Sun ORACLE | SUN SPARC ENTERPRISE SERVERS

An Oracle White Paper
September 2010

Optimizing Single Instance Oracle Databases on Oracle's T-series Servers

ORACLE

Introduction

This document is intended for Oracle Database Administrators (DBAs) who want to optimize single instance Oracle Database performance on Oracle's T-series servers.

Oracle's T-series servers are based on Oracle's CMT processors which combine chip multiprocessing (CMP) and hardware multithreading (MT) with an efficient instruction pipeline to enable chip multithreading. The resulting processor design provides multiple physical instruction execution pipelines and several active thread contexts per pipeline.

The Oracle Database scales well on T-series servers. For instance, Oracle Database 11g with Oracle Real Application Clusters demonstrated excellent horizontal scalability across 12 Sun SPARC Enterprise T5440 servers on Oracle Solaris 10 running the industry-standard TPC-C workload. Middleware and business applications deployed on Oracle database instances on T-series servers have consistently performed well . For instance, Siebel CRM 8.0 application, an OLTP application, scaled linearly from 5000 concurrent users on a single Sun Fire T5220 system to 10,000 concurrent users on two such systems running Oracle database 10gR2. Similarly, Oracle Weblogic Server deployed on Oracle Database 11g on a Sun SPARC Enterprise T5440 server delivered world record numbers. Oracle BI EE, a component of Oracle Fusion Middleware, workload demonstrated linear scalability and world record performance across two Sun SPARC Enterprise T5440 servers running on Oracle Database 11g.

However, certain Oracle Database operations that are single threaded do not currently perform well on T-series servers as the current generation of T-series servers are designed for throughput rather than being focused on the performance of a single thread. Oracle Database provides functionality to exploit concurrency and also to perform these operations in parallel. The purpose of this whitepaper is to explain how the concurrency and the parallel features of Oracle Database can be exploited to improve the throughput of the database running on T-series servers. It also introduces the virtualization options available on T-series servers which can further increase concurrency and overall system utilization.

This whitepaper assumes that the reader is familiar with deploying Oracle Database on T-series servers. For those not familiar with Oracle Database deployment on Oracle's T-series servers, refer to the reference section for a list of collateral.

Oracle's T-series Systems

Oracle's currently shipping Chip Multithreading processors are the UltraSPARC T2 and UltraSPARC T2 plus processor. Understanding the architecture of Chip Multithreading (CMT) processors enables one to understand that the T-series servers, which are based on Chip Multithreading processors, are designed for throughput and their strengths can be exploited by enabling parallelism in the applications.

Oracle's Chip Multithreading (CMT) processors

There is a huge disparity between processor speeds and memory access rates. While processor speeds continue to double every two years, memory speeds have typically doubled only every six years. As a result, memory latency now dominates real-world application performance, erasing even very impressive gains in clock rates. Creating multicore processors by simply replicating existing single threaded cores does not significantly improve aggregrate chip performance as it ignores key performance issues such as memory latency and hardware thread context switching.

First introduced with the UltraSPARC T1 processor, chip multithreading takes advantage of multicore advances, but adds a critical capability-the ability to scale with threads rather than frequency. Unlike traditional single-threaded processors and even most current multicore (CMP) processors, hardware multithreaded processor cores allow rapid switching between active threads as other threads stall for memory. A CMT processor differentiates itself from other multicore – and even from other multicore and multi-threaded - processors by providing a large number of hardware threads per core and efficiently managing hardware thread context switching. Figure 1 illustrates the difference between CMP, fine-grained hardware multithreading (FG-MT) and chip multithreading. The key to this approach is that each core in a CMT processor is designed to switch between multiple threads on each clock cycle. As a result, the processor's execution pipeline remains active doing real useful work, even as memory operations for stalled threads continue in parallel.

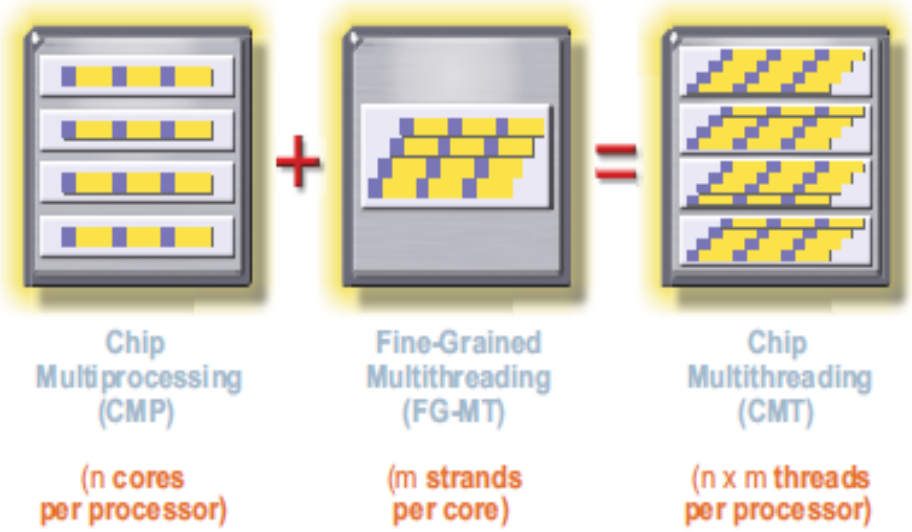

Figure 1: Chip multithreading combines CMP and fine-grained hardware multithreading

The UltraSPARC T2 processor family is the second generation processor from Sun based on chip multithreading.

