Paper SAS406-2014

Empowering the SAS® Programmer: Understanding Basic Microsoft Windows Performance Metrics by Customizing the Data Results in SAS/GRAPH® Software

John Maxwell, SAS Institute Inc.

ABSTRACT

Typically, it takes a system administrator to understand the graphic data results that are generated in the Microsoft Windows Performance Monitor. However, using SAS/GRAPH® software, you can customize performance results in such a way that makes the data easier to read and understand than the data that appears in the default performance monitor graphs.

This paper uses a SAS® data set that contains a subset of the most common performance counters to show how SAS programmers can create an improved, easily understood view of the key performance counters by using SAS/GRAPH software. This improved view can help your organization reduce resource bottlenecks on systems that range from large servers to small workstations.

The paper begins with a concise explanation of how to collect data with Windows Performance Monitor. Next, examples are used to illustrate the following topics:

- converting and formatting a subset of the performance-monitor data into a data set
- using a SAS program to generate clearly labeled graphs that summarize performance results
- analyzing results in different combinations that illustrate common resource bottlenecks

INTRODUCTION

When a SAS program takes longer than expected to run, the reason is often unknown. Using the Windows command line, you can create a list of computer resources to record what is really happening on the computer while a SAS program is running. The Windows Performance Monitor records these resources outside of SAS. However, understanding the graphic data results that are generated in the Microsoft Windows Performance Monitor typically requires the assistance of your system administrator.

Importing this data into SAS® Enterprise Guide® lessens the need to engage your system administer. The SAS Enterprise Guide engine simplifies the task of moving the resource data from a Windows log to a SAS data set. Then you can use SAS Enterprise Guide to generate graphs with SAS/GRAPH software to customize performance results and analyze the performance data at a glance. This enables you to better manage your SAS jobs with respect to system resources.

This paper shows how SAS programmers can use SAS Enterprise Guide and SAS/GRAPH software to create an improved, easily understood view of the key performance counters. Although there are many resources that can be monitored using this approach, this paper shows a subset of the most common performance counters. Note that the name *sasbasic* that is used throughout this paper is a fictitious name to represent a basic group of resources.

The approach that is presented in this paper involves these steps:

- Create a Data Collector Set.
- Record and collect performance data.
- Import and view the data with SAS Enterprise Guide.
- Generate graphs that summarize performance results.

The examples that are shown use command-line access with Windows 7, SAS 9.4, and SAS Enterprise Guide 6. However, the techniques will work on most current operating systems and releases of SAS.

The approach that is introduced here assumes a basic knowledge of customizing graphs and manipulating data in SAS Enterprise Guide.

CREATING A DATA COLLECTOR SET

Windows Performance Monitor is built into all recent Microsoft operating systems. It is often called *perfmon*, which is the command that is used to launch Windows Performance Monitor. Perfmon records how the computer and SAS use resources in Data Collector Sets. A *Data Collector Set* is "…the building block of performance monitoring and reporting in Windows Performance Monitor. It organizes multiple data collection points into a single component that can be used to review or log performance" (Microsoft 2014a).

There are two ways to use perfmon: through a command-line interface, such as logman, or the interactive Performance Monitor application. If you want to use the interactive Performance Monitor application instead of the command-line method, see the "Performance Tools" chapter in the *SAS® 9.4 Companion for Windows, Second Edition* (SAS Institute Inc. 2014a). Logman is a straightforward command-line interface to perfmon for adding counters for performance analysis. It is the method that is used in the examples in this paper.

The first step in creating a Data Collector Set is to create a list of specially formatted performance counters in a text file. In the examples throughout this paper, you will be storing eight performance counters in a Data Collector Set called sasbasic. The eight counters in sasbasic are grouped by one of four basic objects (LogicalDisk, Processor, Paging File, and Memory). Here are the formats for these objects:

```
\LogicalDisk\
\Processor\
\Paging File\
\Memory\
```

Note: Later in this paper you are shown how to use another common object called *Process*. This object is not collected with sasbasic and should not be confused with the object called *Processor*, which looks at CPU.

After the object name, you include specifics about the object. For example, it is important to think about which Logical Disk should be included when collecting data about the file storage system. Note that you should include each file storage system that is used by SAS. Here are the formats for collecting data on the C:\ and D:\ drives:

```
\LogicalDisk(C:)\
\LogicalDisk(D:)\
```

You can also total the drives with an object using this format:

```
\LogicalDisk(_Total)\
```

However, avoid using (_Total) for the Disk object because the average of the drives will hide important details.

Next, add the counter that will measure performance for the areas in which you are interested. For example, here is the format for the Current Disk Queue counter for the LogicalDisk object that will collect data on the C:\ drive:

```
\LogicalDisk(C:)\Current Disk Queue
```

If you also collect the same data on the D:\ drive, add this format:

```
\LogicalDisk(D:)\Current Disk Queue
```

There are many advanced metrics that you can use to pinpoint performance bottlenecks and hardware and operating system issues in a Windows operating environment. The counters that are recommended by SAS Technical Support for monitoring Windows performance are listed in SAS Note 52017 "Suggested perfmon counters for technical support" (SAS Institute Inc. 2014b).

