

Capacity Planning Basics for OLTP

Oracle customers frequently ask us questions like: "How many users can I get on my machine X?" Of course, we have to respond with the favorite answer of performance analysts everywhere: "I don't know." There are always answers for questions like this; however, you must acquire vast amounts of information to make accurate estimates. You not only need to know the exact hardware configuration, but must also have a good idea of what users are doing on the computer and how often they are doing it.

Many RDBMS users perform On-Line Transaction Processing (OLTP), interacting directly with a computer to get responses in real time. This article will discuss a simple technique for modeling OLTP performance.

What Is a Transaction?

Everyone talks about "transactions" very casually, yet different people and different software environments use different definitions for the word. You must have a clear idea of what their transactions are before you can model or evaluate the performance of an OLTP system.

In a block-mode OLTP environment such as the MVS Customer Information Control System (CICS), a transaction typically begins when the user presses the ENTER key (or a Program Function key) on the terminal, and the transaction ends when the screen is updated and the keyboard unlocked.

In other environments, such as a desktop workstation, transactions must be carefully defined in terms of a user task. For instance, you may want to define a transaction as "successfully entering all of the fields in a SQL*Forms block," or "adding a new account to the database." This type of logical transaction definition is often useful when you are interviewing users, because your idea of a transaction maps to a real-world task that the user can relate to and help quantify.

When planning for a client-server environment, you generally need to define transactions in user terms, then figure out what resources are consumed on the server to complete each user transaction. You can then apply the techniques in this article to model server requirements for the application. Client systems are usually dedicated to one user, so there are no capacity issues to speak of (except perhaps: "does this machine run the client application adequately?").

Transaction Rate

The key to understanding OLTP capacity and performance modeling is the concept of a "transaction rate." Each OLTP transaction requires a certain amount of computer resources; for example, a quarter-second of CPU time and five I/O operations. You can calculate the resources needed for a specific transaction type if you know how much computer resource each transaction needs and how often the transactions arrive. You can also calculate the resource requirement for an entire OLTP application if you have these figures for each type of transaction in the system.

It's difficult to estimate the transaction rate accurately, since it depends on how many users are on the system, how much time each user spends "thinking" between transactions, and how long each transaction runs. These last two numbers are called "think time" and "response time," respectively. For example, if a user "thinks" for an average of 55 seconds between transactions, and transaction response time is 5 seconds, then the user will enter one transaction per minute, or about 0.017 transactions per second. If 100 users average 55-second think time and 5-second response time, then the transaction rate will be 100 per minute, or 1.7 per second. This simple equation enables you to calculate the transaction rate from the number of users, their average think time in seconds, and the average response time in seconds:

$$\text{transactions per second} = \text{users} / (\text{think time} + \text{response time})$$

For simple data-entry transactions, think time can be as low as 20 seconds, although think time more commonly averages 100 to 600 seconds. Since the accuracy of the results depends directly on the accuracy of think time estimates, you should interview and/or observe the users carefully to come up with realistic think time estimates. If you believe users when they say they enter five transactions per minute all day long, you will probably overestimate the resources needed for the application system by a factor of three or five. On the other hand, use peak transaction rates (see Figure 1) to plan for capacity. Using the average transaction rate for the entire day to compute resource usage results in poor system performance during the peak interval, where the transaction rate may be twice the daily average.

