Tivoli software

IBM

Performance of GPFS/HSM – Tivoli Storage Manager for Space Management

Part 1: Automigration

By Hai-Yann Hwang and Dominic Mueller-Wicke

Version 1.0

Copyright Notice

Trademarks

IBM, the IBM logo, Tivoli, the Tivoli logo, AIX, Cross-Site, NetView, OS/2, Planet Tivoli, RS/6000, Tivoli Certified, Tivoli Enterprise, Tivoli Enterprise Console, Tivoli Ready, and TME are trademarks or registered trademarks of International Business Machines Corporation or Tivoli Systems Inc. in the United States, other countries, or both.

Lotus is a registered trademark of Lotus Development Corporation.
Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
UNIX is a registered trademark of The Open Group in the United States and other countries.
C-bus is a trademark of Corollary, Inc. in the United States, other countries, or both.
PC Direct is a trademark of Ziff Communications Company in the United States, other countries, or both and is used by IBM Corporation under license.
ActionMedia, LANDesk, MMX, Pentium, and ProShare are trademarks of Intel Corporation in the United States, other countries, or both. For a complete list of Intel trademarks, see http://www.intel.com/sites/corporate/trademarx.htm.
SET and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC. For further information, see http://www.setco.org/aboutmark.html.
Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Notices

References in this publication to Tivoli Systems or IBM products, programs, or services do not imply that they will be available in all countries in which Tivoli Systems or IBM operates. Any reference to these products, programs, or services is not intended to imply that only Tivoli Systems or IBM products, programs, or services can be used. Subject to valid intellectual property or other legally protectable right of Tivoli Systems or IBM, any functionally equivalent product, program, or service can be used instead of the referenced product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by Tivoli Systems or IBM, are the responsibility of the user. Tivoli Systems or IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, New York 10504-1785,

U.S.A.

About the Tivoli Field Guides

Sponsor

Tivoli Customer Support sponsors the Tivoli Field Guide program.

Authors

Those who write field guides belong to one of these three groups:

- Tivoli Support and Services Engineers who work directly with customers
- Tivoli Customers and Business Partners who have experience using Tivol i software in a production environment
- Tivoli developers, testers, and architects

Audience

The field guides are written for all customers, both new and existing. They are applicable to external audiences including executives, project leads, technical leads , team members, and to internal audiences as well.

Types of Field Guides

Two types of Tivoli Field Guides describe how Tivoli products work and how they are used in real life situations:

- Field Guides for technical issues are designed to address specific te chnical scenarios or concepts that are often complex to implement or difficult to understand, for example: endpoint mobility, migration, and heartbeat monitoring.
- Field Guides for business issues are designed to address specific business practices that have a high impact on the success or failure of an ESM project, for example: change management, asset Management, and deployment phases.

Purposes

The Field Guide program has two major purposes:

- To empower customers & business partners to succeed with Tivoli software by documenting and sharing product information that provides accurate and timely information on Tivoli products and the business issues that impact an enterprise systems management project
- To leverage the internal knowledge within Tivoli Customer Support and Services and the external knowledge of Tivoli customers and Business Partners

Availability

All completed field guides are available free to registered customers and internal IBM employees at the following Web site:

http://www.ibm.com/software/sysmgmt/products/support/Field_Guides.html

Authors can submit proposals and access papers by e -mail:

mailto:Tivoli_eSupport_Feedback@us.ibm.com

Table of Contents

1

Performance of GPFS/HSM – Tivoli Storage Manager for Space Management

Introduction

With the explosive growth in data and information, the demand for efficient storage management across multiple tiers of storage of various speeds and costs has been increasing rapidly.

The IBM Tivoli Storage Manager for Space Management client for UNIX and Linux (the **HSM** client) migrates files from local file systems to distributed storage and can then recall the files either automatically or selectively. Migrating files to storage frees space for new data on local file systems and takes advantage of lower-cost storage resources that are available in the network environment.

To free space quickly when needed, HSM periodically searches for files suitable for migration and stores the candidates in pool files [1]. Starting with Version 5.5, a scout daemon is designed to build metadata files, i.e., hash tables of file information with multiple phases of search strategy so migration candidates can be provided with high efficiency [2].

This paper is arranged in four sections:

- Section 1 gives a brief overview of the purpose, content, and structure of this paper.
- Section 2 discusses the HSM performance.
- Section 3 discusses the GPFS storage management and performance.
- Section 4 compares the two storage managements and concludes.

In summary, the GPFS policy engine is a very powerful tool to replace the normal HSM candidate selection in GPFS environments. We recommend to all our GPFS customers to switch to this mechanism for the threshold migration. Nevertheless, HSM supports other file system types like JFS2 and VXFS, which cannot provide such an advanced inode scan mechanism . For multiple platform support, it is needed to support HSM implemented scan and candidate selection functionality.

A second paper will be prepared in the near future to accompany this paper and will address the following topics:

- The migration and recall with the new grouped migration feature in TSM 6.2.2.
- The partial and streaming recall.
- The policy based reconciliation of file systems.

2

TSM for Space Management for UNIX (aka HSM)

For HSM Storage Management, this section describes its design and internal mechanism, with the performance measurement for the critical operations: scan and migration.

Mechanism

Metadata Files

Starting with Version 5.5, HSM builds a hash table for all the files in a managed file system to store their metadata information. There would be two files as follows:

/<FileSystemName>/.SpaceMan/Metadata/.<FileSystemName>.meta1

/<FileSystemName>/.SpaceMan/Metadata/.<FileSystemName>.meta2

Where the second file is 8-15% of the size of the first file.

The size of the hash table is estimated by the maximum number of files the file system could support, i.e., the capacity of the file system divided by the fr agment size. This worst case design leads to significant space consumption, especially for file systems containing a lot of big files.

