CPU has to perform Undo operations for any aborted transaction to restore the state of the database.

Seven classes of transactions can now be identified:

<table>
<thead>
<tr>
<th>Class</th>
<th>Transaction</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Terminal Originated</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Lock Access</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Blocked Transactions</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>Data Item Access</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>Lock Release</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>Undo Operations</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>Lock Release*</td>
<td>7</td>
</tr>
</tbody>
</table>

* This is the lock release for aborted transactions

Table 2

Similar to the last model, each Class-1 transaction enters the system if number of active transactions is smaller than MPL. The newly admitted transaction changes to Class-2 and initiates a lock request for its first granule. In the absence of conflicts, lock is set and the transaction changes to Class-4 for data item accesses which involve both CPU and disks. When the data items in the granule have been processed, the transaction changes back to Class-2 to set lock for next granule. This process will be repeated until all the data items of a given transaction have been processed. In case of lock conflicts, the requesting transaction changes to Class-3 and is placed in the Block Queue where it will stay until its requesting lock is released by the seized transaction. At that time, it is changed to Class-2 for lock setting. The Time-out method is used as a mean for deadlock resolution. Under this method, the CPU will periodically check whether there are any transactions have been blocked for a time greater than the predefined time-out interval and are holding some locks. The selected transaction will be changed to Class-6. All the database operations performed for this transaction have to be Undone and then it is changed to Class-7 for lock releasing. The aborted transaction is then changed back to Class-2 and has to be delayed for a period of time in the Delay Server before being restarted again to avoid the chance of immediately getting into a deadlock situation again.

In contrast to Pun’s (1987) Claim-as-needed model, the present model for Dynamic Two Phase Locking database systems includes the workload for Undo operations whenever a transaction is aborted as a result of deadlock. The deadlock resolution method adopted by Pun (1987) was the Wait-For-Graph whereas time-out method is used in the
model. In addition, aborted transactions are delayed for a fixed time period before being restarted so as to minimize deadlock induced thrashing.

**Model Parameters and Performance Measures**

The model parameters are defined as followed:

- **DS**, number of data items in the database
- **NT**, number of terminals
- **TS**, transaction size, which is measured in terms of number of data items to be accessed by a transaction.
- **Z**, mean think time of the terminals
- **MPL**, multiprogramming level of the system
- **Ng**, number of granules in the database
- **Nl**, number of granules to be accessed by a transaction
- **Tcpu**, mean CPU computation time for a data item.
- **Tio**, mean I/O time to access a data item
- **Tch**, mean lock checking time
- **Tset**, mean set lock time
- **Trel**, mean time to release a lock
- **Tout**, the time-out interval used for deadlock resolution

The service times of the service centres are assumed to be exponentially distributed. In order to simplify the model, the database size is assumed to be fixed. A relatively small database comparing with transaction sizes is used to increase the degree of conflicts among transactions.

A sensitivity analysis has been carried out to determine the effect of different parameter values on the system performance. Results that if the value of Tcpu is equal to Tio, the CPU will be the bottleneck resource under all cases if one CPU and two or more I/O units are used in the model. The hardware configuration was chosen to simplify the analysis of the results.

The following system parameters have been chosen for the simulation experiments:

- **DS** = 5000 data items
- **NT** = 100 terminals
- **Z** = 5 seconds
- **Tcpu** = 50 ms
- **Tio** = 50 ms
- **Tch** = 5 ms
- **Tset** = 5 ms
- **Trel** = 5 ms
- **Tout** = TS * MPL

The variance of the waiting time in the block queue increases with MPL values (Agrawal, Carey & McVoy, 1987). To minimize the probability of aborting transactions not involved in deadlock, the value of Tout for deadlock resolution is chosen to be proportional to the MPL values. Different values for MPL, Ng and TS are used in the experiments:

- **MPL** = 1 to 32
- **Ng** = 1 to 5000 (maximum number of granules)
- **TS** = 5, 20, 100 and 200 (in data items)

In determining the Nl for each transaction, it is assumed that transactions access data items randomly and the data items are fully packed in granules as far as possible for the same transaction (Pun & Belford, 1987). The number of granules to be accessed by a transaction can be calculated by:

\[ Nl = \frac{(TS * Ng)}{DS} \]

Since DS is constant, for a fixed transaction size the value of Nl can be expressed in terms of Ng.

