Throughput measurement using a synthetic job stream*

by DAVID C. WOOD and ERNEST H. FORMAN

The MITRE Corporation
McLean, Virginia

INTRODUCTION

Managers of computer facilities frequently need to be able to measure the throughput of their installations. Throughput is based on the amount of work a system can perform in a given time in a particular environment, and is usually measured as the reciprocal of the running time of selected jobs. The throughput of a computer system is a function of the hardware configuration, the operating system software and the workload characteristics. Once the characteristics of the workload are identified, the configuration and operating system can be optimized in terms of throughput for that environment. Improvements resulting from system changes can be evaluated using throughput measurements.

With a given workload and system configuration, CPU utilization is a crude measure of relative throughput. More accurate results are obtainable by careful experimentation with benchmarks which are representative of the workload. A benchmark usually refers to a single job, such as a matrix inversion which may be typical of certain scientific applications. Benchmarks are frequently used to evaluate different computer systems prior to their selection.¹ With a multiprogramming computer system, the throughput depends on the job mix, the scheduling algorithm, and many parameters in the operating system. A job stream is a collection of independent jobs which can be used to determine the relative throughput of a multiprogramming system based on the time taken to execute all the jobs.

A job stream could be gathered from the actual workload by using a sampling procedure and verifying the resulting job stream by comparing features with a total workload. Such a study at the University of Iowa has been reported.² This is not feasible in all installations. The following difficulties have been experienced in assembling and using a job stream composed of actual jobs:

- users of a service facility are reluctant to supply programs, data bases and operating instructions
- security considerations prevent many jobs from being included in the job stream
- it is difficult to closely match the overall characteristics of the jobstream to the workload characteristics without being able to select from a large number of jobs since the characteristics of each job are fixed
- it is extravagant on storage space to duplicate large data bases
- new releases of the operating system and changes in catalogued procedures make it difficult to keep complex jobs viable.

An alternative is to create synthetic jobs reflecting characteristics of the workload. This paper describes experience in defining the characteristics of a workload and generating a synthetic job stream based on the workload characteristics. There is little published work in either of these areas, and there are many practical difficulties to be solved. The synthetic job stream has been validated by comparison with a job stream comprised of actual jobs, and with the use of a hardware monitor. It has been used to measure the throughput of alternative hardware configurations and alternative software options.

CHARACTERIZATION OF WORKLOAD

There are no established standards for describing characteristics of a workload. Characteristics which are defined must be obtainable quantitatively for the workload, and since they are to be used as a basis for assembling a job stream, they should be easily trans-
Designing the job stream specifications based on the workload characteristics is the more difficult task. The required running time of the job stream is determined from the average job time and the accuracy desired. The number of jobs required can then be deduced. Several different types of synthetic jobs may be used; for example, jobs requiring tape or disk or neither. Core size, CPU times, and I/O times can be defined for each job in order to match the overall workload distributions for each characteristic, but it is not obvious how the various characteristics should be correlated for individual jobs. For example, although the probability of larger I/O time increases as the CPU time of a job increases, this is not always the case.

An approach to the correlation problem is to base the specifications of the synthetic jobs on actual jobs. A sampling technique can be used to select jobs in the actual workload, and the sampling procedure can be verified by comparing the characteristics identified previously. However, instead of obtaining the actual jobs, with the attendant practical difficulties noted earlier, only their characteristics are used. The core requirement, CPU and I/O time, peripheral requirements, printer output, etc., of each job can be used as the specifications for creating a synthetic job which looks identical.

GENERATION OF SYNTHETIC JOB STREAM

The synthetic job used in this study is based on the type of program suggested by Buchholz. A listing of the program, which is written in PL/I, appears in the Appendix. Parameters in the program are adjusted to vary the CPU time, I/O time and number of lines of printer output. Region size and tape and disk requirements of each job are determined by suitable specification of the job control language (JCL). Most synthetic jobs consist of compile, link edit and go steps of the PL/I program, but in those cases where a small region size or short CPU time is required, only the go step of a previously compiled and link edited program is executed.