Table 2.1 The Size of HSM's Main Metadata File (file .meta1)

Number of Files	Workload **	File system Capacity	.meta1 File Size	Time for Creation
47	1 GB each file	74 GB	2.38 GB	40 seconds
1 million	W1, total 10 GB	74 GB	2.36 GB	64 seconds
10 million	W1, total 100 GB	149 GB	4.76 GB	57 seconds
10 million	W2, total 500 GB	596 GB	19.06 GB	4 minutes
30 million	W1, total 300 GB	596 GB	18.97 GB	3 minutes
100 thousand	100 GB each file	10 TB	Est. 304 GB*	Est. 60 minutes

* Size estimated for 2500 million files (10TB / 4KB fragment size) .

** Test Workloads W1 and W2 defined in Appendix A.

Tune "ulimit" for Metadata Files

Large metadata files exceed common ulimit setup easily. Then HSM would start to build .meta1 file from scratch and fail repetitively. To resolve the problem, adjust the file size of ulimit, then restart HSM with “dsmmigfs stop/start” if necessary.

Option “maxfiles infs”

To reduce the space consumption, a new option “maxfilesinfs” has been implemented to reduce the size of the metadata files. With the option set to 11 million, instead of the estimated 149 million (596 GB / 4KB fragment size), the size of the hash table is reduced proportionally.

Table 2.2 Option “maxfilesinfs” Impacts the Size of HSM’s Main Metadata File *

Number of Files	Workload	File system Capacity	maxfilesinfs	.meta1 File Size	Time for Creation
10 million	W2, total 500 GB	596 GB	Not set	19.06 GB	4 minutes
			11000000	1.41 GB	1 minutes

Unfortunately, the current setup for the option is global for all managed file systems on the client machine, thus it can not be tuned for various file systems.

The option is available for HSM 5.5.2, 6.1.3 and thereafter.

The option is reported to lead to significant improvements for scan and search activities . Our tests indicate about 10% enhancement.

NOTE : With HSM 6.2.0, a new functionality is introduced to define the hash table size. This functionality improves the possible configuration tasks, documented in detail in TSM manuals.

Scan

Scan Scheduling

Starting with Version 5.5, once a file system is added to HSM’s management, HSM scout daemon would immediately scan the whole file system, to collect file information into a hash table, as the base for searching and selecting migration candidates. HSM can not handle automigration before the first scan finishes.

A second scan would be run some time later, as fast as one hour after the end of the first scan, not only to update the file information, but also to gauge the volatility of the file system. A very stable file system could have a third scan after eight hours since the end of the first scan. Later scans would be run with increasing intervals.

The scan schedule can be checked with command

```
dsmscoutd scanplan <FileSystemName>
```

Option “candidatesinterval”

The behavior of the candidatesinterval option changed with TSM version 6.1.3. In previous versions of HSM, it was not possible for the user to specify a exact time interval for the scan activities of the HSM component dsmscoutd.

Starting with TSM 6.1.3 the option knows three defined states which can be modified by different values of the option.

CANDIDADTESINTERVAL 0

The dsmscoutd will scan the file system continously.

CANDIDADTESINTERVAL 1

The dsmscoutd will rescan the file system in intervals calculated based on the changes in the file system since the last scan was started. That means number of new created files number of deleted files and number of changed files. The behavior of the calculation is the following:

1. Minimum changes (less than 30%) : The previous scan interval will be doubled.

2. Medium changes (more than 30% but less than 50%) : reuse the previous scan interval.

3. Many changes (more than 50% but less than 90%) : halve the previous scan interval.

4. Maximum changes (more than 90%) : reduce the scan interval to the thenth part of the
previous interval.

CANDIDADTESINTERVAL 2 - 9999

The value of the scaninterval option reflects the interval in hours which will be used until the next scan will be started.

Scan Performance

As Figure 2.3 shows (below), the HSM scan rate reaches highest a li ttle while after start, then continues dropping till the end.

The performance also deteriorates when the number of files in the file system increases.

