

DB2 self-tuning memory manager log parser

Skill Level: Intermediate

[Askari Naqvi \(askarin@ca.ibm.com\)](mailto:askarin@ca.ibm.com)

Software Development Analyst
IBM

[Christian Garcia-Arellano \(cmgarcia@ca.ibm.com\)](mailto:cmgarcia@ca.ibm.com)

Software Developer
IBM

[Haysam Alsayed \(halsayed@ca.ibm.com\)](mailto:halsayed@ca.ibm.com)

Software Tester
IBM

[Adam Storm \(ajstorm@ca.ibm.com\)](mailto:ajstorm@ca.ibm.com)

Software Developer
IBM

09 Aug 2007

Updated 23 Oct 2008

Starting in IBM® DB2® for Linux®, UNIX®, and Windows® 9, a new memory tuning feature, self-tuning memory manager (STMM), simplifies the task of memory configuration by automatically setting values for several critical memory configuration parameters. This article introduces a simple tool to parse the STMM log files to simplify the task of monitoring the decisions made by the memory manager. It has been updated to describe significant improvements to the log parser.

Important: Read the [disclaimer](#) before reading this article.

Introduction

When enabled, the memory tuner dynamically distributes available memory resources among several memory consumers, including sort heap, package cache, lock list, and buffer pools. The feature works by iteratively modifying the memory configuration in small increments with the goal of improving overall system performance.

All the changes made by STMM are logged in two places: the db2diag.log and the STMM log files. The following describes the contents of both log files and how to monitor the changes made by STMM using the db2diag tool and the parseStmmLogFile.pl tool.

How does STMM work?

STMM makes its decisions with the help of new internal metrics that predict the effect that additional memory has for a given heap. These metrics, when combined with STMM's advanced tuning algorithms, can in most cases, tune a system from an out-of-the-box configuration to near-optimal memory usage in an hour or less. However, in tuning the system STMM can make hundreds of tuning decisions, each of which result in a change to a configuration parameter or buffer pool size. Since only the most recent of these changes is reflected in the configuration files, determining the historic values for configuration parameters or buffer pools requires an investigation of the db2diag.log and STMM log files.

Monitor the changes in the db2diag.log file

The db2diag.log file is the repository for simple information on each configuration change made by the memory manager. In this file, you see records that indicate a configuration change such as shown in Listing 1:

Listing 1. Record indicating a configuration change

```
2006-10-17-19.10.00.912218-240 I408210A457          LEVEL: Event
PID      : 946302                TID   : 1          PROC  : db2stmm (MYDB1) 1
INSTANCE: ewhhr                 NODE  : 001        DB    : MYDB1
APPHDL   : 1-52                 APPID : *N1.cgarciaa.060809150048
AUTHID   : CGARCIAA
FUNCTION : DB2 UDB, config/install, sqlfLogUpdateCfgParam, probe:20
CHANGE   : STMM CFG DB DEWHR000: "Sheapthres_shr" From: "109306" <automatic>
                                         To: "105115" <automatic>
```

Note that the record above prefaces the configuration change with "STMM CFG" to indicate that this change originated from STMM as opposed to a user-initiated configuration update. You also see records that indicate a buffer pool change such as this shown in Listing 2:

Listing 2. Record indicating a bufferpool change

```
2006-10-17-19.03.58.672185-240 I395047A488          LEVEL: Event
PID      : 946302                TID : 1          PROC : db2stmm (MYDB1) 1
INSTANCE: ewhhr                 NODE : 001
APPHDL   : 1-52                 APPID: *N1.cgarciaa.060809150048
AUTHID   : CGARCIAA
FUNCTION: DB2 UDB, buffer pool services, sqlbAlterBufferPoolAct, probe:90
MESSAGE  : Altering bufferpool "BUFFERPOOL_16K" From: "117268" <automatic>
                                                To: "109666" <automatic>
```

These records can be easily filtered out from the db2diag.log with the help of the db2diag tool. For example, the following command can be used to look for all the changes to the buffer pool size:

Listing 3. db2diag command to look for bufferpool changes

```
db2diag -g "message:=Altering bufferpool" db2diag.log
```

In the case of a database with multiple partitions using the data partitioning feature of DB2, the changes made on each of the partitions can be filtered out using the `-node` option. For example, the following command filters out all the database configuration updates on partition 1:

Listing 4. db2diag command to look for configuration changes

```
db2diag -node 1 -g "changeevent:=CFG DB" db2diag.log
```

The STMM log

In addition to the log entries in the db2diag.log file, changes are also logged in much greater detail in the STMM logs, which are stored in a directory named stmmlog, in the same directory as the db2diag.log file. The STMM logs are mostly meant to be used by DB2 support for problem determination. However, some of the tuning information in STMM logs could help DBAs understand the tuning decisions made by STMM. Each entry in the STMM log records the statistics collected before making a tuning decision and the actions performed based on those statistics. The STMM log is split into a maximum of five files, each of which has a maximum size of 10MB. These log files are maintained in a circular fashion, always removing the oldest one before creating a new file.

