IBM **Innovation Center**
for SAP® Solutions

IBM **Power**

Planning Guide

Using AIX 64K Active Memory Expansion with SAP

IBM Deutschland Research & Development GmbH
Created on March 21st, 2023 – Version 1.0

Note: Before using this information and the product it supports, be sure to read the general information under "Copyrights and Trademarks" as well as "Disclaimer and Special Notices".

Edition Notice and Version Information

IBM Corporation
New Orchard Road
Armonk, NY 10504

Edition Notice: this is version 1.0 of this document.
Focus: IBM AIX® Active Memory Expansion (AME) 64K and SAP
Target:

- SAP NetWeaver Application Server ABAP on AIX
- SAP S/4HANA Application Server ABAP on AIX

Document Version	Changes
1.0	Initial version

Preface

Running SAP application servers on AIX offers enterprise customers the reliability and flexibility that they need to keep their business running smoothly.
One of the unique features of AIX is Active Memory Expansion (AME). It allows customers to configure their AIX servers with less real memory than actually required by the SAP application - letting AME do the memory expansion under the covers. AME is transparent to applications and as a net result it can be very beneficial in memory constrained systems.
This paper introduces and evaluates the 64K variant of AME for use with SAP.

About This Document

This document is intended for architects and specialists planning for SAP to run on AIX servers. It gives guidance how to plan and configure 64K Active Memory Expansion.
This guide includes results of performance tests with AME and SAP NetWeaver application servers running on AIX on Power8, Power9 and Power10 processor–based systems.

Go to the SAP topic group at the IBM Power Community site for up-to-date materials complementary to this guide.

The most recent version of this document can be found here: https://www.ibm.com/support/pages/node/6962405

Feel free to provide feedback and change requests for this document via email to enable.SAP@de.ibm.com.

To view the SAP Notes that are referenced in this document you need a valid SAP user ID.

Table of Contents

Figures

Tables

Overview - Active Memory Expansion and SAP

Active Memory Expansion (AME) is a separately purchasable feature of IBM® Power® servers that has been available for a long time. You can check in the Hardware Management Console (HMC) if it is available on your server.
If it is available on the server, then you can switch it on for any AIX LPAR on that server. AME allows you to 'expand' the memory of your AIX logical partition (LPAR) without actually adding real memory to it.
AIX and the applications running inside the LPAR work with the expanded memory while AME does its memory expansion under the covers. A very good overview of the technical concepts of AME can be found in the AIX documentation (Active Memory Expansion (AME)).

Initially, AME became available based on POWER7® with AIX 6.1. At that time, AME was only able to work with 4K memory page sizes. This variant will be called ***AME4K*** in the following article. AME4K was initially released for usage with SAP via SAP Note 1464605 - Active Memory Expansion (AME) with the restriction that it should not be switched on with applications that required 64K memory page sizes.

With POWER8® servers and AIX 7.2 TL1, AME was enabled for 64K memory page sizes. This variant will be called ***AME64K*** in the following. For Power10 processor–based servers running AIX 7.3, AME64K has even become the new default.

We validated AME64K with SAP workloads on Power8, Power9 and Power10. SAP Note 1464605 - Active Memory Expansion (AME) was updated and AME64K is now the recommended option and fully supported in SAP environments.

The following chapters show the methodology and the results of our validation tests and give customers guidance when to use and how to optimally configure AME for their SAP workloads.

Test Setup with SAP Workload

To measure the effects that AME has on SAP workloads we needed an SAP workload that contains real-life SAP transactions, and which could be executed in a repeatable fashion.

Traditionally, the SAP Sales and Distribution (SD) Benchmark (https://www.sap.com/about/benchmark.html) is used to measure and compare the performance of servers and database platforms on which SAP can run.
In our case we used the SAP SD benchmark to generate a stable and repeatable SAP workload pattern. The two numbers that we collected from the SD benchmark were the **SAPS (SAP Application Performance Standard) number** which corresponds to the transactional throughput of the benchmark run and the **dialog response time** which measures the responsiveness of the system as seen by the simulated SAP GUI users.
It was not our aim to optimize the system for the highest possible number of SD users, for lowest response times or highest SAPS numbers (for an explanation of SAPS see https://www.sap.com/about/benchmark/measuring.html). The results of the SD benchmark served only as an indicator for the effects of the different AME expansion factors and the AME variants on SAP workload. Decreasing SAPS numbers and increasing response times indicate that the AME expansion factor was chosen too aggressively –

leading to lower overall throughput and slower responsiveness of the SAP system, which is something you want to avoid.

Benchmark Server Setup

Our test environment consisted of a 3-tier SAP system configuration, running SAP NetWeaver 7.40 with SAP kernel 753 (PL1000) with Db2 LUW as the database backend.

Two alternative configurations were used as SAP application servers: one running on a Power9 and the other running on Power10. Both servers were defined as dedicated LPARs with an entitlement of one CPU with SMT8. The LPARs were running AIX 7.2 TL4 SP5 (Power9) and AIX 7.3 TL1 SP1 (Power10).