UltraSPARC T2 Processor Family

The UltraSPARC T2 processor family consists of two processors, UltraSPARC T2 processor and UltraSPARC T2 Plus processor. They are the industry's first systems on a chip (SoC) -- integrating computing, security, and I/O on to a single chip.

Both UltraSPARC T2 and UltraSPARC T2 Plus processor have the same core design. They provide up to eight cores, with each core able to switch between up to eight threads (64 threads per processor). In addition, each core provides two integer execution units, so that a single UltraSPARC core is capable of executing two threads at a time.

Figure 2 provides a simplified high-level illustration of the thread model supported by an eight-core UltraSPARC T2 processor family.

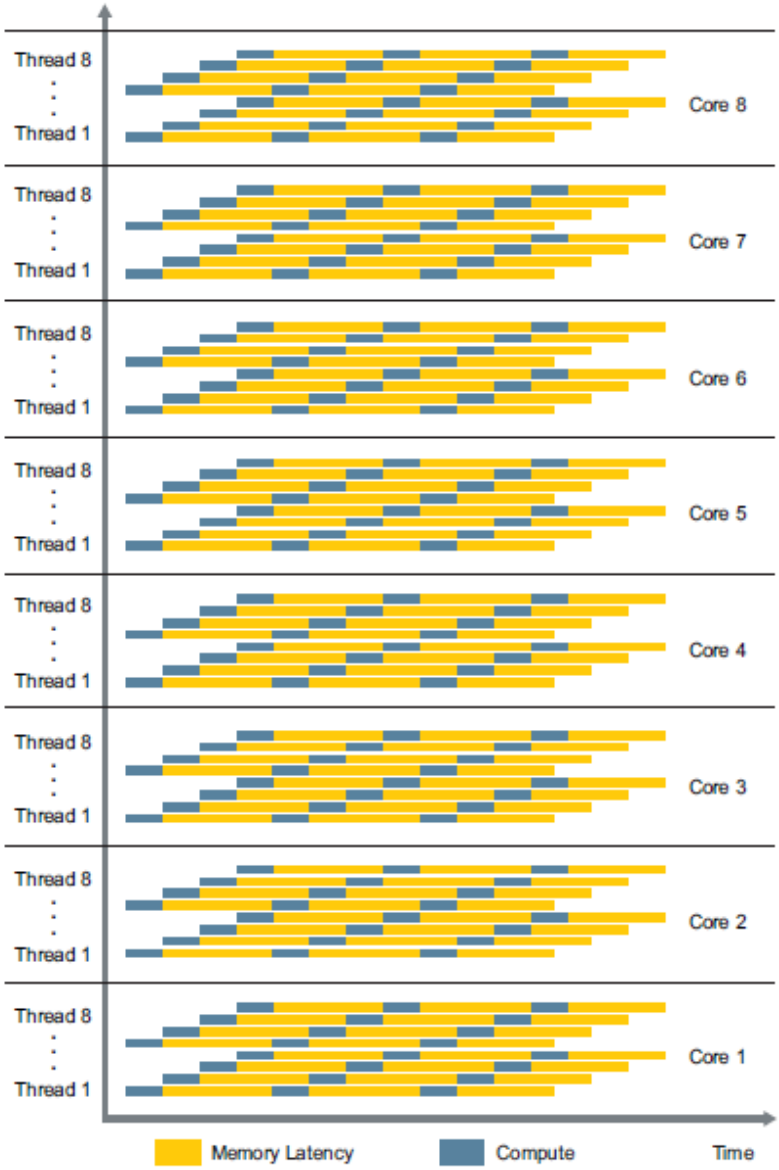

Figure 2: An eight core UltraSPARC T2 or UltraSPARC T2 Plus processor supports up to 64 threads, with up to 2 threads running in each core simultaneously.