Open Windows Notepad and add the list of counters that you want to collect. For this example, add the following:

```
\LogicalDisk(C:)\Current Disk Queue Length
\LogicalDisk(C:)\Disk Read Bytes/sec
\LogicalDisk(C:)\Disk Write Bytes/sec
\Processor(_Total)\% Processor Time
\Processor(_Total)\% User Time
\Memory\Available MBytes
\Memory\System Cache Resident Bytes
\Paging File(_Total)\% Usage
```

Next, save the file as any filename with a .txt extension. For this example, name the file counters.txt.

The order of counters in this file will be the same order that is used by SAS Enterprise Guide when you import the data. This might be helpful later, as you will see in the "Importing and Viewing the Data with SAS Enterprise Guide" section, if you need to rename the columns of data when you import the data to create customized graphs.

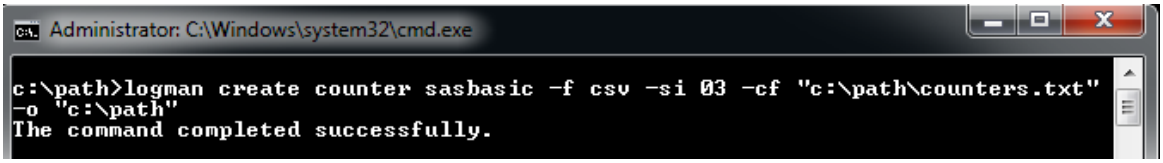
Notice that the counters that you are using do not start with the machine name. This is to ensure that the current machine is used. If the machine name was included, it would look like this:

```
\\machine-name\object(instance or index)\counter
```

Machine-name is the name of the actual machine where you want to collect counters. As long as you run the **`logman create`** command on the same machine that will be collecting data, you can omit the machine name. Doing so allows the counters.txt file to be portable from machine to machine.

You are ready to create the Data Collection Set. Open a command prompt (**`cmd.exe`**) and use the **`logman create`** command to generate the Data Collector Set using a command like this (Display 1):

```
logman create counter sasbasic –o "c:\path\for\csvfilename" -f csv -si 03 -cf
"c:\path\to\counters.txt"
```



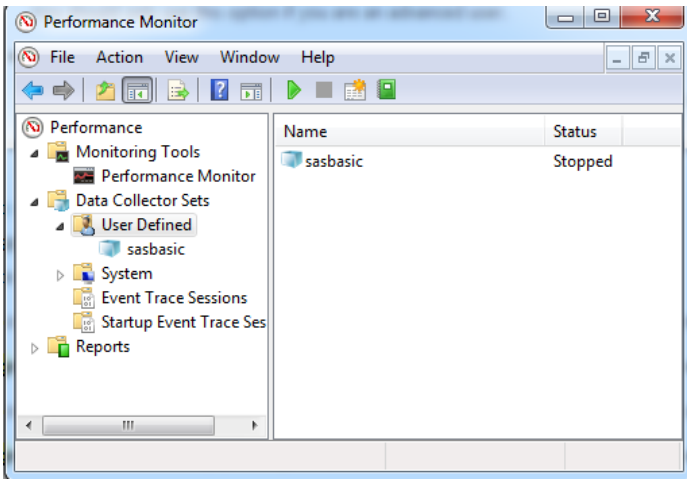
Display 1. Using the `logman create` command in the Command Prompt Window

In this command, **`logman`** is called to **`create`** a Data Collector Set of **`counter`** (s) called **`sasbasic`** where the output filename **`(-o)`** is *csv-file-name*.csv, the file type **`(-f)`** is comma-separated values **`(csv)`**, the sample interval **`(-si)`** is every **`03`** seconds, and the counter file list **`(-cf)`** is **`counters.txt`**.

Depending on what is monitored, you can alter this command to collect performance data more or less frequently or to create a binary (BLG) file instead of a comma-separated values (CSV) file. SAS Technical Support typically requests a BLG file, but a CSV file is required for the examples in this paper. For example, if you wanted the Data Collector Set to collect the status of the machine every hour, you would change the **`–si`** argument to **`-si 01:00:00`**. That interval would be useful for obtaining a high-level overview of a machine over weeks without creating a large log. The example in this paper is using a 3-second sample interval, which is a typical interval to use when monitoring SAS.

After you submit the command, you see "`The command completed successfully`." This indicates that the perfmon Data Collector Set called sasbasic is now created and ready to use.

The user interface to perfmon will show that sasbasic has been created under Data Collector Sets in the User Defined container (Display 2).



Display 2. The sasbasic Data Collection Set Is Created in the Performance Monitor

Sasbasic will persist on the machine, so you do not need to run the **`logman create`** command again until you need to use a different Data Collector Set. You do not have to open perfmon. This step is just to illustrate that the collection set was created in the interactive interface of Performance Monitor. This step also helps show that Performance Monitor and logman are the same tool with different interfaces.