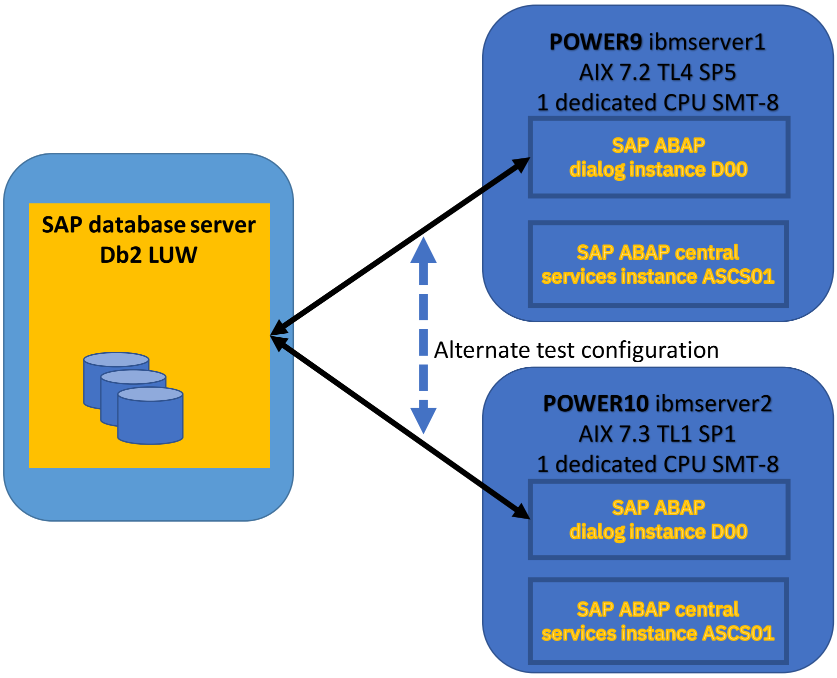

Figure 1 - Benchmark Server Setup

Benchmark Configuration

The SD benchmark was configured to run with 330 simulated SAPGUI users on both the Power9 or Power10.
To guarantee a reproduceable environment, the server was rebooted, and the SAP system was started before each run. A complete run consisted of four benchmark loops. We picked the last loop to measure CPU and memory consumption, and to collect the SD results. The following screenshot (***Figure 2***) shows the typical CPU utilization on the Power9 application server during such a run.

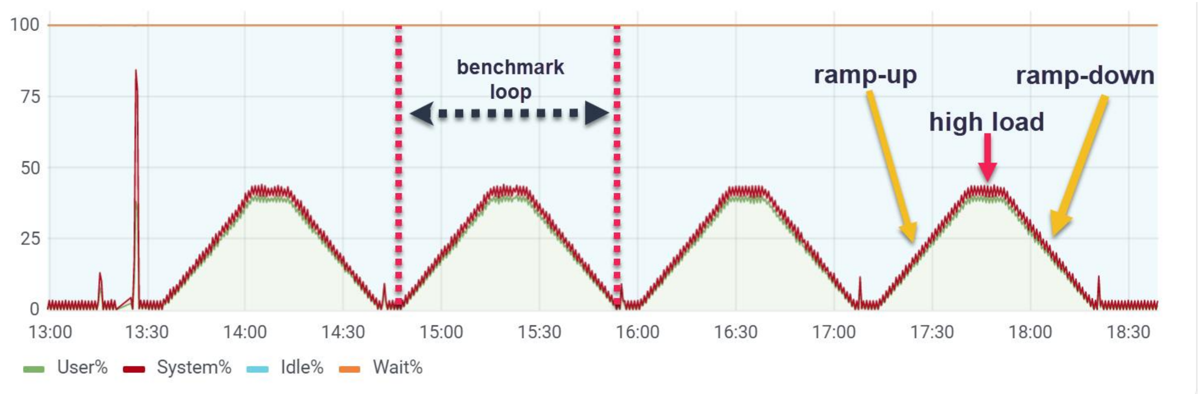

Figure 2 - CPU utilization during SD benchmark

Each benchmark loop starts with a ramp-up phase in which the user number is increased to the maximum number of simulated SD benchmark users. The interesting phase for our AME evaluation is the high-load phase during which all simulated users are active. It is followed by a ramp-down phase during which the simulated users finish their work and log out.

The initial memory configuration was chosen such that there was some headroom between the available memory of 14GB (plus 500MB of paging space) and the maximum amount of active virtual memory of about 10GB that was needed during the high load phase. ***Figure 3*** shows the typical memory consumption for a benchmark run where AME was active, but running with an expansion factor of 1.0.

