MODULE: 4

CONTENTS:

- **MemoryManagement:**

  - Memorymanagementstrategies:Background;
  - Swapping;
  - Contiguousmemoryallocation;
  - Paging;
  - Structureofpagetable;
  - Segmentation.

- **VirtualMemoryManagement**:

  - Background;
  - Demandpaging;
  - Copy-on-write;
  - Pagereplacement;
  - Allocationofframes;
  - Thrashing

MEMORYMANAGEMENT

MainMemoryManagementStrategies

- Everyprogramtobeexecutedhastobeexecutedmustbeinmemory.Theinstructionmustbefetchedfrommemorybeforeitisexecuted.
- Inmulti-taskingOSmemorymanagementiscomplex, because asprocesses areswappedinandout of theCPU, theircodeanddata mustbeswappedinandoutof memory.

BasicHardware

- Main memory, cache and CPU registers in the processors are the only storagespacesthatCPUcanaccessdirectly.
- The program and data must be bought into the memory from the disk, for the process torun. Each process has a separate memory space and must access only this range of legaladdresses. Protection of memory is required to ensure correct operation. This preventionisprovidedbyhardwareimplementation.
- Two registers are used - a base register and a limit register. The base register holds thesmallestlegalphysicalmemoryaddress;thelimitregisterspecifiesthesizeoftherange.
- For example, The base register holds the smallest legal physical memory address; thelimit register specifies the size of the range. For example, if the base register holds300040 and limit register is 120900, then the program can legally access all addressesfrom300040through420940(inclusive).

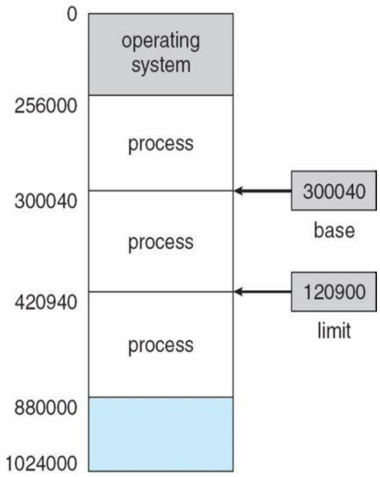


Figure:A baseandalimit-register definealogical-addressspace

- The base and limit registers can be loaded only by the operating system, which uses aspecial privilegedinstruction. Since privileged instructions can beexecutedonly inkernelmodeonlytheoperatingsystemcanloadthebaseandlimitregisters.

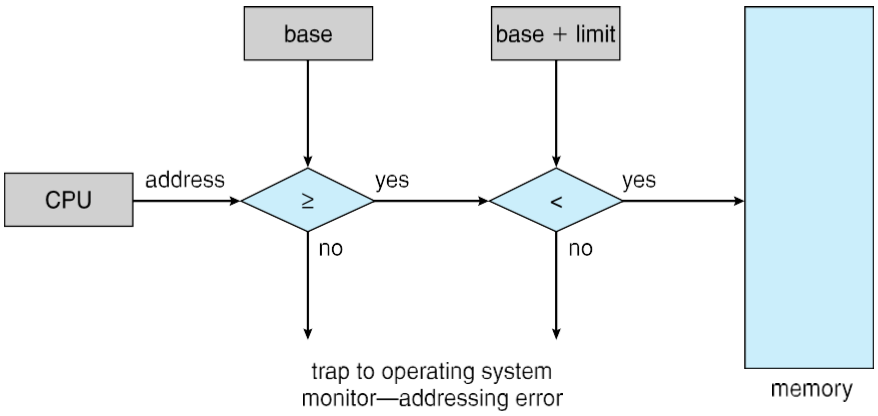


Figure:Hardwareaddressprotectionwith baseandlimit-registers

AddressBinding

- User programs typically refer to memory addresses with symbolic names. These symbolicnamesmustbemappedorboundtophysicalmemoryaddresses.
- Addressbindingof instructionsto memory-addressescanhappenat3 differentstages.

