Chapter 3

Characterization of Tuning Parameters, Workload-types and Memory modeling

3.1 Introduction

The success of effective and robust self-tuning of DBMS for performance gain depends largely on the strength of the knowledge-component of the autonomic computing architecture. The two important knowledge-components of the autonomic computing namely the sensors and effectors have to be identified and their effect is to be established in the DBMS context. Hence, it is important to identify the indicators[84] of performance degradation called sensors. One obvious sensor would be user load as the users increase, the load on the server would increase and thus affect the response-time. However, other sensor inputs that are not so obvious have to be experimentally identified. It is also necessary to identify key tuning parameters called effectors that have significant impact on the query response-time under various workload types. Though, there have been attempts to identify
sensor and effectors in the past, these experiments were not comprehensive as they were limited to ranking the tuning parameters [12] and also determine the impact only on one or two workload types [13]. The literature review also reveals that no research work has been done to study the effect of user-load and scaling factors on the query response-time. Hence, there is a need to systematically carry out a series of experiments taking into account workload types, User-load variations etc. and establish certain facts that help in developing the adaptive tuning techniques discussed in the following chapters.

From the literature survey it is evident that understanding the workload characteristics, establish the impact of tuning parameters. Determining the extent of interference between the tuning parameters and their effect on the system performance is critical in devising adaptive self-tuning techniques. The experiments have been categorized as Workload characteristics, tuning parameter characteristics and tuning parameter inter-dependency.

There have been attempts to model the DBMS memory to establish relationship between the buffer-hit-ratio [63] and the database-cache using analytical approaches and use this equation for self-tuning. However, DBMS being a complex software system, the analytical approaches presented are not only difficult, but also limited in their scope as they do not directly relate the response-time with sensors and effectors. Hence, an attempt is made as part of this research work to develop a mathematical model that relates the query response-time with Buffer size and User-load.

3.1.1 Workload Types & their characteristics

Before the details of the experiments are discussed, it is important to present the standard workload types and their characteristics used in experimentation. The Transaction Performance Processing Council(TPC) has classified the workloads as TPC-C, TPC-H, TPC-E, TPC-W etc. each representing certain application scenario. Table 3.1 describes the workload type and their details. The workloads were simply named as A, B, C, D etc. that represent different application scenarios. The workload types TPC-A, TPC-B are obsolete as they have been refined and accommodated in other workload types like TPC-C, TPC-D etc.
Table – 3.1 Workloads and their characteristics

<table>
<thead>
<tr>
<th>Sl. No.</th>
<th>Workload Type &amp; Description</th>
<th>Scaling Factor</th>
<th>Database Size</th>
</tr>
</thead>
</table>
| 1       | TPC-C
OLTP (Order Processing System)         | 2              | 138.66MB         |
|         |                                          | 5              | 334.92MB         |
|         |                                          | 10             | 662.01 MB        |
| 2       | TPC-E*
OLTP (Brokerage House)              | 1              | 8.55GB           |
|         |                                          | 2              | 17.09GB          |
| 3       | TPC-H*
DSS (Adhoc Business Queries)        | 1              | 956.02MB         |
|         |                                          | 2              | 1.87GB           |
| 4       | TPC-D
DSS (Broad range of DS Systems)        | 1              | 956.02MB         |
|         |                                          | 2              | 1.87GB           |

[TPC ➔ Transaction Performance Processing Council *H ➔ Heavy workload, E ➔ E-Commerce workload]

As can be seen from Table – 3.1, the database size has wide variation over the range of 138.6MB to 17.09GB for the above workload types. Larger the size of the database longer it takes to answer end-user queries. Therefore, to reduce frequent disk accesses, it is advantageous to keep the memory subcomponents larger for databases of TPC-H and TPC-E workloads and thus improve the query response-times.

3.1.2 Scope of the Experimental work:

Due to limitations of the Workload generation tool, and the large number of sensor-effect combinations, the scope of the experiments is limited to:

1. User-load range is limited to 2-100 users as the workload generation tool has limited user-load support.
2. The Scaling factors are limited to 2-10 for TPC-C workload and 1-2 for other workload types.
3. Characterization is limited to only the dynamic tuning parameters of Shared memory segment of Oracle 10g.