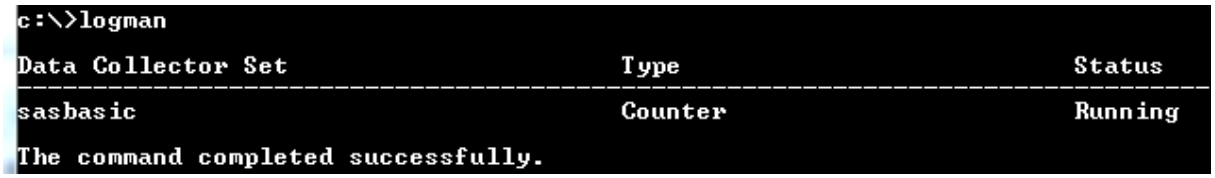
RECORDING AND COLLECTINGPERFORMANCE DATA

Now that you have created your sasbasic Data Collection Set, you are ready to record performance. Start the performance data collection with logman referencing the sasbasic Data Collector Set (Display 3):

```
logman start sasbasic
```

Display 3: Starting the Performance Data Collection in the Command Prompt Window

You are now collecting performance data. You can verify this by running logman without any options. It will return the Data Collector Sets that you have created and will show a status of **`Running`** (Display 4). This is a good way to get a list of Data Collector Sets and see which ones are running.



Display 4. Using the `logman` Command in the Command Prompt Window to Identify Data Collector Sets

For more advanced monitoring, you might often include counters with the Process object. You should start the SAS programs that are going to be collected before you start the Data Collector Set if the Data Collector Set contains a Process object with a wildcard, such as the following:

```
\Process(*)\% User Time
```

Performance information about the SAS process will not be included if you start the SAS process after you start the Data Collector Set. Sasbasic does not contain a Process object, so you can start the SAS program before or after the SAS programs that are monitored.

The example below is included in the "Performance Tools" section of the *SAS® 9.4 Companion for Windows, Second Edition* (SAS Institute Inc. 2014a). It is imperative that you not use sample programs like this as a standard or point of performance reference. You should use real-life production code for performance testing.

```
data a (drop=s);
   do i = 1 to 5e7;
      x = ranuni(i);
      y = x*2;
      z = exp(x*y);
      output;
   end;
      /* The sleep helps to delineate the subsequent  */
      /* sort in the Performance Monitor graph.       */
   s = sleep(15);
run;

proc sort data = a noduplicates;
   by z y x i;
run;
```

This sample code is sufficient for using enough system resources to demonstrate how to collect performance data. Different SAS procedures and statements will stress different machine resources; this is why you should use real-world data and production code when analyzing Performance Monitor results.

Long sleep statements between SAS steps within a program help mark boundaries when you are looking at a graph (Figure 1). This works because the system will have reduced activity while sleeping.

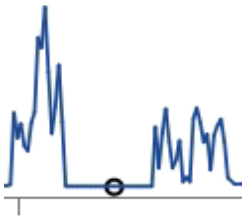


Figure 1. A Graph Showing Long Sleep Statements between SAS Steps

So the dip in the middle of this graph might be when **`s = sleep(15);`** was running from the step above.

After the program or activity has completed, use the **`logman stop`** command to stop the performance data collection (Display 5):

```
logman stop sasbasic
```

```
c:\path>logman stop sasbasic
The command completed successfully.
```

Display 5. Using the `logman stop` Command to Stop Performance Data Collection in the Command Prompt Window

You can use logman without any options to see whether the status of the collector set is stopped.

Now the performance metrics are captured in the CSV file, which was specified with the **`-o`** option when the Data Collector Set was created. The **`logman create`** command used **`-f`** to create a CSV file, which is now ready to be imported into SAS Enterprise Guide. The default format for perfmon data is BLG. This is often the preferred format when you are sending information to other parties for analysis. For example, SAS Technical Support will typically ask for the data in BLG format. There might be situations where a BLG file is the only file available. If you have only a BLG file, the Microsoft **`relog`** command can be used to convert the file to CSV. Here is an example from the command prompt (Display 6):

```
relog C:\path\sasbasic_000001.blg -f csv -o d:\path\basic1.csv
```

```
Administrator: C:\Windows\system32\cmd.exe

c:\path>relog C:\path\sasbasic_000001.blg -f csv -o c:\path\sasbasic.csv

Input
----------------
File(s):
     C:\path\sasbasic_000001.blg (Binary)

Begin:    1/7/2014 11:02:59
End:      1/7/2014 11:07:29
Samples:  19

100.00%

Output
----------------
File:     c:\path\sasbasic.csv

Begin:    1/7/2014 11:02:59
End:      1/7/2014 11:07:29
Samples:  19

The command completed successfully.

c:\path>
```

Display 6. Using the `relog` Command to Convert a BLG File to a CSV File in the Command Prompt Window

The **relog** command has many other uses, such as grabbing a subset of the time range originally used or grabbing every nth value. Additional information about the **relog** command is available on the Microsoft "Relog" web page (Microsoft 2014b).

IMPORTING AND VIEWING THE DATA WITH SAS® ENTERPRISE GUIDE®

Open a new project in SAS Enterprise Guide and select **File ► Import Data**. Then navigate to the CSV file that you created. Use the SAS Enterprise Guide Import Data Task to correctly import the CSV file. Be sure to select **Rename columns to comply with SAS naming conventions** on Page 2 of the Import Wizard (Figure 2).