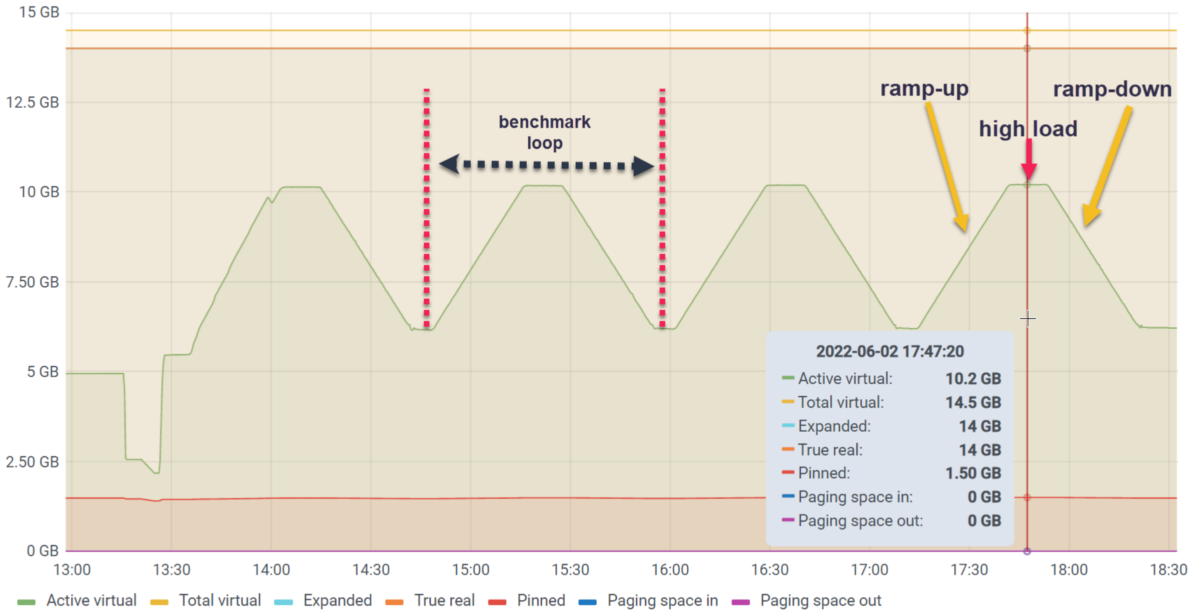

Figure 3 - Memory usage during SD benchmark

The high-load phase is a good model for a workload peaks that might occur in any SAP system and for which the AIX administrator should ensure that all SAP users see optimal throughput and response times.

To evaluate the effect of AME on SAP workload, the CPU and memory consumption were measured only during the high-load phase and only during the last (fourth) SD benchmark loop. The high-load phase of the benchmark had a duration of about 15 minutes.
All numbers that are listed in the tables below like the average CPU load, the memory consumption, and the average AME CPU load were collected during this high-load phase (see ***Test Iteration***).

SAP kernel and 64K application page sizes

Before switching on AME, it is important to understand the effects that memory page sizes have on the performance of the SAP kernel.

In general, applications on AIX may see performance benefits when using 64K instead of the default 4K pages, see 64 KB page size support.

If not configured otherwise, the SAP kernel runs with 4K page sizes as a default. You can configure the SAP kernel to run with 64K page sizes by setting LDR_CNTRL environment variable in the SAP application server profile:

```
SETENV_XX = LDR_CNTRL=DATAPSIZE=64K@TEXTPSIZE=64K@STACKPSIZE=64K@SHMPSIZE=64K
```

Running the SAP kernel with 64K pages is in most cases recommended, because it lowers the CPU load. This allows the SAP application server, for example, to support more SAP users.
There is a moderate increase in SAP virtual memory consumption that comes with this option. For more details see SAP Note 1387644 - Using 64 KB Virtual Memory Pages Sizes on AIX .

We executed the first tests without AME being active to demonstrate the positive effect that 64K page sizes have in our SD benchmark setup.
See ***Table 1*** for the differences in CPU utilization for both Power9 and Power10:

- the CPU savings when using 64K pages compared to 4K pages are:
  - 4 percentage points on Power9 (42% vs. 46%)
  - 2 percentage points on Power10 (28% vs. 30%)
- The SAPS numbers and the response times remain basically the same.

	Page Size	Active virtual memory	CPU load	SAPS	Response Time
Power10	64K	10.4 GB	28%	1979	0.05s
Power10	4K	9.1 GB	30%	1976	0.05s
Power9	64K	10.2 GB	42%	1973	0.07s
Power9	4K	8.9 GB	46%	1971	0.06s

Table 1 – 4K and 64K page sizes, AME switched off

Enabling Active Memory Expansion (AME)

If AME is not yet enabled for the LPAR, it has to be switched on in the Hardware Management Console (HMC). Enabling AME requires a reboot of the LPAR.

The two parameters which determine AME behavior are the expansion factor and the configured (true) memory size. The (expanded) memory that AIX will see is calculated as:

expanded_memory_size = expansion_factor * true_memory_size

For example, with an expansion factor of 2.0 and a true memory size of 7GB, the AIX operating system and the applications will see the expanded memory size of 14GB.
For more details see the AIX documentation at: Active Memory Expansion (AME).
If AME is enabled for an LPAR, then the AME expansion factor and memory size can be changed dynamically – without reboot - from the HMC using dynamic LPAR operations.

As a default, AME is enabled for 4K pages (AME4K), the exception being Power10 with AIX 7.3 where AME64K is enabled by default. To check the current page size for which AME is enabled, issue the vmo command to display the Virtual Memory Manager (VMM) tunable parameter **`ame_mpsize_support`**:

```
vmo -o ame_mpsize_support
```

If the value of this variable is 0, then the AME4K variant is active.

For enabling 64K page sizes (AME64K), issue the following command:

```
vmo -ro ame_mpsize_support=1
```

Please note that any change of the AME page size support requires a reboot!

The following sections describe the effects that AME4K and AME64K have on SAP workload and how they interact with the application page size setting of the SAP kernel.

AME4K and SAP

You might consider combining the benefits of using 64K application pages sizes with AME4K. But this is the downside:

- AME4K forces every application to use 4K pages, thereby nullifying the CPU savings that the use of 64K application page sizes would have.

In a running SAP system, you can check the effective page size that is used by the SAP kernel by issuing the following **ps** command and checking the output for the **`disp+work`** executable:

```
#ps -eZ
      PID    TTY  TIME DPGSZ SPGSZ TPGSZ SHMPGSZ CMD
  5571064      -  0:01    4K    4K    4K      4K disp+work
```

The following table shows this negative side effect when AME4K is switched on. The CPU load column shows the same numbers when running with 4K or 64K application page sizes. The positive effect on CPU usage that 64K application page sizes have (see ***Table 1***) can no longer be observed. Even though the SAP kernel is configured to use 64K pages, it is effectively only using 4K pages.