The PL/I program performs the following tasks:

(a) Creates a data set of master records (data set DD name is MASGEN)
(b) Processes records from this data set (data set DD name is MASTER)
(e) Exercises a compute kernel a number of times for each record processed
(d) Outputs from one to three data sets after each record has been processed (data set DD names are OUT1, OUT2, OUT3. If less than three output data sets are required, OUT2 and/or OUT3 may be declared "DUMMY")

DESIGN OF JOB STREAM SPECIFICATIONS

The process of translating the workload characteristics into a synthetic job stream has two steps. First, the specifications of the individual jobs are determined so that they match the workload. Second, the synthetic jobs themselves are generated with the specifications determined previously.

Core requirement is easily measured for a real memory, in contrast to a virtual memory machine. The distribution of time spent in various region sizes can be calculated. CPU utilization can be expressed as the percentage of time the CPU is active. Accounting statistics almost always give a measure of I/O activity, but in some machines, such as the IBM 360, the measure may be inadequate for throughput studies. The measure given with the Systems Management Facility (SMF) of OS is the number of data transfers, not the quantity of data transferred. To resolve this deficiency, a hardware monitor has been used to measure the average channel activity per data transfer.

Although CPU and I/O channel utilization are important, they are not easily translated into jobs which will produce this activity. This can best be achieved by considering the ratio of I/O channel time to CPU time, both as a total for the workload, and as a distribution. For example, in a particular installation, short jobs may generally be dominated by I/O time, whereas long jobs are more often CPU bound. The utilization of tape units and exchangeable disk units can be measured in terms of the percentage of time these units are assigned to some job. The amount of activity while these units are assigned to a job and of public devices such as work disk packs is already included in the I/O channel utilization. The demand on printers, card readers and punches is easily calculated in terms of lines of print or number of cards.

Characteristics of the workload are defined in terms of the resource requirements imposed on the system. Resources considered are:

- Core requirement
- CPU utilization
- I/O channel activity
- Peripheral device utilization

Characteristics introduced in this paper relate to the IBM 360 series operating under OS/MVT; similar terms could be defined for other machines. Most of the characteristics can be obtained by an analysis of system accounting data.

Resources considered are:

- Core requirement
- CPU utilization
- I/O channel activity
- Peripheral device utilization

The process of translating the workload characteristics into a synthetic job stream has two steps. First, the specifications of the individual jobs are determined so that they match the workload. Second, the synthetic jobs themselves are generated with the specifications determined previously.
(e) Prints some number of lines after each record is processed.

The content of the master records, the type of calculations within the compute kernel, the content of the output records and the content of the printed lines are to a large extent arbitrarily selected. Some effort was made in these selections to include a variety of operations. The execution time of the compute kernel is approximately 30 milliseconds on a 360/50.

There are five input parameters to the synthetic PL/I job:

(a) NMASTER: The number of master records to be created.
(b) R: The number of records to be processed from the master file \((R \leq NMASTER)\).
(c) N: The number of times the compute kernel is to be executed.
(d) L: The number of lines to be printed.
(e) LPR: The number of lines to be printed out for each record processed until the total number of lines of printed output is equal to \(L\).

The four data sets in the PL/I program (MASGENMASTER, OUT1, OUT2, OUT3) are assigned unit and volume designations in the JCL according to the tape and user-disk requirements for the job. Blocking factors for these data sets are also specified in the JCL.

The I/O time reported by the System Management Facility (SMF) routine under OS/MVT (release 18) is based on the number of execute channel programs (EXCPs). The larger the blocking factor, the smaller the number of job EXCPs for a given number of records processed. By increasing both the number of records processed and the blocking factors, it is possible to keep the number of EXCPs for a job constant while increasing the amount of actual I/O taking place. There are many variations that can be applied to the synthetic job. For example, the amount of CPU time used between each I/O operation can, instead of being fixed, be made a random variable subject to the constraint that the total job CPU time is as specified. The degree of sophistication of the synthetic job must be weighed against the resulting complexity of the experiments and the effect on the validity of the results.

