UNICOS/mk Performance Measurement

Stephan Gipp, Cray Research, Eagan, MN USA

ABSTRACT: *In order to support the complexity of a distributed operating system such as UNICOS/mk many of the performance measurement tools familiar from UNICOS underwent some changes. These changes range from subtle changes to the meaning of some numerical values presented by tools such as 'sar/tsar' and 'acctcom' over adding attempts to analyze the multi PE performance and balance to 'ja' to the implementation of several new displays to 'xsam/xmppview' and the definition of an additional concept of CPU time. This paper will give an overview on these changes and try to explain how to interpret the numbers and displays.*

1 Compatibility

The purpose of any performance measurement within the operating system is to aid with the identification of either software or hardware related system bottlenecks. In order to do this, internal details of the underlying hardware and software architecture need to be externalized. As a result of this necessity, one of the primary goals of the design of the UNICOS/mk operating system was broken: compatibility. Instead of maintaining true binary compatibility with UNICOS, a *compatibility in spirit* was the development goal.

As a result, no new tool in the area of performance measurement was written for UNICOS/mk. Instead tools where taken from UNICOS and adapted or modified to support the different needs of a T3E architecture.

2 Times

The core of most or all performance measurements are several definitions of time. In UNICOS/mk four basic times are defined to describe basic system performance:

- User CPU time
- System CPU time
- I/O wait time and
- Remote CPU time

These times are accumulated on a per process as well as on a per site basis.

2.1 User CPU time

User CPU time is defined as the time connected to a CPU at the local node while executing in user mode.

This definition is identical to the definition under UNICOS.

2.2 System CPU time

System CPU time is defined as the time connected to a CPU at the local node while executing in system mode.

Although this sounds very much like the UNICOS definition, the distributed nature of UNICOS/mk causes many of the system calls to execute on other nodes than the local node for a large percentage of their overall execution time. Since the above definition of system CPU time does not include these times, the per process accumulated system time will often be smaller under UNICOS/mk than under UNICOS.

2.3 I/O wait time

I/O wait time is defined as the time not connected to a CPU at the local node with outstanding I/O requests.

Unlike UNICOS, this is no longer broken into locked and unlocked time. The reason for this change was not so much a philosophical decision, but rather forced upon us by some implementation details of UNICOS/mk.

2.4 Remote CPU time

Remote CPU time is defined as the time connected to a CPU on non-local nodes.

This time does not have any equivalent feature in UNICOS. It is simply a result of the serverization of operating system components that was used to implement UNCOS/mk. For all practical purposes this time also reflects high degree of parallelism within UNICOS/mk.

The relation between these times is best explained by going through the basic activities of system calls such as reada() and read() as shown in Figure 1. In this scenario a user processes is running on PE-A issuing a reada() system call that is followed

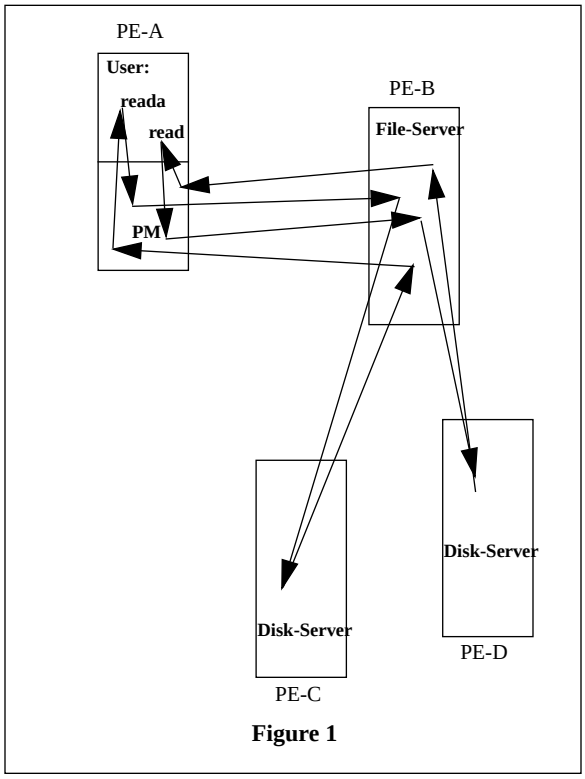

Figure 1

some time later is followed by a read() system call. Both calls are first processed by the local PM, are forwarded to a file server running on PE-B and finally processed by disk servers running on PE-C and PE-D.

From the point of view of the user process, times are accumulated as follows:

- While executing in user mode or in the local PM, either user CPU time or system CPU time is accumulated.
- Once the read() request is forwarded to the file server, I/O wait time is accumulated until the request is completed.
- For the request handled by the disk servers, system CPU time is accumulated. This time is charged back to the caller (i.e. the file server running on PE-B) as part of the reply messages from the disk servers to the file server.
- The charged back time received by the file server is accumulated as remote CPU time in the file server. Also, while executing on PE-B, the file server accumulated system CPU time. The sum of these two times (system CPU time and remote CPU time) is charged back to the user process running on PE-A as part of the reply message from the file server to the user process.
- The charged back time received by the user process is accumulated as remote CPU time. This remote CPU time now includes sum of all the system CPU times used to process the request on PE-B and PE-C or PE-D.