3.2 Database Memory Architecture

It is important to understand the DBMS memory architecture in order to establish the impact of various tuning parameters on system performance. Since, DBMS requires memory for diverse set of needs, the memory is divided into segments.
Figure 3.1 Memory Structure of Oracle 10g: Source Oracle Technology Network

Each segment has several subcomponents and each subcomponent has a specific purpose. The Oracle 10g memory architecture that is divided into Shared memory and Non-shared memory is shown in figure 3.1. In this section, a detailed memory structure of Oracle 10g is presented. Though the memory structure in other database systems is similar, however, the purpose and sizes of the subcomponents may vary. To speedup data access by avoiding expensive disk accesses every DBMS has its own memory structure, which is divided into Shared and Private memory subcomponents.

The shared memory is used by all the user processes while private memory is used by individual user processes. It is very difficult to manage the shared memory compared to private memory as the user processes would be accessing this memory concurrently. Further, allocating appropriate amount of memory to the subparts of the Shared memory is not a trivial task [25]. Following are the important subcomponents of the DBMS Shared memory and non-shared memory.

3.2.1 Buffer Cache

The database buffer cache holds blocks of data from disk that have been recently read to satisfy a select statement or that contain modified blocks that have been changed or added from a DML statement. In Oracle10g[115], the memory area in the SGA that holds these data blocks is dynamic. As the processing and transactional needs change during the day or during the week, the values of DB_CACHE_SIZE and DB_nK_CACHE_SIZE can be dynamically changed without restarting the instance to enhance performance for a table. Oracle can use two
additional caches with the same block size as the default (DB_CACHE_SIZE) block size: the KEEP buffer pool and the RECYCLE buffer pool.

3.2.2 Shared Pool

The shared-pool contains two major sub caches: the library cache and the data dictionary cache. The shared pool is sized by the SHARED_POOL_SIZE initialization parameter. This is another dynamic parameter that can be resized as long as the total SGA size is less than SGA_MAX_SIZE or SGA_TARGET.

3.2.3 Library Cache

The library cache holds information about SQL and PL/SQL statements that are to be executed by the execution engine of the DBMS. The PL/SQL statements in the library cache are shared by all users and hence, many different database users can potentially share the same SQL statements. The library cache also holds execution plan and parse tree along with the SQL statements. If an identical SQL statement is run, by the same user or a different user, the execution plan then the parse tree that is already computed is reused, thus improving the execution time of the query. If the size of library cache is too small, then the execution plans and parse trees are flushed out of the cache, requiring frequent reloading of SQL statements from the disk into the library cache thus leading to increase in query execution time.

3.2.4 Data Dictionary Cache

The data dictionary contains database tables, that belong to SYS/SYSTEM schemas. It also contains the metadata about the database, database structures, and the privileges of the users. The data dictionary cache holds cached blocks from the data dictionary. These Data blocks are used frequently to assist in processing user queries and other DML commands. A very small dictionary cache leads to frequent requests for information from the data dictionary and will cause disk access. Hence, the size must be appropriately set so that the execution of queries is sped-up.

3.2.5 Redo Log Buffer

The redo-log buffer holds the most recent changes to the data blocks in the data files. When the redo-log buffer is one-third full, or every three seconds, redo-log records are written. The entries in the redo log buffer, are used to recovery the database if the database instance crashes. For faster database recovery the redo log buffer must be large enough.
3.2.6 Large Pool

The large-pool is an important memory area of the SGA. It is used for transactions that involve large update operations, processes performing parallel queries, and backup/restore operations. The large pool provides large chunks of memory for heavy database operations such as query that is part of Decision support system. The initialization parameter LARGE_POOL_SIZE controls the size of the large pool and is a dynamic parameter.

3.2.7 Java Pool

The Java pool is used by the Oracle JVM (Java Virtual Machine) for all applications that run Java code and use data within a user session. Storing Java code and data in the Java pool is similar to SQL and PL/SQL code cached in the shared pool.

3.2.8 Program Global Area

The Program Global Area belongs to a private process. The role of the PGA depends on whether DBMS is configured as shared or dedicated server. In a shared server configuration, multiple users share a connection to the database, minimizing memory usage on the server. However, this might affect the response time for user requests. The PGA also includes a sort area that is used whenever a user request requires a sort, merge, or hash join operation.

