This paper discusses programming techniques for continuously gathering performance data in a complex multijobbing environment. The program takes advantage of computing system and support program characteristics to count and time data processing activities.

The data gathered by the program enables installation managers to measure the effects of modifications to equipment and programming support configurations and to define work loads.

Measurement of system operational statistics

by W. I. Stanley

Current computing systems are often comprised of large amounts of computing equipment, do a large volume and variety of work, and are supported by complex operating systems. Such computing systems are sometimes expected to process data on a real-time basis, as well as provide conversational mode operation and teleprocessing. One effect of these trends is that computer installation managers have an increased need for quantitative data about work load characteristics (number of compilations, executions, etc.) and processing activities (frequency and duration of use of CPU, channels, etc.) in order to evaluate system performance. Providing this data requires complex techniques for measuring the internal activities of the systems and associating these activities with specific units of work.

This paper presents a special-purpose program called the job accounting system (JAS) for continuously and automatically monitoring the activities of a modern real-time operating system.[1,2] The system being monitored is used both for multijobbing jobshop operations and real-time support of NASA manned space flight activities.

The monitoring system gathers statistics that are used to bill users of the computer facilities and to evaluate system performance. Emphasis in this paper is on performance evaluation. Specifically, the data are used to assess the suitability of the existing configuration for the work load being processed, to determine whether operating procedures are resulting in effective use of system resources, and to anticipate needed configuration changes to accommodate a changing work load. The data, which are

provided in terms of resource usage per job and per job step,[3] are gathered over relatively long periods (days or weeks of processing time).

The data obtained by JAS, which are called operational statistics in this paper, are used in two ways associated with performance evaluation. Operational statistics are interpreted directly. For example, monitored central processing unit usage might be low because insufficient advantage is being taken of the multijobbing capabilities of the operating system. Operational statistics are also used to characterize work loads in simulating computing system operation. Experience with the real-time computer complex indicates that, given suitably parameterized models calibrated with detailed measured performance data,[4] few statistics are needed to characterize the work load. These data include types of jobs and job steps (FORTRAN compilations, assemblies, linkage-editor processing, etc.) and some information about the equipment (main storage size, CPU speed, etc.). Using operational statistics rather than hypothesized work loads enhances the accuracy of system simulation.[5]

This paper first considers operation of JAS, including the kinds of data gathered, and then discusses some of the factors taken into account in its design. The final topic in the paper is resolution of the timer used to measure elapsed times and the effect of this resolution on results.

The job accounting system

history

The job accounting system arrived at its present state of development by an evolutionary process. In 1966, JAS was implemented to define the workload at a real-time computer complex (RTCC). Statistics from this effort were primarily job names, step names, I/O device usage, CPU usage, and elapsed job time. This information was used to derive a record of system performance (jobs per hour, percent of operating time per day, etc.) and to define job mix (percent assemblies, FORTRAN compilations, program executions, etc.). The statistics were used as input to General Purpose Simulation System (GPSS) models of the computer system. Even at that time, the potential for expanding the use of the statistics obtained by JAS was apparent. Therefore, the JAS output was defined to be unprocessed statistics stored in a data base. By post-processing the data base with utility programs, performance reports were generated. Since 1966, the JAS data base has been expanded to include additional information for billing computer users and for providing management information on program development costs.

monitored data

The data base contains three types of operational statistics: static accounting data, dynamic statistics on job execution, and special operator information entered through the computer console. However, the operator-supplied information is not presently used in the performance reports. These operational statistics are

summarized by the collection program at three points during job execution: at the start of a job, at the end of a job step, and at the end of a job. Records of I/O activity are added at the start of each job to indicate I/O activity of the operating system for each job initiated, at the end of each step to indicate I/O activity for the step, and at the end of each job to record operating system I/O activity from the end of the last step in a job to the end of the job.

Static information is supplied by the programmer and obtained from job control statements. It includes job name, programmer name, and similar accounting information. Items such as start-of-job time, date, and computer identification are included in this record.

