
Rule WLM355: Device DISC time was a major cause of DASD I/O delay

Finding: CPEXpert has determined that device DISC time was a major cause of delay in DASD response for the I/O operations of the service class.

This finding applies only to MVS versions prior to OS/390 Release 3, and to MVS versions with OS/390 Release 3 if I/O Priority Management has **not** been specified.

Impact: This finding may have a MEDIUM IMPACT or HIGH IMPACT on the performance of the service class.

Logic flow: The following rules cause this rule to be invoked:

- Rule WLM350: I/O activity may have caused significant delays
- Rule WLM351: I/O activity may have caused significant delays
- Rule WLM352: I/O activity may have caused significant delays for server service class
- Rule WLM353: I/O activity may have caused significant delays for server service class

Discussion: DISC means that there is some delay that is often (but not always) associated with a mechanical movement during which the device disconnects from the control unit (or the control unit disconnects from the channel).

With legacy systems (e.g., 3380 drives attached to 3990-2 control units), the DISC time of most concern was associated with seek (arm movement) and rotational position sensing (time waiting for the disk platter to rotate to the location where desired data resides). Considerable performance improvement efforts were directed at reducing the seek activity and reducing the rotational position sensing (RPS)¹ delays for the legacy systems. These two mechanical delays still exist for most modern *redundant array of independent disks* (RAID)² systems, but their impact can not be directly reduced with normal methods.

With modern disks, data is cached into device cache buffers that contain

¹RPS delays were caused by a path not being available when the required data came under a device read head. Since a path was not available, the device could not reconnect to the channel or control unit. Consequently, data could not be read and transmitted, and another rotation of the platter was experienced until the data again came under the device read head. Multiple rotations might be required, depending on the busy level of the path.

²An array is an ordered collection of physical devices (disk drive modules) that are used to define logical volumes or devices.

data read from a track on the disk platter. Using device cache buffers containing the track data eliminated the multiple-RPS delays caused by a path busy when the device tried to reconnect. Required data is read into the device cache buffer during a single rotation and stored until a path is available to transfer the data.

In addition to the cache buffer design, modern control units such as the 3990-6 or 2105 have very large cache memory installed. With cache in the control units, data to be read can be transferred in a variety of ways, depending on where the data resides.

For a read operation, desired data often is found in the control unit cache. If the required data is in cache, the data can be transferred between the control unit cache and the channel, and this transfer is done at channel speed. If the required data is not in cache, the data can be transferred between the device and channel (and concurrently placed into the control unit cache for subsequent access).

For write operations, data can be placed into Non-volatile Storage (NVS) as a part of the control unit. Write operations normally end as the data to be written is placed in the NVS; and the storage processor writes the data to the device asynchronous with other activity (as a “back end” staging operation). See subsequent discussion for more detail about read and write operations.

The storage director can simultaneously transfer data between the channel and device and manage the data transfer of different tracks between the cache and channel, and the cache and the device. With large amounts of cache memory, a high percent of data accesses normally will be resolved from the fast cache memory and the relatively slow device will not cause significant delays.

As a result of the above improvements, DISC time for modern systems is a result of *cache read miss* for read operations, back-end staging delay for write operations, peer-to-peer remote copy (PPRC) operations, and other miscellaneous reasons³. DISC time often can be very small with adequate cache. For example, there would be zero disconnect time for a cache read hit (the record was found in the cache). However, DISC time can be large and can cause serious delay to I/O operations.

C Read operations. With devices attached to cached controllers, a read operation finds required data in the cache (a “read hit”) or does not find required data in the cache (a “read miss”).

³Artis has described a “sibling PEND” condition that results from collisions within the physical disk subsystem of RAID devices. See “Sibling PEND: Like a Wheel within a Wheel,” www.cmg.org/cmgap/int449.pdf. While this condition is titled “sibling PEND,” the time properly belongs in DISC time, rather than PEND time .

If a read operation *finds data in the cache*, acquiring the data involves only the transfer of data from cache. In this case, the data transfer takes place at channel speeds. Channel speeds can vary, depending on the channel type, from about 4.5 MB per second (parallel channels), up to 18 MB per second (ESCON channels), to over 100MB per second (FICON channels).

If a read operation *does not find data in the cache*, the data must be read from the physical disk device⁴. With the IBM-3390-3 controller and the initial release of the IBM-3390-6 controller, an entire track would be read into cache for a direct read. This algorithm was changed to read only the record required in a direct read; the change eliminated unnecessary activity by the controller⁵.