3.3 Experimental Setup:

Figure 3.2 shows the complete experimental setup. The experiments were conducted on PowerEdge Dell Server R410, having Dual CPU, 6 core running at 2.8GHz, 16GB RAM and 500 GB Hard-disk space. In this setup, Oracle 10g is used as a case study and the database under test running on Windows 2003 Server operating system. The three tuning parameters namely, Db_cache_size(DCS), Shared_pool_size(SHP) and Large_pool_size(LRP) of the System Global Area(SGA) are used as variables in the setup. The WorkbenchFactory Version 6.8.1 is used as the workload generation and performance measurement tool. This tool allows the variation in the User-load(N), Scaling Factor(SF) and choosing the required Workload Types(WL) as the readings are taken. The workload generation tool is a trial version and has certain limitations. The important limitations being the number of virtual users are limited to 100 users. Hence, test for higher user-load could not be carried out.
For heavy workloads like TPC-H and TPC-D the user range can’t be specified. As a result, the readings have to be taken on a per-user basis and have to be recorded and plotted using a separate tool. Further, the workload generation tool also doesn’t have the facility to measure and display the throughput and hence, only response-time is recorded in all the experiments. Even though, this tool has these limitations, it is a very versatile workload generation tool as compared to other tools like HammerOra, Swingbench, etc. because of the variety of workloads and scaling factors it supports, ready results it provides and most importantly the ease of use.

### 3.3.1 Effect of Database Cache on Query Response-time

The **Db_cache** tuning parameter is primarily used for caching the disk pages as it helps in avoiding expensive Disk I/Os. Hence, this parameter affects the query response-time the most. In this experiment, the other two tuning parameters are set at default values(Shared_pool=80MB and Large_pool=8 MB) and the Buffer_cache size is varied from 32MB to 1200MB and the average query response time is noted for a given workload type and scaling factor. These experiments were carried out for user loads of 20, 40, 60 & 100 and the graphs were plotted. Following graphs in figure 3.3 show the impact of **DB_cache** size on the query response-time for TPC-C workload type and scaling factor of 2. As can be seen from the graph in figure 3.3, this parameter has significant impact on the query response-time.
The change in response-time is in the range 124-212 msec to 8-23 msec when the DB_Cache is changed from 32MB to 556MB. Further, this parameter has impact over a wide range 32MB-1200MB. The response-time rapidly increases with increase in number of users as expected. The user-load there can be used as a sensor input in detecting the degradation in performance and take appropriate measures to restore the performance of the system.

3.3.2 Effect of Shared pool on the Query Response-time

Shared pool is also an important subcomponent of SGA and is mainly used for resources that are shared by all the users of the DBMS. This memory area contains shared compiled queries, shared locks and other important database resources for answering end user queries. The user load was set at 20 users and the workload type set to TPC-C scaling factor 2. The values of Buffer cache and Large-pool were set at their default values of 24MB and 8MB respectively.

In this experiment the shared pool was varied from 80MB to 184MB and the average query response time is noted. The experiment was repeated for user load 40, 60 and 100 users. As can be seen from graphs in Fig. 3.4, the response-time changes rapidly from 165-180 msec. range to 72-88 msec. range when shared-pool changed from 80 to 88MB. However, beyond
88MB the reduction in response-time was not significant and beyond 132 MB the response time remained constant for user loads 20, 60 and 100.

For user-load 40, increasing the shared pool size may adversely affect the response time, though marginally.

### 3.3.3 Effect of Large pool on the Query Response time

The Large pool plays a vital role in executing batch jobs during off-peak hours. It is also an important subcomponent of SGA and is mainly used for supporting queries that need large memory space, processing parallel queries and backup/restore operations.

In this experiment, the values of Buffer cache and Shared pool were set at their default values of 24MB and 80 MB respectively. The workload type was set to TPC-C and scaling factor to 2. The Large pool tuning parameter was varied from 8MB to 184MB and the query response time was noted. Here again, the readings were taken for the user loads 20, 40, 60 & 100 users and the graphs were plotted. As can be seen from the graphs in figure 3.5, Large pool has moderate effect on the response-time. The response-time was found to vary from 96-113 msec. to 31-33 msec. range and also the tuning range is limited to 8MB-184MB.
3.3.4 Effect of DB_Cache on Response-time for TPC-H Workload type

In this experiment, the a workload type TPC-H(Decision Support System) with a scaling factor of 1 was chosen and the Db-cache tuning parameter was varied from 24MB to 1384MB and the response-time was noted. As can be seen from the graph in figure 3.6, the response-time shows very small change when, db_cache value was changed from 24MB to 184MB. But beyond 184MB a sharp change in response-time was observed. The response-time can be brought down to as low as 8 msecs from 90 msecs and hence, this parameter must be more rapidly varied to accomplish the desired improvement in response-time. On the other hand the shared-pool was found to have sharp change in response-time in the range 80MB to 108MB. Beyond 108 MB this parameter has no effect on the response-time.
In another experiment, the Large-pool was varied from 8MB to 1024MB and the response-time was noted. As can be seen from figure 3.8, this parameter has no effect on the query response-time.