1. **Compile Time**- If it is known at compile time where a program will reside in physicalmemory,thenabsolutecodecanbegeneratedbythecompiler,containingactualphysical addresses. However, if the load address changes at some later time, then theprogramwillhavetoberecompiled.
2. **Load Time**- If the location at which a program will be loaded is not known at compiletime,thenthecompilermust generaterelocatablecode,whichreferencesaddressesrelative to the start of the program. If that starting address changes, then the programmustbereloadedbutnotrecompiled.
3. **Execution Time**- If a program can be moved around in memory during the course of itsexecution,then bindingmustbe delayed untilexecutiontime.

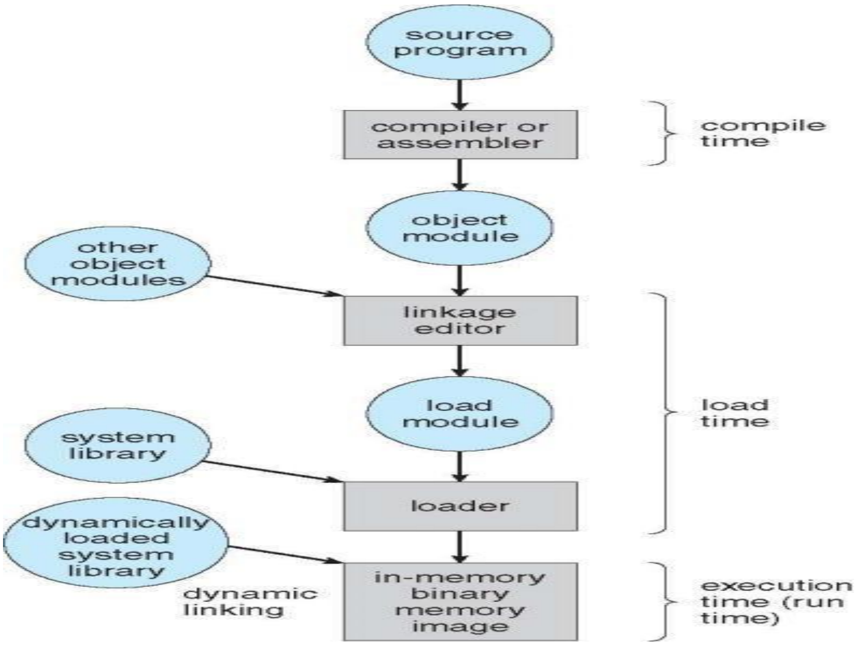


Figure:Multistep processingof auserprogram

LogicalVersusPhysicalAddressSpace

- TheaddressgeneratedbytheCPUisalogicaladdress,whereasthememoryaddresswhereprogramsareactuallystoredisaphysicaladdress.
- Thesetofalllogicaladdressesusedbyaprogramcomposesthelogicaladdressspace,andthesetofallcorrespondingphysicaladdressescomposesthe physicaladdress space.
- Theruntimemappingoflogicaltophysicaladdressesishandledbythememory-managementunit (MMU).
    - Oneofthesimplestisa modificationofthebase-registerscheme.
    - Thebaseregisteristermedarelocationregister
    - Thevalueintherelocation-registerisaddedtoeveryaddressgeneratedbyauser-processatthetimeitissent tomemory.
    - Theuser-programdealswithlogical-addresses;itneverseestherealphysical-addresses.

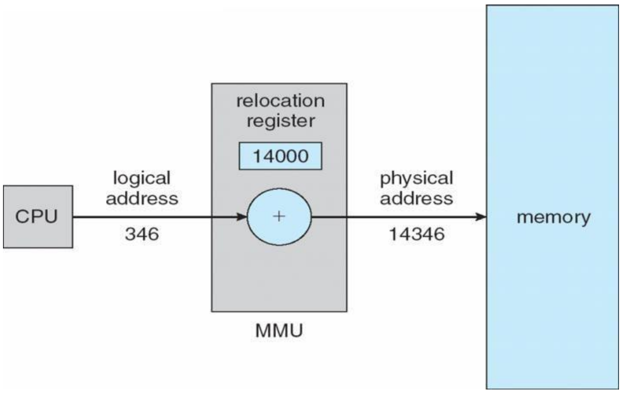


Figure:Dynamicrelocationusing a relocation-register

DynamicLoading

- Thiscanbe usedto obtainbetter memory-spaceutilization.
- Aroutineisnotloadeduntilitiscalled.