In addition to supporting 64 threads per processor, the UltraSPARC T2 processor family also provides

- On-chip Level-1 and Level-2 caches
- Per-core floating point capabilities
- Per-core cryptographic acceleration
- On-chip PCI Express Interface

Through SoC design, the UltraSPARC T2 processor family significantly enhances the general purpose nature of the CPU – building in eight floating point units (1 per core). Enhanced floating point capabilities open the UltraSPARC T2 processor family to the world of compute-intensive applications as well as traditionally CMT-friendly datacenter throughput applications. Low-cost security and cryptograpic acceleration is provided by the on-chip, per-core cryptographic accelerators. In addition, the ability to move data in and out of the UltraSPARC T2 processor family is significantly aided by an integrated on-chip PCI Express interface andmemory management units.

The UltraSPARC T2 Plus processor is an enhanced UltraSPARC T2 processor supporting multi-socket implementations. The UltraSPARC T2 plus processor has on-chip coherency logic and links to support two and four socket implementations.

Table 1 provides a comparison between the UltraSPARC T2 and UltraSPARC T2 Plus processors

TABLE 1. ULTRASPARC T2 AND ULTRASPARC T2 PLUS FEATURES

FEATURES	ULTRASPARC T2 PROCESSOR	ULTRASPARC T2 PLUS PROCESSOR
Cores per processor	Up to 8	Up to 8
Threads per core	8	8
Threads per processor	64	64
Sockets supported	1	2 or 4
Memory	4 memory controllers, up to 16 FB-DIMMs	2 memory controllers, up to 16 FB-DIMMs
Floating point	1 FPU per core 8 FPUs per chip	1 FPU per core 8 FPUs per chip
Cryptography	Stream processing unit per core, support for the 10 most popular ciphers	Stream processing unit per cores, support for the 10 most popular ciphers
Integer resources	2 integer execution units per core	2 integer execution units per core
Additional on-chip resources	Dual 10 Gb Ethernet interfaces, PCI Express interface (x8)	PCI Express interface (x8), Coherency logic and links (6.4 GB/second)

Oracle's T-series Servers

T-series Servers with UltraSPARC T2 processors

The UltraSPARC T2 processors are available with four, six, or eight cores, each of which can handle eight concurrent threads for a total of 64 threads per processor.

The Sun SPARC Enterprise T5120 and T5220 servers are based on the UltraSPARCT2 processor. The Sun SPARC Enterprise T5120 server supports up to four internal disk drives and offers one x8 PCI Express and two x4 PCI Express or XAUI combo slots. The Sun SPARC Enterprise T5220 server supports up to eight internal disk drives, two x8 and two x4 PCI Express slots and two x4 PCI Express or XAUI combo slots. Both servers support up to 64 GB of main memory.

The following table summarizes the features of Sun SPARC Enterprise T5120 and T5220 servers.

TABLE 2. SUN SPARC ENTERPRISE T5120 AND T5220 SERVER FEATURES

FEATURE	SUN SPARC ENTERPRISE T5120 SERVER	SUN SPARC ENTERPRISE T5220 SERVER
CPUs	Four-, six-, or eight-core 1.2 GHz or 1.4 GHz UltraSPARC T2 processor	Four-, six-, or eight-core 1.2 GHz or eight-core 1.4 GHz UltraSPARC T2 processor
Threads	Up to 64	Up to 64
Memory Capacity	Up to 64 GB	Up to 64 GB
Maximum internal disk drive	Up to four 73 GB or 146 GB 2.5 inch SAS hard drives,RAID 0/1	Up to eight 73 GB or 146 GB 2.5 inch SAS hard drives ,RAID 0/1
PCI	One eight lane PCI Express , Two four lane PCI Express or XAUI combo slots	Two eight lane PCI Express , Two four lane PCI Express •,Two four lane PCI Express or XAUI combo slots
Form Factor	1 rack unit (1U)	2 rack units (2U)

T-series Servers based on UltraSPARC T2 Plus processors

The UltraSPARC T2 Plus processor is available with four, six, or eight cores, each of which can handle either 8 threads or up to 64 threads per processor. Sun offers a set of rack-mount server and blade products based on each processor. The Sun SPARC Enterprise T5240 and T5440 servers are based on the UltraSPARC T2 plus processor. The Sun SPARC Enterprise T5240 server is a two-socket system, which supports up to 16 internal disk drives and offers four dedicated 8-lane PCI Express expansion slot and two 8-lane PCI Express or XAUI expansion slots. The maximum supported memory for the Sun SPARC Enterprise T5240 server is 256GB. The Sun SPARC Enterprise T5440 server is a 1–4 processor system, which supports up to four internal disk drives, four dedicated 8-lane PCI Express expansion slots with 8- lane connectors, two 8-lane PCI Express slots with 16-lane connectors and two

additional 8-lane PCI Express or 10 Gigabit Ethernet expansion slots. The maximum supported memory for the Sun SPARC Enterprise T5440 server is 512GB.