The STMM log file parser has the goal of filtering out important tuning information and formatting it so that the evolution of the memory configuration can be easily discerned.

Parse the STMM log file

The parsing tool `parseStmmLogFile.pl` presented here has the following syntax:

Listing 5. `parseStmmLogFile.pl` syntax

```
parseStmmLogFile.pl <log file> <database name> <options>
```

The output generated by the tool is in the form of a table. The first four columns of this table are the same regardless of the options chosen by the user. The first four columns are as follows:

- The tuning interval number
- The time at which the tuning interval occurred
- The total number of seconds from the first tuning interval in the log file
- The amount of time in seconds that had elapsed since the beginning of the previous tuning interval. This value includes the time spent resizing the memory heaps and the time spent collecting statistics to make the tuning decision.

Listing 6 shows an example of the first four columns with two tuning intervals.

Listing 6. `parseStmmLogFile.pl` sample output

```
[ MEMORY TUNER - LOG ENTRIES ]
[ Interv ] [ Date ] [ totSec ] [ secDif ]
[ 1 ] [ 02/01/2006 09:45:02 ] [ 76 ] [ 76 ]
[ 2 ] [ 02/01/2006 09:46:03 ] [ 137 ] [ 61 ]
```

Based on the option selected, the tool parses through the STMM log file and collects specific details for the options specified. There are a total of four options available to choose from:

- (s) — Displays each consumer's new size (the default option).
- (m) — Displays each consumer's minimum size.
- (b) — Displays each consumer's benefit data. The benefit data is an indication of how much each consumer would benefit from additional memory.
- (o) — Displays DATABASE_MEMORY tuning information.

Additionally, there are two optional flags (which if specified, must be supplied in addition to one of the flags above) that modify the output:

- (4) — Converts all memory consumers to 4KB pages in size.
- (d) — Produces output that is delimited with semicolons. This option is useful if you are planning on inputting the parser output into a spreadsheet.

Example 1. History of heap resizes

This option is used to display the configuration parameters and buffer pools changes performed by STMM.

In Listing 7, the `parseStmmLogFile.pl` command displays the information from two tuning intervals. The first tuning interval started 76 seconds after the STMM log file is created, and the two parameters being tuned are `SHEAPTHRES_SHR` and `PCKCACHESZ`. The second line displays the second tuning interval that started 61 seconds later, and results in a transfer of 1000 pages from `PCKCACHESZ` to `SHEAPTHRES_SHR`.

Listing 7. Sample output for tuning intervals

```
$ parseStmmLogFile.pl stmm.0.log mydbname s

[ MEMORY TUNER - LOG ENTRIES ]
[ Interv ] [ Date ] [ totSec ] [ secDif ] [ newSz ]
[ ] [ ] [ ] [ ] [ SHEAPTHRES_SHR PCKCACHESZ ]
[ 1 ] [ 02/01/2006 09:45:02 ] [ 76 ] [ 76 ] [ 31482 19438 ]
[ 2 ] [ 02/01/2006 09:46:03 ] [ 137 ] [ 61 ] [ 32482 18438 ]
```

Example 2. History of database memory resizes

The following command is used to output the basic information about the database memory tuning decisions. This includes a column indicating the total amount of memory on the system as determined by the memory tuner (`configMem`), the amount of physical memory available to be used by DB2 (`memAvail`), and the current size of the database shared memory set that is sized using the `DATABASE_MEMORY` configuration parameter (`setConfSz`).

Listing 8. Sample output for database_memory tuning

```
$ parseStmmLogFile.pl stmm.0.log mydbname o

[ MEMORY TUNER - DATABASE MEMORY AND OVERFLOW BUFFER TUNING - OG ENTRIES ]
[ Interv ][ Date ] [ totSec ][ secDif ][ configMem ][ memAvail ][ setCfgSz ]
[ 1 ][ 02/01/2006 09:45:02 ][ 76 ][ 76 ][ N/A ][ N/A ][ N/A ]
[ 2 ][ 02/01/2006 09:46:03 ][ 137 ][ 61 ][ 4194304 ][ 1559966 ][ 62224 ]
```

Example 3. History of SORTHEAP resizes

This command is used to output a summary of the information used to tune the value of the SORTHEAP configuration parameter. Each line represents a successful automatic update to the value of the SORTHEAP value. The columns include the previous value of the SORTHEAP configuration parameter (OLD), the current value (NEW), the minimum and maximum values computed by the memory tuner (min and max).