	Configured Application Page Size	CPU load	SAPS	Response Time
Power10	64K (effective: 4K!)	31%	1975	0.05s
Power10	4K	31%	1974	0.05s
Power9	64K (effective: 4K!)	48%	1969	0.08s
Power9	4K	48%	1970	0.07s

Table 2 - 4K and 64K application page sizes, AME4K with expansion factor 1.0

This is the reason for the:
Warning: do not switch on AME4K with applications that run with 64K page sizes (see also SAP Note 1387644 - Using 64 KB Virtual Memory Pages Sizes on AIX).

There is another negative effect that can be seen in ***Table 2***.
Compared to ***Table 1*** where AME was inactive, the CPU load increased by switching on AME4K even though the expansion factor was at 1.0.

- On Power10, the CPU load increased from 30% to 31%
- On Power9, the CPU load increased from 46% to 48%

On Power8, the CPU load increased even more from 71% to 77%.

Warning: when running with AME4K, make sure that your expansion factor is **not** permanently set to 1.0. If you do not plan to effectively use memory expansion with expansion factors greater than 1.0, you should rather disable AME to save CPU cycles.

AME64K and SAP

With AME64K, the limitations of AME4K are lifted:

- Applications which are configured to use 64K application page sizes are fully supported and run with 64K page sizes (check this with the `ps -eZ` command)
- CPU load does not increase when running with AME and an expansion factor of 1.0. This can be seen when comparing the CPU load in ***Table 3*** with the numbers from ***Table 1***.

	Configured Application Page Size	Active virtual memory	CPU load	SAPS	Response Time
Power10	64K	10.3 GB	28%	1977	0.05s
Power9	64K	10.2 GB	42%	1973	0.05s

Table 3 - AME64K with expansion factor 1.0

Contrary to the situation with AME4K, you can switch on AME64K and leave the expansion factor at 1.0 without any increased CPU load.
Obviously, you do not save any memory when running with a factor of 1.0. But this setting can be helpful for collecting data about your workload via **`amepat`** to determine suitable AME expansion factors (see ***Finding a suitable AME expansion factor***).

Another advantage of AME64K compared to AME4K becomes apparent at higher AME expansion factors:

- **AME64K has lower CPU overhead** in compression/expansion operations.
- **AME64K leads to more efficient compression** allowing for higher expansion factors and lower CPU overhead.

Table 4 shows these advantages on the Power9 test system. The CPU column shows the percentage values of both the total CPU and the share the AME operations have. Both the total CPU load and the AME share are reported in the amepat report. For a sample report see ***Amepat report for expansion factor 4.7***.
The numbers in ***Table 4*** were generated with true memory set to 5GB, an expansion factor of 2.8 and the resulting expanded memory at 14GB:

- with AME64K, the AME compression works on 64K instead of 4K pages which leads to a higher compression ratio of 7.26 instead of the 4.45 that can be achieved with AME4K.
- with AME64K, the overall CPU utilization is 49% whereas with AME4K we see 71%
- with AME64K, AME operations account for only 5% of the total CPU load - which is much lower than the 16% which we see with AME4K.

Furthermore, AME4K causes the SAPS numbers to decrease slightly compared to the unconstrained (no AME) case and the AME64K scenarios. The GUI response times suffer as well.

AME	Application page size	Active Virtual Memory	Compression Ratio	SAPS	Response time	CPU % Total	AME
64K	64K	9741 Mb	7.26	1969	0.07s	49%	5%
4K	4K	8244 Mb	4.45	1948	0.18s	71%	16%

Table 4 - AME64K and 4K with expansion factor 2.8

As a conclusion, in this environment, AME64K works fine with an expansion factor of 2.8 with enough CPU capacity left for unexpected workload spikes.
Whereas with AME4K, true memory would have to be increased, and the expansion factor should be lowered to avoid any negative impact on SAP throughput and responsiveness and to lower CPU usage.

Recommendation: use AME64K in combination with 64K application page sizes for best performance and lowest CPU load.

Finding a suitable AME expansion factor

After switching on AME for an LPAR, the basic question is:
What is a good choice for true memory and for the AME expansion factor?

The AME configuration should:

- run with the least amount of true memory to maximize memory savings.
- have little or no impact on throughput and responsiveness of the workload.
- leave some headroom in CPU capacity and memory for unexpected workload spikes.

The answer depends on the characteristics of the workload that you want to run on the LPAR. You need to have a good understanding of your workload:

- what are the times with peak usage of CPU and memory?
- what is the impact on your workload when your LPAR is CPU or memory constrained?

The following sections describe in detail the steps towards an optimal expansion factor for our test environment. You might want to use a similar approach for finding the best AME configuration for your environment as well.

Our test focus was on the AME64K variant with 64K application page sizes. For all tests, we chose the expansion factors and the configured true memory such that the resulting expanded memory was always 14GB:

```
expansion_factor * true_memory_size = 14GB
```

Monitoring with the amepat Tool

To see the impact of AME, we need to monitor:

a) the behavior of the SAP workload
b) CPU and memory load on the LPAR.

In the SAP SD benchmark, the SAPS numbers serve as a measure for throughput, and the dialog response times are a good measure for the responsiveness of the system.
For your SAP application, you would need to collect metrics that are specific to your application and which show how your application is behaving (for example in SAP transaction ST03).

For monitoring the system behavior, you can use many AIX monitoring tools like *topas* or *lparstat* (see ***Further Recommendations***)
SAP transaction ST06 also collects and displays operating system and AME metrics. More suitable during the initial setup of AME is the ***amepat*** utility. (see amepat Command)

In addition to the monitoring capability, ***amepat*** tool also model and forecast the potential effects of different AME expansion factors.
To get realistic measurements of the resource usage, it is important that you run *amepat* during high workload phases. It also helps *amepat* to better model the effects that different expansion factors will have with the given workload.
Note: with AME64K you need to invoke the *amepat* command with the *-M* option to enable modelling of 64K page usage.