Thisworksasfollows:

1. Initially,all routinesarekeptondiskina relocatable-loadformat.
2. Firstly,themain-programisloadedinto memoryandisexecuted.
3. When a main-program calls the routine, the main-program first checks to see whether theroutinehasbeenloaded.
4. Ifroutinehasbeennotyetloaded,theloaderiscalledtoloaddesiredroutineintomemory.
5. Finally,controlispassedtothenewlyloaded-routine.

Advantages:

1. Anunused routineisneverloaded.
2. Usefulwhenlargeamountsof codeareneededtohandleinfrequentlyoccurringcases.
3. Although the total program-size may be large, the portion that is used (and hence loaded)maybemuchsmaller.
4. DoesnotrequirespecialsupportfromtheOS.

DynamicLinkingandSharedLibraries

- With **static linking**library modules get fully included in executable modules, wastingboth disk space and main memory usage, because every program that included a certainroutine from the library would have to have their own copy of that routine linked intotheirexecutablecode.
- With **dynamiclinking**, however, only a stub is linked into theexecutablemodule,containingreferences to theactuallibrarymodulelinkedin atrun time.
  - The stub is a small piece of code used to locate the appropriate memory-residentlibrary-routine.
  - This method saves disk space, because the library routines do not need to be fullyincludedintheexecutablemodules,onlythestubs.
  - An added benefit of dynamically linked libraries (DLLs, also known as sharedlibrariesor sharedobjectsonUNIXsystems)involveseasyupgrades andupdates.

Sharedlibraries

- Alibrarymaybereplacedbyanewversion,andallprogramsthatreferencethelibrarywillautomaticallyusethenewone.
- Versioninfo.isincludedinbothprogram&librarysothatprogramswon'taccidentallyexecuteincompatibleversions.

Swapping

- Aprocess mustbe loadedintomemoryin ordertoexecute.
- If there is not enough memory available to keep all running processes in memory at thesame time, then some processes that are not currently using the CPU may have theirmemoryswappedouttoafastlocaldiskcalledthe***backingstore***.
- *Swapping is the process of moving a process from memory to backing store and movinganotherprocessfrombackingstoretomemory*.Swappingisaveryslowprocesscomparedtootheroperations.
- A variant ofswapping policy isused forpriority-based scheduling algorithms.Ifahigher-priority process arrives and wants service, the memory manager can swap out thelower-priority process and then load and execute the higher-priority process. When thehigher-priority process finishes, the lower-priority process can be swapped back in andcontinued. This variantofswappingis***calledrollout,rollin.***

Swappingdepends uponaddress-binding:

- Ifbindingisdoneatload-time,thenprocesscannotbeeasilymovedtoadifferentlocation.
- Ifbindingisdoneatexecution-time,thenaprocesscanbeswappedintoadifferentmemory-space,becausethephysical-addressesarecomputedduring execution-time.

Majorpartofswap-timeistransfer-time;i.e.totaltransfer-timeisdirectlyproportionaltotheamountofmemoryswapped.

Disadvantages:

1. Context-switchtimeisfairlyhigh.
2. If we want to swap a process, we must be sure that it is completely idle.Twosolutions:
   i) NeverswapaprocesswithpendingI/O.
   ii) ExecuteI/OoperationsonlyintoOSbuffers.

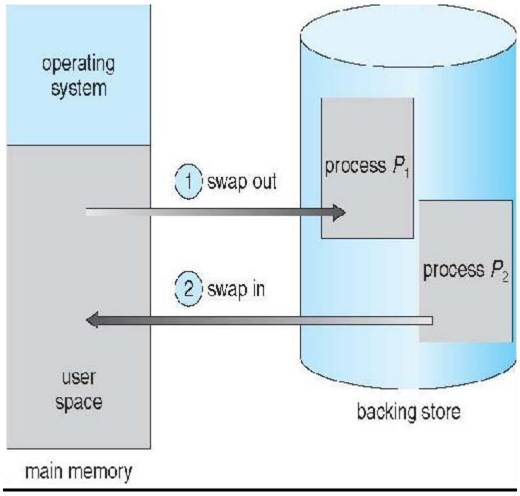


Figure:Swappingoftwoprocessesusingadiskasabackingstore

Example:

Assume that the user processis 10MB in sizeandthe backingstore is astandard harddisk withatransfer rate of40 MB per second.