Table 3 compares the features of Sun SPARC Enterprise T5140, T5240 and T5440 server features

TABLE 3. SUN SPARC ENTERPRISE T5140,T5240 AND T5440 SERVER FEATURES

FEATURE	SUN SPARC ENTERPRISE T5140 SERVER	SUN SPARC ENTERPRISE T5240 SERVER	SUN SPARC ENTERPRISE T5440 SERVER
CPUs	Dual four-, six-, or eight core 1.2 GHz or eightcore1.4 GHz UltraSPARC T2 Plus processors	Dual four-, six-, or eight core1.2 GHz or eightcore 1.4 GHz or 1.6 GHz UltraSPARC T2 Plus processors	One to four eight-core 1.2 GHz, 1.4 GHz, or 1.6 GHz UltraSPARC T2 Plus processors
Threads	Up to 128	Up to 128	Up to 256
Memory Capacity	Up to 128 GB	Up to 256 GB	Up to 512 GB
Maximum internal disk drive	Up to eight SFF 2.5-inch SAS 73, 146, or 300 GB disk drives, or 32 GB SSDs, RAID 0/1	Up to sixteen SFF 2.5- inch SAS 73, 146, or 300 GB disk drives, or 32 GB SSDs, RAID 0/1	Up to four SFF 2.5-inch SAS 73 or 146 GB disk drives, or 32 GB SSDs, RAID 0/1
PCI	One x8 PCI Express slot Two x8 PCI Express or XAUI combo slotsa	Four x8 PCI Express slots Two x8 PCI Express or XAUI combo slotsa	Six x8 PCI Express slots, Two x8 PCI Express or XAUI combo slotsa
Form Factor	1 rack unit (1U)	2 rack units (2U)	4 rack units (4U)

Parallel Database Operations

Oracle Database deployed on T-series servers can be optimized by exploiting the parallel features offered by Oracle Database. SQL parallel execution was first introduced in Oracle more than a decade ago and has been enriched and improved since. The Oracle Database Operations that can be executed in parallel includes:

- SQL loader and SQL-based data loads
- SQL Queries
- RMAN backups
- Index Creation
- Statistics

This section provides a very brief overview of how SQL parallel execution works in the Oracle database. Refer to references section for a detailed discussion.

By default the Oracle Database is configured to support parallel execution out-of-the-box. The most relevant database initialization parameters are:

- parallel_max_servers: the maximum number of parallel servers that can be started by the database instance. In order to execute an operation in parallel, parallel servers must be available (i.e. not in use by another parallel operation). By default the value for *parallel_max_servers* is set to 10 times the value of *cpu_count* parameter.
- parallel_min_servers: the minimum number of parallel servers that are always started when the database instance is running. *parallel_min_servers* enables you to avoid any delay in the execution of a parallel operation for the startup of parallel servers.

You can verify that parallel execution is enabled for your database instance by viewing the value of *parallel_max_servers* parameter. A non-zero value indicates that parallel execution is enabled for your database instance

There are several ways to enable a query to execute in parallel. You can enable the table for parallel execution thereby making all operations accessing those tables to execute in parallel. You can also use a parallel hint to a SQL query to run that particular query in parallel. Or you can alter the session to run parallel query enabling all queries in that session to run in parallel.

Once you enable parallelism in the query, you can set the degree of parallelism(DOP) by one of the following ways :

- Default DOP : Oracle database supports default parallel capacities where Oracle chooses the DOP. By default, Oracle sets Default DOP to be twice the value of the *cpu_count* parameter.
- Fixed DOP : Unlike the DEFAULT parallelism, a specific DOP can be requested from the Oracle database. For example, you can set a fixed DOP at a table or index level by issuing the alter table command with parallel option. The following command sets the degree of parallelism to 4 for customer table.
    - alter table customers parallel 4 ;

Oracle 11gR2 introduced two new features in parallel query execution to further improve performance and scalability.

- Auto Degree of Parallelism and Querying

When activated, Oracle determines the optimal degree of parallelism (DOP) for any given SQL operation based on the size of the objects, the complexity of a statement, and the existing hardware resources.

Auto DOP is controlled by the initialization parameter `PARALLEL_DEGREE_POLICY`. This parameter can take the following values:

- MANUAL: The default value for `PARALLEL_DEGREE_POLICY` is `MANUAL`, which means Auto DOP is not active.
- LIMITED: When set to `LIMITED`, Auto DOP is only applied to statements that access tables or indexes decorated with the `PARALLEL` clause but without an explicit DOP. Tables and indexes that have a specific or fixed DOP specified will use that specified DOP.
- AUTO: When set to AUTO, Auto DOP will be applied to ALL SQL statements executed on the system, regardless if they access an object decorated with a `PARALLEL` clause or not. Given this behavior, it is possible for more SQL statement to execute in parallel on Oracle Database 11g Release 2 then did on previous releases

In-Memory Parallel Execution

Until the Oracle 11gR2 release, the parallel server processes by-passed the buffer cache and read the necessary data directly from disk. However, In-Memory Parallel Execution (In-Memory PX) tries to take advantage of larger buffer caches wherever possible. Hence, it improves query performance by minimizing or even completely eliminating the physical I/O needed for a parallel operation. In-memory parallel execution is enabled by setting the initialization parameter *parallel_degree_policy* to auto.