**Comparison of Results**

The effect of the following factors on performance for database systems with the two locking protocols has been studied:

- (a) granularity of the database;
- (b) multi-programming level;
- (c) lock selection methods;
- (d) percentage of read/write operations;
- (e) time-out interval for deadlock detection.

**Granularity and Multiprogramming Level**

Granularity Ng of the database system can have a significant impact on the performance of the database systems. Its value determines the number of data items in a granule and the number of locks to be set by a transaction. One would expected that the concurrency and the throughput of the system to increase with Ng. However, for fine granularity, i.e. large value of Ng, both the locking overhead and the degree of data contention become significant factors as now each transaction has to set a greater number of locks. It is to be expected that throughput should initially increase with Ng and decrease for larger values of Ng. In other words, there should be an optimum value of granularity where throughput has maximum values. In Pun’s study (1987), he postulated that the optimum granularity is given by:

\[ [Ng]_{opt} = [DS/TS] \] (1)
Figure 3: S2PL-Different Granularity & TS

Figure 4: D2PL-Different Granularity & TS

Figure 5: Granularity & Throughput Bound

Figure 6: D2PL-Different Granularity & MPL
where \([N_g]_{opt}\) is the value of \(N_g\) for maximum throughput (rounding up to the nearest integer). Throughput for Static Two Phase Locking (S2PL) and Dynamic Two Phase Locking (D2PL) as a function of different transaction sizes are shown in figures 3 and 4 respectively. Despite the difference in model assumptions, the results seem to be in excellent agreement with Pun’s results for both locking protocols. The figures also suggest that larger transactions have more sharply defined optimum values and thus are more sensitive to changes in granularities. The initial increases in throughput is due to the decrease in degree of data contention as \(N_g\) increases without increasing the number of locks for each transaction. After the optimum granularity \([N_g]_{opt}\), the decrease in system throughput is the result of greater degree of data contention and locking overhead for each transaction as now each transaction has to set more locks.

In the absence of data contention, throughput should be bounded by resource contention (Zahorjan, Lazowska & Graham, 1984). The maximum throughput, \(X_{max}\), is given by the equation

\[
X_{max} = \frac{1}{D} \tag{2}
\]

where \(D\) is the service demand of the bottleneck resource and includes the mean service time for a transaction and its associated locking overhead. The degree of data contention depends to a large extent on the values of granularity \(N_g\) which in turn affects the values of \(X_{max}\). The effect of granularity, \(N_g\), on system throughput for different values of multiprogramming level for the Static Two Phase Locking is given in figure 5. Analysis of the results show that for large values of MPL and \(N_g\), \(X_{max}\) is bounded by :

\[
X_{max} = \frac{1}{D} - K' \cdot \ln N_g \quad MPL > 20 \tag{3}
\]

\[
N_g > [N_g]_{opt}
\]

Where \([N_g]_{opt}\) and \(K'\) are constants which have the approximate values of 1000 and 95 respectively for the set of input parameters.

In the case of Dynamic Two Phase Locking, equation 3 is no longer valid. Figure 6 shows that for values of \(N_g\) greater than \([N_g]_{opt}\), there is a dramatic decrease in system throughput. This can be explained in terms of occurrences of deadlocks which affect throughput in two ways. Firstly, as a result of deadlock, CPU has to perform Undo operations for aborted transactions. Secondly, the time-out method used in deadlock resolution results in the blocking of deadlocked transactions. A more detailed discussion on the effect of the time-out parameter on performance will be given later. The results also suggest that Dynamic Two Phase Locking has similar performance characteristics to Static Two Phase Locking up to a value of \(N_g\) which is dependent on MPLs. As the probability of deadlock should increase with MPL, deadlock induced thrashing should have greater impact on systems with larger MPL values. Table 3 gives estimates of values of \(N_g\) for different values of MPL when deadlock induced thrashings become significant.