The actual transfer of the 10-MB process to or from main memory

takes10000KB/40000KBpersecond=1/4second

=250 milliseconds.

Assuming that no head seeks are necessary, and assuming an average latency of 8 milliseconds,the swap time is 258 milliseconds. Since we must both swap out and swap in, the total swaptimeisabout516milliseconds.

ContiguousMemoryAllocation

- The main memory must accommodate both the operating system and the various userprocesses. Therefore we need to allocate the parts of the main memory in the mostefficientwaypossible.
- Memory is usually divided into 2 partitions: One for the resident OS. One for the userprocesses.
- Eachprocessiscontainedinasinglecontiguoussection of memory.

1. MemoryMappingandProtection

- Memory-protectionmeansprotectingOSfromuser-processandprotectinguser-processesfromoneanother.
- Memory-protectionisdoneusing
  - Relocation-register:containsthevalueofthe smallestphysical-address.
  - Limit-register:contains therangeoflogical-addresses.
- Eachlogical-addressmustbeless thanthelimit-register.
- TheMMUmapsthelogical-addressdynamicallybyaddingthevalueintherelocation-register.Thismapped-address issent tomemory
- WhentheCPUschedulerselectsaprocessforexecution,thedispatcherloadstherelocationandlimit-registerswiththecorrectvalues.
- BecauseeveryaddressgeneratedbytheCPUischeckedagainsttheseregisters,wecanprotecttheOSfromtherunning-process.
- Therelocation-registerschemeprovidesaneffectivewaytoallowtheOSsizetochangedynamically.
- TransientOScode:Codethatcomes&goesasneededtosavememory-spaceandoverheadforunnecessaryswapping.

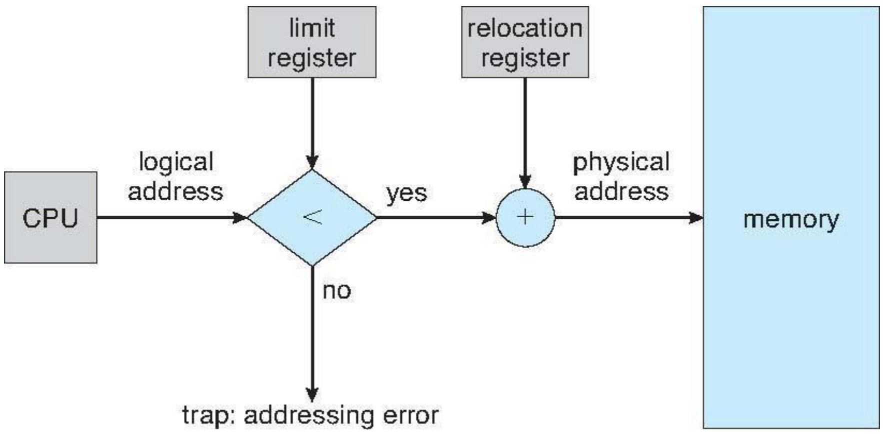


Figure:Hardwaresupportforrelocationandlimit-registers

2. MemoryAllocation

Twotypesof memorypartitioningare:

1. Fixed-sizedpartitioning
2. Variable-sized

partitioning1.Fixed-

sizedPartitioning

- Thememoryisdividedintofixed-sizedpartitions.
- Eachpartitionmaycontainexactlyoneprocess.
- Thedegreeofmultiprogramming isboundbythenumberofpartitions.
- Whenapartitionisfree,aprocessisselectedfromtheinputqueueandloadedintothefreepartition.
- Whentheprocessterminates,thepartition becomesavailableforanotherprocess.

2. Variable-sizedPartitioning

- TheOSkeepsatableindicating whichpartsofmemoryareavailable andwhichpartsareoccupied.
- Aholeisablockofavailablememory.Normally,memorycontainsasetofholesofvarioussizes.
- Initially,allmemoryis availableforuser-processesandconsideredonelargehole.
- Whenaprocess arrives,theprocessisallocated memoryfromalargehole.
- Ifwefindthehole,weallocateonlyasmuchmemoryasisneededandkeeptheremaining memoryavailabletosatisfyfuturerequests.