Dynamic statistics are stored in both the end-of-job and end-of-step records. Dynamic statistics are measured use of resources, such as CPU time per job and per job step, frequency of I/O device usage, main storage usage, job and job step elapsed execution time, and three system statistics. These three statistics are: system I/O wait time, when the CPU is in the wait state but at least one I/O device is operating; system idle time, when the CPU is in the wait state and no I/O devices are active; and system task CPU time, when the CPU is being used to perform system tasks, such as reading a job stream into the system or accepting operator commands.

performance reports

To create a performance report, statistics are taken from the start-of-job, end-of-step, and end-of-job records, plus the associated I/O records in the data base. The CPU time used to execute the operating system job scheduler can be calculated from the data base records. (Job scheduler CPU time equals job time minus the sum of the job step times.) The associated I/O records reveal the I/O activity performed for the job scheduler and for job steps. The step names and the number of steps per job characterize the work load. The time of day recorded in each end-of-job and start-of-job record is used to calculate the time between the end of one job and the beginning of another job entered via the same job initiator task.[6] Also, the number of active job initiators reveals the level of multijobbing at any given time, which, in turn, can be related to system resource usage.

The amount of time that the computer system is actually in operation is indicated by the number of times that the operating system must be loaded into the computer (called initial program loading or IPL in this system) and the length of time the computer is not operating prior to being restarted by an IPL. Another measure of unused computing capacity is the amount of system idle time, which is accumulated and recorded in each end-of-step and end-of-job record. Operator information may be needed to account for unproductive time.

The utility program that processes the data base to provide performance information summarizes these statistics and produces an eight- to ten-page report. Much of the report is self-explanatory.

Table 1 General job statistics

Total counts
Jobs run
IPL's
Abnormally terminated jobs
Operator accounting messages
Background utilities
Concurrent initiators
Total times (hr/min/sec)
CPU time for job stream
CPU time for system tasks
CPU time for utilities
System I/O wait time
System idle wait time
Job run time
Nonjob time
Sample time
Job averages
Number of jobs per hour (based on sample time)
Number of jobs per hour (based on job run time)
Average time (hr/min/sec)
Job elapsed time
Job CPU time
Initiator between-job time
Time to IPL
Time between IPL's

For example, job statistics, as shown in Table 1, give a general breakdown of how time was spent during a sampling period. Included are statistics on installation throughput, measured in jobs processed per hour. The use of computer time as a function of CPU utilization is recorded in:

- Percentages of job run time
  - CPU time for job stream
  - CPU time for system tasks
  - CPU time for utilities
  - Time CPU is waiting for I/O
  - Time CPU is idle
- Percentages of total sample time
  - Job run time
  - Nonjob time

The ideal is for 100 percent of the CPU capacity to be used for 100 percent of the total sample time. Because of limited main storage and I/O device resources, this goal can never be reached, and the report shows the degree of success in achieving the goal. As a basis for determining the makeup of the work load of the installation, the report provides statistics on the kind of job step executed:

- General step statistics
  - Number of completed steps
  - Average steps per job
  - Average steps per hour
  - Average step elapsed time
  - Average step CPU time
- Step statistics by step name
  - Average CPU time
  - Number in sample
  - Percent of step type
  - Step-to-job CPU time

This information is used as input for models of new equipment configurations. Usage of computer resources, other than the CPU, are not defined for each job step. The information presented is not particularly detailed; additional information is needed to define the basic configuration model. The frequency with which job steps refer to I/O devices is also provided in the report. The collection program does not collect statistics of references per data set. In actuality, I/O statistics are reported for each job step.

Design and implementation considerations

The statistics that might be collected during normal job processing appear endless. An effort was made to implement a collection system to gather specific information at a reasonable cost in productive use of the collection system. Comparisons of computer

system performance with and without the job accounting system indicate that main storage and CPU requirements reduce processing capabilities by about one percent. Several factors were considered to minimize throughput degradation.

resource usage

Computer resource usage can be counted at less cost in CPU execution time and main storage buffer space than the duration of resource utilization can be measured. For example, the frequencies with which data sets are accessed for a job or job step can be counted with less performance degradation than occurs when the time is measured during which an I/O device is utilized in accessing data sets used in a job or job step. The frequency statistic may be just as useful in an analysis of a new hardware configuration, since an approximate I/O device utilization figure can be determined by simulating the allocation of data sets and the accesses to them.