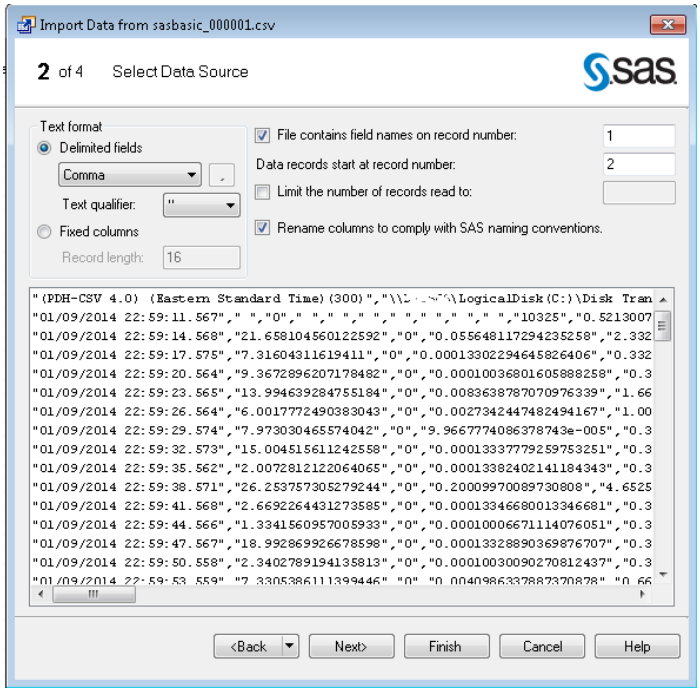


Figure 2. Selecting `Rename columns to comply with SAS naming conventions` in the Import Wizard

You can either change the column names and labels in the Modify step of the Import Wizard or keep the names the same and do it after the import is complete. You should rename the columns because the default names that are used by perfmon will be truncated. For example, the first column **_PDH_CSV_4_0_Eastern_Standard** is renamed **Time** in Figure 3.

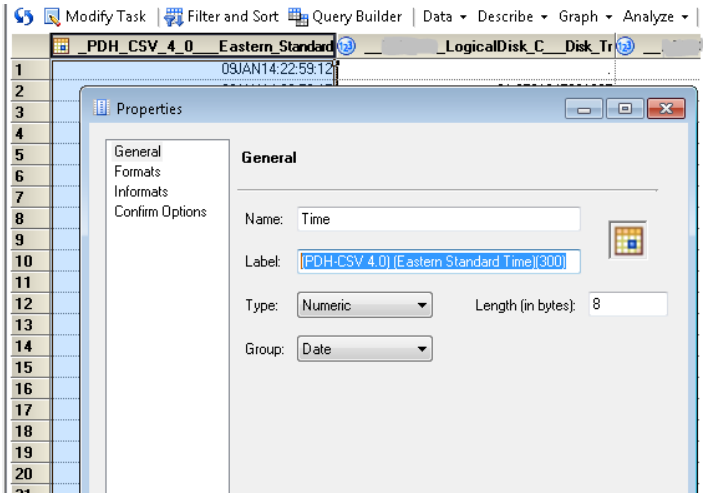
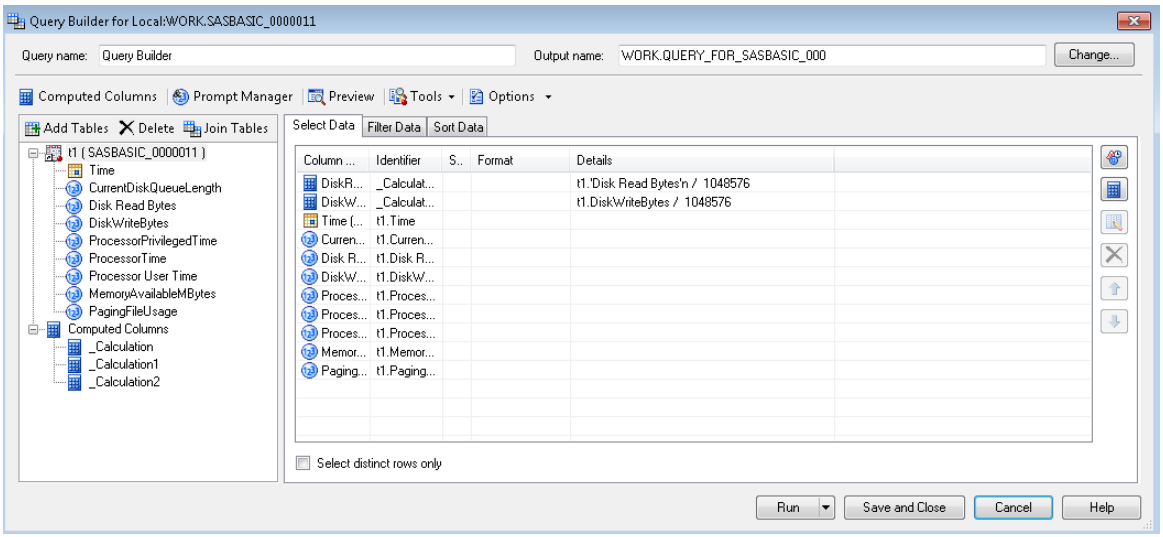


Figure 3. Renaming the First Column as `Time` in the Import Data

This ensures that the graphs are clearly labeled and also helps when choosing columns. Remember, the columns will be in the same order as they are in the counters.txt file that was used to create the Data Collection Set.