As can be seen from figures 3.6, 3.7 and 3.8 that Db_cache has the highest impact on response-time beyond 184MB and Shared-pool has marginal effect only up to 108 MB and Large-pool has no positive impact on the response time. As can be seen from figure 3.8 the response-time deteriorates beyond 512MB of large-pool tuning parameter.
3.4.4.1 Effect of DB_Cache on Response-time for TPC-E Workload type

TPC-E represents broader form Decision Support System (DSS) type workload. As can be observed from figure 3.8.1 that db_cache tuning parameter has quite a significant impact on response time. However, it was observed that Shared_pool and Large_pool had no effect on the response-time.

![Response-time V/s Db_cache - TPC-E, SF=1](image)

Figure 3.8.1 Response-time V/s Db_cache under TPC-E workload

3.4 Interdependency between Tuning parameters

As has been pointed out in [13], the interaction between various tuning parameters needs to be established. Following experiments determine the extent of interference one tuning parameter has on the other. It is observed through these set of experiments that the tuning parameters affect each other only to a very small extent. And this effect is confined to a smaller range of tuning parameter values. Hence, the tuning parameters can be altered independently of each other while fine tuning the database system.

3.4.1 Effect of Large-pool on Db-cache

Figure 3.9 shows the effect of changing the large pool size from 8MB to 184MB on the response-time while the DB_Cache size is varied from 8MB-184MB. As can be seen from the graph, the response-time deteriorates only slightly (9.45%) as the available memory for the db_cache is reduced by the increase in large pool size. The increase in response time is observed only in first half of the graph i.e. Db_cache less than 72MB. Beyond this, there is hardly any change in response-time. So, for larger values of tuning parameters the interference between these two tuning parameters can safely be ignored.
3.4.2 Effect of Shared-pool on Db-Cache

In this experiment, the shared pool was varied from 80MB to 160MB while response-time was measured as a function of increase in Db_cache size from 8MB to 144MB.

The query response-time has been found to rise in the range 8MB-64MB of DB_cache by 14.76% and beyond 64MB value of shared pool does not have any affect the response-time and hence, for large values of db_cache tuning parameter, it can be safely assumed to be independent of the other two tuning parameters.
3.4.3 Effect of Large-pool on Shared-pool

In this experiment, the large pool is changed from 8MB to 164MB while the shared pool is varied from 80MB to 164MB.

![Figure 3.11 Effect of Large pool on Shared_pool](image)

As can be observed from the graph in figure 3.11, the response-time slightly deteriorates in the range of 80MB-124B and beyond 124MB there is no effect of changing the large pool on response-time. This result is on expected lines as purpose of large pool and shared pool are very different, hence, the interference between them would not be significant.

3.4.4 Effect of Shared-pool on Large-pool

In this experiment, Shared pool was changed from 80MB to 164MB while the large pool was varied from 8MB to 144MB and the response-time was noted.

![Figure 3.12 Effect of Shared pool on large pool](image)
As can be seen from the graph in figure 3.12, the shared pool has marginal effect on the response-time over the entire range of the large pool. Hence, the interference effect of Shared pool on Large pool can be ignored.

3.4.5 Effect of Db-cache on Large-pool

In this experiment, Db_cache was changed from 24MB to 244MB and the large pool is varied from 16MB to 182MB and the response-time was noted.

![Figure 3.13 Effect of Db_cache on Large Pool](image)

As can be seen from the graph in figure 3.13, the response-time is negatively affected by the change in Db_cache. However, the adverse effect is small and can be ignored during the tuning of the system.

3.4.6 Effect of Database cache on Shared pool

In this experiment, Db_cache was changed from 24MB to 244MB and the shared pool is varied from 80MB to 164MB and the response-time was noted.

![Figure 3.14 Effect of Db_cache on Shared Pool](image)
As can be seen from the graph in figure 3.14, the response-time is negatively affected by the change in Db_cache in the range 80MB-112MB and beyond 112MB, the effect however, is negligible. The above set of experiments were repeated for TPC-E, TPC-D and TPC-H workloads and the results were noted. The impact of DB_Cache on response-time was highest for all the workload types. The Shared pool had no effect on TPC-H workload whereas large pool had a very small impact and over a very small range (8-64MB).