<table>
<thead>
<tr>
<th>Log ((N_g))</th>
<th>MPL</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.50</td>
<td>8</td>
</tr>
<tr>
<td>2.70</td>
<td>4</td>
</tr>
<tr>
<td>2.90</td>
<td>2</td>
</tr>
</tbody>
</table>

The impact of multiprogramming level has been briefly mentioned. In figure 7, throughput as a result of changing the multiprogramming level for the Static Two Phase Locking database system is given. The increase in system throughput diminishes as MPL increases and is bounded for larger values of MPL. The saturation in throughput is the combined effect of data and resource contention. For values of \(N_g\) greater than \([N_g]_{opt}\) but less than \([N_g]_{m}\), the maximum system throughput \(X_{max}\) has the form

\[
X_{max} = \frac{1}{D} - K'' \cdot N_g \quad [N_g]_{opt} < N_g < [N_g]_{m} \tag{4}
\]

Equations 3 and 4 suggest that the maximum throughput should have the approximate theoretical form

\[
X_{max} = \frac{1}{D} - K \cdot \ln (a + N_g) \tag{5}
\]

where \(a\) is another constant.

For large value of \(N_g\), i.e. \(N_g >> a\), equation 3 will hold, whereas for smaller values of \(N_g\), one can use equation 4. In figure 8, the maximum system throughput \(X_{max}\), as a function of \(N_g\) for all \(N_g\) greater than \([N_g]_{opt}\), is presented.

The constants \(K, K'\) and \(K''\) in general depend on the processing time for each data item \(T_{cpu}\) and the locking overhead for each granule \(T_1\). As observed from figure 9, the small locking overhead can have only marginal effect on system throughput. From figure 10, for moderate values of \(T_{cpu}\), its effect on system throughput is far from negligible. Measurements suggest that \(K''\) of equation 4 has the form

\[
K'' = C/T_{cpu} \quad \text{for} \quad T_{cpu} >> T_1 \tag{6}
\]

where \(C\) is another constant.

In the case of Dynamic Two Phase Locking, figure 11 shows that increasing MPL beyond some optimum values can sharply reduce system throughput. Such system behaviour can be again explained in terms of occurrences of deadlock. The decrease in system throughput depends on the granularity of the database and the optimum multiprogramming level \([MPL]_{opt}\), which is defined as the value of MPL.
Figure 7: S2PL: Effect of MPL

Figure 8: S2PL: $X_{\text{max}}$ in Different Granularity

Figure 9: S2PL: $X_{\text{max}}$ and Different Locking Overhead

Figure 10: S2PL: $X_{\text{max}}$ and Different $T_{\text{cpu}}$
when deadlock induced thrashing occurs. Estimates of $[\text{MPL}]_{\text{opt}}$ for different values of $N_g$ is given in table 4. It is of great importance for database designers to choose the appropriate MPL values for database systems based on the Dynamic Two Phase Locking.

<table>
<thead>
<tr>
<th>$N_g$</th>
<th>$[\text{MPL}]_{\text{opt}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>315</td>
<td>7</td>
</tr>
<tr>
<td>500</td>
<td>6</td>
</tr>
<tr>
<td>714</td>
<td>5</td>
</tr>
<tr>
<td>1000</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 4

The performance of Dynamic Two Phase Locking database systems is more susceptible to the number of locks per transaction $N_g$ than MPL. By breaking up large transactions into smaller ones, one can actually suppress occurrences of thrashing to higher MPL values. The results in figure 12 demonstrated that significant improvements can be achieved by halving the transaction size.

**Lock Selection Methods**

The simulation results described in the last subsection has made the assumption of random selection of granules. However, real database applications rarely access data in a purely random fashion and some spatial locality usually exist. A sequential granule selection strategy was adopted for the following set of experiments to study the effect of spatial locality on system performance. In this selection strategy, for a given transaction, the first granule is selected at random, but subsequent granules are selected sequentially. The performance for the two protocols under this lock selection strategy are shown in Figures 13 and 14 respectively. In the case of Static Two Phase Locking, there is marked improvement as a consequence of the reduction in data contention. The probability of conflicts among different concurrently executed transactions is smaller under this lock selection method as compared to the random selection method. For Dynamic Two Phase Locking, the improvement is more dramatic because of the absence of deadlock under this new strategy. In figure 15, the throughput for the two locking protocols are compared. It can be easily observed that the difference is minimal. This is also due to the small degree of data contention under this lock selection method for the two locking protocols.