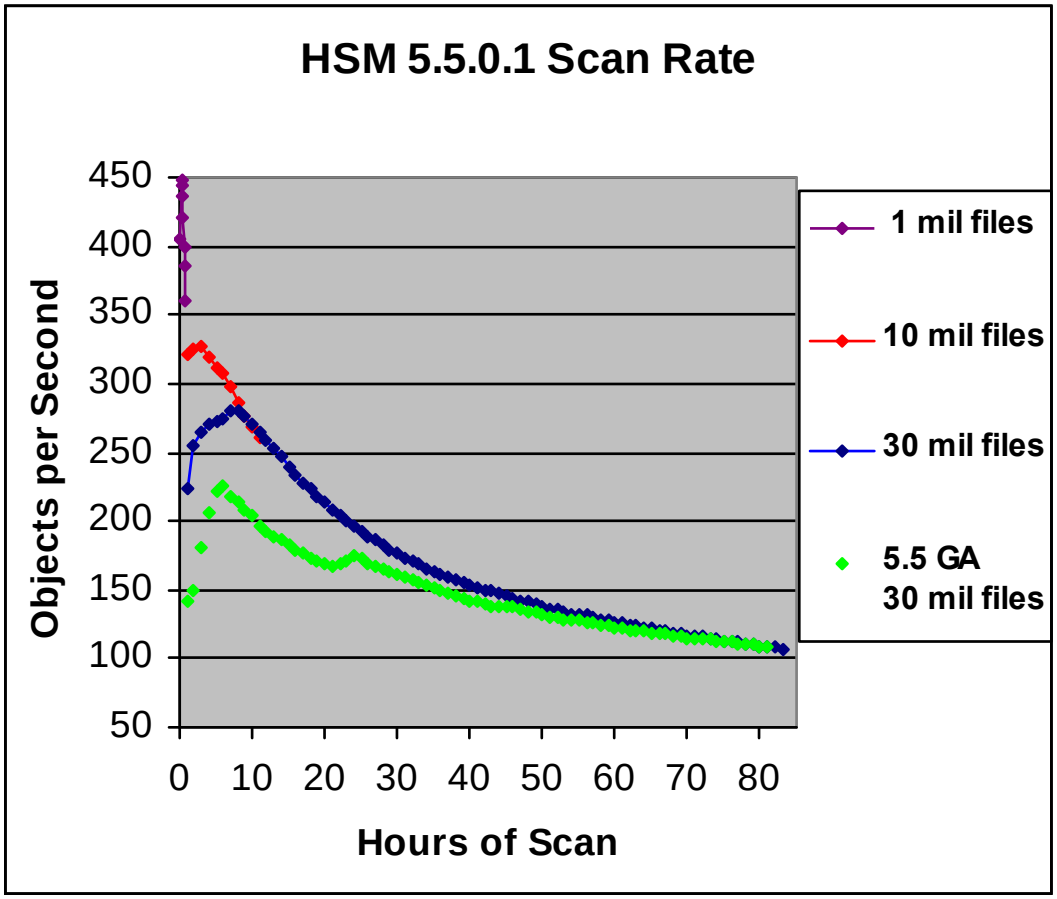


Figure 2.3 HSM Scan Rate with GPFS 3.2, On p630 6M2 *

* 2-way 1.2GHz Power4

With GPFS 3.3, the overall scan rates become more scalable, since the 30 million file system can be scanned in 13 hours instead of 83 hours for GPFS 3.2, due to more efficient DMAPI processing.

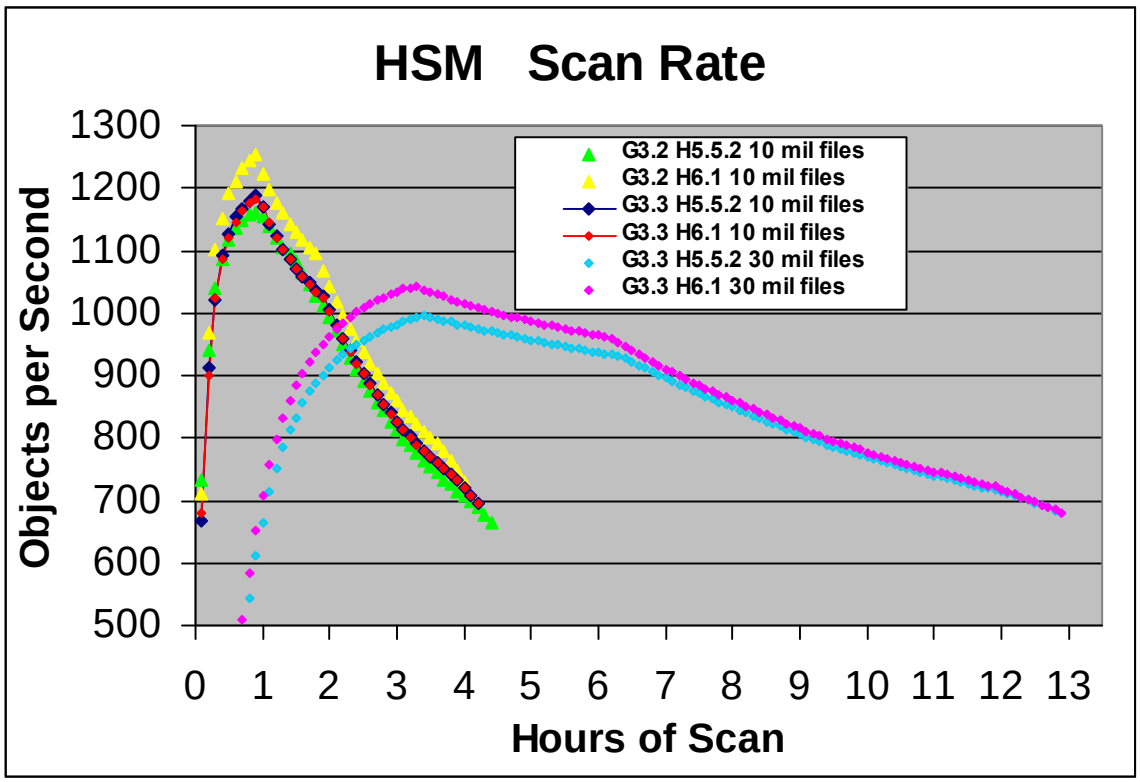


Figure 2.4 HSM Scan Rate with GPFS 3.2, 3.3 On pSeries p5 561

* pSeries p5 561, 4-way 1.8GHz Power5 LPAR, AIX 6.1

Table 2.5 (below) summarizes the scan performance with various settings, workload, and environment:

- The major influence factor for scan rates is the number of files.
- The size of files does not impact significantly. For example, the data size of the 10 million, Workload 2 case is 5 times as large as that of the 10 million, Workload 1 case, whereas their scan rate difference is within 10%.
- The size of the hash table does not impacts significantly. For example, setting Option maxfilesinfs to 11 millions shrinks the hash table size by 94%, whereas the scan rate only decreases by 10%.

GPFS VERSION	HSM VERSION	WORKLOAD #FILES WORKLOAD DATASZ	SCAN TIME	OVERALL SCAN RATE (OBJECTS / SEC)	PEAK SCAN RATE (OBJECTS / SEC)	CPU USAGE USER SYS IDLE IOWAIT
On p630 6M2, 2-way 1.2GHz Power4						
3.2	5.5.0.1	1 million W1 10GB	0.8 hour	360	444	
3.2	5.5.0.1	10 million W1 100GB	11.2 hours	259	326	18 12 58 12
3.2	5.5.0.1	30 million W1 300GB	83.1 hours	107	280	
On pSeries p5 561, 4-way 1.8GHz Power5 LPAR						
3.2	5.5.2 / 6.1	1 million W1 10GB	0.3 hour	- / 922	- / 1447	9 5 76 10
3.2	5.5.2 / 6.1	10 million W1 100GB	4.4 hours	742 / 664	1253 / 1165	
3.3	5.5.2 / 6.1	10 million W1 100GB	4.2 hours	695 / 693	1188 / 1182	14 8 65 13
3.3	5.5.2	10 million W2 500GB	4.6 hours	629	740	14 8 65 13
3.3	5.5.2 maxfilesinfs*	10 million W2 500GB	4.1 hours	716	1010	15 10 60 15
3.3	5.5.2 / 6.1	30 million W1 300GB	12.9 hours	679 / 680	996 / 1041	12 7 72 8

Table 2.5 HSM Scan Performance Summary

* Option maxfilesinfs set to 11000000

During the scan, the cpu utilization is 20 – 30%, split by the HSM scout daemon and the GPFS mmfsd64 process. The I/O wait time is around 10%.