MEASUREMENT PROCESS

The running time of the job stream has been calculated as the elapsed time from starting the card reader until all processing is completed. The order in which the jobs are arranged may have some effect on the running time. It is preferable to avoid having a long job running last when there are no other jobs to multiprogram with it. The job stream should be run in two or three different orders and the average time calculated. If the job stream is long enough, the variations will be relatively small. Maximum differences rarely exceeding seven percent have been experienced with a three hour job stream where the average running time of a job is seven minutes.

It is important to start each run with a clean system to ensure consistency; for example, work disk packs should be scratched. Any delays which may be caused by operator intervention, such as mounting user disk packs, should be avoided as far as possible.

COMPARISON OF ACTUAL AND SYNTHETIC JOB STREAMS

A potential disadvantage of a synthetic job stream is that it may not be representative of the actual workload because of the lack of characteristics not explicitly designed into it. The validity of using a synthetic job stream to measure throughput has been confirmed by comparison with a representative job stream. The representative job stream, running for about three hours on a 360/50, was comprised of actual jobs selected from the workload so as to reflect the workload characteristics defined previously. A synthetic job stream was created with each job having the same characteristics (CPU time, I/O time, region size, lines of output, and tape and disk requirements) as the corresponding representative job.

Both job streams have been run on a 360/50 and 360/65, both with and without HASP. The elapsed time for the synthetic job stream runs varied between 71 and 76 percent of the elapsed time for the corresponding representative job stream. Table I shows the activity data obtained with a COMRESS DYNAPROBE hardware monitor. The percentage of time the selector channels were busy with the synthetic job stream running was approximately half that for the representative job stream. This is because the synthetic job

<table>
<thead>
<tr>
<th>TABLE I—Activity Monitored with Hardware Monitor (Minutes)</th>
<th>Synthetic</th>
<th>Representative</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU Active</td>
<td>32.4</td>
<td>34.2</td>
</tr>
<tr>
<td>Supervisor (OS/HASP) Time</td>
<td>22.7</td>
<td>19.0</td>
</tr>
<tr>
<td>Selector Channel 1 Busy</td>
<td>12.4</td>
<td>26.6</td>
</tr>
<tr>
<td>Selector Channel 2 Busy</td>
<td>13.5</td>
<td>30.4</td>
</tr>
<tr>
<td>Selector Channel 4 Busy</td>
<td>14.6</td>
<td>35.1</td>
</tr>
</tbody>
</table>
stream was tuned with the number of EXCPs used to represent I/O activity. More data must be transferred per EXCP in order that the selector channels are kept suitably busy. This can be done by using larger blocking factors in the synthetic jobs. The other activities recorded by the hardware monitor showed close agreement between the two job streams, as can be seen from the figures given in Table I.

Both job streams have been used to determine the relative throughput of a 360/65 with 1024K bytes of core in relation to a 360/50 with 512K bytes of core. The performance of HASP on each of these configurations has also been evaluated using both job streams.

The throughput factor for the 360/65 in terms of the 360/50 was measured with the representative job stream to be 2.18. The throughput factor measured with the synthetic job stream was 2.33. This difference can be attributed to the fact that the synthetic job stream is less I/O bound and was therefore able to make better use of the faster CPU of the 360/65.

The improvement in performance obtained with HASP on each of the two machines as measured by the representative and synthetic job streams is shown in Table II. These measures are in quite close agreement. The deficiency of I/O in the synthetic job stream had little effect in this case since the major performance benefit of HASP is derived from "SYSIN" and "SYSOUT": I/O which did not differ substantially between the two jobs streams.

SUBSEQUENT RESULTS WITH A SYNTHETIC JOB STREAM

Further use has been made of a synthetic job stream to determine throughput factors of three different IBM 360 configurations, to determine relative CPU speeds (as reported by the accounting routine) of these configurations, and to evaluate the effect of a hardware and operating system change on one of these configurations. In order to equitably charge users for attended time on each of the three different configurations, a 360/50-I (512K bytes of core), a 360/65-I (512K bytes of core), and a 360/65-J (1024K bytes of core), a measure of the relative throughput of these machines was needed. A synthetic job stream was used to obtain this measure. In order to assure a true representation of the I/O activity in the synthetic job stream, hardware monitor measurements were made of actual workload channel activity for a two week period and the I/O of the synthetic job stream was made to accurately reflect the true workload I/O. The resulting synthetic job stream had more I/O than the original synthetic job stream and more even than the representative job stream. This indicates that the representative job stream itself had too little actual I/O as compared to the workload. The throughput factor of the 360/65-J in terms of the 360/50-I as measured with the new synthetic job stream was 1.75. This decrease from the throughput factors measured with the representative and original synthetic job streams clearly illustrates the strong dependence of computer performance on the I/O workload characteristics.