For our tests with SAP application servers, we used the SAP SD benchmark as workload and identified the high-load phases of the benchmark as our workload peaks.
We ran *amepat* only during these high-load phases of the SD benchmark.

Initial Test Setup

The following checklist makes sure that the LPAR on which we want to conduct our AME64 tests is set up correctly:

- check if AME is enabled for the LPAR with the **`lparstat -c`** command:

```
#lparstat -c
System is not running in AME memory mode, -c flag is not valid on the Hardware
```

  If you receive this message, go to the Hardware Management Console (HMC) to enable AME. Enabling AME requires a reboot of the LPAR. To reduce the number of reboots, you can combine this reboot with the one needed to change the ame_mpsize_support parameter.
- check that **`ame_mpsize_support=1`** is set. A reboot is needed if the ame_mpsize_support parameter needs to be changed.
- make sure that the SAP instance are started with 64K application page sizes by checking the LDR_CNTRL setting in SAP profiles.
- start the SAP instance and check with the **`ps -eZ`** command that the SAP kernel actually uses 64K application page sizes.

Test Iteration

For our tests, the AME expansion factor and the true memory size were set such that the resulting expanded memory was always 14GB.

This sequence can be used as a general recipe for finding the optimal AME configuration:

- Start the SAP workload and wait for the high-load phase.
- Monitor the performance characteristics of the workload.
- As soon as the high-load phase begins, start *amepat* recording:
  `amepat -M -R` ***`amepat_recording_file`***
- After the recording has finished, generate the *amepat* report:
  `amepat -P` ***`amepat_recording_file`*** **`>`** ***`amepat_report`***
- Check the performance data and the modelling recommendations in the *amepat* report. For a sample report see ***Amepat report for expansion factor 2.0*** on page 23.
- Decrease the true memory size and increase the AME expansion factor dynamically via a DLPAR operation and make sure that: **`expansion factor * true memory size = 14GB`**
- Re-run the test with the new expansion factor until one or more of these conditions are encountered:
  - the SAP application performance is no longer satisfactory.
  - the CPU load is too high.
  - an AME memory deficit reported. For details see the section “Memory Deficit” in Active Memory Expansion (AME).

Analysis of the Test Results

The above sequence (see ***Test Iteration***) was used on Power10 to generate the numbers in ***Table 5*** below.
The first line in the table shows the results without AME being enabled. It serves as the baseline against which we want to compare. In all AME tests, the expanded memory was 14GB and the "Memory Savings" column shows how much memory we saved while running with the specified expansion factor.

AME expansion factor	True Memory GB	Memory Savings	SAPS	Response time	CPU % Total	CPU % AME	AME deficit MB
Off	14	0%	1979	0.05s	28%	n/a	n/a
1	14	0%	1977	0.05s	28%	n/a	n/a
1.6	8.75	37.5%	1979	0.05s	28%	0%	0
2.0	7.0	50%	1979	0.05s	29%	1%	0
2.44	5.75	59%	1979	0.05s	32%	3%	0
2.8	5.0	64%	1973	0.06s	34%	5%	0
3.5	4.0	71%	1975	0.06s	35%	6%	0
3.75	3.75	73%	1974	0.06s	36%	6%	0
4.0	3.5	75%	1971	0.07s	37%	8%	0
4.35	3.25	77%	1970	0.08s	43%	12%	0
4.7	3.0	79%	1253	4.1s	85%	43%	1645

Table 5 - AME64K with increasing expansion factors and 14GB of expanded memory on Power10

For a wide range of expansion factors between 2.0 and 4.0, the CPU load increase is in the single-digit range. Let us have a closer look at the individual results:

The **expansion factor of 2.0** allows us to run with only 50% of the required memory while AME has a very low (1%) impact on CPU load compared to running without AME.
Even with an **expansion factor of 2.8**, the CPU impact is still in the single-digits (6%), and it allows us to save nearly two-thirds (64%) of the memory which is required when running without AME.

Increasing the expansion factor much higher does not save substantially more memory and it increases the risk that we run into memory shortages and an AME deficit.
With higher expansion factors you run the risk that even a minor increase of the workload might push you over the 'tipping point' which we see in our example at expansion factor of 4.7.

The extreme case with an **expansion factor of 4.7** and only **3.0 GB of true memory** is shown in the last line:

- CPU load increases substantially, amepat reports a combined user+system CPU time of 85%. In addition, there is a 15% CPU wait time which is caused by waits for the hardware compression unit. This adds up to a combined CPU load of 100%.
- overall SAP performance suffers extremely with SAPS throughput decreasing from 1970 to 1253 and dialog response time at 4.1 seconds.
- an AME memory deficit of 1645MB is shown in the *amepat* report (see ***Amepat report for expansion factor 4.7***). This is a clear indication that the expansion factor was chosen too high (or the configured true memory too low) for the workload.

According to ***Table 5***, expansion factors between 3.5 and 4.0 add only little CPU load. But we would run the risk that any change in the workload or memory usage would bring us too close to a memory deficit situation.

Conclusion: For our SAP SD benchmark workload, any expansion factor between 2.0 and 2.8 is a good choice for productive use - saving a substantial amount of memory with no impact to SAP users while leaving some room for increased memory or CPU demands.

Further Recommendations

When you have found a suitable expansion factor and true memory setting for your workload, you should continue to monitor your system. Over time, the CPU or memory requirements of your workload may change, or memory fragmentation may occur. These effects make it harder for AME to reach its expansion target and might eventually force you to adapt your initial choice of expansion factor and true memory.