statistical data base

Another factor considered is uses of the statistics, which can affect their exact definition. For instance, at many computer installations, it is necessary to implement accounting routines for billing users of a multijobbing system. Many of the fundamental statistics useful for billing are also useful for performance analysis. However, if the objective for statistics collection is strictly oriented toward billing, the recorded statistics may be of limited value for analysis. For example, a statistic that combines CPU time and I/O wait time may be suitable for billing but not for performance analysis. In measuring for billing purposes, this statistic may be further confused by an algorithm that weights the combined time depending on the amount of main storage used, the use of peripheral equipment, etc. In order to avoid such problems, the fundamental statistics—CPU time, I/O wait time, and computer resource usage—are collected and saved in the statistical data base.

In order to reduce the volume of output supplied to the data base, some compromise is necessary between producing a chronological record of every interesting event and creating event summaries. A multipurpose data base of fundamental, unprocessed statistics can answer a greater variety of needs. The cost of this approach may not be any greater than the cost of providing information useful for a single purpose. The data base can be subsequently processed according to whether the interest is in billing computer users, in performance analysis, or in general management information.

levels of detail

Another design consideration is the implementation of optional levels of statistical detail, which avoids continuously gathering statistics that are not necessary for most purposes. Statistics gathering is selectable from the console at two levels of detail. At the more general level, which is normally used for billing, data are collected for the start-of-job and end-of-job records, supplemented by information from the operator's console. At the other level, which is used for performance evaluation, all statistics possible using JAS are collected, including job step and I/O device

usage data. Thus, at the more detailed level, both sampling frequency and the types of data monitored increase. The basic reason for allowing data to be collected at more than one level of detail is that the cost of collecting the more detailed statistics can be incurred only when the data are needed. The sampling interval should be sufficiently long so that the data are statistically valid. Such statistics may then be used to monitor changes in processing requirements or for planning configuration changes.

collection points

Another consideration is the requirement for flexibility. Because it is impossible to collect one set of statistics that can satisfy every need, the capability should exist to augment the data base to suit particular needs. This flexibility is obtained by having entries to the collection program at key collection points in the operating system and operating system options to enable installation personnel to add information to the data base. The first feature implies the need to anticipate those statistics that might be useful. The second feature implies development of simple methods for logging installation information in the data base.

Two types of collection were considered: a computer console command language for use by the operator and a special job control statement or accounting field in an existing statement. The console language permits the operator to add information concerning the operating environment, such as the reasons for computer "down time." The operator information is not presently used in the performance reports. The job control statement, or accounting field, provides a means of identifying the jobs processed and for defining information about them.

operating system modifications

A basic design goal for the monitoring system was the ease with which operators could control the system. The console language provided with JAS permits the operator to select the level of statistical detail, choose the output device (punch or tape drive), add information to the data base, or stop statistics collection entirely. Implementing these facilities requires adding code to the operating system nucleus, the console command modules, and the job scheduler. Main storage space is also necessary for statistics buffers (132 bytes or 588 bytes per active job, depending on the level of detail selected) and the job accounting system output routines. Since the output routines reside in main storage during job accounting system operations, approximately 5000 to 7000 bytes of main storage (depending on the level of statistical detail selected) are required for single jobbing. This figure excludes both the code in the job scheduler that does not normally reside in main storage and the additional statistics buffers needed during multijobbing operation. Most job accounting system code resides in external storage until it is needed.

statistics collection

Even though the job accounting system operates with several load modules that are brought into main storage only as needed and provides optional levels of statistical detail, normal processing of jobs could be significantly delayed without proper integration into the operating system structure. Use of the punch as an output

device and the conversational use of the computer console offer prime examples of potential throughput delays caused by slow I/O devices. To avoid such delays, the monitoring function is accomplished as a series of system tasks. The output task involves recording statistics asynchronously with their collection, while the rest of the tasks are concerned with the collection of statistics. Communication among the tasks involves WAIT/POST logic,[6] a chain of statistics buffers ready for output, and statistics collection buffers associated with each job as it is executed.