The CPU times reported by the accounting routine for synthetic job stream runs on each configuration were compared and CPU factors derived using the 360/50-I as a reference. Table III contains the CPU time for runs on each machine and the CPU factors. It is interesting to note that the CPU of the model 65 with the smaller core appears to be faster than the 65 with the megabyte core. Although it is known that CPU time for a given job does vary from run to run on a given 360, it is statistically significant that all 25 jobs of the synthetic job stream were charged with less CPU time on the 360/65-1 than on the 360/65-J. Two primary factors why the 360 accounting time varies are improper distribution of CPU time for I/O interrupt processing and cycle stealing. These factors are directly proportional to the amount of multiprogramming and I/O activity respectively, both of which are greater on the 360/65-J machine.

The managers of the computer installation felt that the throughput of one of the machines had decreased following a hardware change and a transition to a newer release of the operating system. The hardware

---

**TABLE II—HASP Performance**

<table>
<thead>
<tr>
<th></th>
<th>360/50I</th>
<th>360/65J</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synthetic</td>
<td>126</td>
<td>54</td>
</tr>
<tr>
<td>Representative</td>
<td>166</td>
<td>76</td>
</tr>
<tr>
<td>Time with</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HASP</td>
<td>141</td>
<td>82</td>
</tr>
<tr>
<td>Time without</td>
<td>194</td>
<td>109</td>
</tr>
<tr>
<td>HASP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>% (HASP/ Non-HASP)</td>
<td>89.4</td>
<td>65.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>85.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>69.7</td>
</tr>
</tbody>
</table>

---

**TABLE III—CPU Times and Factors**

<table>
<thead>
<tr>
<th>Machine</th>
<th>CPU Time</th>
<th>CPU Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>360/50-I</td>
<td>3334</td>
<td>1.00</td>
</tr>
<tr>
<td>360/65-J</td>
<td>1070</td>
<td>3.12</td>
</tr>
<tr>
<td>360/65-J</td>
<td>823</td>
<td>4.05</td>
</tr>
</tbody>
</table>
change involved the replacement of a selector channel which was used for tape I/O, by multiplexor sub-channels. Data from the hardware monitor had shown that the selector channel that was removed and the multiplexor channel had very little usage so that a cost savings could be realized with no predicted degradation in performance.

The synthetic job stream previously run on this machine was run in the new environment. Contrary to there being a degradation in performance, the synthetic job stream ran in less time thus showing an increase in throughput for the same workload characteristics.

CONCLUSIONS

The PL/I synthetic program described can be used as the basis for a job stream reflecting the workload characteristics which are considered important in affecting throughput: CPU utilization, I/O channel activity, core requirement, printer output, and tape and disk requirements. Experimentation with a synthetic job stream designed with these characteristics matching those of a job stream comprised of actual jobs has shown comparable results. This supports the choice of characteristics on which the synthetic job stream is based. A hardware monitor is invaluable in determining I/O channel activity in the absence of such a measure for 360 computers.

A synthetic job stream is a more practical tool than a job stream composed of actual jobs for measuring throughput. The synthetic job stream is easier to assemble and does not require extensive data bases. It is particularly advantageous where the regular workload is classified. Synthetic jobs are not dependent on compilers or data management systems and therefore do not require maintenance because of system changes in those areas.