If you do not need the recording or modelling capabilities of *amepat*, other quick and easy ways to monitor AME operations from the AIX command line are the *topas* or *lparstat* commands (see topas Command - IBM Documentation and lparstat Command - IBM Documentation). Furthermore, SAP GUI shows you in the expert view of transaction ST06 the relevant AME metrics.

Monitoring with topas

When monitoring via *topas*, the numbers to watch out for are the **T**arget and **A**ctual **E**xpansion **F**actors in the EF[T/A] field which is highlighted in the below sample output in red. If AME is no longer able to reach its target expansion factor, this is an indication that you should adapt the expansion factor or true memory setting.

```
Topas Monitor for host:ibmserver2                EVENTS/QUEUES    FILE/TTY
Thu Mar  9 16:57:47 2023   Interval:2            Cswitch     3841 Readch      373
                                                 Syscall     6791 Writech  244.7K
CPU     User% Kern% Wait% Idle%                  Reads          0 Rawin         0
Total    17.6   4.9   0.3  77.2                  Writes       177 Ttyout      373
                                                 Forks          0 Igets         0
Network    BPS  I-Pkts  O-Pkts    B-In   B-Out   Execs          0 Namei         3
Total    1.30M   1.28K   1.08K    607K    727K   Runqueue    1.00 Dirblk        0
                                                 Waitqueue    0.0
Disk    Busy%      BPS     TPS  B-Read  B-Writ                    MEMORY
Total     0.0        0       0       0       0   PAGING           Real,MB   14336
                                                 Faults      8596 % Comp     52
FileSystem          BPS     TPS  B-Read  B-Writ  Steals      4749 % Noncomp   0
Total              224K   434.5    185K   39.0K  PgspIn         0 % Client    0
                                                 PgspOut        0
Name           PID  CPU%  PgSp Owner             PageIn         0 PAGING SPACE
disp+wor  10224032   4.5 75.5M b13adm            PageOut        0 Size,MB    2048
disp+wor  12124628   3.1 77.1M b13adm            Sios           0 % Used     10
disp+wor  12583228   3.0 76.6M b13adm                             % Free     90
cmemd       655640   2.0  704K root              AME
disp+wor   9765296   1.5 81.0M b13adm            TMEM        3.50GWPAR Activ    0
disp+wor   9437670   1.4 72.4M b13adm            CMEM      722.16MWPAR Total    0
lrud        131336   1.4  512K root              EF[T/A]   4.0/4.0Press: "h"-help
disp+wor  10289602   0.8 47.9M b13adm            CI:4.66KCO:4.58K        "q"-quit
disp+wor   9568564   0.8 47.8M b13adm
disp+wor   9306584   0.6 75.6M b13adm
disp+wor   9175394   0.5 49.5M b13adm
```

Monitoring with lparstat

An alternative monitoring capability is available with the *lparstat* command using the *-c* option which is only available when AME is enabled. This is a sample output with a 30 second monitoring interval and a count of 5:

```
# lparstat -c 30 5

System configuration: type=Dedicated mode=Capped mmode=Ded-E smt=8 lcpu=8 mem=14336MB tmem=3584MB

%user  %sys  %wait  %idle  %xcpu xphysc    dxm
----- ----- ------ ------ ------ ------ ------
 16.9   4.0    0.2   78.9    4.0 0.0397      0
 16.7   4.1    0.2   79.1    3.7 0.0373      0
 15.5   3.6    0.2   80.6    3.6 0.0360      0
 15.6   4.0    0.2   80.2    3.7 0.0374      0
 14.4   3.5    0.1   82.1    3.5 0.0346      0
```

The most important column to watch out for is the **dxm** column which shows the AME deficit memory in MB. For a detailed explanation see the section “Memory Deficit” in Active Memory Expansion (AME). The **dxm** value should always be 0!

If you start seeing non-zero values in the **dxm** column, then AME is no longer able to compress and expand memory as expected. You will see high AME CPU utilization **%xcpu** which might slow down your application.
To avoid such an AME deficit, you should lower the expansion factor and increase true memory such that you reach the same expanded memory as before. This change can be made dynamically from the HMC without interruption for the SAP workload.

Monitoring with in SAP GUI with ST06

In SAP GUI, you can use transaction ST06 to monitor AME metrics. In the Memory display, switch to 'Expert View' as shown in the screenshot below. Check if the actual expansion factor is equal to the target factor or if there is a memory deficit:

Memory Virtualization Virtual System	Memory Mode	Dedicated Expanded	
	AME Target Factor	4,00	
	AME Actual Factor	4,00	
	AME Expanded Memory	14.336	MB
	AME True Memory	3.584	MB
	AME Deficit Memory	0	MB

Figure 4 - Transaction ST06 with AME metrics

AME Benefits

Switching on and exploiting AME can result in significant cost savings and in addition serve as a 'safety net' helping to avoid paging or outages caused by out-of-memory situations.

Cost savings

The most obvious benefit of AME is that it allows you to run your LPAR with less real (physical) memory while applications can use the 'expanded' memory. This frees up physical memory for other LPARs or allows your workload to run on a server which does not have enough real (physical) memory available.

The achievable memory savings depend heavily on the workload that runs in the LPAR. In our tests we saw (see ***Analysis of the Test Results***) that an SAP workload can be run with AME expansion factor of 2.0 or even higher – which results in memory savings of 50% or more. Depending on your specific workload, the maximum expansion factor needs to be adapted and the optimal factor could be higher or lower (see ***Finding a suitable AME expansion factor***).