**Percentages of Read/Write Operations**

Depending on the mode of operation to be acted on data granules, two types of locks, namely exclusive locks for read/write operations and share locks for read only operations are possible. All exclusive read/write locks can be in conflict. In this section, the effect of the number of read locks to the number of write locks in transactions are being investigated. The read/write ratio is defined as the percentage of read-only operations to read/write operations in a transaction. Figures 16 and 17 show that for the case of Static Two Phase Locking, increasing the number of read-only operations can help in improving system performance because of the reduction in lock conflicts. As suggested in the result, the increase in throughput is affected by the granularity of the database and the read/write ratio of the transactions. The increase is greater if $N_g$ is larger and the ratio is small. The result is consistent with Tay (1987). Similar results hold for Dynamic Two Phase Locking as shown in figures 18 and 19. The improvement is greater under high MPL as the probability of deadlock is also smaller for greater the read/write ratio.

**Time-out Interval**

Time-out method is the adopted method for deadlock resolution for the Dynamic Two Phase Locking database systems. The time-out interval is seen as a very critical design parameter. It is affected by a number of factors such as MPL, granularity of database, and some of the hardware characteristics also play a significant role in determining its optimum value. Figure 20 shows the throughput as a function for time-out intervals for different values of $N_g$. It is observed that the optimum values of time-out, $T_{\text{opt}}$, is affected by the granularity of the database system. For large $N_g$, a small time-out should be chosen.

**Conclusions**

This paper presents a performance comparison between Static Two Phase Locking and Dynamic Two Phase Locking for concurrency control in database systems. Based on the simulation results, the authors arrived at the following conclusions:

1. Under the set of model assumptions, for both locking protocols, the optimum granularity for maximum throughput depends on transaction size in a manner as postulated by Pun & Belford (1987).

2. Increasing MPL always improve performance for Static Two Phase Locking whereas for Dynamic Two Phase Locking, under fine granularity, deadlock induced thrashing may degrade system performance at high MPL.

3. The degree of resource and data contention in the system are seen as the main factors in determining the maximum system throughput. The maximum system throughput for Static Two Phase Locking is bounded by a value dependent on the granularity of the database.
(4) Breaking down large transactions into smaller transactions improves system performance for both protocols, especially for Dynamic Two Phase Locking.

(5) The performance of both protocols, especially Dynamic Two Phase Locking, can be improved by adopting the sequential lock selection method which is deadlock free.

(6) Although increasing the percentage of read-only operations in transactions improves the system performance, the effect is marginal and decreased with increase in read-only operations. Furthermore, it is affected by the granularity of database. In the case of Dynamic Two Phase Locking, dramatic improvement is attainable through suppressing thrashing to higher MPL.

(7) Time-out is a simple method for resolving deadlocks. Results show that there exist optimum time-out intervals which are affected by the granularities of database accompanied with other factors. Although the results has been confined to a limited set of parameters used in the simulation experiments, they can serve as useful guidelines for the design of such systems. However, a more exhaustive study is needed to investigate the effect of different parameters set on performance such as to quantity the effect of different proportion of read/write operations. Other factors, such as MPL, transaction sizes, think time and some other hardware characteristics on the optimum time-out interval should be studied in order to have better understanding of the factors determining the optimum time-out interval. Furthermore, an area worthy of greater attention and has not been addressed is the effect of semantics of transactions on system performance.

References


Dr. S. L. Hung received his Ph.D degree from the University of Warwick, U.K. He is currently principal lecturer at the Department of Computer Science of the City Polytechnic of Hong Kong. His current research interests include Distributed Systems, Software Metrics and Neural Networks.

Mr. K. Y. Lam received the BSc degree from the Department of Computer Science, City Polytechnic of Hong Kong, Hong Kong. He is currently a Master of Philosophy candidate in the City Polytechnic of Hong Kong. His current research interests include Performance Evaluation, Distributed Systems and Software Engineering.
Related Content

Cache Management for Web-Powered Databases
Dimitrios Katsaros and Yannis Manolopoulos (2003). Web-Powered Databases (pp. 203-244).
www.irma-international.org/chapter/cache-management-web-powered-databases/31429/

Common Information Model
www.irma-international.org/chapter/common-information-model/11126/

GeoBase: Indexing NetCDF Files for Large-Scale Data Analysis
www.irma-international.org/chapter/geobase/85460/

Dimensions of UML Diagram Use: A Survey of Practitioners
www.irma-international.org/article/dimensions-uml-diagram-use/3379/

Agility Facilitators for Contemporary Software Development
www.irma-international.org/article/agility-facilitators-for-contemporary-software-development/160349/