3.5 Workload Characterization

The objective of these set of experiments was to establish the relationship between the Workload type and the Buffer-Hit-Ratio(BHR). Since BHR is a performance degradation indicator, which can be easily computed and used as sensor input to the tuning system. The Buffer-Hit-Ratio is an indication of what percentage of access to the requested page are served from the Buffer pool as compared to the total number of logical read operations.

The total number of logical reads includes the number of actual physical reads and the reads from the buffer cache. Higher the value of this ratio, faster would be the execution of the queries. The typical range of the BHR is in the range 55-98.21% depending on user-load, scaling factor and the workload type. The following set of experiments measure the BHR for user-load in the range of 10-100, scaling factor 1-10 and the Workload types used are TPC-C, TPC-H, TPC-D and TPC-E and establish the correlation between BHR and workload types. The outcome of these experiments helped in developing self-tuning techniques that are workload-aware [52].

3.5.1 Effect of TPC-C workload on BHR

A series of experiments were carried out to study the effect of Workload type[87], Scaling Factor and User-load on Buffer-Hit-Ratio(BHR). Table – 3.2 shows the values of BHR that were noted when the user-load was varied from 10 to 100 for TPC-C and TPC-E whereas the userload was varied from 1-10 for TPC-H and TPC-D. The scaling factor was varied from 1 to 10 for TPC-C where for other workload type scaling factor was varied from 1-2. As can be observed from the table 3.2, the BHR values range from 98.21 to 82.4 for TPC-C workload type. The user-load and scaling factor range was from 10-100 and 1-10 respectively for the workload type TPC-C.
Table – 3.2  BHR values for User load V/s Scaling Factor

<table>
<thead>
<tr>
<th>Workload Type</th>
<th>User-Load range</th>
<th>Scaling Factor range</th>
<th>BHR range</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPC-C</td>
<td>10-100</td>
<td>1-10</td>
<td>82.4-98.21</td>
</tr>
<tr>
<td>TPC-H</td>
<td>1-10</td>
<td>1-2</td>
<td>58.56-75.23</td>
</tr>
<tr>
<td>TPC-D</td>
<td>1-10</td>
<td>1-2</td>
<td>56.73-78.54</td>
</tr>
<tr>
<td>TPC-E</td>
<td>10-100</td>
<td>1-2</td>
<td>82.63-89.51</td>
</tr>
</tbody>
</table>

3.5.2 Effect of TPC-H workload on BHR

Table 3.2 shows the values of BHR that were noted when the user load was varied from 1 to 10 and the scaling factor is varied from 1 to 2 for the workload type TPC-H. TPC-H is a heavy workload type, and hence only user load in the range of 1-10 and scaling factor 1-2 was chosen. As can be observed from the following table, the BHR value ranges from 72.23 to 58.56 over the user load and scaling factor ranging from 1-10 and 1-2 respectively. It is clear from Table 3.2, that the two workloads namely TPC-C and TPC-H can easily be distinguished from the BHR values as the table values are mutually exclusive. TPC-H being a heavy workload only two scaling factors are considered.

3.5.3 Effect of TPC-E workload on BHR

Table 3.2 shows the values of BHR that were noted when the user-load is varied from 1 to 10 and the scaling factor is varied from 1 to 2 for the workload type TPC-E. As can be observed from table 3.2, that the BHR value ranges from 89.51 to 82.63 over the user and scaling factor range of 1-10 and 1-2 respectively. It is clear from Table, that the two workloads namely TPC-C and TPC-E overlap with each other over a small range and for some set of scaling factors. These results are on expected lines as TPC-C and TPC-E belong to the same workload category the OLTP type. Therefore the two workloads can’t be distinguished clearly. However, TPC-E and TPC-H workloads can easily be distinguished from the BHR values as the table values are mutually exclusive.

3.5.4 Effect of TPC-D workload on BHR

Table 3.2 shows the values of BHR that were noted when the user-load was varied from 10 to 100 and the scaling factor varied from 1 to 2. The workload type chosen was TPC-D. As can be observed from table 3.2, BHR value ranges from 78.54 to 56.73 over the user-load and scaling factor range of 10-100 and 1-2 respectively. As can also be seen from Table 3.2, that the BHR values of workloads namely TPC-H and TPC-D overlap with each other. This is
because, TPC-D represents a broader range of Decision Support System type workload as compared to TPC-H.