Using AME as a safety net

Another way to exploit AME is that it can make your system more robust against unexpected memory demands that would otherwise lead to heavy paging or even cause an outage of the SAP application.

Consider the following scenario:

You have properly sized and configured the memory for your SAP application server and have defined a reasonable amount of paging space.
Now, an unexpected workload spike, an SAP configuration change or a third-party application causes the memory demand to increase.
When the active virtual memory approaches 100% of real memory, your system will start to page out memory to paging space and SAP application performance is going to suffer considerably.
When the memory demand approaches the limit of your virtual memory (real memory + paging space) the consequences are even more severe:
To avoid running completely out of virtual memory, AIX will start to kill processes – which includes SAP processes.
This will ultimately cause an outage of the SAP application server.
See more details on this topic in SAP Notes 2630086 - Avoid signal 33, out of memory on AIX and 1121904 - SAP on AIX: Paging space and physical memory resources.

You want to avoid this outage and the following example shows how AME can help:
Consider an SAP application server which is configured with 50GB of real memory and an additional 10 GB of paging space. The memory usage during 'normal' operations is 40GB. This leaves some headroom for unexpected memory demands of either SAP or another workload running on the system.

Without AME:
If the total memory usage approaches and goes beyond 50GB, you will see a significant slowdown of SAP performance caused by paging. With a further increase up to 60GB you will run into an outage of the SAP application server instance.
Adding an additional 50GB of real memory to the LPAR would help – bringing the total available virtual memory to 110GB. The additional 50GB of real memory would act as a 'safety net' and most of the time they would not be used.

With AME:
With AME switched on, you would leave true memory defined at 50GB + 10 GB of paging space and define an AME expansion factor of 2.0 – resulting in 110GB of available virtual memory. (2.0 x 50GB true memory + 10GB paging space)
During 'normal' operation where the memory demand is at 40GB, you would not see any AME activity and your workload would run as before.
When the total memory usage exceeds 50GB, you will see some AME activity but there will be nearly no impact to the SAP application.
Only in the case where total memory demand approaches 100GB will you see a significant slowdown SAP throughput due to paging. But your SAP system will not stop until memory demand approaches the 110GB limit.

Benefit: instead of spending the money to add an additional 50GB of real memory to the LPAR, you can use AME to reach the same 'safety net' effect.

Summary and recommendations

Our tests demonstrated the benefits of AME for SAP ABAP application servers supporting SAP NetWeaver or SAP S/4HANA. Expansion factors of 2.0 or even higher can be reached without impact on SAP application performance. For optimal results, the AME64K variant in combination with 64K application page sizes should be used.

To determine the optimal settings of AME expansion factor and true memory setting for your specific workload you should use the *amepat* tool. Both the expansion factor and true memory can be changed dynamically without disruption for the workload.

What are the scenarios where AME is not helpful?
AME operates very efficiently on memory that allows for high compression ratios. Applications that have their own built-in memory compression or memory encryption are not very well suited for AME. Examples are database systems like, for example, IBM Db2, where database memory areas contain compressed data.
Another category are applications which do their own memory management like, for example, the SAP JAVA application server.

Appendix: Sample amepat Reports

This appendix contains two sample *amepat* reports from the tests that we used to find the optimal expansion factors for our SAP SD workload.

The first report ***Amepat report for expansion factor 2.0 with modelling*** includes the modelling part of *amepat* reporting. The recommended new expansion factor is highlighted in blue. In this case, the recommended expansion factor of 3.74 is really helpful and real life measurements with this factor (see ***Table 5***) show that it is a valid choice.
We saw other cases where the recommendation from the report was not a good choice, so you always need to verify any choice of expansion factors and true memory with the 'real' workload.
Furthermore, *amepat* modelling adds a certain amount of CPU load. In this case the CPU load is show as 32% compared to the 29% in ***Table 5*** when modelling was switched off. Modelling in *amepat* is active as a default, use the *-N* option to switch it off.

The second sample report ***Amepat report for expansion factor 4.7*** shows the extreme case where AME is no longer able to reach the desired expansion factor. The numbers to watch out for are highlighted in red: the CPU load, the AME share of the CPU load and – most important - the memory deficit.