Oracle Database provides parameters that define the threshold of parallel operations. The default values for these parameters are derived from the *cpu_count* parameter. By default, Oracle sets the *cpu_count* parameter to the number of CPUs in the system, or current processor set (if one is in use).

A T-series server exposes each hardware thread as a CPU to the system. For example a four socket Sun Enterprise T5440 system has 256 hardware threads each of which is exposed as a CPU to the system. Hence, the *cpu_count* parameter is set to 256 on Sun Enterprise T5440 server. Certain parallel execution related parameters, which are derived from the *cpu_count* parameter, have unexpectedly high default values.

Table 4 shows the default values of parameters related to parallel query execution that are derived from the *cpu_count* parameter on a four socket Sun Enterprise T5440 system.

TABLE 4. DEFAULT VALUES OF CERTAIN ORACLE PARAMETERS IN SUN ENTERPRISE T5440 SERVER

PARAMETER_NAME	PARAMETER DESCRIPTION	FORMULA TO DERIVE DEFAULT VALUE	DEFAULT VALUE
Cpu_count	No. Of CPUs	No. Of CPUs	256
Parallel_max_servers	The maximum number of parallel servers that can be started by an instance	10*cpu_count	2560
Parallel_servers_target	Number of parallel server processes allowed to run parallel statements before statement queuing will be used	4*CPU_COUNT*parallel_threads_per_cpu	2048

Parallel_degree_limit (when set to cpu)	Limit the degree of parallelism used to ensure parallel servers processes do not flood the system	Cpu_count * parallel_threads_per_cpu	512

As indicated in table 4, statements can be executed with 512 degree of parallelism on a 4 socket Sun SPARC T5440 server by default. Furthermore, 2048 parallel server processes are allowed to run parallel statements before statement queuing functionality of Oracle Database 11gR2 kicks in.

A recommended best practice is to set parameters that define the threshold of parallel server processes to reasonable limits. The following section describes the effect of tuning parameters that define the threshold of parallel server processes on a single socket Sun SPARC Enterprise T5220 server.

Parallel Index Creation

The following `create index` statement was executed from an online transaction processing workload on the `customer` table of size 2 GB on an 8 core Sun SPARC Enterprise T5220 server.

```
create unique index icust2 on cust  (c_last, c_w_id, c_d_id, c_first, c_id ) pctfree 1 initrans 3
storage ( buffer_pool default ) compute statistics tablespace icust2_0;
```

The `create index` query was executed by letting Oracle 11gR2 decide the optimal degree of parallelism. This was achieved by setting `parallel_degree_policy` parameter to `auto`. Two runs were done after setting `parallel_degree_policy` parameter to auto: one with default `parallel_servers_target` parameter and another with setting the `parallel_servers_target` parameter to 32.

Table 5 summarizes the execution time for running *create index* statement by enabling Oracle Database 11gR2 automatic degree of parallelism

TABLE 5. EFFECT OF ENABLING ORACLE DATABASE 11GR2 AUTOMATIC DOP

PARALLEL_SERVERS_TARGET	PARALLEL_MAX_SERVERS	EXECUTION TIME	%CHANGE	CPU UTILIZATION (IDLE/USR/SYS)
Default (256)	Default(640)	00:07:15.88	-	81/10/9
32	64	00:02:44.27	63%	75/15/10

As indicated in table 5, setting the `parallel_max_servers` parameter to 64, the `parallel_servers_target` parameter to 32 and enabling optimal DOP parameter feature reduced the execution time by over 60%.

The AWR report indicated I/O as a bottleneck when the run was executed with default value of `parallel_servers_target` parameter. Over 70% of the total call time was related to I/O wait events.

Snippet of Top 5 Timed Events:

Event	Waits	Time (s)	Avg wait(ms)	%Total	Call Time
db file sequential read	1,701,702		7,639	4	57.8
kfk: async disk IO	40,147		1,889	47	14.3

The tablespace I/O stats section of AWR report indicated that the large number of producers reading from the *CUST_0* tablespace generated a read bottleneck on this tablespace.

Snippet of *CUST_0* tablespace read statistics :

	Avg Reads	Avg Reads/s	Waits Wt(ms)
CUST_0	1,692,891	2,879	25.0

Reducing the value of the `parallel_servers_target` parameter to 32, completely eliminated the read bottleneck on the `CUST_0` tablespace.

Along with AWR report, Solaris tools including `vmstat`, `mpstat` and `corestat` can be used to understand cpu utilization on T-series server. Refer to Appendix 1 for additional details.