Here are some steps to help make the data more meaningful. In this example, Query Builder was used to change DiskRead and DiskWrite bytes per second to DiskRead and DiskWrite megabytes per second (Display 7).



Display 7. Changing How Data Is Displayed in the Query Builder

Windows Performance Monitor does not have the option to record Read and Write times in megabytes, so the Query Builder **`Computed Columns`** feature is used to make this conversion. The original column is divided by 1,048,576 bytes to convert it to megabytes.

Once the data is formatted as desired, it is time to graph the results.

GENERATING GRAPHS THAT SUMMARIZE PERFORMANCE RESULTS

The monitoring resources that are discussed in this section are a subset of the most common performance counters.

To begin creating graphs, select the imported data set in your Process Flow, and then click **Select Task ► Graph ► Line Plot**. This will write the SAS/GRAPH code for you.

LOGICALDISK COUNTERS

Figure 4 shows a line plot with the newly converted DiskReadMB and DiskWriteMB combined on the same graph.

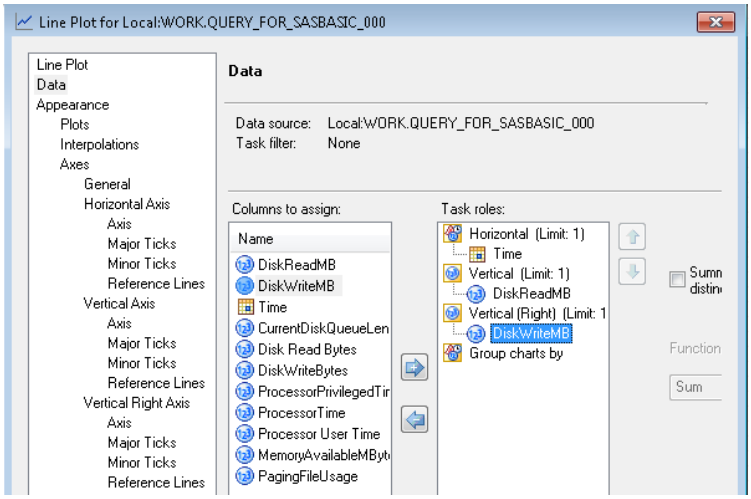


Figure 4. The Line Plot for DiskReadMB and DiskWriteMB

The file system throughput is the most common bottleneck for SAS programs and the most difficult to find unless you use a tool such as perfmon to collect the performance data. Here, the Read times were much slower than the healthy Write times, which are close to 80 MB per second (Figure 5). If the same scale (0-80 MB/sec) was used for the Read times, it would look flat.

