Module placement effect on application performance

- Modules begin to run most quickly when all conditions are true:
  - They are already loaded in virtual storage
  - The virtual storage they are loaded into is accessible to the programs that call them
  - The copy that is loaded is usable
  - The virtual storage is backed by central storage

- Modules that are accessible and usable (and have already been loaded into virtual storage but not backed by central storage) must be returned to central storage from page data sets on DASD or SCM.

- Modules in the private area and those in LPA (other than in FLPAs) can be in virtual storage without being backed by central storage.

- When using virtual storage access, these modules will begin to run much faster when they are in central storage.

- Modules that are anywhere in LPA are loaded into virtual storage, and modules placed in FLPAs are also always in central storage.

- The results of paging out and paging stealing are usually the same; unless stolen pages are reclaimed before being used for something else, they will not be in central storage when the module they contain is called.

- LPA modules must be referenced very often to prevent their pages from being stolen.

- When a page in LPA (other than in FLPAs) is not continually referenced by multiple address spaces, it tends to be cold.
  - One reason these pages might be cold is that address spaces often get swapped out (without the LPA pages to which they refer), and a swapped-out address space cannot refer to a page in LPA.

- When all the pages containing an LPA module (or its first page) are not in central storage when the module is called, the module will begin to run only after its first page has been brought into central storage.

- Modules can also be loaded into CSA by authorized programs.

- When several address spaces load the same module, the performance considerations are similar to those for modules placed in LPA. (However, unlike LPA pages, CSA pages must be paged out when the system relinquishes them.)

- When a usable and accessible copy of a module cannot be found in virtual storage, either the request must be deferred or the module must be loaded.

- When a module is loaded, it can be loaded from a VLF data space used by LLA, or from load libraries or PDSEs residing on DASD.

- Modules in LPA must always be loaded the first time they are used by an address space.

- How long this takes depends on:
  - Whether the directory for the library in which the module resides is cached
  - Whether the module itself is cached in storage
  - The amount of time the DASD subsystem on which the module resides at the time the I/Os load the module
  - The address space cache directories entry for all the modules in the data space in the linklist concatenation (defined in PROGxx and LNKLSTxx) by default.

- The time it takes to bring a module from DASD to LPA (or from CSA) depends on how busy all of the resources needed to complete the I/Os are.

- The time it takes to read a module from a load library (not a PDSE) on DASD is minimized when the modules are written to a data set by the binder/linkage-editor, or an ECB/CCOPY/MOD edit operation when the data set has a block size equal to or greater than the size of the largest load module, or if the library contains data sets larger than 32 kilobytes, set to the maximum supported block size of 32760 bytes.

Access time for modules

- From a performance standpoint, modules not already loaded in an address space will usually be available to a program in the least time when found at the beginning of the following list, and will take more time to be available when found later in the list:
  1. LPA
  2. Link list concatenation (all directory entries and some modules cached automatically)
  3. TASKLIB/STEPLIB/JOBLIB (with LLA caching of the library)
  4. TASKLIB/STEPLIB/JOBLIB (without LLA caching of the library)
  5. TASKLIB/STEPLIB/JOBLIB

- For best application performance, you should place as many frequently-used modules as high on this list as you can.

- However, if CPU time and system-wide factors must be considered when you decide how many load modules to place in LPA:

  - A program that is frequently run from the source code level is usually loaded into LPA.

- Performance: When a module is loaded into central storage, the module placement effect on application performance depends on when it is loaded into central storage.

- Module placement effect on system performance

  - Whether the placement of a module affects system performance depends on how many address spaces use the module and how much of virtual storage is used.

  - Placement of infrequently-used modules that are used by few address spaces have little effect on system-wide performance or on the performance of address spaces that do not use the modules.

  - Placement of frequently-used modules placed by a program, particularly those used by a number of address spaces at the same time, can substantially affect system performance.

Module placement considerations

- More virtual storage can be used when a large number of address spaces each load their own copy of a frequently-used module, because multiple copies are more likely to exist in central storage at any time.

- One reason these pages might be cold is that address spaces often get swapped out (without the LPA pages to which they refer), and a swapped-out address space cannot refer to a page in LPA.

- When several address spaces load the same module, the performance considerations are similar to those for modules placed in LPA. (However, unlike LPA pages, CSA pages must be paged out when the system relinquishes them.)

- When a usable and accessible copy of a module cannot be found in virtual storage, either the request must be deferred or the module must be loaded.

- When a module is loaded, it can be loaded from a VLF data space used by LLA, or from load libraries or PDSEs residing on DASD.

- Modules in LPA must always be loaded the first time they are used by an address space.

- How long this takes depends on:
  - Whether the directory for the library in which the module resides is cached
  - Whether the module itself is cached in storage
  - The amount of time the DASD subsystem on which the module resides at the time the I/Os load the module
  - The address space cache directories entry for all the modules in the data space in the linklist concatenation (defined in PROGxx and LNKLSTxx) by default.

- The time it takes to bring a module from DASD to LPA (or from CSA) depends on how busy all of the resources needed to complete the I/Os are.

- The time it takes to read a module from a load library (not a PDSE) on DASD is minimized when the modules are written to a data set by the binder/linkage-editor, or an ECB/CCOPY/MOD edit operation when the data set has a block size equal to or greater than the size of the largest load module, or if the library contains data sets larger than 32 kilobytes, set to the maximum supported block size of 32760 bytes.