Threestrategiesusedtoselectafreeholefromthesetofavailableholes:

1. FirstFit:Allocatethefirstholethatisbigenough.Searchingcanstarteitheratthebeginning ofthesetofholesoratthelocationwherethepreviousfirst-fitsearchended.

2. BestFit:Allocatethesmallestholethatisbigenough.Wemustsearchtheentirelist,unlessthelisti sorderedbysize.Thisstrategyproducesthesmallestleftoverhole.

3. WorstFit:Allocatethelargesthole.Again,wemustsearchtheentirelist,unlessitissortedbysize. Thisstrategyproducesthelargestleftoverhole.

First-fitandbestfitarebetterthanworstfitin termsof decreasingtimeandstorageutilization.

3. Fragmentation

Twotypesof memoryfragmentation:

1. Internalfragmentation
2. Externalfragmentation

1. InternalFragmentation

- Thegeneralapproachistobreakthephysical-memoryintofixed-sizedblocksandallocate memoryinunits basedonblock size.
- Theallocated-memorytoaprocessmaybeslightlylargerthantherequested-memory.
- Thedifferencebetweenrequested-memoryandallocated-memoryiscalledinternalfragmentationi.e.Unusedmemorythatisinternaltoapartition.

2. External Fragmentation

- External fragmentation occurs when there is enough total memory-space to satisfy arequestbuttheavailable-spacesarenotcontiguous.(i.e.storageisfragmentedintoalargenumberofsmallholes).
- Boththefirst-fitandbest-fitstrategiesformemory-allocationsufferfromexternalfragmentation.
- Statistical analysis of first-fit reveals that given N allocated blocks, another 0.5 N blockswillbelosttofragmentation.This propertyisknownasthe50-percentrule.

Twosolutionstoexternalfragmentation:

- Compaction:Thegoalistoshufflethememory-contentstoplaceallfreememorytogether in one large hole. Compaction is possible only if relocation is dynamic and doneatexecution-time
- Permit the logical-address space of the processes to be non-contiguous. This allows aprocesstobeallocatedphysical-memorywhereversuchmemoryisavailable.Twotechniquesachievethissolution: 1)Pagingand2)Segmentation.

Paging

- Pagingisamemory-managementscheme.
- Thispermitsthephysical-addressspaceofaprocesstobe non-contiguous.
- Thisalsosolvestheconsiderableproblemoffittingmemory-chunksofvaryingsizesontothebacking-store.
- Traditionally: Support forpaginghasbeen handledbyhardware.
- Recentdesigns:Thehardware&OSarecloselyintegrated.

BasicMethodofPaging

- The basic method for implementing paging involves breaking physical memory intofixed-sized blocks called frames and breaking logical memory into blocks of the samesizecalled pages.
- When a process is to be executed, its pages are loaded into any available memory framesfromthebackingstore.
- The backing store is divided into fixed-sized blocks that are of the same size as thememoryframes.

Thehardwaresupportforpagingisillustrated inFigure1.