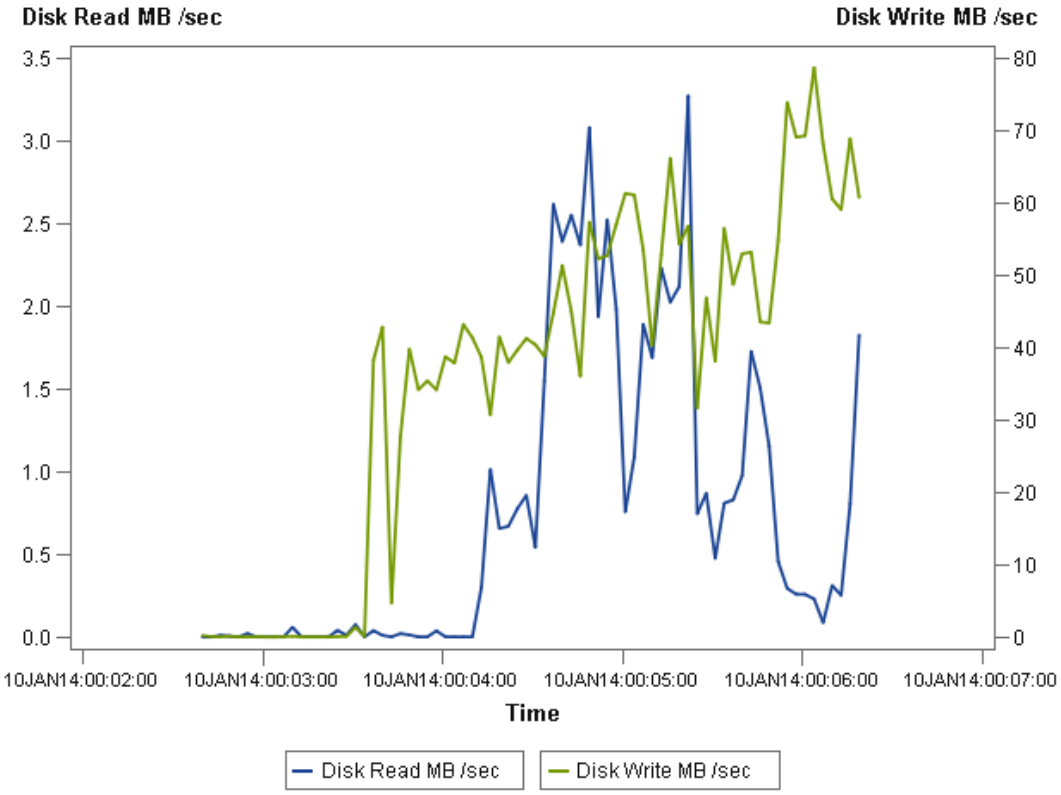


Figure 5. Read Times for Disk Read and Disk Write

SAS Enterprise Guide automatically scales to Read times to 0-4 MB/sec on the left. The different scales make it easier to see that Read times are not sustained at a maximum value in this example. This indicates that the file system could be capable of reading more megabytes per second. Note that the maximum value that is recorded in the perfmon log for Disk Read and Disk Write does not necessarily mean this is as fast as this disk system can read or write.

If the SAS log shows a large difference between real time and CPU time and the graph plateaus for a sustained amount of time at some maximum Read and Write value, the disk could be a bottleneck. When this happens, it is time to look at more counters, such as the following:

```
\LogicalDisk(C:)\Current Disk Queue Length
```

A needle plot from the **`Line Plot`** selection in SAS Enterprise Guide was chosen to show information about Current Disk Queue Length (Figure 6).

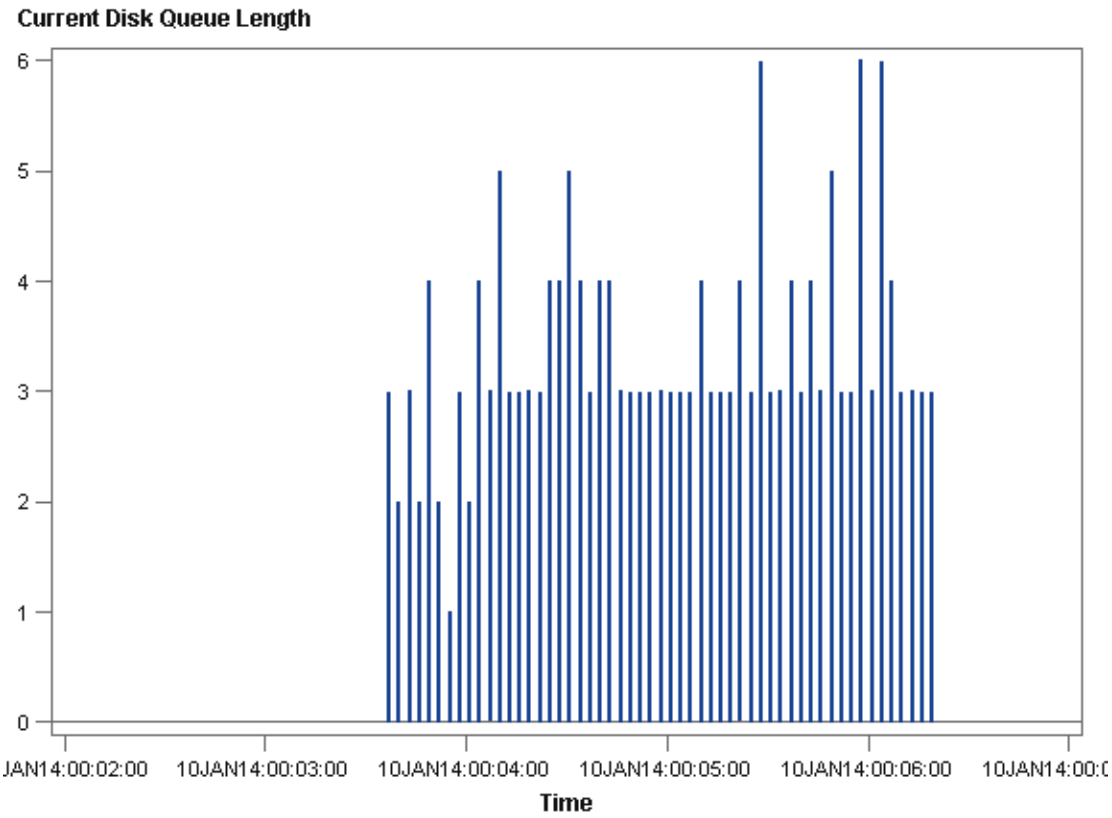


Figure 6. A Needle Plot for Current Disk Queue Length

Current Disk Queue Length is a quick snapshot in time of the Read and Write request that are in the queue. As a rule, you do not want sustained values higher than 2 Disk Queues.