  This implies that it is possible for a process that the sum of all CPU times and the I/O wait time is larger than elapsed time.

From the point of view of the nodes, times are accumulated as follows:

- While executing in user mode or system mode, nodes accumulate the respective times.
- After processing a request from a different node[1], the total CPU time[2] required to process this request is charged back to the caller as part of the reply message[3].
- After receiving a message reply, the charged back time contained in that message is accumulated as remote CPU time at the receiving node.
- This implies that the system wide accumulated remote CPU time maybe greater than the system wide accumulated system CPU time.

3 Tools

As said before, no new performance measurement tools where implemented for UNICOS/mk.

One tool is no longer supported (*hpm)*, and some features in other tools are deferred:

- the curses display of mppview and
- some displays in csam and xsam.

3.1 ja

A simple cluster analysis has been added to ja in order to assist the interpretation of multi-PE application accounting data.

For each member of a multi-PE-application user CPU time, combined system and remote CPU time and the number of bytes transferred is collected. On each of these three categories a simple cluster analysis is performed.

The analysis is limited to 5 clusters. For each cluster values for minimum, maximum, average and standard deviation are presented. This will allow for an easy identification of imbalances in applications.

3.2 sar and tsar

In order to ease use as well as maintenance, sar and tsar are no longer different tools. Instead, sar[4] is a wrapper around tsar using a set of pre-defined tsar scripts to provide sar style functionality and output.

A new option (-G) was added to sar that provides summary node utilization separated by node type (OS, CMD and APP). This avoids problems where overall system utilization could be dominated by a single type (such as APP nodes).

---

[1] as a result of an IPC call

[2] user CPU time + system CPU time + remote CPU time

[3] In addition, this charged back time is accumulated at the sending site as *service time*. The *Top Remote Display* in xsam shows both the per node remote CPU time and the per node service time.

[4] and sadc, sa1 and sa2

3.3 csam and xsam

Five new displays where added to xsam to improve the visualization of T3E specific activities. These displays also try to reduce the amount of information shown by focusing on the nodes with the highest activities. Figure 2 shows the Top Site[5]

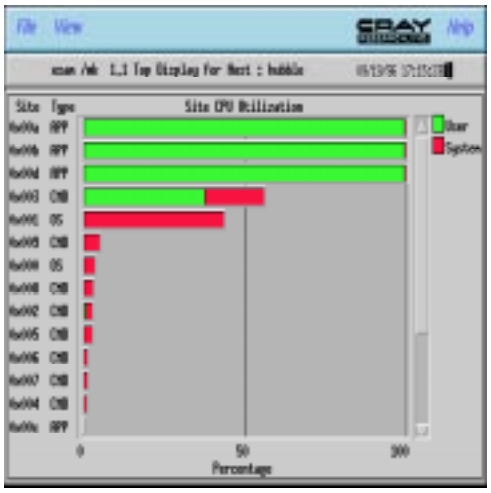

Figure 2

Display with three application nodes running 100% in user mode and one command and one support node running at non-trivial load levels.

Other displays provide

- information about thread[6] and memory[7] utilization at a specific node,
- system wide statistics about locality of operating system execution (remote CPU time received vs. delivered, amount of IPC traffic) and
- system call utilization.

Some displays in both csam and xsam are deferred. These include:

- logical device activities
- swap maps
- memory maps in csam
- process activities in csam and
- process snapshots

It is no longer possible to run samdaemon (the data acquisition daemon) on the SWS; it has to run on the T3E. Xsam, however, can execute on either the T3E or an SWS.

3.4 xmppview

This tool has been greatly enhanced compared from it's predecessor in UNICOS-MAX. Color is used as a 4th dimension allowing the implementation of displays showing not only the allocation of nodes, but also detailed information such as

- node utilization (CPU times, memory usage, process counts, run queue data and IPC statistics),
- GRM configuration data (service type and application requirements)
- hardware configuration data (memory sizes and clock speeds).

Figure 3 shows a lightly loaded system with 64 nodes. Ten

Figure 3

colors show different amount of CPU usage (i.e. user + system CPU time) and a little histogram in the upper left corner shows global statistics.

Like xsam, xmppview can execute either on a T3E system or an SWS. Due to the large amount of graphical data, especially when the node display is rotated, it is recommended to execute xmppview on the SWS.

[5] The term *site* is derived from the original Chorus µkernel and for the scope of this paper is identical to terms *node* and *PE*.

[6] The term thread is derived from the original Chorus µkernel and for most parts is equivalent to a UNCIOS/mk process, but also include the OS internal servers.

[7] UNICOS/mk does not require a contiguous space for processes. This makes memory fragmentation a less important issue. The memory maps shown in xsam are logical and not physical memory maps, i.e. the maps neither show the fragmentation of process space nor the actual physical location of a process in memory.