Migration

Manual Migration

The following command can be used to migrate specified files manually:

dsmmigrate [-Detail] [-filelist=<filename>] [<files to migrate>]

Automigration

On a HSM-managed file system, when the disk usage reaches the high threshold, HSM will

1. **Migrate** files until the disk usage reaches the low threshold,

2. **Premigrate** files, i.e., copy files to the server storage pools, but still keep them on disk, until the percentage of the file system occupied by the premigrated files reaches the option **Pmpercentage** (the default value is the difference of high threshold and low threshold). This design is to expedite later migration. Note this step will not decrease the disk usage.

Option Maxmigrators

Starting with Version 5.5, HSM scout daemon would search the hash table to select candidates for migration, deliver to a 'dsmautomig' process when the candidate count reaches the option **Maxcandidates** (the default value is 100).

The number of the subordinate 'dsmautomig' processes is specified by the option **Maxmigrators**. When Maxmigrators increases, migration rate increases, with moderate CPU and proportional memory increase. Based on the following test results, we recommend to set option Maxmigrators to 5.

NOTE : If HSM will be used to migrate files directly to tape it is essential to ensure free tape drives for transparent recall's of files. This must be reflected at the MAXMIGRATORS option .

Max-migrators	**Migration Rate**	**mmfsd64 CPU**	**scoutd CPU**	**automig CPU**	**Memory of automig processes** (~ 20MB/process)
1	6 files/sec	6%	3%	9%	20 MB
3	11 files/sec	9%	10%	13%	60 MB
5	14 files/sec	12%	9%	16%	100 MB
7	15 files/sec	12%	11%	15%	
10	16 files/sec	12%	11%	17%	
20	17 files/sec	15%	8%	20%	400 MB

Table 2.6 Option Maxmigrators Impacts Migration Rate and Resource Utilization *

* HSM 5.5.0.1 on pSeries p5 561, 4-way 1.8GHz Power5 LPAR, AIX 6.1

Candidate Search Strategy

HSM scout daemon searches the hash table for candidates in the following order; a step would be executed only if all of its predecessor steps do not generate enough candidates for migration or premigration:

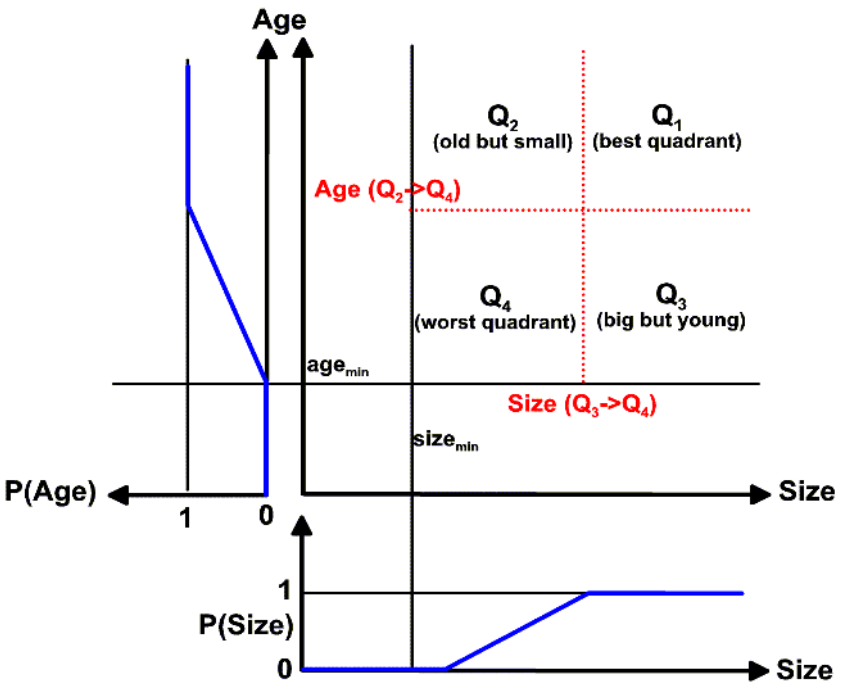
1. Premigrated Search: The premigrated files are assumed to be found and converted to migrated files rapidly, with the data transfer to the server already done.

2. Level 1 Search: This search is to find the largest and the oldest candidate files.

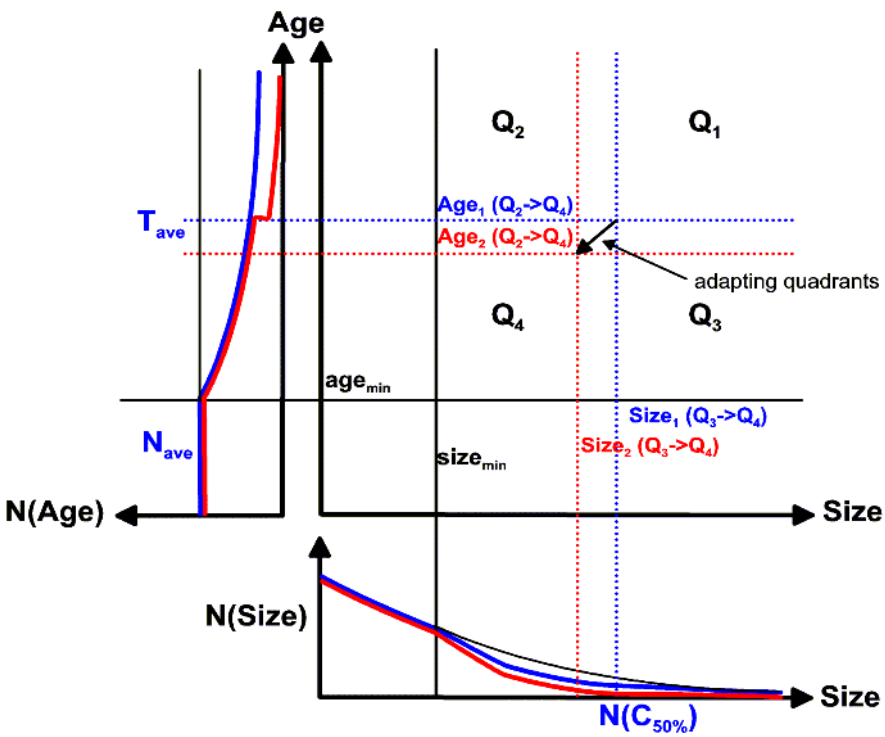
3. Level 2 Search: This search is to find the largest or the oldest candidate files.

4. Level 3 Search: This search is to find the rest of the candidate files.

The level search isn't static. It depends on the current situation on the file system. The following graph displays the internal behaviour of the dsmscoutd candidates selection algorithm:



A criteria for good candidates are files that are big and old. With this, we classify all files into four quadrants. To create the four quadrants Q1 – Q4 we need to compute the Age and Size criterions that divide the files and more or less “good” candidates. To calculate the boundaries we take a random sample of files from the CFI. The random sample is representative for the distribution of size and age in the file system.



First all files that belong to Q1 are migrated because these are the best candidates. When no files are left in Q1 the quadrants Q2 and Q3 will be queried next, because one criterion (age or size) is already fulfilled. The other one can be seen as a probability P(Age) and P(Size) between 0 and 1.

The criteria Age and Size for distinguishing the four quadrants are adapted automatically after each scan of the file system.

Search/Migration Performance

Table 2.7 lists the performance of a typical migration scenario :

- The premigration is designed ongoing in the background and the premigrated files can be read without recall. If the number of premigrated files is very high, HSM is able to free space in a very short time frame. This is the case for both HSM -driven migration and GPFS-driven migration. Premigrated files will be delivered at first to free space and prevent ENOSPACE condition as soon as possible.

  However, the tests show that
  - Even with no premigrated files, the premigrated search still takes 8 minutes.
  - The premigrated file processing is 20% slower than the usual file processing.

  Therefore, adding the time to premigrate and the time to convert to migrated, premigration actually is 2.2 times as expensive as direct migration.

- The first level 1 search picks up 22% files to migrate i.e., the files of 100KB or larger.

4KB	57.8%
8KB	20%
100KB	20%
1 MB	2%
2 MB	0.1%
3 MB	0.1%

- The subsequent level 1 searches could move fast to the fi les delivering the same migrating
rate as the beginning, so the areas already searched are remembered for efficiency, see the
minute “dsmdf” output for the second level 1 search:

Time	**#FilesSearched**	**#Candidates***	**#TotalMigrated**	**InstantMigrateRate**
08:27:19	65491	65491	495909	0 Files/sec
08:28:21	540977	65500	496009	0 Files/sec
08:29:23	540977	65500	496009	0 Files/sec
08:30:25	540977	65500	496009	0 Files/sec
08:31:27	1825600	65956	496409	13 Files/sec
08:32:29	1829009	66700	497209	13 Files/sec

* total candidates including those for the preceding PreMigrated search

Table 2.7 HSM Migration Performance of 10-million File System with Pmpercentage=2 *

SEARCH TYPE	#FILES SEARCHED, %SELECTED	DURATION (HOUR)	THRUPUT FILES /SEC	MB /SEC	TOTAL MIGRATED #FILES	SIZE	AVGFILESIZE	PREMIGRATED FILES	SIZE	DISK USAGE	CPU USAGE USER SYS IDLE IOWAIT
At 79% Disk Usage, Drop Thresholds to High=73%, Low=71%											
PreMigrated	0 0%	0.14 hr	0	0	0	0	0	0	0	79%	
Level 1	2.2 mil 22%	10.53 hr	13.1	2.2	430k	74GB	170KB	65k	11GB	71%	
At 71% Disk Usage, Drop Thresholds to High=69%, Low=67%											
PreMigrated	65k 100%	1.38 hr	13.1	2.3	496k	86GB	173KB	0	0	70%	
Level 1	2.8 mil 8%	4.83 hr	16.5	2.7	651k	112GB	172KB	66k	11GB	67%	
At 67% Disk Usage, Drop Thresholds to High=65%, Low=63%											
PreMigrated	66k 100%	1.39 hr	13.1	2.3	717k	124GB	172KB	74	14MB	66%	
Level 1	3.6 mil 6%	4.83 hr	16.5	2.7	872k	151GB	173KB	65k	11GB	63%	
At 63% Disk Usage, Drop Thresholds to High=61%, Low=59%											
PreMigrated	65k 100%	1.54 hr	11.7	2.0	938k	162GB	172KB	4	1MB	62%	
Level 1	4.4 mil 5%	4.94 hr	16.3	2.7	1097k	190GB	174KB	66k	11GB	59%	
At 59% Disk Usage, Drop Thresholds to High=57%, Low=55%											
PreMigrated	66k 100%	1.39 hr	13.1	2.3	1162k	201GB	173KB	86	19MB	58%	
Level 1	5.2 mil 4%	4.96 hr	16.2	2.7	1321k	229GB	173KB	66k	11GB	55%	

* Data Workload W2, maxfilesinfs set to 11 million, on pSeries p5 561, 4-way 1.8GHz Power5 LPAR, AIX 6.1

Table 2.8 lists the performance of a migration scenario with Pmpercentage=0 to avoid the expensive hourly premigration. However, the premigrated searches still take 5 minutes.

SEARCH TYPE	#FILES SEARCHED, %SELECTED	DURATION (HOUR)	THRUPUT FILES /SEC	MB /SEC	TOTAL MIGRATED #FILES	SIZE	AVGFILESIZE	PREMIGRATED FILES	SIZE	DISK USAGE	CPU USAGE USER	SYS	IDLE	IOWAIT
At 88% Disk Usage, Drop Thresholds to High= 87%, Low=83%														
PreMigrated	0 0%	0.08 hr	0	0	0	0	0	0	0	88%	16	11	73	0
Level 1	0.9 mil 22%	4.18 hr	13.1	2.2	197k	38GB	192KB	0	0	83%	10	4	74	12
At 83% Disk Usage, Drop Thresholds to High= 82%, Low=78%														
PreMigrated	0 0%	0.08 hr	0	0	0	0	0	0	0	83%	16	11	73	0
Level 1	1.7 mil 13%	4.82 hr	16.5	2.7	421k	81GB	192KB	0	0	78%	9	4	74	12
At 78% Disk Usage, Drop Thresholds to High= 77%, Low=73%														
PreMigrated	0 0%	0.08 hr	0	0	0	0	0	0	0	78%	16	11	73	0
Level 1	2.5 mil 9%	4.95 hr	12.9	2.2	651k	126GB	193KB	0	0	73%	10	4	74	11
At 73% Disk Usage, Drop Thresholds to High= 72%, Low=68%														
PreMigrated	0 0%	0.08 hr	0	0	0	0	0	0	0	73%	16	11	73	0
Level 1	3.3 mil 7%	4.98 hr	12.8	2.2	882k	170GB	193KB	0	0	68%	10	4	74	12
At 68% Disk Usage, Drop Thresholds to High= 67%, Low=63%														
PreMigrated	0 0%	0.08 hr	0	0	0	0	0	0	0	68%	16	11	73	0
Level 1	4.1 mil 6%	4.91 hr	12.9	2.2	1110k	214GB	194KB	0	0	63%	10	4	74	11

Table 2.8 HSM Migration Performance of 10-million File System with Pmpercentage=0

* Data Workload W2, maxfilesinfs set to 11 million, on pSeries p5 561, 4-way 1.8GHz Power5 LPAR, AIX 6.1

3

GPFS Using TSM HSM as External Pool

For GPFS Storage Management, this section describes its design and internal mechanism with the performance measurement for scan and migration.

Mechanism

GPFS provides a feature of policy-driven tiered storage management, including

- Three types of **storage pools**:
    1. A system storage pool where metadata is stored.
    2. User storage pools for data.
    3. External storage pools managed by external applications, such as HSM.
- **policy rules** to manage data placement and movement.
- **filesets** to manage data at a finer granularity than file systems.
- **User-defined exit scripts** to be executed when rules are fired, for example, call HSM migrate commands with specified files when storage pool usage reaches certain thresholds.

A Simple Example

For example, the following **volume specification** assigns volume "*gpfs1nsd*" to the system storage pool, volume "*gpfs2nsd*" to a user pool named "*mail*",,and the rest of the two volumes to a user pool named "*data*":

gpfs1nsd:::dataAndMetadata:1::

gpfs2nsd:::dataOnly:1::*mail*

gpfs3nsd:::dataOnly:1::*data*

gpfs4nsd:::dataOnly:1::*data*

The following **data placement rules** specify all files with name ending ".*mail*" should be put to the "*mail*" storage pool, and the "*data*" storage pool is the default pool to store data:

RULE '*2mail*' SET POOL '*mail*' WHERE name like '*%.mail'*

RULE 'default' SET POOL *'data'*

The following **external pool rule** defines a "*hsm*" external storage pool, located at the server "*mortimer*", and the program or script "*hsmScript"* will be executed when the external pool is used:

```
RULE EXTERNAL POOL 'hsm' EXEC './hsmScript' OPTS '-server mortimer'
```

The script "*hsmScript*", accepting the first argument as the command, the second argument as a file containing the list of files to operate on, could call 'dsmmigrat e' for command "MIGRATE", etc.

```
case $1 in
LIST) ls $2 ;;
MIGRATE)
  cp $2 tmpFile
  cut -d" " -f7 tmpFile >$2
  dsmmigrate -Detail -filelist=$2 ;;
TEST) echo "==== HSMtest " $2 $3 ;;
*)   ;;
esac
```

The following **exclude rule** defines the HSM files to exclude from processing:

```
RULE 'Exclude HSM System Files' EXCLUDE WHERE PATH_NAME LIKE
'%/.SpaceMan%'
```

Finally, the following **migrate rule** defines that when the "*data*" storage pool reaches 80% usage, files should be migrated the "*hsm*" external storage pool, until the usage drops to below 75%. The migration candidates should be chosen based on the size of the files:

```
RULE 'Mig2HSM' MIGRATE
    FROM POOL 'data'
        THRESHOLD(80,75)
        WEIGHT(KB_ALLOCATED)
    TO POOL 'hsm'
```

An Advanced Example

The following example illustrate how to use the "DEFINE" command to adapt the settings.

Change stub size for GPFS from Default 0:

```
DEFINE (
   stub_size, 0
)
```

Differentiate premigrated and migrated files by checking the file size (the managed flags indicating that the file is co-managed by both GPFS and HSM):

```
DEFINE (       /*===   '%M%'–type file larger than stub size ===*/
    is_premigrated,
   (MISC_ATTRIBUTES LIKE '%M%' AND KB_ALLOCATED > stub_size)
)
    DEFINE (       /*===   '%M%'–type file equal to stub size ===*/
    is_migrated,
   (MISC_ATTRIBUTES LIKE '%M%' AND KB_ALLOCATED  == stub_size)
)
```

Define useful attributes:

```
DEFINE (       /*===   the number of days since last last access ===*/
    access_age,
   (DAYS(CURRENT_TIMESTAMP)  - DAYS(ACCESS_TIME))
)

DEFINE (       /*===   unit conversion ===*/
    mb_allocated,
   (INTEGER(KB_ALLOCATED / 1024))
)
```

Extend the exclude list:

```
DEFINE (
    exclude_list,
   (PATH_NAME LIKE '%/.SpaceMan/%'   OR NAME LIKE '%dsmerror.log%')
)
```

Modify the weight expression that the policy engine uses to deliver the candidates in the optimal order:

Best candidates : premigrated files which were premigrated for a while

Medium candidates : large and old resident files

Worst candidates : young and small files , or files which were premigrated today.

```
DEFINE (
```

```
    weight_expression,

   (CASE
     WHEN access_age <= 1  THEN 0                          /*=== The file is very young, the
ranking is very low ===*/
     WHEN mb_allocated < 1  THEN access_age                /*=== The file is very small, the
ranking is low ===*/
     WHEN is_premigrated     THEN mb_allocated * access_age * 10    /*=== The file is
premigrated and large and old enough, the ranking is very high ===*/
     ELSE                              mb_allocated * access_age          /*=== The file is resident
and large and old enough, the ranking is standar d ===*/
   END)
)
```

Set up rules:

```
RULE EXTERNAL LIST 'candidatesList' EXEC
'/tmp/HSM++/borodin.gpfs.exec.list'

RULE EXTERNAL POOL 'hsm' EXEC '/tmp/HSM++/borodin.gpfs.exec.hsm'

RULE 'bestCandidates' LIST 'candidatesList'

   WEIGHT (weight_expression)

   SHOW (weight_expression)

   WHERE NOT (exclude_list)

      AND NOT (is_migrated)
```

Migrate files based on the theshold settings and defined optimizing criteria:

```
RULE 'default' SET POOL 'system'

RULE 'TM1' MIGRATE FROM POOL 'system'

      THRESHOLD(10,8,5)

      WEIGHT (weight_expression)

      TO POOL 'hsm'

      WHERE NOT (exclude_list)

         AND NOT (is_migrated)
```

Scan

The design of GPFS scan follows a completely different paradigm from HSM's:

1. GPFS only scans when needed, that is, rules are applied or triggered and file information is in need to select candidates.

2. The GPFS scan mechanism bases on a subsequent inode scan. This method is very fast; wheras HSM is bases on the DMAPI interface, hence it need to use the DMAPI implemented directory scan, the performance is slower.

3. GPFS does not build permanent data structure or save scan results.

The following are the performance of the directory scan and the subsequent inode scan. The directory scan features with high I/O wait.

Number of Files	Directory Scan Time	Dir Scan CPU Usage User Sys Idle IOWait	Inode Scan Time	Inode Scan CPU Usage User Sys Idle IOWait
7 million	8.0 minutes	27 7 29 37	3.1 minutes	25 6 65 4
10 million	12.0 minutes	20 7 33 40	4.0 minutes	22 6 60 13

Table 3.1 GPFS Scan Performance

***** GPFS 3.3, On pSeries p5 561, 4-way 1.8GHz Power5 LPAR, AIX 6.1

Migration

GPFS calls HSM to migrate candidate files. When a storage pool reaches the defined threshold or when a **mmapplypolicy** command is invoked, GPFS processes the metadata, generates a list of files, and invokes a user provided script or program which initiates the appropriate commands for the external data management application to process the files. This allows GPFS to transparently control offline storage and provide a tiered storage.

The mmapplypolicy command uses the HSM command dsmmigrate for the migration of files based on filelists. Per default, the GPFS policy engine starts 24 migration processes per node and generates file lists with 100 candidate files included. With GPFS 3.3, the following new attributes are added to the mmapplypolicy command to allow customers to tune for their specific need:

-n DirThreadLevel

The number of threads that will be created and dispatched within each mmapplypolicy process during the directory scan phase. The default is 24.
It is responsible for the number of dsmmigrate processes which will be started in parallel on each node. It is very important for customers who migrate directly to tape.

- B MaxFiles

The maximum number of files that will be placed into any of the temporary filelist files that are passed to the programs or scripts invoked via the external pool and external list mechanism. When more than MaxFiles are chosen, mmapplypolicy invokes the external program multiple times. The default is 100.

Table 3.2 lists the resultant performance:

1. GPFS always selects the best candidates available in its fast full scan, in contrast to HSM's partial scan. For example, in the first test that thresholds are decreased to trigger migration, all the files selected to migrate are the largest 3 MB files.

2. The number of files migrated per second is 3 – 4 times to that of HSM, and the byte throughput is 40 - 58 MB/sec, versus HSM's best 2.2 – 2.7 MB/sec, partly due to the candidate files are 5 – 10 times larger than HSM's.

Migration	Migration Time	MigrationThruput		Migrated			CPU Usage			
		Files/sec	MB/sec	#Files	Size	AvgFileSize	User	Sys	Idle	IOWait
7-million file system (***Overall performance need to add **Scan Time 11 minutes**)										
94% - 91% (small load)	4.2 minutes	11	40	3k	10 GB	3.1 MB	11	22	30	38
91% - 85%	8.0 minutes	33	51	16k	25 GB	1.6 MB	27	27	22	25
85% - 80%	7.9 minutes	42	47	20k	22 GB	1.1 MB	23	36	28	13
80% - 75%	8.4 minutes	42	47	21k	23 GB	1.1 MB	13	33	33	21
75% - 70%	8.4 minutes	41	47	21k	23 GB	1.1 MB	28	34	23	16
70% - 60% (double load)	15.1 minutes	47	49	40k	42 GB	1.1 MB	18	32	36	14
10-million file system (***Overall performance need to add **Scan Time 16 minutes**)										
86% - 81%	10.1 minutes	18	57	11k	34 GB	3.0 MB	17	24	23	34
81% - 76%	12.7 minutes	30	45	23k	34 GB	1.5 MB	25	33	27	15
76% - 71%	9.9 minutes	54	58	32k	34 GB	1.1 MB	23	38	24	15
71% - 66%	11.7 minutes	52	56	33k	36 GB	1.1 MB	21	47	10	22
66% - 61%	10.4 minutes	53	58	33k	36 GB	1.1 MB	18	45	14	22
61% - 56%	11.6 minutes	48	52	33k	36 GB	1.1 MB	19	41	20	21

Table 3.2 GPFS Initiated HSM Migration Performance

4

Comparison of HSM-driven and GPFS-driven Space Management

Performance Comparison

The following table compares the performance of the following automatic space managements:

1. The sophisticated CFI-based HSM (since 5.5) space management
2. The simple and straightforward GPFS space management

Table 4.1 and Figure 4.2 demonstrate the performance of the two systems in a file system containing 10 million files. When the data in the file system gradually grows to exceed the specified high threshold, if the thresholds and monitor mechanism are based on HSM, HSM would use the information stored from the previous scan to start migration, as the red line shows; whereas if the thresholds and monitor mechanism are based on GPFS, GPFS would run the scan first to choose the best candidates, then send the candidat e list to HSM to migrate, as the green line shows. It is clear that GPFS mechanism is much more efficient than HSM's, with faster scan, better candidate selection, and faster migration.