It is okay if the Current Disk Queue Length values briefly reach large numbers. The large numbers are reached because the current queue length could include input or output in transit that is not really waiting. The needle plot makes it easy to see this example at a glance. Three disk queues are sustained once a significant amount of input and output occur. If this needle plot shows data for a workstation, then there are no apparent problems. If you wanted to improve performance more, you could add another drive for the SAS temporary WORK library. If Current Disk Queue Length becomes large and stays large, then it is time to consult the advanced principles that are listed in SAS Note 42197 "A list of papers useful for troubleshooting system performance problems" (SAS Institute Inc. 2011).

If you add another drive to the system, you should create a new Data Collector Set that also includes counters for the new drive, as shown below:

```
\LogicalDisk(D:)\Current Disk Queue
```

Then rerun the test to see whether the Current Disk Queue Length is better for sustained periods.

MEMORY COUNTERS

In this next example, available memory and paging file usage are combined (Figure 7).

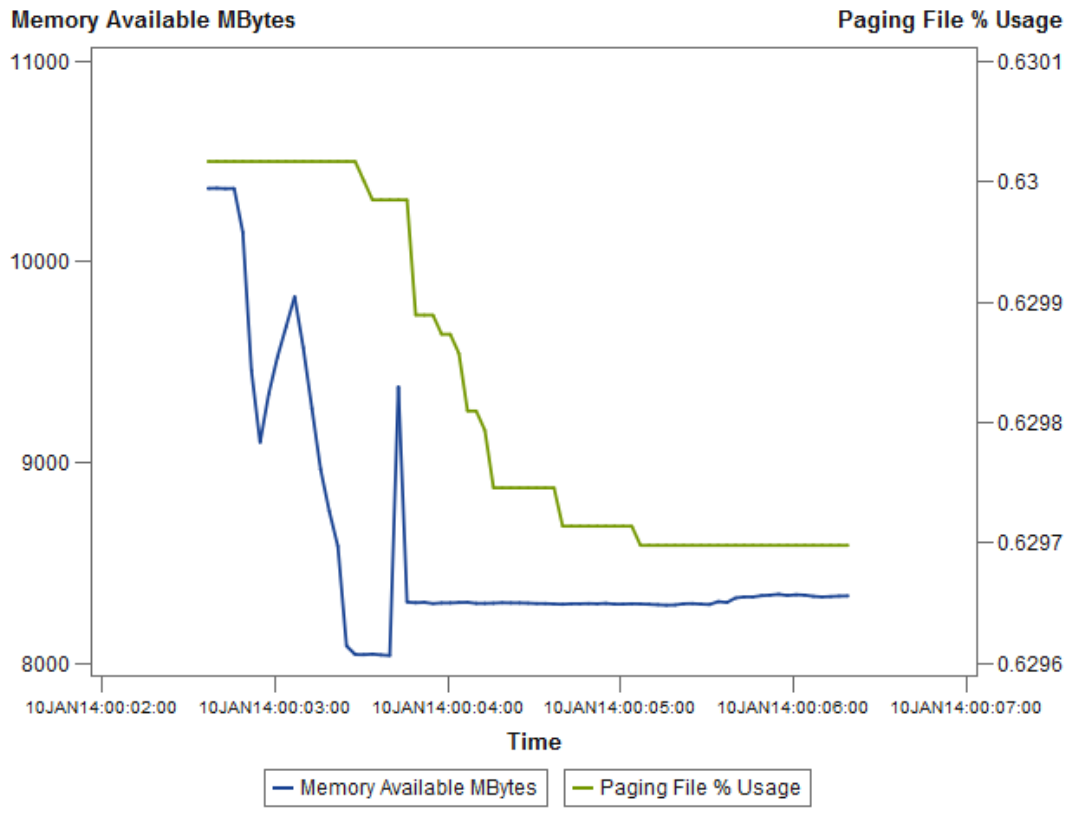


Figure 7. Memory and Paging File Usage

The paging file usage that is displayed in Figure 7 is normal, and plenty of memory remains through the run. This indicates that there is not a problem with memory. If the paging file was used more heavily, the available memory would probably also approach zero at the same time, indicating a memory constraint.

If a machine is running out of memory, one obvious idea is to add more memory. Consider which processes are using the memory and why. There are solutions to known memory issues, such as the problem with Microsoft Cache Manager described in SAS Note 39615 "Input/output performance in SAS is degraded due to excessive memory usage on Windows" (SAS Institute Inc. 2010). If memory problems are unexpected and related to a SAS program, it is a good idea to contact SAS Technical Support to help determine why you have these results.

PROCESSOR/CPU COUNTERS

The processor total graph in Figure 8 demonstrates a general overview of the processor usage over time.