3.5.5 Comparison Chart:

The chart in figure 3.15 shows the minimum and maximum values of the BHR for all the Workload types. From the graph it is evident that for TPC-C workload the BHR varies in the range of 92.82 to 99.8 for the scaling factors ranging from 1 to 10.

![Comparison Chart](image)

Figure 3.15 Effect of workload type on Buffer Hit Ratio (BHR)

As can be seen from figure 3.15, the BHR values of TPC-E type workload overlap with those of TPC-C over a small range as both of these are of OLTP type workloads. On the contrary the TPC-H workload type has a very different range of BHR starting from 57.33 to 74.33 indicating a very poor value of BHR. TPC-E workload has a BHR range of 58-82% overlapping only over a very small range with the BHR range of TPC-H workload as these two belong to the same workload category. From these graphs, we can conclude that by observing the range of BHR, it is possible to identify the workload type and accordingly adjust the tuning parameters so as to meet the performance goals of self-tuning systems by adapting well to the change in workload types. To further improve the accuracy of identifying the workload-type is to use the database size in conjunction with the BHR, as the workloads differ in their database sizes.

Experimental Findings:

1. Db_Cache has the highest impact on response time across all workload types.
2. User load & scaling factors also affect the system performance significantly.
3. BHR can be used as sensor input to determine degradation in performance.
4. BHR can be used to identify current workload type.
5. Inter-tuning interference is in the range 5-15% and is limited to a small range.
6. On certain workloads for example, TPC-H, the shared pool and large pool have very small or no effect on query response-time.

3.5 Modeling the DBMS memory for performance tuning

The performance of Database Management System (DBMS) is significantly affected when the key tuning parameters are altered. Most DBMS come along with several tuning parameters that can be dynamically altered. It is therefore important to identify only a few important tuning parameters and evaluate their effect on the system performance. There have been attempts [39] [28] to develop a mathematical model[91] that describes the relationship between Buffer Hit Ratio and Db_cache size. However, the most useful analytical solution would be to relate the query response-time to tuning parameters, userload, scaling-factor and workload type. Ideally we must find an analytical solution for query response time as:

\[ Q_{rt} = f(N, WrkldType, Db_{Cache}, Sh_{pool}, Large_{pool}) \] \hspace{1cm} (3.1)

Where N is user load, WrkldType represents the current workload type and the other three variables are the dynamic tuning parameters. However, due to the highly complex and non-linear nature of the DBMS, it is not an easy task develop a mathematical model that relates all the above mentioned variables. To begin with, just to keep the analytical approach simpler, only two parameters namely user load N and Buffer size are considered in finding solution to the above problem.

In this section, a mathematical model is presented to predict the effect of one of the most important tuning parameter, namely, the buffer cache size and also the number of users on the query response time of the end user queries. The model’s output is compared with the experimental results and the results show very close match for some of the workload types. The mathematical models that have been presented in the literature [64] [39] [28] are limited in their scope for using them as they do not directly relate the response-time and tuning parameters, Userloads etc. Further, the proposed models are not validated with different workload types. Hence, there is a need for a systematic investigation into the effect of each tuning parameter on
the query performance based on fundamental principles. A formal mathematical model is presented in the following sections that describes the effect of one of the most important tuning parameters, namely the buffer cache and the user load N on the query response time. The equation is not only easy to compute but also validated with standard workload types.

3.5.1 Memory Architecture of DBMS

The memory structure of DBMS is segmented with each segment meant for a specific task. For instance, the buffer-cache holds the pages of active transactions; the redo-log-buffer holds the records necessary for recovery. Shared pool is used for storing same SQL and PL/SQL statements that are shared by user transactions. This also stores execution plans and meta data structures, privileges and roles of all users. However, the sizes of all these segments put together cannot exceed a fixed maximum called System Global Area(SGA). Apart from the system global area, the individual user processes have their Personal Global Area(PGA) and the size of this area also has significant effect on the query response-time.

3.5.2 Model Description

To understand the importance of various tuning parameters and their effect on the DBMS performance, a mathematical model is desirable. In this work, the effect of two important parameters, namely buffer cache & User load on response-time are considered. The DBMS is presented with n number of queries q1, q2, q3,……qn with each requiring varying number of disk blocks d1, d2, d3, ……dn. The query execution engine of the DBMS generates query execution plans and each plan needs memory for the queries to be executed. The standard memory model of DBMS consists of System Global Area(SGA) of fixed size and this memory is allotted for holding disk blocks for answering the queries, updating fields, rolling back results, holding common queries of different transactions.