The implications of reading the data from the physical disk device differ depending on the type of channel:

- C With parallel channels and ESCON channels, the control unit *disconnects* from the channel while the data is being read. After the data has been read, the control unit attempts to reconnect to the channel. The channel must be available when the control unit attempts to reconnect, or additional overhead results. Consequently, channel busy is an important metric with parallel channels and ESCON channels. IBM suggests that these channel types should not have a consistent busy greater than 50% to avoid unacceptable overhead.
- C With FICON Native channels and control units, the control unit does *not* disconnect from the channel while the data is being read, as disconnect and reconnect protocols have been eliminated with FICON. When the frames of data read from DASD are ready to be presented to the channel, the frames simply queue along with any other frames of data (from other I/O operations transferring data) and the data frames are interleaved at channel transfer rates.

While the device delays caused by cache miss operations do not result in disconnect/reconnect protocol exchanges between channel

⁴The data is read into cache, unless *Inhibit Cache Loading* had been specified. With *Inhibit Cache Loading*, the cache is searched to see whether the record is in cache (from a previous I/O operation). If the requested track is not in cache, the channel program operates directly with DASD. Applications can use *Inhibit Cache Loading* when it is known that records read would not likely be accessed again.

⁵The initial design did not consider that the device and the controller would be “busy” during the transfer of the track from the device to the controller. The belief was that the transfer of the track would be “off line” and not adversely impact performance. However, while the track was being transferred to the controller, the device and controller were busy and other I/O operations were constrained. With very active systems, this constraint could seriously degrade performance. By moving to record-level transfer for direct I/O, this constraint was removed.

and control unit, the actual device delay time exists nonetheless⁶. These device delays are timed by a FICON control unit, and the time is reported to RMF as DISC time. Thus, the delay time is available with FICON channels and control units and titled "DISC" time, even though the actual disconnect and reconnect activities do not occur.

In order to improve the probability of a read hit, the controller can *prestage* data into its cache. Prestaging means that data is read into the controller's cache ahead of its actually being required for use by an application. The amount of data that is prestaged depends on (1) whether the data is being accessed in a direct (random) mode or in a sequential mode and (2) the controller model and the enhancements made to the controller.

For *direct mode*, the 3990 Model 6 (with record cache) stages only the records requested into cache, eliminating the balance of the track staging as was implemented on initial versions of 3990-6 and on the 3990-3. As examples of prestaging for *sequential mode*, the 3990-3 reads up to two tracks into the cache⁷ before they are required, while the ESS 2105 sequential staging reads up to two cylinders ahead.

Applications can indicate (using Define Extent) that data is to be processed in a sequential mode. With the 3990-6, IBM included a *sequential detection algorithm* that automatically detects whether data is being read sequentially, even if the user did not indicate that reads were in sequential mode. If the algorithm detects sequential access, data is prestaged automatically. For example, with the ESS 2105, when the algorithm detects that 6 or more tracks have been read in succession, the algorithm triggers the sequential staging process.

During prestaging operations, the control unit regularly checks to see whether other I/O requests are waiting to be processed. If any are waiting, the control unit interrupts the prestage operation, processes the queued requests, and continues with the prestage.

C Write operations. With devices attached to cached controllers, a number of options are available to help improve performance for particular applications. Use of these options vary depending on the data access characteristics of records being written, performance goals

⁶This might seem a moot point; if the device delay exists, why should it matter whether the time is a result of disconnect between the channel and control unit or simply device delay time? The difference is that the exchange of disconnect and reconnect protocol traffic between the channel and control unit is eliminated with FICON. This exchange of protocol can add considerable overhead, and it is this overhead that is eliminated with FICON. The FICON controller times the device delays that occur simply for RMF reporting.

⁷With the Sequential Staging Performance Enhancement, the 3990-3 can prestage up to a full cylinder (15 tracks) into the cache.

associated with the applications, amount of cache and NVS that is available, etc. Some of the common options are Bypass Cache Mode, Normal Caching Mode, Cache Fast Write Mode, and DASD Fast Write Mode.

- C **Bypass Cache Mode.** The Bypass Cache Mode causes the data in the cache to be bypassed. The I/O write request is sent directly to DASD, but a search of the cache is performed because the track in the cache could have been modified by a previous I/O operation. If the track is in the cache, the corresponding cache slot is marked invalid to prevent a read hit by a subsequent I/O operation. If the cache slot had been modified by a previous cache fast write hit or a DASD fast write hit, the track is destaged and the slot is marked invalid.

The performance of an I/O operation with Bypass Cache Mode is almost the same as if the write were performed via a noncache storage control. The Bypass Cache operation is slightly longer than a write via a noncache controller, because a directory search of the controller's cache is required to determine whether the track is in cache.

The controller presents channel end and device end only after the transfer operation is complete. Since the I/O write operation deals directly with the device, disconnect time can be significant.