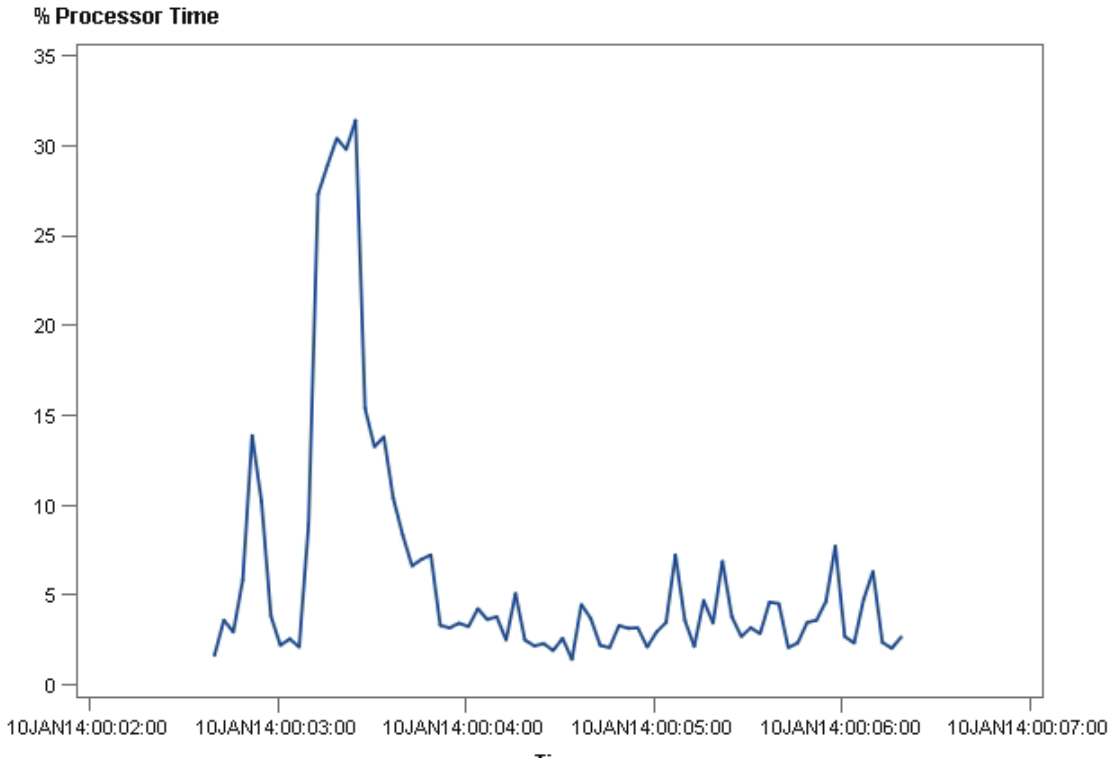


Figure 8. Processor Usage over Time

In this example, the processor usage does not exceed 35%. Sustained high levels of processor activity signify a potential bottleneck. If there are multiple processors, then one idle processor could hide a problem when it is averaged with the total. For this type of problem, look at each processor by adding the following to the counters.txt file:

```
\Processor(*)\% Processor Time
```

The wildcard (*) records every processor instance. If other problems are ruled out and processor usage is very high, this normally means that the processor is not adequate for the selected task.

CONCLUSION

It is very common for a SAS programmer to also have to be their own system administrator. Creating counters and Data Collection Sets with logman is an easy way to find data on performance metrics. Importing this data into SAS Enterprise Guide helps bridge the gap between the SAS programmer and system administer. SAS/GRAPH provides ways to analyze the performance data at a glance, which gives programmers the power to know how to better manage their SAS jobs with respect to system resources. It can also help you decide when the system can handle more concurrent jobs and which jobs can be paired to run at the same time or scheduled for a slow time.

This method also provides the framework for the SAS statisticians to work in a comfort zone of data sets and SAS graphs that go beyond using a simple line graph or dividing a column by megabytes. SAS and SAS Enterprise Guide offer infinite possibilities for analyzing your Windows performance data.

REFERENCES

Microsoft. 2014a. "Creating Data Collector Sets". Microsoft TechNet Library. Available at technet.microsoft.com/en-us/library/cc749337.aspx.

—. 2014b. "Relog." Microsoft Technet Library. Available at technet.microsoft.com/en-us/library/bb490958.aspx.

SAS Institute Inc. 2014a. "Performance Tools". *SAS® 9.4 Companion for Windows, Second Edition*." Available at support.sas.com/documentation/cdl/en/hostwin/67241/HTML/default/viewer.htm#n0dhcfs54ax4kkn1t6bmm3mrjsq3.htm.

—. 2014b. SAS Note 52017: "A Complete list of SAS Recommended Performance Counters." Available at support.sas.com/kb/52/017.html.

SAS Institute Inc. 2010. SAS Note 39615 "Input/output performance in SAS is degraded due to excessive memory usage on Windows." Available at support.sas.com/kb/39/615.html.

RECOMMENDED READING

The resources in this section provide suggestions for performance tuning and guidelines and settings for improving performance. You can also find information about comparing the history of graphs so you know whether SAS setting changes are having an impact on system resources.

Crevar, Margaret and Brown, Tony. 2011. "Best Practices for Configuring your IO Subsystem for SAS®9 Applications." Available at support.sas.com/rnd/papers/sgf07/sgf2007-iosubsystem.pdf.

SAS Institute Inc. 2011. SAS Note 42197: "A list of papers useful for troubleshooting system performance problems." Available at support.sas.com/kb/42/197.html.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

John Maxwell
SAS Institute Inc.
Work Phone: 919-677-8008
E-mail: support@sas.com
Web: support.sas.com