Listing 9. Sample output for SORTHEAP tuning

```
$ parseStmmLogFile.pl stmm.0.log mydbname v
[ SORTHEAP TUNING - SORTHEAP CHANGE VALIDATION RECORDS ]
[ Date           ][ totSec ][ secDif ][ SHEAPTHRES_SHR ][ OLD ][ NEW ][ min ][ max ]
[ 02/01/2006 14:51:01 ][ 184 ][ 184 ][ 11212 ][ 373 ][ 560 ][ 224 ][ 2243 ]
```

Hints and Tips

- Ensure that the database name specified upon executing the tool exists in the STMM log file.
- For optimal results, specify only one option (m, s, or o) for each execution of the tool to ensure that the results are easily readable.
- If no option is specified upon execution, the default displays the new size or s option.
- A detailed list of options is provided inside the script itself, including the same examples presented above.
- The tool requires a Perl interpreter to be installed on the system where it is run. If the Perl interpreter is not present on your system, you can download one from <http://www.perl.org>. Please make sure you check your organizations' policies on use of third party software before downloading and installing this software.
- The tool was developed in the Perl scripting language to allow DBAs to modify it for their specific needs. For example, it can be modified to use a different format that can be imported into other tools, for example, to allow the plotting of the historical data.

Disclaimer

IBM makes no representations, warranties, or other commitments whatsoever about any non-IBM Web sites or third-party resources that may be referenced, accessible

from, or linked from this document. A link to a non-IBM Web site does not mean that IBM endorses the content or use of such Web site or its owner. In addition, IBM is not a party to or responsible for any transactions you may enter into with third parties, even if you learn of such parties (or use a link to such parties) from an IBM site. Accordingly, you acknowledge and agree that IBM is not responsible for the availability of such external sites or resources, and is not responsible or liable for any content, services, products, or other materials on or available from those sites or resources. Any software provided by third parties is subject to the terms and conditions of the license that accompanies that software.

Downloads

Description	Name	Size	Download method
Tool to parse the STMM log files	parseStmmLogFile.4k	4KB	HTTP

[Information about download methods](#)

Resources

Learn

- "[Self-tuning Memory Manager Roadmap](#)": Find additional information about DB2 V9 Self-tuning Memory Manager.
- "[Self-tuning Memory Manager Whitepaper](#)": Read more about STMM.
- [developerWorks Information Management zone](#): Learn more about DB2. Find technical documentation, how-to articles, education, downloads, product information, and more.
- Stay current with [developerWorks technical events and webcasts](#).

Get products and technologies

- Build your next development project with [IBM trial software](#), available for download directly from developerWorks.

Discuss

- Participate in [developerWorks blogs](#) and get involved in the developerWorks community.

About the authors

Askari Naqvi

Askari Naqvi is a member of the DB2 organization at the IBM Toronto Lab. In the last six years, he has worked on several critical features and enhancements for DB2 across multiple versions. Some of the notable projects include DB2 Satellite, DB2 Query Patroller, Autonomics, Automatic Tablespaces, and STMM. Currently, he has been focusing on Workload Management and improved Storage Management in DB2. Askari has an MS in Computer Science from McGill University, Montreal.

Christian Garcia-Arellano

Christian Garcia-Arellano is a member of the DB2 Engine Development team at the IBM Toronto Lab. In the last five years, he has mainly worked on several memory related enhancements for DB2, including the design and development of the STMM. Other projects have included enhancements to the configuration advisor, sort memory management, and exploring the effects of online memory tuning on a running workload. He has also worked on autonomic projects like the design advisor.

Haysam Alsayed

Haysam Alsayed is a member of the DB2 System Verification Team at the IBM Toronto Software Lab. Haysam recently joined IBM in May 2006 and was studying Software Engineering at the Schulich School of Engineering at the University of Calgary. Up until now, Haysam has focused his efforts on testing the DB2 Connect product with various host servers such as z/os and iSeries. Haysam has also been involved in testing DB2 9, which includes focusing on integrating features of STMM, DB2 Connect, and Security.

Adam Storm

Adam Storm is a member of the DB2 Engine Development team at the IBM Toronto Lab. In the last seven years, he has worked on several memory related enhancements for DB2. These projects have included enhancements to the DB2 Configuration Advisor, the creation of the DB2 Memory Tracker, and work on the DB2 Memory Visualizer. He has also worked several other Autonomic projects at IBM including the db2support tool and the DB2 Design Advisor. Most recently, Adam was the lead developer on the STMM for DB2 9.