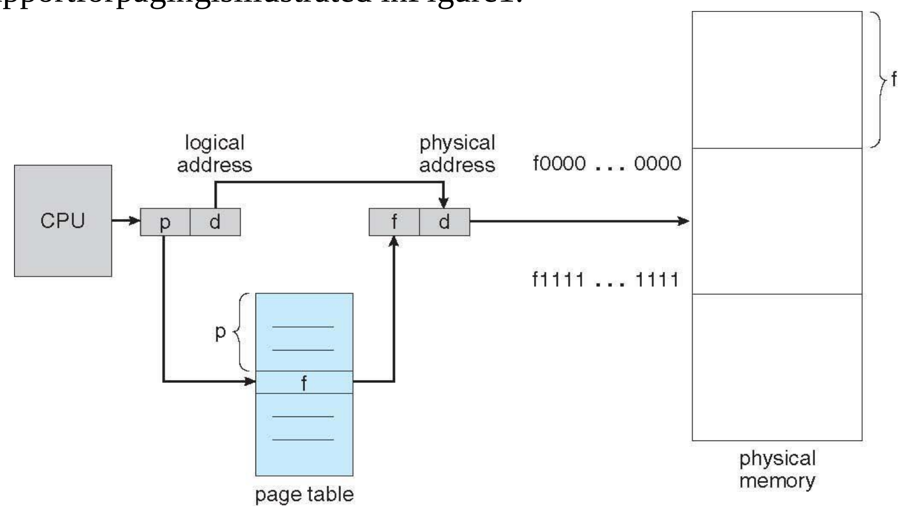


Figure1:Paging hardware

- AddressgeneratedbyCPUisdividedinto2parts(Figure 2):
    1. Page-number (p) is used as an index to the page-table. The page-table contains thebase-addressofeachpageinphysical-memory.
    2. Offset(d)iscombinedwiththebase-addresstodefinethephysical-address.Thisphysical-addressissenttothememory-unit.
- Thepagetable mapsthepagenumbertoaframenumber,toyielda physicaladdress
- Thepagetablemapsthepagenumbertoaframenumber,toyieldaphysicaladdresswhichalsohast woparts:Theframenumberandtheoffsetwithinthatframe.
- Thenumberofbitsintheframenumberdetermineshowmanyframesthesystemcanaddress,and thenumber ofbitsin theoffsetdetermines the sizeofeachframe.

Thepaging model ofmemoryisshownin Figure2.

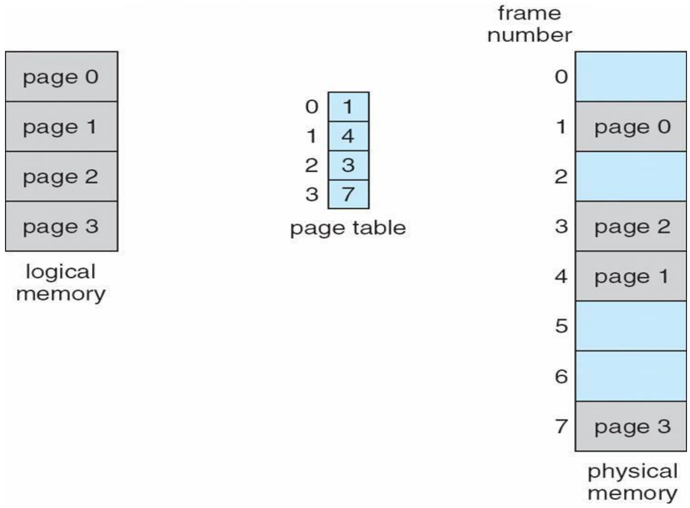


Figure2: Pagingmodeloflogicalandphysical memory.

- Thepagesize(liketheframesize)isdefinedbythehardware.
- The size of a page is typically a power of 2, varying between 512 bytes and 16 MB perpage,dependingonthecomputerarchitecture.
- The selection of a power of 2 as a page size makes the translation of a logical addressintoapagenumberandpageoffset.
- If the size of logical address space is 2^m and a page size is 2^n addressing units (bytes orwords), then the high-order m – n bits of a logical address designate the page number,andthe nlow-orderbitsdesignatethepageoffset.

Thus,thelogicaladdressisasfollows:

pagenumber	pageoffset
p	d
m-n	n

- When a process requests memory (e.g. when its code is loaded in from disk), free framesareallocated fromafree-framelist, andinsertedintothatprocess'spagetable.
- Processes are blocked from accessing anyone else's memory because all of their memoryrequests are mapped through their page table. There is no way for them to generate anaddressthatmaps into anyother process'smemoryspace.
- The operating system must keep track of each individual process's page table, updating itwhenever the process's pages get moved in and out of memory, and applying the correctpage table when processing system calls for a particular process. This all increases theoverheadinvolvedwhenswappingprocessesinandoutoftheCPU.