Parallel Create table as Select

The `create table … as select` statement (CTAS) is a powerful tool for manipulating large sets of data. Many data transformations can be expressed in standard SQL, and CTAS provides a mechanism for efficiently executing a SQL query and storing the results of that query in a new database table.

In a data warehouse environment, CTAS is typically run in parallel using `nologging` mode for best performance.

The `create table … as select` statement contains two parts: a `create` part (DDL) and a `select` part (query). Oracle Database can parallelize both parts of the statement. The `create` part is parallelized if *one* of the following is true:

- A `parallel` clause is included in the `create table … as select` statement
- An `alter session force parallel ddl` statement is specified

The query part is parallelized if *all* of the following are true:

The query includes a parallel hint specification (`parallel` or `parallel_index`) or the create part includes the parallel clause or the schema objects referred to in the query have a parallel declaration associated with them.

- At least one of the tables specified in the query requires either a full table scan *or* an index range scan spanning multiple partitions.

- If you parallelize the creation of a table, that table then has a parallel declaration (the `parallel` clause) associated with it. Any subsequent DML or queries on the table, for which parallelization is possible, will attempt to use parallel execution.

In a data warehouse environment, CTAS is typically run in parallel using `nologging` mode for best performance.

A CTAS statement from decision support system workload was executed in the following scenario:

- Running the CTAS statement with default DOP
- Setting the parallel_max_servers parameter to 64 with default DOP
- Running Oracle Database 11gR2 decide the optimal degree of parallelism

Table 5 summarizes the execution time for running `CTAS` statement in the above scenario.

TABLE 5. RUNNING CTAS IN DIFFERENT SCENARIO

11GR2 AUTO DOP	PARALLEL_MAX_SERVERS	EXECUTION TIME	%CHANGE	CPU UTILIZATION (IDLE/USR/SYS)
Not Enabled	Default (640)	00:01:31	-	97/3/0
Not Enabled	64	00:01:18	14%	96/4/0
Enabled	64	00:00:50	45%	95/5/0

As indicated in table 5, reducing the parallel_max_servers parameter to 64 and enabling auto DOP feature of Oracle Database 11gR2 reduced the execution time by over 40%.

Parallel Recovery Manager (RMAN) Backup

Recovery Manager (RMAN) is an Oracle utility that performs backup and recovery tasks on your databases.

You can configure RMAN to specify backup devices, type of backup, location for storing backup and others. RMAN also allows you to configure connection to the backup device, referred to as RMAN channel. However, all these settings are optional; the RMAN backup and recovery environment comes preconfigured for each target database with a default value for all these settings. By default, RMAN takes backups on disk, opens a single channel to disk and stores the backup at *$ORACLE_HOME/dbs* directory.

Since RMAN allocates just a single disk channel for backup in default configuration, RMAN cannot read or write in parallel by default. This behavior can be changed by opening multiple channels for the required device type. The *configure device* command with *parallelism* clause is used to specify the number

of channels required for particular device type. For example, the following command opens 6 channels for tape device type

- Configure device type tape parallelism 6

The following command opens 8 channels for disk device type:

- Configure device type disk parallelism 8.

However, each channel can backup only one datafile at a time. For example, if you have 10 datafiles to backup and you create 20 channels, RMAN would use only 10 channels and start the backup of each datafile in parallel. Prior to Oracle Database 11gR1, there was no way to parallelize the backup of an individual datafile. Oracle Database 11gR1 introduced a new feature, Multisection backup, which enables one to parallelize the backup of individual files as well. Multisection backup enables RMAN channels to break the datafiles into chunks known as sections. Each RMAN channel then backups chunk of datafile thereby parallelizing the backup of individual datafiles if enough channels are available.

For example, you can parallelize the backup of 4 GB tablespace by dividing the datafile into 500m sections and opening 8 RMAN channels.

However, the true advantage of parallel backups exists when the datafiles and backup destinations are stored in ASM (ie data is or can be spread across multiple disks). If you section a datafile that resides on only a single disk, the disk head has to move constantly to address different sections of the file, outweighing the benefits of sectioning.

Compressed database backup with varying degree of channel parallelism

Backup performance while creating compressed backupsets is CPU bound. On T-series servers, you can use increased parallelism to run jobs on multiple CPUs and thus improve performance**.**

We took compressed backup of entire database on an ASM disk group that had 6 physical disks. The size of the database was 32 GB and there were 17 datafiles in that database. The datafiles were of varying size.