The area of SGA that holds the disk blocks for executing the queries is called Buffer cache. The larger the size of this portion, faster will be the queries as the disk block need not be brought afresh from the disk if it is already there in the buffer cache. To better understand the role of Buffer cache in the query execution and improving throughput, there is a need to develop a model that describes the inner working of the DBMS precisely in terms of the tuning parameters, number of users etc. Figure 3.16 shows the logical view of the memory wherein the disk blocks required by each transaction is brought into the buffer cache before being processed. If page required is not found in the cache, it would be treated as a buffer miss and a
process is initiated to get it from the disk. On the contrary a buffer-hit results when, the page requested is already in the memory.

![Figure 3.16 Memory Model for Query Execution in DBMS](image)

It is evident that the buffer space being limited and as the query arrival rate increases, there would be many buffer-misses resulting in prolonged query response-times [65]. Therefore, it is desirable to have a self-tuning system, that proactively monitors the performance indicators like the query response-time or buffer-hit-ratio and initiates corrective measures automatically.

Whenever a particular performance indicator such as buffer-hit-ratio falls below a designated threshold level, the buffer-cache size will be increased to scale-up the system performance. The DBMS is presented with a variety of workload types. Important among them include OLTP(On-Line Transaction Processing), DSS(Decision Support System), Web Work Load etc. Hence there is a need to establish the impact of buffer cache for every kind of workload. As a preliminary step, only two workload types namely TPC-C and TPC-H were used to verify the model.

### 3.5.3 Mathematical Model

Let the $Q_{in\_avr}$ be the average query arrival rate and $Q_{out\_avr}$ be the average query execution rate. In a fairly loaded DBMS we can assume that $Q_{in} > Q_{out}$. In this situation, if there is a very small buffer cache then, frequent misses will result in excessive disk I/Os resulting in poor throughput and query response-times. To study the effect of size of buffer-cache on the Query response-time, the activities in the DBMS memory are modeled as under:
Let $\Delta Bf$ be the change in the buffer size as result of which the average query response time will be reduced by a factor $\Delta Qrt$. As can be seen from the graph in Figure 3.17, change in Query response-time will directly vary with the increase in Buffer size $\Delta Bf$. It can be argued that, larger the number of queries (larger user-load), smaller will be the change in query response-time $Qrt$ w.r.t the change in buffer size.

![Figure 3.17 Response-time V/s DB Cache graph](image)

The user-load $N$ (Number of active users) also contributes to the increase in the Query response-time, hence it is inversely proportional to $\Delta Qrt$. Therefore, the following relation holds.

$$\text{----------------------------- (3.2)}$$

For this derivation to hold, the other tuning parameters like shared-pool, redo-log –buffer, large-pool are assumed to be constant.

$$\text{----------------------------- (3.3)}$$

Separating the terms and integrating on both the sides we get

$$\text{----------------------------- (3.4)}$$

Integrating we get:

$$\text{----------------------------- (3.5)}$$
Using the boundary conditions

\[ \text{Qrt} = \text{Qrt\_max} \quad \text{at} \ Bf = Bfo \ (\text{Initial buf size}) \]

\[ \text{Qrt} = \text{Qrt\_min} \quad \text{when} \ Bf \rightarrow \infty \ (\text{Large buf size}) \]

And substituting them in equation (3) we get,

\[ \text{------ (3.6)} \]

It is clear from the eqn. (3.6) that the query response-time decreases exponentially with increase in buffer size and increases with increase in the number of users using the DBMS concurrently.

3.5.4 Result Analysis

The tests were carried out with OLTP and DSS workload type with a scaling factor of 2 & 1 respectively. Since, Buffer-cache provides quicker references to data by caching the frequently used pages, the response-time must decrease with increased sizes of the Buffer cache. The results obtained under the two workload types corroborate this fact.

3.5.4.1 Result validation under TPC-C (OLTP) Workload

In this experiment, TPC-C(OLTP) load with a scaling factor of 2 (Database size of 1.85GB) and a user load of 20 and 40 were used.
The experiment was carried out using Benchmark Factory load generation and analysis tool running on IBM 330X with CPU speed @2.8GHz having 8GB of RAM and 500GB of Hard-disk space. The DBMS used was Oracle 9i running on an Windows 2003 operating system. The effect of buffer size on the response time is estimated for different values of buffer size ranging from 4 MB to 56 MB.