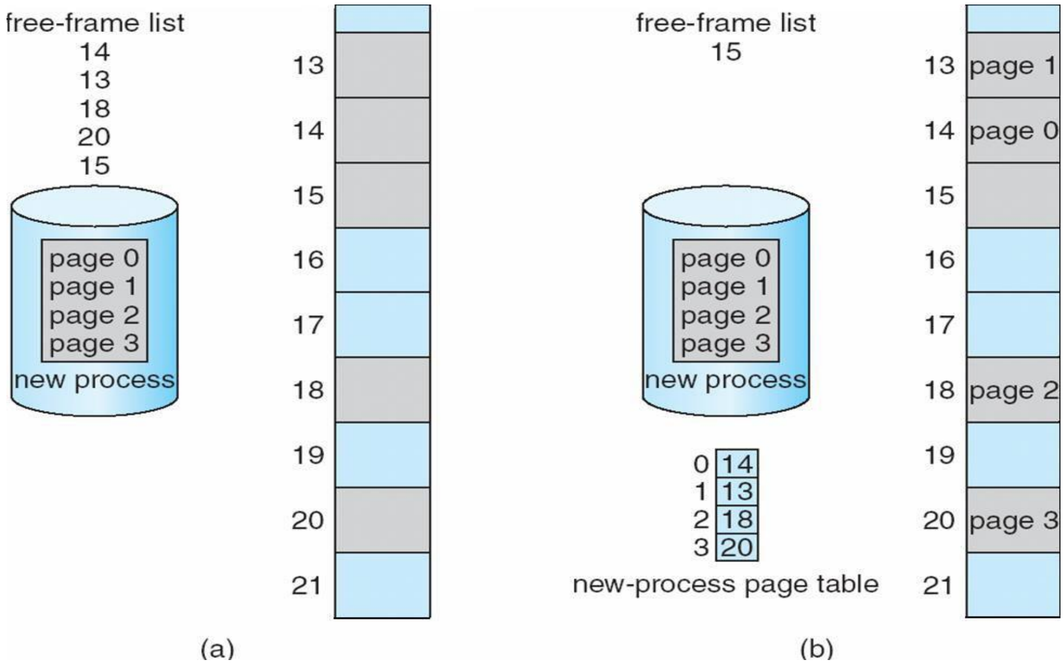


Figure:Freeframes(a)beforeallocationand(b)afterallocation.

HardwareSupport

TranslationLookasideBuffer

- Aspecial,small,fastlookuphardwarecache,calledatranslationlook-asidebuffer(TLB).
- EachentryintheTLBconsistsoftwoparts:akey(ortag)andavalue.
- When the associative memory is presented with an item, the item is compared with allkeys simultaneously. If the item is found, the corresponding value field is returned. Thesearch is fast; the hardware, however, is expensive. Typically, the number of entries in aTLBissmall,oftennumberingbetween64and1,024.
- TheTLBcontains onlyafewofthepage-tableentries.

Working:

- Whenalogical-addressisgeneratedbytheCPU,itspage-numberispresentedtotheTLB.
- If the page-number is found (TLB hit), its frame-number is immediately available andusedtoaccessmemory
- If page-number is not in TLB (TLB miss), a memory-reference to page table must bemade.Theobtainedframe-numbercanbeusedtoaccessmemory(Figure1)

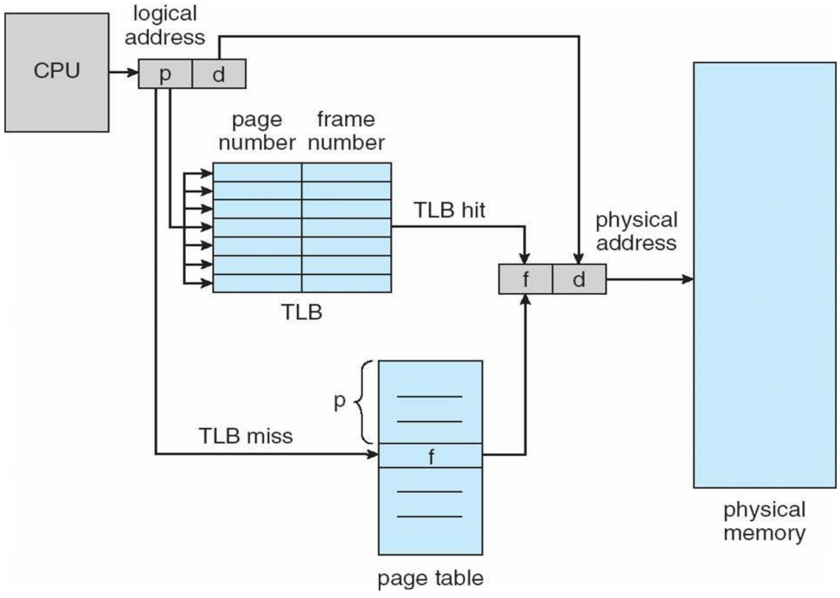


Figure1:Paging hardwarewithTLB

- Inaddition,weaddthepage-numberandframe-numbertotheTLB,sothattheywillbefoundquicklyonthenextreference.
- If theTLBisalreadyfullofentries,theOSmustselectoneforreplacement.
- Percentageoftimesthataparticularpage-numberisfoundinthe TLB iscalledhitratio.