To begin statistics collection, either the output task is created at IPL time as a result of a program switch in the operating system or the operator initiates the output task via the console. In either case, the system is initialized for statistics collection. The output task is then delayed until, as part of one of the statistics collecting tasks, a statistical buffer is added to an output chain. The output task is then allowed to proceed with formatting and recording of the statistics.

The collection tasks are generally associated with normal job execution. Each time the job scheduler initiates a new job, a statistics buffer is obtained. The job name, job start time, and other information is collected. The buffer is then added to the output chain, and the statistics are recorded. The job scheduler task is not delayed while the statistics are being recorded; but rather, another buffer is obtained and the job step proceeds normally. This second buffer is used to accumulate CPU time and I/O device statistics for the job step if the job step level of statistics collection was selected. In this case, the buffer is passed to the job accounting system output routine at the end of the step. If statistics are being collected only at the job level, statistics are accumulated for all steps in the job but I/O statistics are not collected. In all cases, the job accounting control section of the job scheduler contains the code to chain statistical buffers for output, to initiate recording of the new statistics, and to obtain a new buffer for continued collection.

The other task that involves collecting information for the data base is the console communications task. In this case, statistics collection involves requesting input from the operator, adding an output buffer to the output chain containing his response, and posting completion of the task.

Obviously, with multiple statistics-collecting tasks, the output chain could contain more than one entry, which is permitted. The statistical buffers are added in chronological order to one end of the output chain. The output task involves removing buffers from the other end of the output chain, formatting the statistics, and recording statistics from each buffer as fast as the selected output device permits. As soon as the contents of one buffer have been recorded, the next is removed, formatted, and recorded. The output task continues (concurrently with other tasks) until all statistics have been recorded. As part of the output task, all buffer space is made available again after use.

Timing source resolution

Operational statistics that include measurements of elapsed time require the use of a clock, which may be read internally by the collection program. Measurement accuracy depends on several variables: the clock resolution (related to the elapsed time between the instances when a clock steps to a new value), the total time measured for one or more events, and the independence of the clock from the event being measured. If clock resolution is small, so that a clock count of one measures time that is insignificant compared to the duration of the event, the second two variables that determine accuracy are unimportant. Otherwise, time must be approximated from a statistical sample.

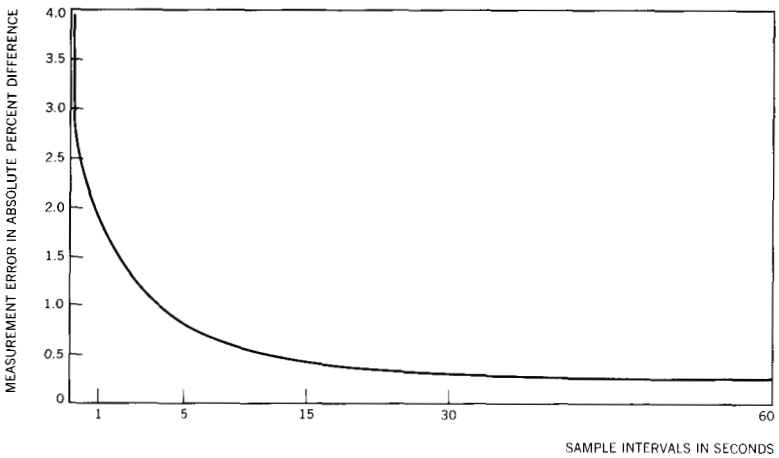
The significance of clock resolution can be seen by considering the measurements of CPU time required to execute a particular job. If an interval timer with, for example, a resolution of 16 2/3 milliseconds per clock count is used to measure CPU time, accuracy problems may occur. Any measurement of CPU execution time on a high-speed computer is likely to give either one of two answers: the clock reading at the start of execution is equal to the reading at the end of execution (zero time), or the clock has counted once between the start and end of execution (16 2/3 milliseconds).