Table 7 summarizes the backup time taken to complete compressed backup of size 32 GB with varying degree of channel parallelism

TABLE 7. EFFECT OF CHANNEL PARALLELISM ON BACKUP TIME

NO OF CHANNELS	MULTISECTION ENABLED	BACKUP TIME (HH:MM:SS)	%CHANGE	CPU UTILIZATION
1	No	2:25:10	-	98/2/0
2	No	1:21:19	44%	97/3/0
3	No	0:58:22	60%	94/6/0
4	No	0:58:46	60%	94/6/0

25	Yes (section size 2 MB)	00:28:31	80%	84/16/0
50	Yes (section size 1 MB)	00:17:00	88%	75/24/1

The minimum time required to backup this database is defined by the time it takes to backup the largest file if multisection backup is not enabled. It is assumed here that there is enough I/O bandwidth available for taking the backup. Enabling multisection backup further reduces the backup time.

As indicated in table 7, multiple channels with multisection improved the backup time by 88% .

Concurrent Database Operations

This section explains how Oracle Database deployed on T-series servers can be optimized by exploiting the concurrency features offered by Oracle Database.

Increasing throughput through concurrency

The large number of hardware threads in T-series servers can be effectively utilized by increasing concurrency.

The following table demonstrates the scalability of users of an online transaction processing workload on a Sun SPARC Enterprise T5220 server.

TABLE 8. EFFECT OF INCREASING CONCURRENT USERS ON TRANSACTION THROUGHPUT

NO. OF USERS	TRANSACTION THROUGHPUT	%IMPROVEMENT	CPU UTILIZATION(IDLE/USR/SYSTEM)
8	72963	-	87/11/2
16	144778	98%	74/23/3
32	246134	237%	50/46/5
64	364965	400%	5/86/9

As indicated in table 8, increasing concurrent users helps improve the transaction throughput. By going from 8 users to 64 users, the CPU utilization went up from 11% to 86% thereby improving the throughput by 400%.

Concurrent data loading

SQL Loader is a high-speed data loading utility that loads data from external files into tables in an Oracle database. SQL Loader accepts input data in a variety of formats, can perform filtering, and can load data into multiple Oracle database tables during the same load session.

SQL*Loader can load data either by executing the SQL INSERT statement to load the data (referred to as conventional path load) or can load data directly bypassing most of the RDMS processing (referred to as direct path load). A direct path load can be executed in parallel which enables multiple direct path load sessions to concurrently load the same data segments. This is usually referred to as parallel direct path load. However, there are cases when you can't use direct prallel load. Refer to Oracle documentation for details.

Data Path Loads can be enabled by setting `direct` parameter to true. Parallel Data Path can be enabled by setting `direct` parameter and `parallel` parameter to true. However, setting the parallel parameter to true only allows multiple concurrent direct path load sessions. To start multiple concurrent sessions, the user needs to explicitly start multiple sessions. To load data concurrently on the same table, the sql loader needs to use the append option and the input file needs to be sliced to multi-parts.

We loaded 725 GB of datafile to a table using sqlldr on a Sun SPARC Enterprise T5220 server. The following table summarizes the time it took using various SQL-Loader options.

TABLE 9. EFFECT OF CHANNEL PARALLELISM ON BACKUP

DIRECT LOAD	PARALLEL	CONCURRENT SESSIONS	TIME TO LOAD	%CHANGE	CPU UTILIZATION(IDLE/USR/SYSTEM)
No	No	-	16:44	-	98/2/0
Yes	No	-	10:12	38%	98/2/0
Yes	Yes	5	2:07	87%	90/9/1
Yes	Yes	10	1:11	93%	84/16/1

As indicated in table 9, by starting multiple SQL-Loader sessions the data loading time reduced by over 90%.

Virtualization options on T-series Servers

T-series servers run on the Oracle Solaris 10 Operation System. The Oracle Solaris 10 Operating System provides Oracle Solaris Containers technology to enable virtualization and consolidation. Oracle Solaris Resource Manager, which is a component of Oracle Solaris Containers, can be used to precisely control system resources available to multiple applications running in a single Operating

System instance. Additionally, the T-series servers provide virtualization capabilities granting full hardware isolation using Oracle VM for SPARC, formerly called Sun Logical Domains.

Each Logical Domain runs its own copy of the Oracle Solaris Operating System, and hardware resources may be moved between Logical Domains without requiring a reboot or even restarting an Oracle 11gR2 Database.

With Logical Domains, domains have roles which define their characteristics. An Oracle Database can be located on a domain with any role, as long as that domain provides the appropriate devices and resources required and supported by the Oracle environment. There is no performance overhead of deploying Oracle Database in an I/O domain. By having direct access to physical I/O devices, an I/O domain provides optimal I/O performance. However, the number of I/O domains that can be created on a single platform is limited by the I/O resources available on that platform; this limit is currently tied to the number of PCI buses available. As a result, only a limited number of I/O domains can be created.

Refer to the references section for a whitepaper that covers virtualization options for Oracle Database deployment on Oracle's T-series servers.