REFERENCES

1 E O JOSLIN
   Computer selection
   Addison-Wesley 1968
2 W L SHOPE K L KASHMARAK
   J W INGHAM W F DECKER
   System performance study
   Proceedings SHARE XXIV March 1970 pp 588-659
3 W BUCHHOLZ
   A synthetic job for measuring system performance
   IBM Systems Journal Vol 8 No 4 1969
APPENDIX

PL/I LISTING OF SYNTHETIC PROGRAM

SYN1: PROCEDURE OPTIONS (MAIN);
    DECLARE L MASTER_REC ALIGNED STATIC,
            2 MASTER_KEY CHARACTER(12),
            2 MASTER_SUM BINARY FIXED(31),
            2 MASTER_CHECK BINARY FIXED(31),
            2 MASTER_DATA (5) CHARACTER(12),
            INTKEY PICTURE 'I6I9'*,
            CHECK BINARY FIXED(31) INITIAL(0),
            PARMS FILE INPUT,
            MASGEN FILE RECORD OUTPUT,
            MASTER FILE RECORD INPUT,
            OUT1 FILE RECORD OUTPUT,
            OUT2 FILE RECORD OUTPUT,
            OUT3 FILE RECORD OUTPUT,
            LINES BINARY FIXED(31) INITIAL(1),
            NREPS BINARY FIXED(31) INITIAL(1),
            N BINARY FIXED(31),
            L MASTER BINARY FIXED(31),
            NMASTER BINARY FIXED(31),
            LPR BINARY FIXED(31),
            LPK BINARY FIXED(31);

    GET R, N, L, NMASTER AND LPR
      GET FILE (PARMS) DATA;
      PUT DATA(R,N,L,MASTER,LPR); PUT SKIP(5);

    /* CREATE NMASTER MASTER RECORDS */
    OPEN FILE(MASGEN); DO J=1 TO NMASTER;
      CHECK = CHECK + J;
      INTKEY = J;
      MASTER_KEY = 'Q00000'; INTKEY;
      MASTER_DATA = '00000'; INTKEY;
      MASTER_CHECK = CHECK;
      MASTER_SUM = 0;
      WRITE FILE(MASGEN) FROM (MASTER_REC);
    END;
    CLOSE FILE (MASGEN);
    OPEN FILE (MASTER);
    OPEN FILE (OUT1);
    OPEN FILE (OUT2);
    OPEN FILE (OUT3);
    DO L1 = 1 TO R;
      READ: READ FILE (MASTER) INTO (MASTER_REC);
    /* EXECUTE KERNEL N-TIMES PER RECORD OR 1 TIME PER RECORD UNTIL N TOTAL REPETITIONS */
    DO L2 = 1 TO ((N-R)/R + 1 + 1) WHILE (NREPS<N);
      DECLARE IX BINARY FIXED(31) INITIAL(13571),
              IY BINARY FIXED(31),
              RN DECIMAL FLOAT;
      ON FIXEDOVERFLOW;
      IY = IX*65539;
      IF IY < 0 THEN IY = IY + 2147483647+1;
      RN = IY;
      RN = RN *.4656613E-9;
      RN = RN * 10.;
      IX = IY;
      INTKEY = SQRT(RN); MASTER_DATA(1) = *000091* INTKEY;
      INTKEY = EXP(RN); MASTER_DATA(2) = *000002* INTKEY;
      INTKEY = LN 3.14; LOG(RN); MASTER_DATA(3) = *000003* INTKEY;
      INTKEY = LN MASTER_DATA(4) = *000004* INTKEY;
      INTKEY = LN MASTER_DATA(5) = *000005* INTKEY;
      MASTER_DATA(6) = MASTER_DATA(6) + MASTER_DATA(1);
      MASTER_CHECK = L1;
      NREPS = NREPS + 1;
    END;
    WRITE: WRITE FILE (OUT1) FROM (MASTER_REC);
    WRITE FILE (OUT2) FROM (MASTER_REC);
    WRITE FILE (OUT3) FROM (MASTER_REC);
    /* PRINT LINES LPR LINES PER RECORD WHILE LINES <L */
    DO K=1 TO LPR WHILE(LINES<L1);
      PUT SKIP DATA(1,L2,LINES,NREPS);
      LINES = LINES + 1;
    END;
    END SYN1;

From the collection of the Computer History Museum (www.computerhistory.org)