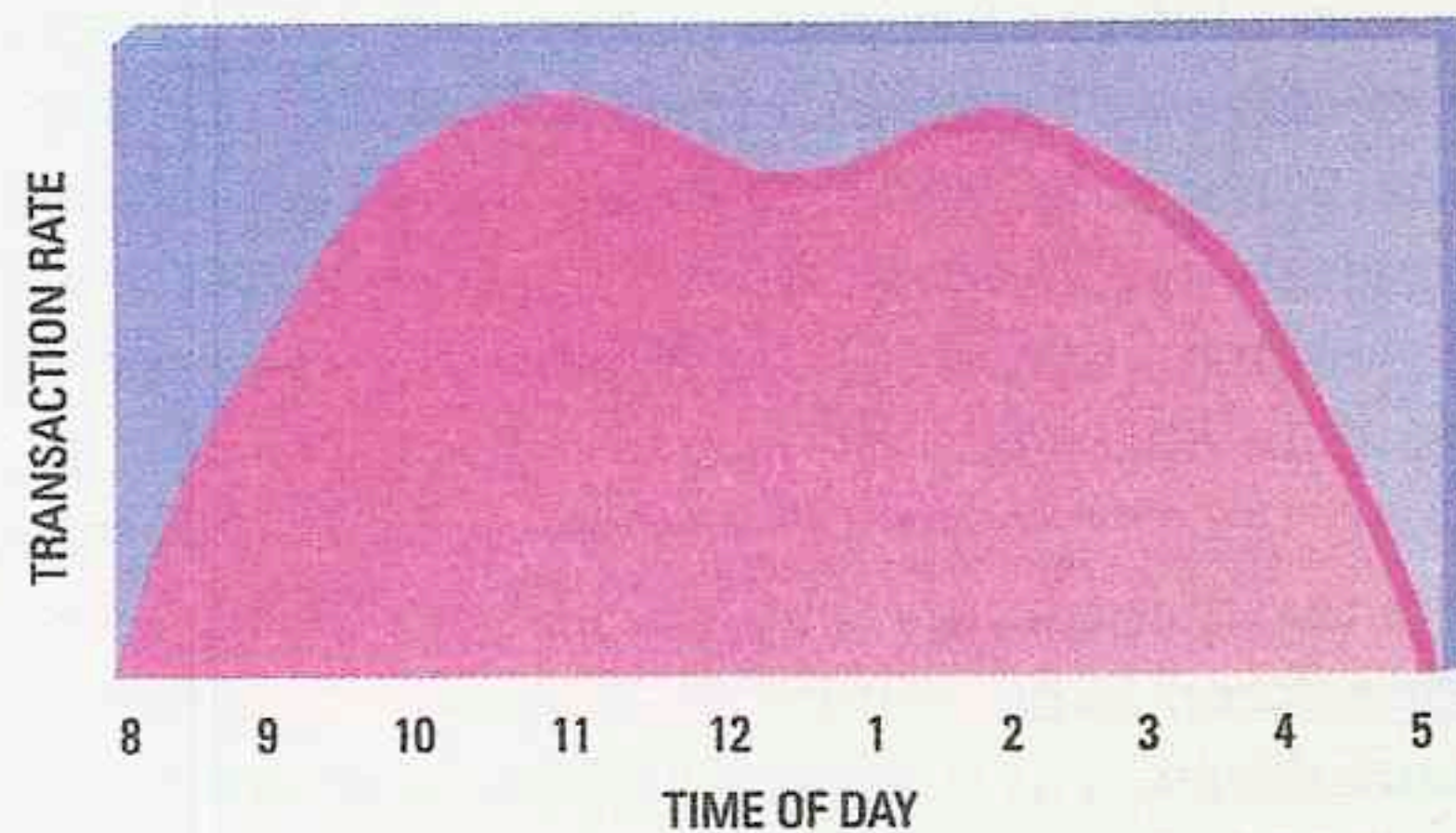


FIGURE 1 PLAN FOR THE PEAK TRANSACTION RATE, WHICH OCCURS AT 11:00AM AND 2:00PM

Resource Per Transaction

The other half of estimating OLTP resource usage is figuring out the resources needed for each transaction. With a good idea of resource requirements, you can multiply the resource usage per transaction by the transaction rate to get total application resource usage.

You may want to estimate I/O and/or memory resources per transaction as well as CPU. However, CPU is usually the most expensive resource (especially on mainframe systems), so the remainder of this article will focus on CPU modeling. Just remember that the same basic techniques used for CPU modeling can be used to project I/O and virtual memory requirements for an OLTP application as well.

CPU resources are best expressed as "path length," or the number of processor instructions required to complete a transaction. CPU seconds are not a useful measure, since the number of CPU seconds used depends on how fast the processors in the system are. It is easy to convert path length into CPU seconds and vice-versa:

$$\text{path length} = \text{CPU seconds} \times \text{processor speed}$$

So a transaction that requires 0.1 CPU second on a 30 MIPS processor will have a path length of $0.1 \times 30,000,000$ or 3 million instructions. Note: use the speed of an individual processor in this calculation, because each transaction only runs on one processor at a time. You cannot employ the MIPS rating for the entire system if it has multiple CPUs.

"Millions of instructions per transaction" can also be expressed as "millions of instructions per second, per transactions per second," or "MIPS per TPS." The 3-million-instruction path length computed above can also be expressed as 3 MIPS per TPS, which is a very convenient number.