Amepat report for expansion factor 2.0 with modelling

```
Command Invoked                : amepat -P /tmp/amepat_record_3_2023030714

Date/Time of invocation        : Tue Mar  7 17:24:11 CET 2023
Total Monitored time           : 9 mins 17 secs
Total Samples Collected        : 3

System Configuration:
---------------------
Partition Name                 : ibmserver2
Processor Implementation Mode  : POWER10 Mode
Number Of Logical CPUs         : 8
Processor Entitled Capacity    : 1.00
Processor Max. Capacity        : 1.00
True Memory                    : 7.00 GB
SMT Threads                    : 8
Shared Processor Mode          : Disabled
Active Memory Sharing          : Disabled
Active Memory Expansion        : Enabled
Target Expanded Memory Size    : 14.00 GB
Target Memory Expansion factor : 2.00


System Resource Statistics:                 Average           Min             Max
---------------------------               -----------     -----------     -----------
CPU Util (Phys. Processors)               0.32 [ 32%]     0.31 [ 31%]     0.35 [ 35%]
Virtual Memory Size (MB)                  9993 [ 70%]     9959 [ 69%]    10011 [ 70%]
True Memory In-Use (MB)                   7154 [100%]     7154 [100%]     7155 [100%]
Pinned Memory (MB)                        1315 [ 18%]     1315 [ 18%]     1316 [ 18%]
File Cache Size (MB)                       202 [  3%]      199 [  3%]      204 [  3%]
Available Memory (MB)                     4527 [ 32%]     4517 [ 32%]     4549 [ 32%]

AME Statistics:                             Average           Min             Max
---------------                           -----------     -----------     -----------
AME CPU Usage (Phy. Proc Units)           0.01 [  1%]     0.01 [  1%]     0.01 [  1%]
Compressed Memory (MB)                    3065 [ 21%]     3059 [ 21%]     3072 [ 21%]
Compression Ratio                         8.15            8.11            8.19

Active Memory Expansion Modeled Statistics:
-------------------------------------------
64K page support               : Enabled
64K page modeling              : Disabled
Modeled Expanded Memory Size   :  14.00 GB
Achievable Compression Ratio   : 6.26

Expansion    Modeled True      Modeled             CPU Usage
Factor       Memory Size       Memory Gain         Estimate
---------    -------------     ------------------  -----------
     1.04         13.50 GB       512.00 MB [  4%]  0.00 [  0%]
     1.48          9.50 GB         4.50 GB [ 47%]  0.00 [  0%]
     1.94          7.25 GB         6.75 GB [ 93%]  0.02 [  2%]
     2.00          7.00 GB         7.00 GB [100%]  0.01 [  1%] << CURRENT CONFIG
     2.44          5.75 GB         8.25 GB [143%]  0.05 [  5%]
     2.80          5.00 GB         9.00 GB [180%]  0.06 [  6%]
     3.30          4.25 GB         9.75 GB [229%]  0.07 [  7%]
     3.74          3.75 GB        10.25 GB [273%]  0.08 [  8%]

Active Memory Expansion Recommendation:
---------------------------------------
The recommended AME configuration for this workload is to configure the LPAR
with a memory size of 3.75 GB and to configure a memory expansion factor
of 3.74.  This will result in a memory gain of 273%. With this
configuration, the estimated CPU usage due to AME is approximately 0.08
physical processors, and the estimated overall peak CPU resource required for
the LPAR is 0.42 physical processors.

NOTE: amepat's recommendations are based on the workload's utilization level
during the monitored period. If there is a change in the workload's utilization
level or a change in workload itself, amepat should be run again.

The modeled Active Memory Expansion CPU usage reported by amepat is just an
estimate.  The actual CPU usage used for Active Memory Expansion may be lower.
```

Amepat report for expansion factor 4.7

```
Command Invoked               : amepat -P /tmp/amepat_record_3_2023020713

Date/Time of invocation       : Tue Feb  7 17:00:08 CET 2023
Total Monitored time          : 9 mins 0 secs
Total Samples Collected       : 3

System Configuration:
---------------------
Partition Name                : ibmserver2
Processor Implementation Mode : POWER10 Mode
Number Of Logical CPUs        : 8
Processor Entitled Capacity   : 1.00
Processor Max. Capacity       : 1.00
True Memory                   : 3.00 GB
SMT Threads                   : 8
Shared Processor Mode         : Disabled
Active Memory Sharing         : Disabled
Active Memory Expansion       : Enabled
Target Expanded Memory Size   : 14.00 GB
Target Memory Expansion factor : 4.70


System Resource Statistics:               Average           Min            Max
---------------------------             -----------     -----------    -----------
CPU Util (Phys. Processors)              0.85 [ 85%]     0.85 [ 85%]    0.85 [ 85%]
Virtual Memory Size (MB)                 9725 [ 68%]     9661 [ 67%]    9758 [ 68%]
True Memory In-Use (MB)                  3063 [100%]     3063 [100%]    3064 [100%]
Pinned Memory (MB)                       1165 [ 38%]     1165 [ 38%]    1165 [ 38%]
File Cache Size (MB)                       80 [  3%]       80 [  3%]      81 [  3%]
Available Memory (MB)                    3433 [ 24%]     3386 [ 24%]    3529 [ 25%]

AME Statistics:                           Average           Min            Max
---------------                         -----------     -----------    -----------
AME CPU Usage (Phy. Proc Units)          0.43 [ 43%]     0.43 [ 43%]    0.43 [ 43%]
Compressed Memory (MB)                   7381 [ 51%]     7380 [ 51%]    7382 [ 51%]
Compression Ratio                        6.76            6.75           6.76
Deficit Memory Size (MB)                 1645 [ 11%]     1633 [ 11%]    1652 [ 12%]
```

AME documentation and references

IBM documentation and Web resources

AIX AME documentation:
https://www.ibm.com/docs/en/aix/7.2?topic=management-active-memory-expansion-ame

AIX documentation of the amepat command:
https://www.ibm.com/docs/en/aix/7.2?topic=amepat-command

IBM support article:
https://www.ibm.com/support/pages/aix-active-memory-expansion-ame

AME FAQs:
https://www.ibm.com/support/pages/active-memory-expansion-ame

AME YouTube videos by Nigel Griffiths:
https://www.youtube.com/watch?v=W85ZRln-2hA
https://www.youtube.com/watch?v=XtWxC5IJWKg

SAP Notes and SAP Knowledge Base Articles

SAP Note	Title / Description
1464605	Active Memory Expansion (AME)
1387644	Using 64 KB Virtual Memory Pages Sizes on AIX
1121904	SAP on AIX: Recommendations for paging space
1972803	SAP on AIX: Recommendations
1492000	General Support Statement for Virtual Environments
1906519	DB6: Use of Active Memory Expansion (AME)
1682104	SAP JVM medium/large page support on AIX

Copyrights and Trademarks

Disclaimer and Special Notices

COPYRIGHT LICENSE: