IBM Software Group

Automatic Database Configuration and Memory Tuning

Adam Storm

Software Developer, IBM Toronto Lab

Agenda

- The configuration challenge
- Automatic configuration of new databases
  - The Configuration Advisor
- Adaptive memory configuration
  - The Self Tuning Memory Manager (STMM)
    - How STMM works
    - Basic tuning
    - Tuning DATABASE_MEMORY
    - STMM in DPF
  - Experimental results
  - STMM configuration changes – FOR REFERENCE ONLY
- Summary

The Configuration Challenge

The Configuration Challenge

- Modern database engines are complex
  - Complexity is the cost of providing a feature rich, flexible product
  - With complexity comes configuration challenges
    - Not just a DB2 issue
- The Configuration Challenge
  - How can DB2, a highly tunable database engine, be configured easily for optimal performance?
  - Must be able to work well for all users regardless of skill level
    - Must not require detailed input from the user
  - Must be adaptive
    - Workload changes should modify configuration

The Configuration Advisor

The Configuration Advisor

- Sets the 36 most important configuration parameters automatically
  - Also sets the sizes of all buffer pools
- Runs in seconds and can generate a configuration that can be very close to optimal
- Accessible through command line, control center and an API
- Existed in version 8 but algorithms were modified for version 9 to improve robustness
  - Advisor is multi-database and DPF aware
- NEW IN VERSION 9 - the advisor automatically configures databases when they are created
  - No longer a need to manually run the advisor

The Configuration Advisor

- Senses the underlying system to provide fitting configuration
  - Number of CPUs, memory available, physical disks, OS Type
- Administrator is also able to provide input to the advisor to influence configuration
  - Percentage of system memory to use
  - Type of workload (decision support, OLTP or mixed)
  - Number of statements per transaction
  - Expected transactions per minute
  - Administrative priority
  - Is database populated with data?
  - Number of expected local and remote applications
  - Isolation level
- If no input is specified, advisor configures system based on sensed information and typical input

Configuration Advisor Output

```
> db2 autoconfigure apply db and dbm

        Former and Applied Values for Database Manager Configuration

 Description                                        Parameter   Former Value        Applied Value
 -------------------------------------------------------------------------------------------------
 Application support layer heap size (4KB)    (ASLHEAPSZ) = 15                    15

 No. of int. communication buffers(4KB)(FCM_NUM_BUFFERS) = 512                     512

 Number of FCM request blocks                (FCM_NUM_RQB) = 256                   256

 Enable intra-partition parallelism     (INTRA_PARALLEL) = YES                     YES

 Maximum query degree of parallelism   (MAX_QUERYDEGREE) = 4                       4

 Max number of existing agents               (MAXAGENTS) = 200                     200

 Agent pool size                        (NUM_POOLAGENTS) = 40                      40

 Initial number of agents in pool       (NUM_INITAGENTS) = 0                       0

 Max requester I/O block size (bytes)          (RQRIOBLK) = 32767                   32767

 Sort heap threshold (4KB)                   (SHEAPTHRES) = 140201                  140201
```

Configuration Advisor Output (cont.)

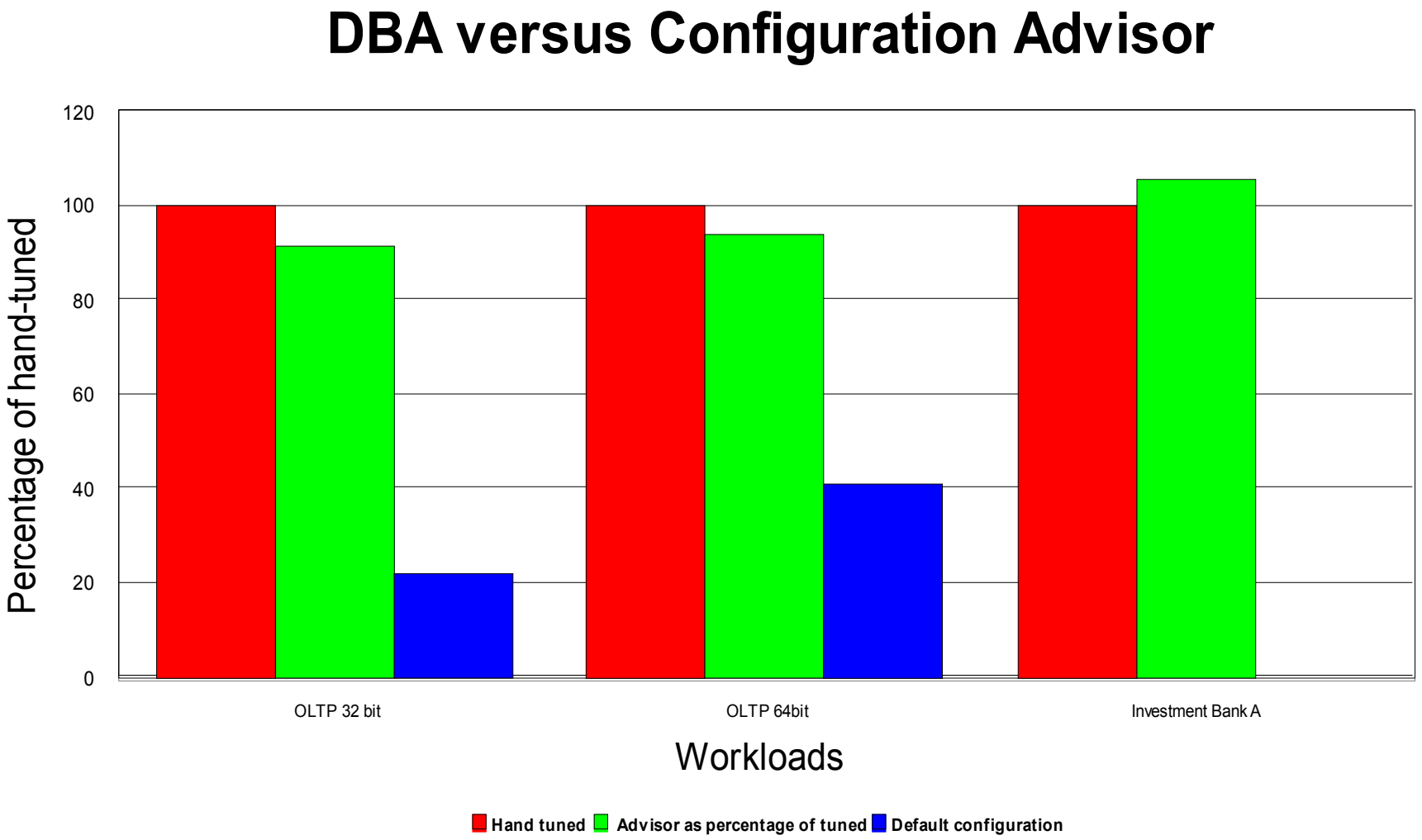
```
Former and Applied Values for Database Configuration

 Description                                    Parameter   Former Value        Applied Value
 ------------------------------------------------------------------------------------------
 Max appl. control heap size (4KB)      (APP_CTL_HEAP_SZ) = 128               160
 Max size of appl. group mem set (4KB) (APPGROUP_MEM_SZ) = 20000             19200
 Default application heap (4KB)              (APPLHEAPSZ) = 256               256
 Catalog cache size (4KB)               (CATALOGCACHE_SZ) = (MAXAPPLS*4)      270
 Changed pages threshold                 (CHNGPGS_THRESH) = 60                60
 Database heap (4KB)                             (DBHEAP) = 1200              4047
 Degree of parallelism                       (DFT_DEGREE) = 1                 1
 Default tablespace extentsize (pages)    (DFT_EXTENT_SZ) = 32                32
 Default prefetch size (pages)          (DFT_PREFETCH_SZ) = 32                32
 Default query optimization class          (DFT_QUERYOPT) = 5                 5
 Max storage for lock list (4KB)                (LOCKLIST) = 100               3775
 Log buffer size (4KB)                          (LOGBUFSZ) = 8                 101
 Log file size (4KB)                           (LOGFILSIZ) = 1000              1024
 Number of primary log files                  (LOGPRIMARY) = 3                 3
 Number of secondary log files                 (LOGSECOND) = 2                 0
 Max number of active applications              (MAXAPPLS) = 40                40
 Percent. of lock lists per application         (MAXLOCKS) = 10                60
 Group commit count                            (MINCOMMIT) = 1                 1
 Number of asynchronous page cleaners    (NUM_IOCLEANERS) = 1                 1
 Number of I/O servers                    (NUM_IOSERVERS) = 3                 4
 Package cache size (4KB)                    (PCKCACHESZ) = (MAXAPPLS*8)      859
 Percent log file reclaimed before soft chckpt (SOFTMAX) = 100               120
 Sort list heap (4KB)                           (SORTHEAP) = 256               7010
 SQL statement heap (4KB)                       (STMTHEAP) = 4096              4096
 Statistics heap size (4KB)                (STAT_HEAP_SZ) = 4384              4384
 Utilities heap size (4KB)                 (UTIL_HEAP_SZ) = 5000              524288


        Former and Applied Values for Bufferpool(s)

 Description                                    Parameter   Former Value        Applied Value
 ------------------------------------------------------------------------------------------
 IBMDEFAULTBP                                Bufferpool size = 1000              2340594
```

Experimental results



Memory Configuration

Special considerations with memory tuning

- Memory configuration can have a dramatic effect on DBMS performance
  - Tuning memory through educated trial and error can be difficult and time consuming
- Optimal configuration may not be static
  - To achieve optimal performance, it may be necessary to adapt the memory configuration as workload shifts
- Configuration advisor only provides static configuration
  - When run on the same system with the same input
- System performance would benefit if memory could be adaptively tuned based on workload requirements

The Self Tuning Memory Manager – Simplified, Automated Memory Tuning

Memory tuning in the past: Analyzing monitor output

Memory tuning in the past: Change configuration and wait

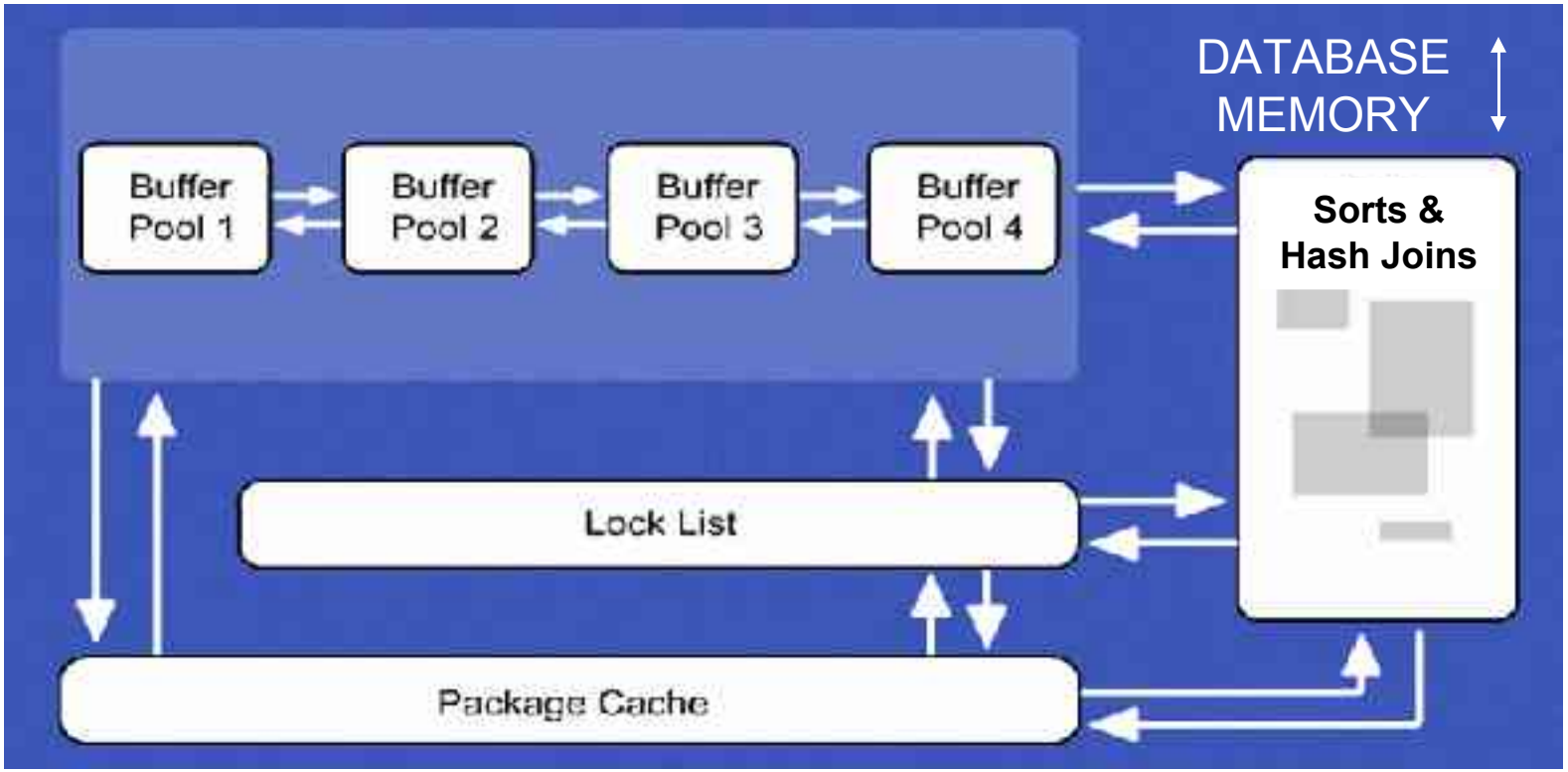
Memory Tuning in Version 9 - STMM

- DB2 Version 9 introduces a revolutionary memory tuning system called the Self Tuning Memory Manager (STMM)
  - Works on main database memory parameters
    - Sort, locklist, package cache, buffer pools, and total database memory
  - Automated online memory tuning
    - Requires no DBA intervention
  - Senses the underlying workload and tunes the memory based on need
  - Can adapt quickly to workload shifts that require memory redistribution
  - Adapts tuning frequency based on workload

STMM Modes of operation

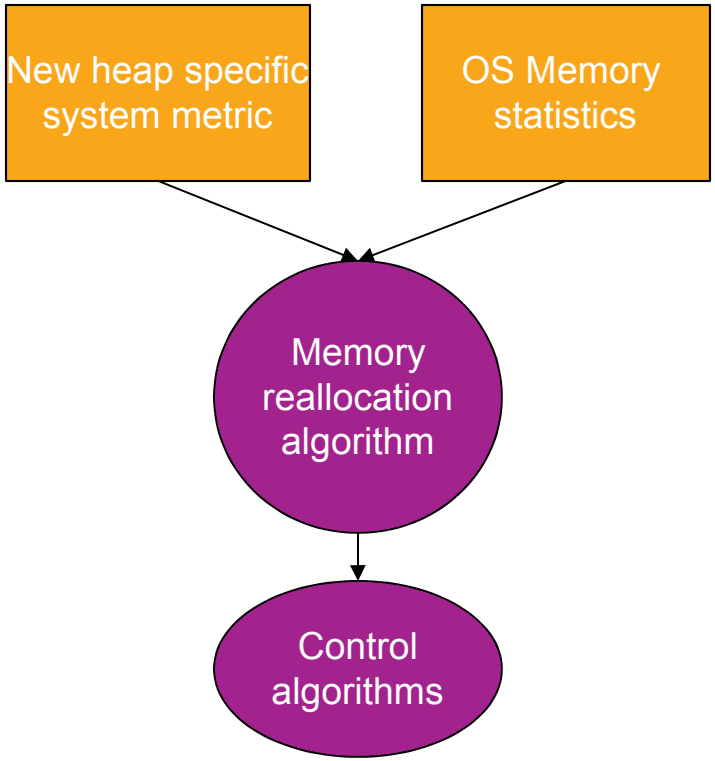
- Works in two different modes
  - Tuning total database memory consumption
    - Takes from, and returns memory to, the OS as necessary
    - Total amount of memory used by DB2 can grow over time
    - Requires only one heap for tuning
  - Fixed total database memory consumption
    - Memory tuning still occurs but total memory used by database is constant
    - For one heap to grow another heap must shrink
      - Requires two heaps to be able to tune
- Is able to tune multiple databases and instances on the same box at the same time
- Works in non-partitioned and in partitioned (DPF) environments

Autonomic Self Tuning Memory



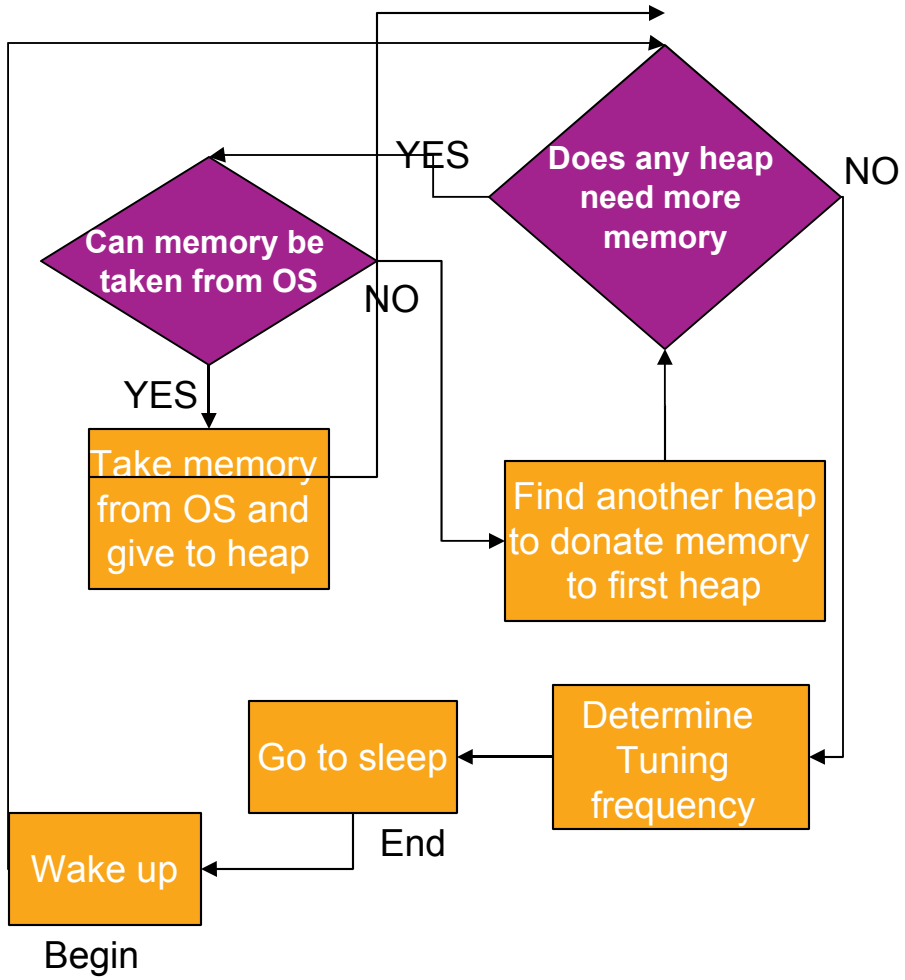
How does STMM work?

- Uses totally new system metrics
- Constantly monitors system to make use of any free OS memory (only if system is set to tune total database memory consumption)
- Works iteratively to determine an optimal memory configuration for all heaps
  - Iterative approach prevents instability
- Control algorithms help determine interval length and prevent oscillations



STMM - In each tuning interval

- Tuner wakes up from sleep
- Determine if memory configuration is sub-optimal
  - Some heaps will be needy, others will have more than enough memory
- If total database memory is being tuned
  - Determine whether OS has free memory that can be used
  - Use this memory to satisfy needy heaps
- If total database memory is not being tuned or if taking memory from the OS couldn't satisfy all heaps
  - Take memory from heaps with more than enough and give to those that are needy
- Continue until no more memory can be moved
  - In each interval each heap can only grow by 50% or decrease by 20%
- Determine tuning frequency based on workload



STMM and the buffer pools

- Trades memory between buffer pools based on relative need
  - New metrics determine where memory is most needed such that total system time is reduced
- Zero, one or more buffer pools can be set to AUTOMATIC
  - In newly created version 9 databases, all buffer pools default to AUTOMATIC
- Works with buffer pools of any page size
  - Transfers from a buffer pool with 8 k pages to one with 4 k are 1:2
- Decreasing the buffer pools can take a lot of time
  - Must write out all dirty pages in memory being freed
  - If pages are in use the resize may wait on locks
  - A large percentage of tuning time could be spent on alter buffer pools
    - Not necessarily a concern, just something to keep in mind

STMM and total database memory consumption

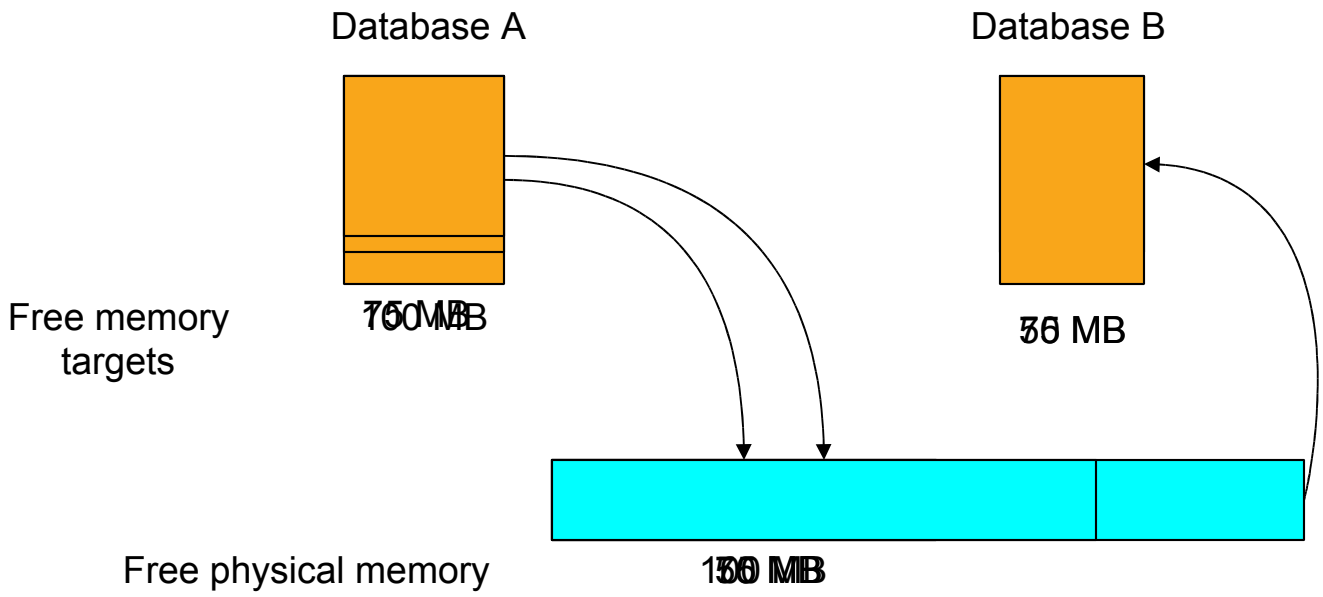
- STMM tunes total database memory consumption if DATABASE_MEMORY is set to AUTOMATIC or a numeric value
  - If set to AUTOMATIC, memory is taken from, and returned to, the OS if required by the database
    - DBA need not know how much memory to allocate to DB2
    - This is the default for newly created version 9 databases
  - If set to a numeric value, memory is given to AUTOMATIC heaps up to the numeric value
    - Allows DBA to set total memory consumption for the database
    - DB2 will then distribute the memory to optimize performance
  - If set to COMPUTED, no DATABASE_MEMORY tuning will occur
    - When database starts, memory requirements are computed based on the heap configuration
    - Once the database starts, the database shared memory set is allocated based on the computation
    - Version 8 AUTOMATIC behavior

Setting DATABASE_MEMORY to AUTOMATIC

- Memory tuner tries to maintain some amount free physical memory on the box at all times
- As long as more free physical memory is available databases are allowed to use more memory
  - Memory is only consumed if database finds that there is a good use for the memory
- If less than enough free physical memory is available databases are asked to free up some memory
  - Memory may not be freed if database needs the memory
    - This should only occur in small amount of cases where database is desperately in need of more memory

Tuning DATABASE_MEMORY – Multiple databases sharing the memory on the box

- Each database sets its own free memory target based on its need for memory relative to other databases
  - New inter-database (inter-instance) communication



Tuning with DPF

- Some assumptions are made on the underlying system
  - All data nodes are similarly distributed and require similar memory
  - All physical nodes contain the same number of logical nodes
    - Or all logical nodes require the same amount of memory
      - Logical nodes/memory ratio must be consistent
- Why the assumptions?
  - One memory tuner for all nodes
  - Tuner tries to maintain a consistent configuration on all nodes
  - In each interval, new configuration is sent out to all nodes
- Tuning node is chosen automatically at first database activation
  - Tuning node selection algorithm tries to find a node that is representative of an average data node
  - Tuning node can be changed through the Admin Stored Procedure

Tuning with DPF

- In each interval
  - Collect memory statistics on tuning node
    - Since statistics are only collected from the tuning node, workload must be similar on all nodes
  - Determine new configuration on tuning node
  - Apply new configuration to all nodes
    - If configuration change fails on any node
      - Rollback change on all nodes
  - Determine sleep time based on tuning node
- Tuning only occurs if database is explicitly activated
  - Must have all nodes active or else configuration will be different

Tailoring STMM with DPF

- If main STMM parameter (SELF_TUNING_MEM) is off at a particular node, no tuning will occur
  - Tuning should be turned off for atypical nodes
    - Catalog nodes with no data
    - Coordinator nodes that don't directly process queries
- Tuning can be turned off for one or more parameters on any given node
  - If STMM configuration update arrives at a node and that parameter isn't set to AUTOMATIC at that node, nothing changes
  - Only parameters set to AUTOMATIC on the tuning node will generate configuration updates

STMM Usage Scenarios

Scenarios where STMM shines – Buffer pool tuning

- Difficult to tune memory when there are multiple buffer pools
- As number of buffer pools increases, possible configurations increases exponentially
- STMM works with multiple buffer pools regardless of page size
- Trades memory between buffer pools
  - Ensures that total memory doesn't change
    - 1 8k page becomes 2 4k pages in transfer
- Works so well that STMM is being used to tune benchmark systems in house
  - Used in several small scale TPC-C benchmarks

Scenarios where STMM shines – Memory varied workloads

- Some workloads have dramatically varied memory demands
  - Periods with high transaction throughput
  - Periods with long running transactions or online utilities
- STMM constantly re-evaluates the memory requirements
  - Can update the memory up to 60 times an hour
- Will optimize the memory usage based on the currently running workload
- Very difficult to perform similar tuning manually

Scenarios where STMM shines – Unknown memory requirements

- New workload with unknown memory requirements
  - Alternatively, new DB2 administrator unfamiliar with memory model
- STMM works deep down in DB2 and is able to sense workload memory requirements
- Tunes quickly enough to bring production systems from out of the box configuration to optimal in an hour or less
- Requires absolutely no DBA interaction once turned on
- Performs several weeks of manual (trial and error) tuning every hour
- Will stop tuning automatically when it reaches optimal configuration

Scenarios where STMM shines – New DPF configurations

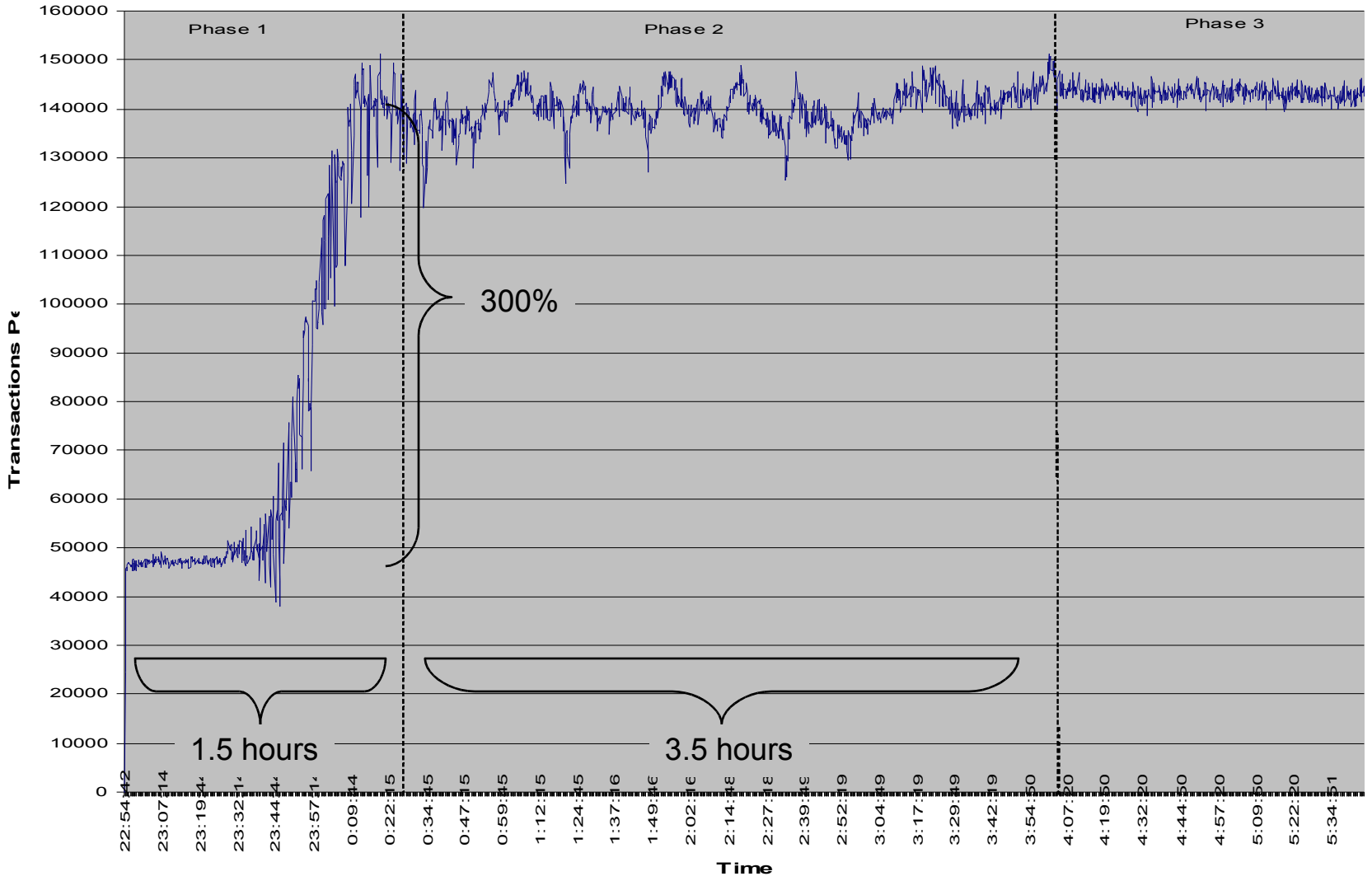
- New system setup with entirely uniform nodes
  - Both data distribution and hardware should be uniform
- Allows DBA to optimize the configuration for all nodes at the same time
- Maintains all nodes at the same configuration for easy future administration
- Chooses tuning node automatically
  - Can be over-ridden if necessary

STMM Experimental Results

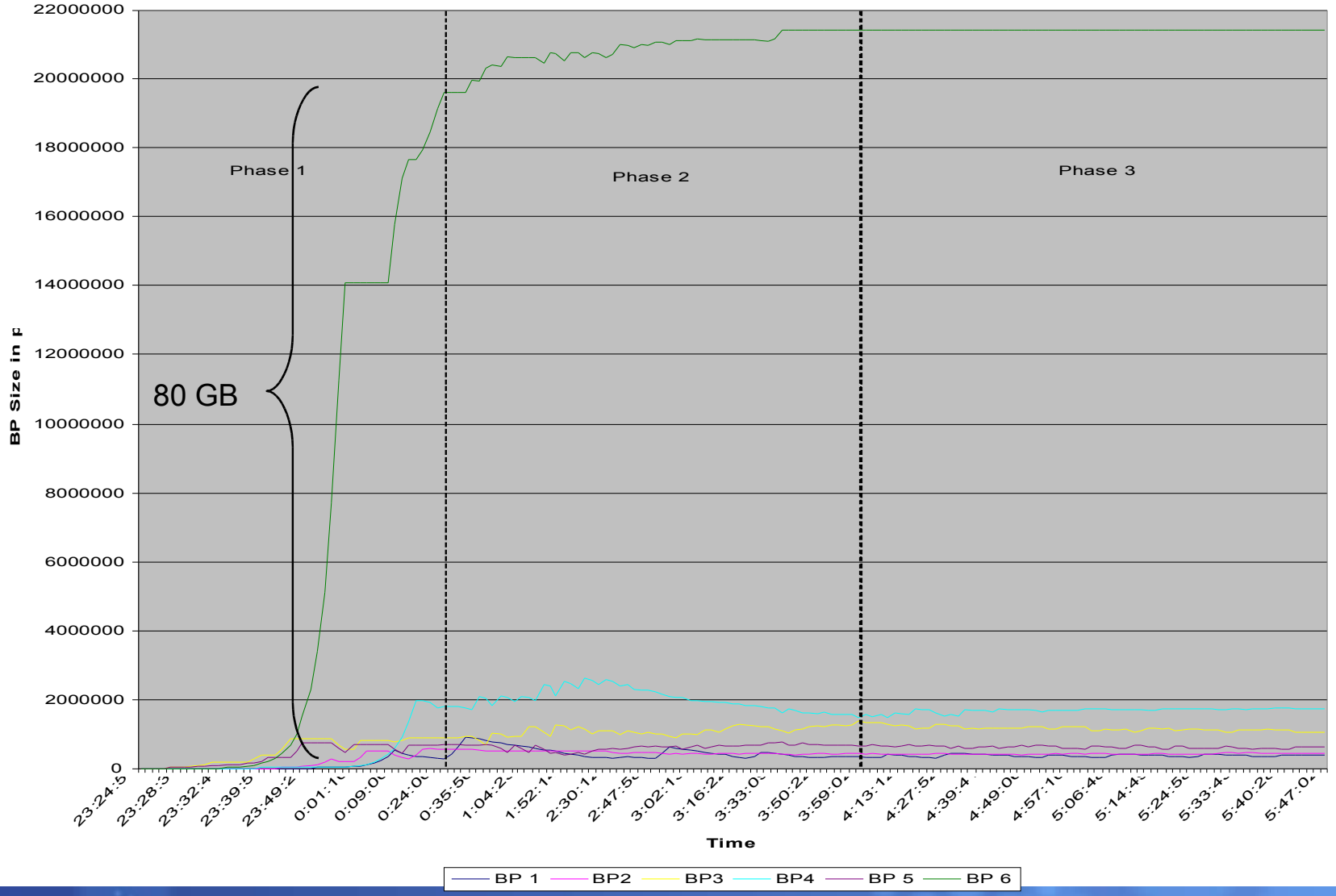
STMM in action – Tuning an OLTP workload

- 370 clients running transaction processing workload
- Running on a machine with:
  - 128 GB of RAM
  - 2 TB database
  - 494 * 36 GB disks
- Workload is very sensitive to buffer pool sizing
- Each of the 13 buffer pools are started with 1000 pages
  - 1000 pages is the default size for a newly created buffer pool
- Workload is started and STMM begins tuning
- STMM should dramatically improve performance…

STMM in action – Tuning an OLTP workload



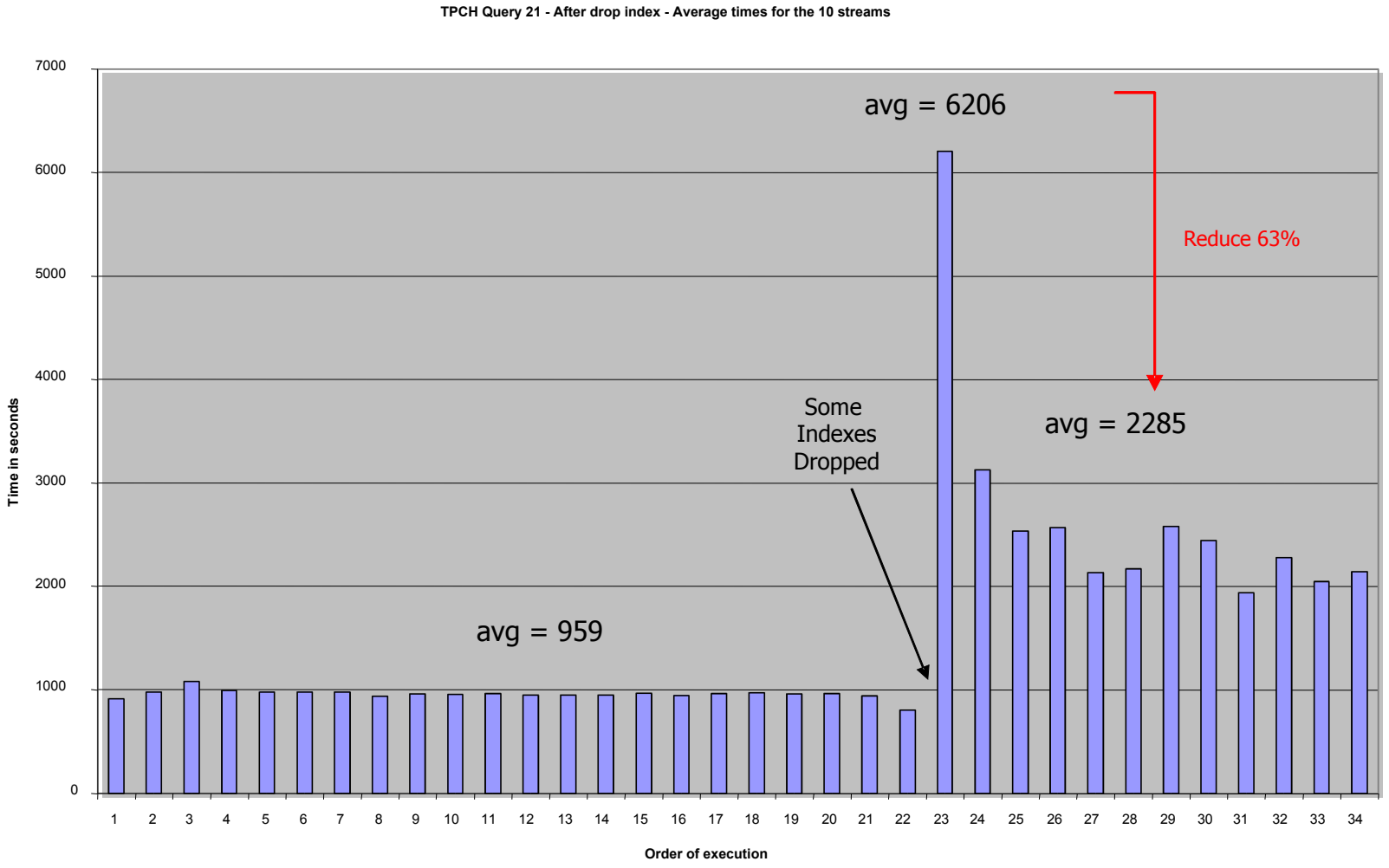
STMM in action – BP sizes during tuning



STMM in action – Oops, dropped my indexes

- 10 agents executing queries with “order by” clause
- At first, queries use indexes to avoid sorting
- After several iterations some of the indexes are accidentally dropped
  - Simulates DBA error in dropping vital indexes
- Lack of indexes forces sorts to be performed
- Dramatically increases the demand on the sort memory
- With only manual tuning, workload will likely get much slower
- STMM should be able to alleviate some of the burden…

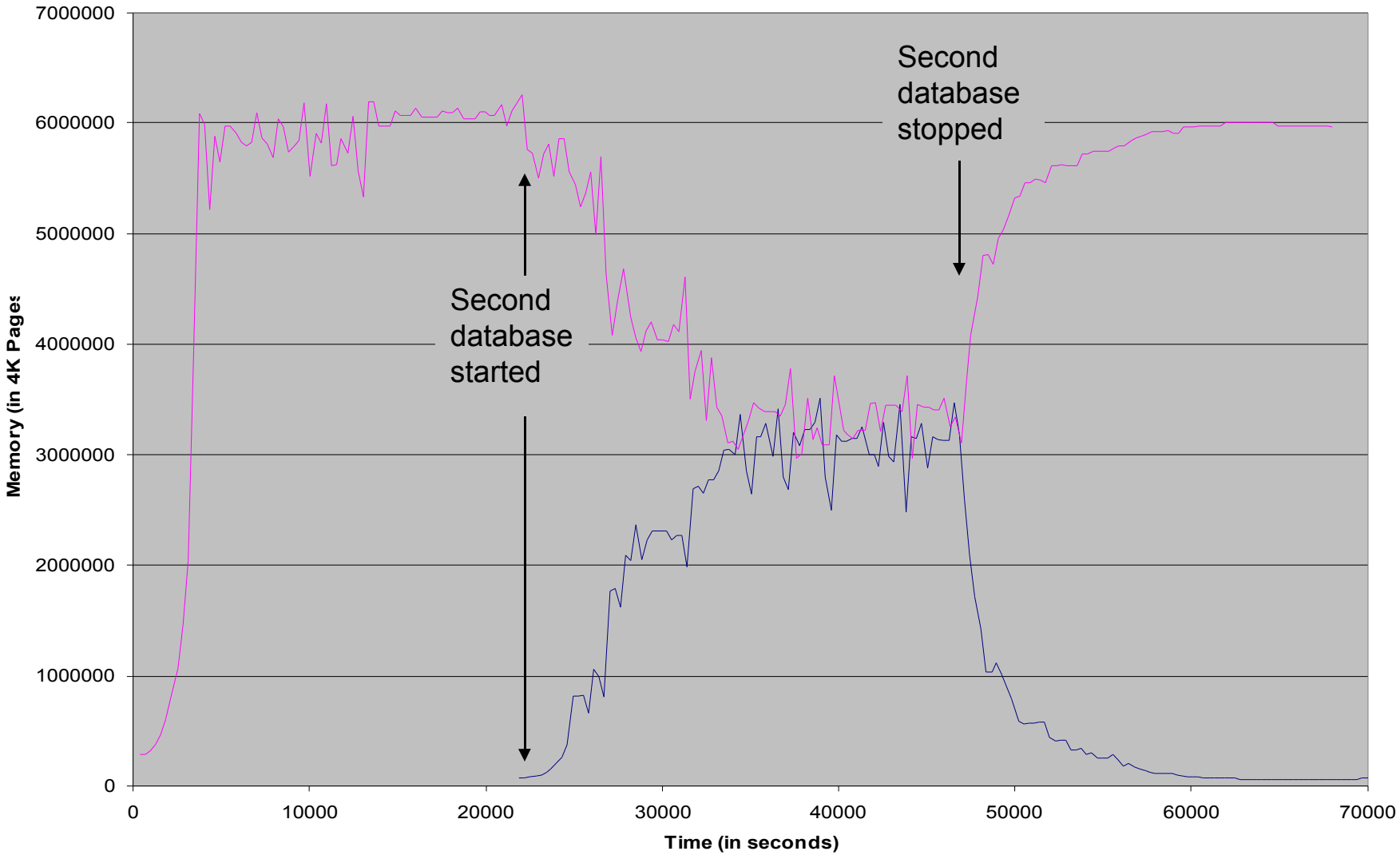
STMM in action – Oops, dropped my indexes



STMM in action – Two database on the same box

- Two databases running the same workload on the same box
  - 4 clients looping through the 21 queries used in TPC-H
  - 15 GB databases
- Running on a machine with 32 GB of RAM
- Demand for memory for each database is equal
- One database is started first and allowed to use up all the memory
- Then, six hours later, the second database is started
  - After both database run together, second database is stopped
- STMM should allow for proper sharing of memory…

STMM in action – Two database on the same box



STMM – When it is most useful

- DBA is inexperienced or is unfamiliar with DB2's memory model
- DBA has workload with unknown memory characteristics
- Multiple database/instances running on the same machine
- System has a lot of free memory
- System has multiple buffer pools of differing page sizes
- Workload is well known but has dramatically differing memory needs from one hour to the next

STMM – When it is less useful

- DBA is very experienced at memory tuning
- Workload's memory requirements are very well understood
- Workload has static memory requirements
  - Run STMM for some time and then freeze the configuration
- DB2 has fixed amount of memory to work with
  - Set DATABASE_MEMORY to the total amount of memory to use
- Only one database running on the machine
- DPF with drastically different memory requirements on each node

Configuration Changes – REFERENCE ONLY

Configuration changes - Activating self tuning memory

- Is on by default for newly created version 9 databases
- New configuration parameter SELF_TUNING_MEM must be set to ON
  - `update db cfg for database <db_name> using self_tuning_mem on`
- Set each parameter that you wish to tune to AUTOMATIC
  - `update db cfg for database <db_name> using locklist automatic`
- In the buffer pool case an alter bufferpool command is necessary
  - `alter bufferpool ibmdefaultbp size automatic`
  - Or for new bufferpools
    - `create bufferpool <bp_name> size automatic`
- Feature can be turned on dynamically

Configuration changes - Activating self tuning memory

- Self tuning memory trades memory between the different consumers
  - Two or more consumers must be tunable for tuning to start
    - A parameter is tunable if it is set to AUTOMATIC
    - The database_memory parameter is also tunable when set to a value
- To query whether or not the system is tuning
  - Connect to the database
  - Get db cfg show detail
  - Check the value for SELF_TUNING_MEM
    - If set to ON (ACTIVE) then system is being tuned
    - If set to OFF, or ON (INACTIVE) then system is not being tuned

Configuration changes - Deactivating Self Tuning Memory

- Two ways to turn STMM off
  - Set SELF_TUNING_MEM parameter to OFF
    - `update db cfg for database <db_name> using self_tuning_mem off`
    - Allows for turning STMM OFF and then ON while maintaining the same set of self tuned parameters
  - Turning an individual parameters to manual (or a value) will stop tuning for that parameter
    - `update db cfg for database <db_name> using locklist manual`
    - `update db cfg for database <db_name> using locklist 1000`
    - `alter bufferpool <bp_name> size manual`
    - `alter bufferpool <bp_name> size 1000`
- All updates can be done dynamically
  - Update may be deferred (sqlcode 1363) in some cases
    - Setting to value causes increase or decrease

Configuration changes – Other changes…

- Added the option to put all sorts in database shared memory
  - To have all sorts run in shared memory
    - Set DBM parameter sheapthres to 0
    - Set DB parameter sheapthres_shr to the desired value
    - This is the default for newly created version 9 instances/databases
    - All sorts must be in shared memory for self tuning of sort memory to work
- All STMM affected memory parameters are now fully dynamic
  - Locklist
    - Was only dynamically increasable in version 8
    - In version 9 can be increased or decreased dynamically
  - Sheapthres_shr
    - Was not dynamic in version 8
    - In version 9 can be increased or decreased dynamically

Summary

- Configuration advisor automatically configures databases as they are created
- STMM goes one step further
  - Advanced automated memory tuning for DB2 version 9
- STMM provides:
  - Online database tuning without any DBA intervention
  - Tuning of total database memory usage
  - Tuning of the four largest consumers of database memory
    - Is activated through database configuration (and alter bufferpool command)
  - A near optimal solution in only an hour
  - A DPF memory tuning solution when nodes are similar

Links

- http://www-128.ibm.com/developerworks/db2/library/techarticle/dm-0605shastry/index.html
  - More information on the configuration advisor
- http://publib.boulder.ibm.com/infocenter/db2luw/v8/index.jsp?topic=/com.ibm.db2.udb.doc/admin/t
  - Configuration advisor command information
- http://www-128.ibm.com/developerworks/db2/library/techarticle/dm-0606ahuja/
  - Overview of Self Tuning Memory
- http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp?topic=/com.ibm.db2.udb.admin.doc/c
  - Self Tuning Memory product documentation

Adam Storm

IBM Canada

ajstorm@ca.ibm.com