Figure 3.18 shows the steady fall in the query response-time with increasing buffer sizes. However, beyond 36 MB and 48 MB the response-time saturates for 20 and 40 users respectively. Hence, the tuning module must alter the buffer_size only within the useful range and avoid over-tuning which will not have any effect on the response-time.

On the contrary it will only lead to poor utilization of system memory. The experiment was repeated twice for checking the consistency in the results. The experimentally obtained data corroborates the proposed memory model. As expected, it is also evident that the query response-time is larger for increased user base. As seen from figure 3.18, the graph obtained through the model and the experimentally established graph, show fairly close match. Hence, we conclude that the buffer-cache has quite a significant impact on the query response-time only over a limited range and tuning DBMS with buffer-cache beyond the threshold has no effect on the query response-time. TABLE - 3.3 shows the constants and boundary values used in Eqn. 3.6

Table - 3.3 Boundary values and other constants

<table>
<thead>
<tr>
<th>SGA</th>
<th>N</th>
<th>K</th>
<th>Qrtmax</th>
<th>Qrtmin</th>
</tr>
</thead>
<tbody>
<tr>
<td>150MB</td>
<td>20/40</td>
<td>0.0015</td>
<td>14.00/17.5</td>
<td>0.25</td>
</tr>
</tbody>
</table>

From the graph in figure 3.18, it is quite clear that the tuning range of the buffer-cache is limited and is most effective only over this limited range. With this knowledge, the DBA can now tune the DBMS far more effectively and confidently.

### 3.5.5.2 Result validation under TPC-H (DSS) Workload

In the second experiment the DBMS was presented with TPC-H kind of workload. As shown in figure 3.19, the response-time behavior is similar, except that the response time saturates at much higher value than that of in TPC-C kind of workload. This is quite on expected lines as the TPC-H load results in retrieval of large number of records.
The memory model is presented here in this section, to describes the relationship between buffer cache, User-load and the query response-time. The model presented is compared with the experimental results and found to match well with the experimental data. This model is useful in deciding the extent of tuning possible. It is clear that arbitrarily increasing the buffer cache size to a large value does not improve the performance beyond a certain point. Though, the model is validated with two workload types, it may not work well for other kind of workloads.

Further, it is observed during validation stage; that when scaling-factor was varied from 2 to 4, 6, 8, 10, the results did not show good match, indicating that the model does not account for large changes in database size. Further, this equation does not include Buffer-hit-ratio. It also did not exhibit good match when the other two tuning parameters namely, Shared-pool and Large-pool were also altered. Thus, it is evident from the above findings that, it is extremely difficult to obtain a single model that describes relationship between so many different variables. Hence, any meaningful solution to fine-tune the DBMS must be based on heuristic and evolutionary approaches that have inherent ability to handle complexity and non-linearity in the system.

**Experimental Findings**:

1. The query response-time shows significant improvement with increased buffer-cache for both the types of workloads, namely, TPC-C and TPC-H.
2. The results show close match with experimental results only for small scaling factors.

This indicates that finding an accurate analytical model that works well under all
conditions is extremely difficult. Therefore, the tuning methods proposed in the following chapters are based on Heuristic approaches that find approximate solution in a reasonable amount of time that is acceptable in most application domains.

Summary

The experiments carried out helped in deeper understanding of the impact of each tuning parameter on the response-time and also the tuning interference from each other. As observed from the graphs that some tuning parameters have greater impact on the response-time than others. Further, the tuning range of each tuning parameter over which they have significant effect on query performance is also different. The Buffer-Hit-Ratio(BHR) can be used as an approximate indicator of the workload type as it has well defined range of values for each workload-type. The identification of workload can be further improved by using database size in conjunction with BHR as the workload types differ in their database sizes. The analytical approach to develop a model that works well under all workload scenarios is extremely difficult. This suggests that analytical self-tuning approaches may not work well under all workload scenarios. This can be ascribed to the fact that, database management systems are both complex and highly non-linear. This experimental data obtained also helped in developing a knowledge-base to define fuzzy tuning range, write fuzzy rules, choose appropriate number of membership functions in the Fuzzy-based tuning approach. Also experimental results were helpful in constructing the training data-set for the Neural networks used in the Neural network based tuning approach.