The Bypass Cache Mode might be used even though the control unit has considerable cache in situations where low priority files are "cache unfriendly" (meaning that they have a poor locality of reference), with very large files with high write activity when the files might "flood" the cache and cause a low read hit or write hit for other (perhaps more important) file accesses.

- C **Normal Caching Mode.** With Normal Caching Mode, all write I/O commands operate directly with the device. In cache operations without cache fast write or DASD fast write, a write operation follows these general rules⁸:
 - C A format write operates directly with DASD. If the track is in cache, it is invalidated. This ensures that a subsequent read will result in a read miss.
 - C If the track modified by an update write operation is in cache, the cache and DASD are updated concurrently (a write hit). This

⁸Source: IBM's 3990 Planning, Installation, and Storage Administration Guide

ensures that the data in cache is current.

- C If the track modified by an update write operation is not in the cache, the operation is a write miss. Only the data on DASD is updated.
- C No *new* tracks are transferred from DASD to cache as the result of a write operation.
- C A track in cache is never made "most recently used" by a write hit in basic caching operations.

If a write hit occurs (the write request updates a record that is already in cache), the controller transfers the data to both cache and DASD. This ensures that the data in cache is current, and is available for a subsequent read operation.

If a write Miss occurs (the write request updates a record that is not in cache) data is transferred from the channel to DASD, and is not placed into cache.

The primary objective of a basic cache write operation is to emulate a DASD write, to ensure that the DASD copy of the data is always valid, and to ensure that any copy of the data retained in cache is valid.

The controller presents channel end and device end only after the transfer operation is complete. Since the I/O write operation deals directly with the device, disconnect time can be significant.

- C **Cache Fast Write Mode.** The Cache Fast Write Mode causes data to be placed into cache immediately, and there is no interaction with the device nor with NVS. Cache fast write is useful in situations where the data that may not be required after the completion of the current job or in situations where the data could be easily reconstructed if necessary (data could be reconstructed if the cache failed).

If the record to be written is already in the cache, this is considered a "write hit" and the entire operation is performed with the cache. With either a write miss (data is not in the cache) or a write hit, no DASD access is required. However, write hits cause the record to be made "most recently used." *When cache space is needed, the controller destages the least recently used data to DASD.*

In most cases when Cache Fast Write Mode is used, the data is only

temporary, and can be discarded when no longer required. For example, sorts would not require permanent data for their sort work files.

If the cache is reinitialized, all cache fast write data is lost and the cache fast write identifier is incremented. Subsequent I/O operations with the old cache fast write identifier are reported to the requesting program as a permanent I/O error.

The controller presents channel end and device end after the data has been placed in the cache. Since the I/O write operation deals only with the cache, disconnect time is eliminated for normal I/O operations⁹.

- C **DASD Fast Write Mode.** In DASD Fast Write Mode, the data is stored *simultaneously* in cache storage and in nonvolatile storage. Since data is stored in NVS, access to a physical DASD is not required for write hits to ensure data integrity. The copy of the data in nonvolatile storage allows storage processor to continue without waiting for the data to be written to DASD. The data remains in cache storage and in nonvolatile storage until the storage control destages the data to DASD. Since completion of the write is indicated when the cache data transfer is complete, DASD Fast Write provides a significant performance enhancement over basic write operations; the DASD fast write hit is as fast as a read hit.

In MVS, activation and deactivation of DASD fast write is provided by a system utilities command with extended function programming support. DASD fast write remains active until explicitly deactivated by another command. DASD fast write is activated at a volume level and is the default for all write operations directed at that volume. DASD fast write can be inhibited at the channel program level.

If DASD fast write is deactivated, the 3990 destages the DASD fast write data to DASD. The 3990 also destages the DASD fast write data to DASD if (1) NVS is deactivated, (2) subsystem caching or device caching is deactivated, and (3) more space is made available in the cache or NVS. These destaging operations are between the cache or NVS and DASD. Consequently, the activity does not result in disconnect time for normal I/O operations (that is, they would not be reflected as DISC time by RMF).

CPExpert computes the average per-second DISC delay time as described in Rule WLM350. Rule WLM355 is produced if the average

⁹There can be considerable device activity if the data is destaged because cache space was needed or after cache fast write is turned off. This destage activity could adversely impact other I/O operations requiring access to the device.

DISC time accounted for a significant percent of the response time of transactions in the service class missing its performance goal.

The following example illustrates the output from Rule WLM355:

RULE WLM355: DEVICE DISCONNECT TIME WAS A MAJOR CAUSE OF DASD DELAYS

A major part of the potential I/O delay to the TSO Service Class could be attributed to device disconnect (DISC) time. Disconnect time normally is caused by missed read hits (the data required was not in the controller's cache), potentially back-end staging delay for cache write operations, peer-to-peer remote copy (PPRC) operations, and other miscellaneous reasons.