Operation	Duration	Comments
HSM Scan	250 minutes	Run regularly to keep metadata hash table up-to-date
GPFS Scan	16 minutes	Run before migration
HSM Migrating 34 GB	231 minutes	Premigrated Search 5 minutes Level 1 Search 226 minutes Candidate Selection: local optimal, avg file size 170KB
GPFS Initiated HSM **Migrating 34 GB**	12 minutes	Migration 12 minutes (based on underlying HSM dsmmigrate) Candidate Selection: optimal for the whole file system, avg file size 3MB

* HSM: 5.5.2, with Maxfilesinfs 11 millions, Maxmigrators 5, Pmpercentage 0
GPFS 3.3 , On pSeries p5 561, 4-way 1.8GHz Power5 LPAR, AIX 6.

Table 4.1 The Performance of HSM and GPFS Operations on a 10-million file system*

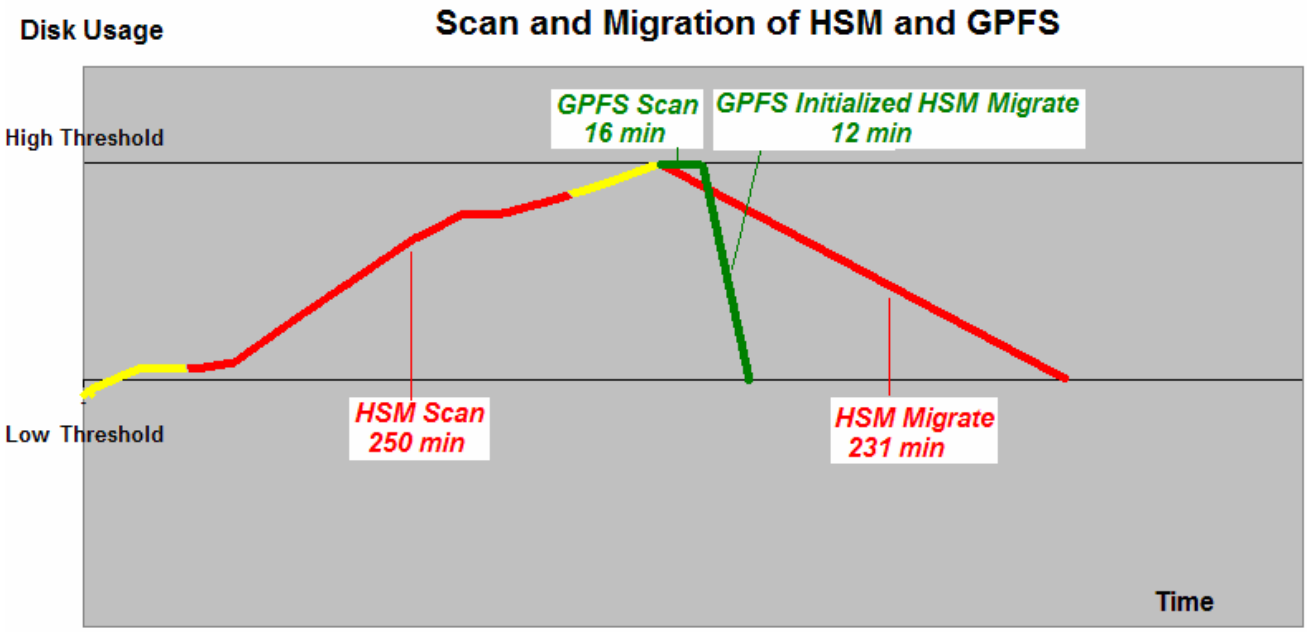


Figure 4.2 The Performance of HSM and GPFS Operations on a 10-million file system

Scalability

We have run the performance tests with various sizes of file systems, and found the performance of scan and migration is mainly scalable, up to 30 million files. We might extend the tests to larger file systems, if resources allow.

An Efficient Space Management

With the performance difference of almost an order of magnitude, we would recommend choose the GPFS space management with using HSM as an external storage pool. This approach lets GPFS processes the metadata, transparently control and provide a tiered storage, with HSM doing the underlying work of moving and managing real data.

With this setup, the current HSM performance could be enhanced significantly, to satisfy the need of the customers to free up the primary storage by migration data to secondary storage in high speed.

In summary, the GPFS policy engine is a very powerful tool to replace the normal HSM candidates selection in GPFS environments. We recommend to all our GPFS customers to switch to this mechanism for the threshold migration. Nevertheless, HSM supports other file system types like JFS2 and VXFS which cannot provide such an advanced inode scan mechanism, thus for the multiple platform support, it is needed to support HSM implemented scan and candidates selection functionality.

5

References

[1] Jens-Peter Akelbein, et al., "*IBM Tivoli Storage Manager for Space Management, Understanding and Optimizing HSM*", Version 2.0.

[2] TSM for Space Management for Unix – Users Guide Version 5.5.

[3] TSM for Space Management for Unix – Users Guide Version 6.3.

[4] TSM for Space Management for UNIX – GPFS Integration – Tivoli Field Guide.

[5] "*General Parallel File System Advanced Administration Guide*", Version 3 Release 2.1.