Now simply multiply the path length number by the number of transactions per second to figure out the CPU resources needed for a particular transaction. In prior

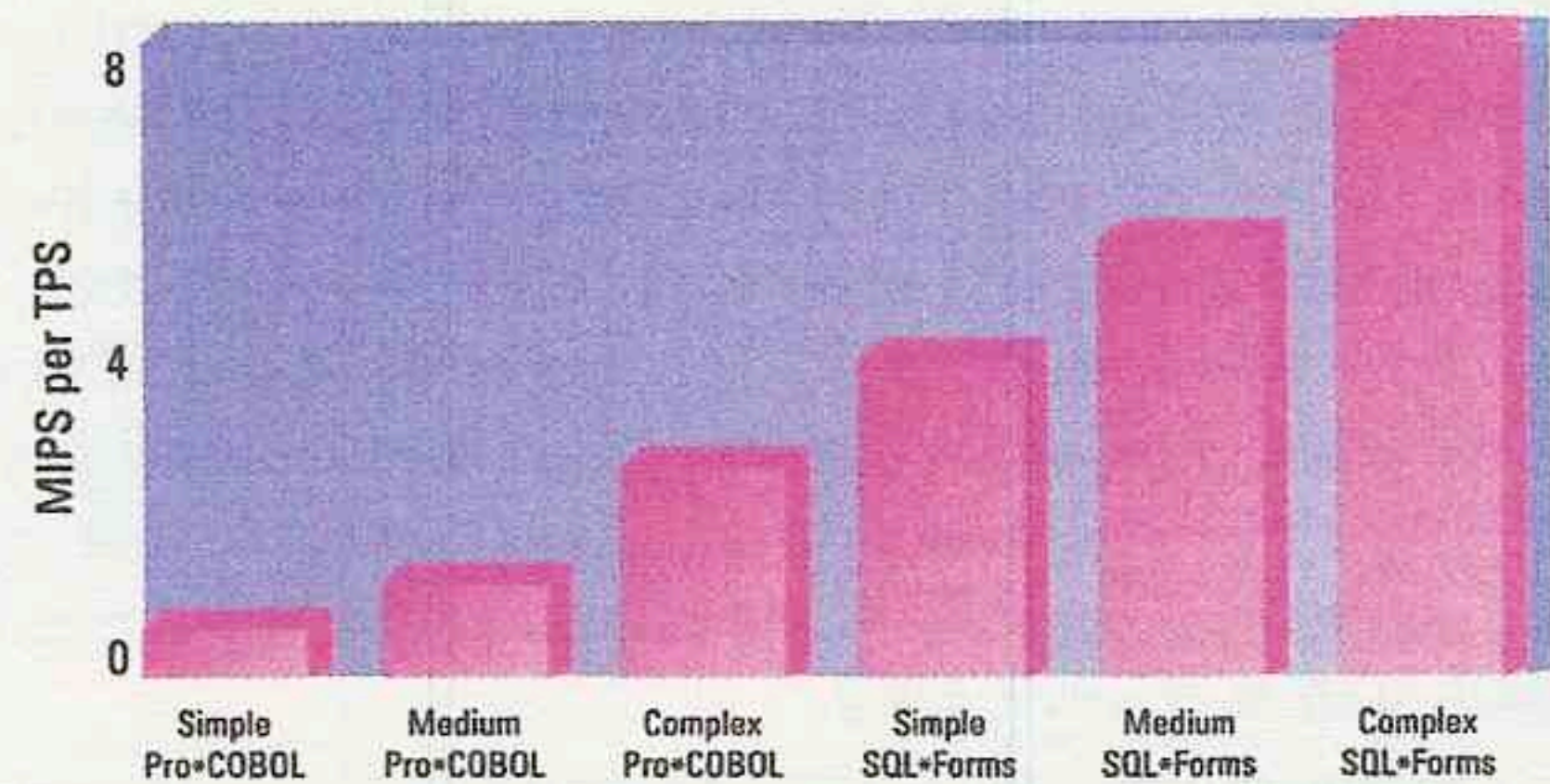


FIGURE 2 TYPICAL PATH LENGTHS FOR ORACLE OLTP APPLICATIONS UNDER CICS

examples, 100 users were generating 1.7 TPS, and each transaction required 3 MIPS per TPS:

$$\text{MIPS} = 1.7 \times 3 = 5.1$$

So the simple single-transaction example might require an average of 5.1 MIPS of processing capacity. Repeat this process for all transactions in an application, then add the numbers together to get an estimate of the total CPU resource the application will require.

Reengineering or converting to Oracle ?

DBAPort - Offers fully configurable, automated DDL schema and data conversion between multiple DBMSs including integrity, index's, views, & privileges. Easily reorganize, add/delete/rename tables/columns from source to target DBMS.

Need to implement Referential Integrity for Oracle 6 or 7?