Conclusion

Sun SPARC Enterprise T-series servers offer a very large number of CPUs along with significant processing power, and there are many ways that this processing capacity can be used up. Oracle Database deployment can leverage parallel and concurrency features offered by Oracle Database to better utilize Oracle's T-series servers. Oracle Database deployment on T-series servers can also take advantage of virtualization options offered by T-series servers and Oracle Solaris operating system to host multiple distinct workload environments to further improve overall system utilization.

Appendix A: CPU measurement tools for T-series Systems

Vmstat

vmstat is usually the first command you run to familiarize yourself with the system. It indicates both the CPU utilization and CPU saturation.

The following fields from vmstat output are important to understand CPU saturation and utilization.

- `cpu:id:`

  This field is used to determine CPU utilization. CPU utilization is calculated by subtracting id from 100.

- `cpu:us + cpu: sy:`

  Boths these fields together represent CPU utilization. Thus, CPU utilization can either be calculated by subtracting cpu.id from 100 or by adding cpu:us + cpu:sy fields.

- `kthr:r:`

  This field represents the total number of runnable threads on the dispatcher queue. It is used to measure CPU saturation. Any sustained non-zero value is likely to degrade performance. The performance degradation is gradual unlike memory saturation, where the performance of the system drops rapidly.

- `faults:in`:

  Number of interrupts per second

- `faults:sy:`

  Number of system calls per second

- `faults: cs:`

  Number of context switches, voluntary and involuntary

Mpstat

Mpstat summarizes the utilization statistics for each CPU on the system.

The mpstat output provides the information about the processor utilization, scheduling and lock related.

The key processor utilization fields are as follows:

- `syscl:` system calls per second

- `usr:`
- `sys:`

The key fields for scheduling related statistics are as follows:

- `icsw` (Involuntary context switches)

  This field display number of threads involuntarily taken off the CPU either through expiration of their quantum or preemption by a higher priority thread. This number indicates if there were generally more LWPs ready to run than physical processors. When run on a CMT processor-based system, the icsw is usually less than it is on traditional systems because there are more hardware threads available for LWPs.

- `migr`:(Migration of threads between processor)

  - This field displays the number of times the OS scheduler moves ready-to-run threads to an idle processor. Migrations on CPUs are bad for performance because they cause a thread to pull its working set into cold caches, often at the expense of other threads.

- `smtx` and srw output provides locking related info. However, more meaningful lock related information can be found by lockstat command

corestat

Oracle Solaris tools mpstat and vmstat provides information about the hardware threads. However, on CMT systems each core (consisting of 8 hardware threads) has its own instruction pipeline Since the instruction pipeline is shared among threads, a stalled thread does not mean stalled pipeline. Hence it is important to understand the core utilization along with utilization of individual threads. Corestat tool provides online monitoring of core utilization.

For non-CMT systems, idle time reported by the Oracle Solaris tools mpstat or vmstat can be used to decide if more load can be added to the system. On a CMT server, corestat output can usefully be referred to along with Oracle Solaris tools mpstat and vmstat to make the same decision.

Corestat reports core utilization for all the available cores by aggregating the instructions executed by all the threads in that core. It's a Perl script which forks cpustat command at run time and then aggregates the instruction count to derive the core utilization. It can be downloaded from http://www.opensparc.net/cool-tools.html.

References

The following table contains links to useful information related to this paper

RESOURCES	
Oracle SQL Parallel Execution	http://www.oracle.com/technology/products/bi/db/11g/pdf/twp_bidw_parallel_execution_11gr1.pdf
Parallel Execution Fundamentals in Oracle Database 11gR2	http://www.oracle.com/technetwork/middleware/bi-foundation/twp-parallel-execution-fundamentals-133639.pdf
Deploying Oracle Database on Oracle Solaris	http://www.oracle.com/ /technetwork/articles/systems-hardware-architecture/deploying-oracle-database-solaris-168405.pdf
Virtualization Options for Oracle Database deployments on Sun SPARC Enterprise T-series systems	http://www.oracle.com/ /technetwork/articles/systems-hardware-architecture/virtualization-options-t-series-168434.pdf
Effective Resource Management on Oracle's Sun SPARC Enterprise Servers using Solaris and Oracle Database Resource Managers	http://www.oracle.com/ /technetwork/articles/systems-hardware-architecture/effective-mgmt-using-oracle-drm-168439.pdf
Dynamic SGA Tuning of Oracle Database on Oracle Solaris with DISM	http://www.oracle.com/ /technetwork/articles/systems-hardware-architecture/using-dynamic-intimate-memory-sparc-168402.pdf

Oracle is committed to developing practices and products that help protect the environment

Optimizing Oracle Database on Sun SPARC Enterprise T-series Systems
September 2010
Author: Ritu Kamboj

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com

SOFTWARE. HARDWARE. COMPLETE.