The fact that an individual time measurement may be inaccurate is not necessarily a problem. If a user job is given the CPU resource 1000 times during its execution, 900 of the time measurements could be zero and 100 of them could be 16 2/3 milliseconds. The total CPU time for the job is $900 \times 0 + 100 \times$ 16 2/3 milliseconds, or 1667 milliseconds. Now, if the clock operates independently of the manner in which the CPU resource is used, 1667 milliseconds is statistically a good approximation of the actual CPU time required for execution. However, if the user job is synchronized with the clock (as can be done using IBM's SYSTEM/360 Operating System STIMER system macroinstruction), the resulting execution time could be biased to produce a lower number than the true value.

The SYSTEM/360 interval timer, with a 16 2/3-millisecond resolution, is used in JAS to measure elapsed times. To verify the reliability of timing statistics using this timer, a model of a multijobbing environment with a SYSTEM/360 Model 75 computer was used. This model was coded in GPSS/360 and contained routines that characterized the operating system and a job stream composed of jobs with assembly, FORTRAN compilation, linkage editing, and user program execution steps. By executing the model, statistics are produced that reflect theoretical performance in that environment.

A model of the SYSTEM/360 standard clock was added to the original GPSS/360 model and used to obtain a secondary measurement of system wait time. The implementation of the secondary measurement in the model paralleled the actual use of the standard timer in the real-time operating system. Each time the simulated computer went into a system wait state, a reading was taken

Figure 1 SYSTEM/360 standard timer accuracy



of the simulated clock; each time the simulated computer was interrupted, another clock reading was obtained. The elapsed time was accumulated over an interval and compared to the absolute wait time normally produced by the model. Several time intervals of different lengths were tried to determine something about the stability of time measurements for a Model 75 using a clock with a resolution of 16 2/3 milliseconds. The results showed that errors were not significant for sampling periods as short as five seconds. Figure 1 shows the absolute deviation of percent system wait time approximated by the 16 2/3-millisecond clock compared to the absolute statistic produced by the model. The time intervals over which system wait time was accumulated were 60 seconds, 30 seconds, 15 seconds, 5 seconds, and 1 second. A deviation of one percent over a 60-second interval would mean that the measurement was in error by 0.6 seconds, or approximately 0.0002 hours.

Summary comment

A variety of statistics useful to installation managers can be obtained continuously at little cost in computing time. The designers of programming support for gathering such information must be cognizant of those areas where unreasonable amounts of machine time can be consumed. For greater efficiency, they must also carefully consider the kinds of information that are actually needed, while providing sufficient flexibility to obtain additional data the need for which was not originally anticipated.

Two general approaches were of major significance in achieving these objectives for the job accounting system. The first was the decision to provide data at optional levels of detail, thus reducing

the time lost in collecting unwanted information. The second was the use of a data base in which to store all monitored information. Post-processing of the data base can then be used to produce reports tailored to particular needs.

ACKNOWLEDGMENTS

The author is pleased to acknowledge the work of L. S. McCollum and J. A. Hudson, who were the principal implementers of the job accounting system for the real-time computer complex project.

CITED REFERENCES AND FOOTNOTE

1. W. I. Stanley and H. F. Hertel, "Statistics gathering and simulation for the Apollo real-time operating system," *IBM Systems Journal* **7**, No. 2, 85–102 (1968).
2. J. L. Johnstone, "RTOS-Extending OS/360 for real-time spaceflight control," *AFIPS Conference Proceedings, Spring Joint Computer Conference,* (1969).
3. B. I. Witt, "The functional structure of OS/360, Part II, Job and task management," *IBM Systems Journal* **5**, No. 1, 12–29 (1966).
4. For example, the detailed statistics of control and application program performance described in Reference 1 are used for analyzing system design and for calibrating models of system programs.
5. Analysis and simulation are discussed in several of the papers in the series "On teleprocessing system design" in the *IBM Systems Journal* **5**, No. 3 (1966).
6. WAIT/POST logic, as well as other types of task interdependencies are discussed by J. W. Havender in "Avoiding dead-lock in multitasking systems," *IBM Systems Journal* **7**, No. 2, 74–84 (1968).