DBATool - Produces OCI functions for Oracle 6 and triggers for Oracle 7 providing Referential Integrity for unique, unique clustered and normal indexes with common, primary and foreign keys using on update, on delete, cascade, restrict, set null rules defined in the CREATE TABLE stmts and much more.

Need a more sophisticated preprocessor?

DBAPrep - Open preprocessor and precompiler that provides the most advanced Embedded SQL tools available. Fully supports standard and extended features of Oracle. Supports Structures, Pointers, C definable host variable data lengths. Automatically allocates memory in variable/column names, fills in SQLDA sqldata, sqlind, null indicator fields and much more.

N Systems tools support the following DBMS platforms: Oracle 6 & 7, DB/2, SYBASE/SQL Server, SQLBase, DBM ES/EE, XDB, Informix, Q+E Database Library, Microsoft ODBC and more. Available for DOS, Windows, OS/2, UNIX variants, VMS, Mac and others.



Insist On The Best

Call: (206) 450-0815

N Systems

2300 103rd Ave. NE, #A
Bellevue, WA 98004

FAX: 206-880-4590

Estimating Resource Requirements

This planning approach is only as good as the numbers put into it, or "garbage in, garbage out," as the saying goes. All of the math in this process is linear, which means that a 10 percent error in any of the estimated numbers will provide a 10 percent error in calculated results. Good guessers, or those who are just plain lucky, will have some estimates that are high and some that are low, so that the errors cancel each other out and the end result will be fairly useful.

Optimistic estimates can be more than 50 percent low, causing users to buy a much smaller machine than needed and making end users very angry as response times increase to over a minute. Pessimistic estimates, on the other hand, may result in users spending thousands or millions of dollars too much for hardware, making you a primary target for the layoff a company may perform to recoup the excess hardware costs.

Figure 2 presents some representative path lengths for both Pro*COBOL and SQL*Forms transactions under CICS. A "simple" transaction, such as a typical data entry application, might select a row from a single table, then update a dozen columns in the row when the user commits the transaction. "Medium" transactions typi-

cally issue several SQL statements, may include a two-way join, update a few rows in one table, and/or update several tables. "Complex" transactions join up to four tables, and might update dozens or hundreds of rows in several tables. Of course, your mileage will vary, and the path lengths in Figure 2 are meaningless for platforms other than MVS CICS.

Note that these representative transactions do not allow for applications that join more than four tables, update very large numbers of rows, or issue DDL statements such as "CREATE TABLE," and generally do not allow for complex SQL processing. The reason for this limitation is simply that good OLTP application design does not allow for this type of operation to occur.

An on-line transaction, by definition, should get good response time. Well-designed OLTP applications should be engineered to run efficiently and quickly. Complex SQL statements can generally be eliminated from OLTP applications by denormalizing the database.

The Bottom Line

There are dozens of other issues to address in any capacity planning effort, and the information in this article is barely enough to get you started. Fortunately, common sense and an organized approach will help you solve most capacity planning problems. The bottom line for planning for future hardware purchases is knowing what the system's users are doing—so don't hesitate to ask them. □

Tom Childers is a Senior Technical Staff member in Oracle's IBM Products division.

Oracle Link To MS Excel and Lotus 1-2-3

Second Wind™ MS-Windows, Macintosh & Presentation Manager

- Access Oracle RDBMS directly from Excel, Lotus 1-2-3 for Windows and other MS-Windows applications at high speed
- Issue SQL command from Excel/Lotus 1-2-3 menu and built-in Query Builder
- Execute multiple SQL statements simultaneously using SQL-Script
- Use Excel or Lotus 1-2-3 macro library for application development
- Link to any Windows applications supporting DDE

ANJU TECHNOLOGIES
4962 El Camino Real, Los Altos, CA 94022
Tel: (415) 962-9670/Fax: (415) 962-9615

ORACLE MAGAZINE WANTS TO HEAR FROM YOU

Are there technical issues and concerns on your platform you would like to see covered in ORACLE Magazine? If you have topic requests or ideas for articles on how users can enhance the performance of the ORACLE RDBMS or other Oracle software on a specific computing platform, then please let us know. We'll pick out the most frequently requested topics and ask Oracle experts to contribute articles for future issues of ORACLE Magazine.

Please send your requests, ideas, or papers you'd like to submit for possible publication (hardcopy and softcopy, in IBM PC or Apple Macintosh format) to:

**Editor, Operating Environments
ORACLE Magazine
500 Oracle Parkway, Box 659510
Redwood Shores, CA 94065**