Advantage: Search operation is fast.Disadvantage:Hardwareisexpensive.

- SomeTLBshavewireddownentriesthatcan'tberemoved.
- SomeTLBsstoreASID(address-spaceidentifier)ineachentryoftheTLBthatuniquelyidentifyeachprocessandprovideaddressspaceprotectionforthatprocess.

Protection

- Memory-protection isachievedbyprotection-bitsforeachframe.
- Theprotection-bitsarekeptinthe page-table.
- Oneprotection-bitcandefineapagetoberead-writeorread-only.
- Everyreferencetomemorygoesthroughthepage-tabletofindthecorrectframe-number.
- Firstly,thephysical-addressiscomputed.Atthesametime,theprotection-bitischeckedtoverifythatnowritesarebeingmadetoaread-onlypage.
- Anattempttowritetoaread-onlypagecausesahardware-traptotheOS(ormemoryprotectionviolation).

ValidInvalid Bit

- Thisbitisattachedtoeachentryin thepage-table.
- Validbit:“valid”indicatesthattheassociatedpageisintheprocess’logicaladdressspace,andisthusalegalpage
- Invalidbit: “invalid”indicatesthatthepageis notinthe process’logicaladdressspace

Illegaladdressesaretrappedbyuseofvalid-invalidbit.
TheOSsetsthisbitforeachpagetoallowor disallowaccesstothepage.

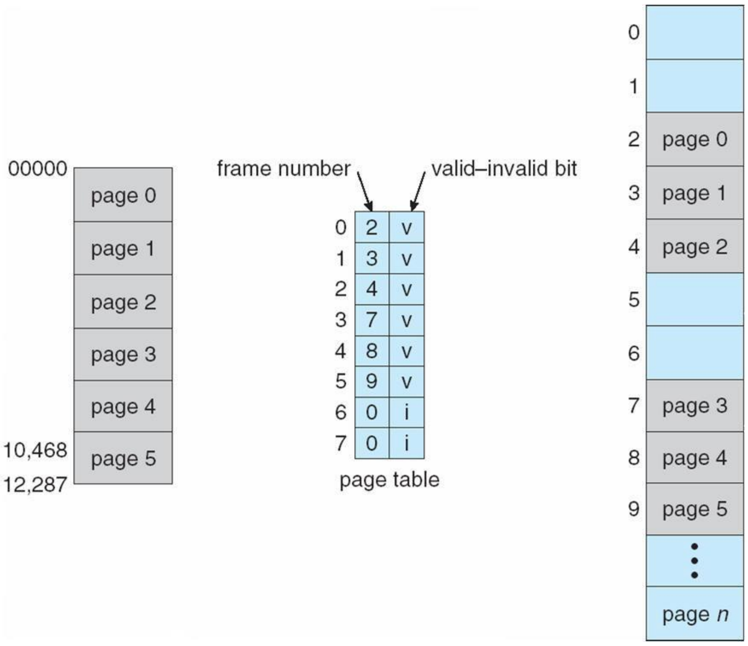


Figure:Valid(v)orinvalid(i)bitinapage-table

SharedPages

- Anadvantageof pagingisthepossibilityof sharingcommoncode.
- Re-entrantcode(PureCode)isnon-self-modifyingcode,itneverchangesduringexecution.
- Twoormoreprocesses canexecutethesamecodeatthesametime.
- Eachprocesshasitsowncopyofregistersanddata-storagetoholdthedatafortheprocess'sexecution.
- Thedatafor2differentprocesseswillbedifferent.
- Onlyone copyoftheeditor need bekept inphysical-memory(Figure 5.12).
- Each user's page-table maps onto the same physical copy of the editor, but data pages aremappedontodifferentframes.

Disadvantage:
Systemsthatuseinvertedpage-tableshavedifficultyimplementingshared-memory.

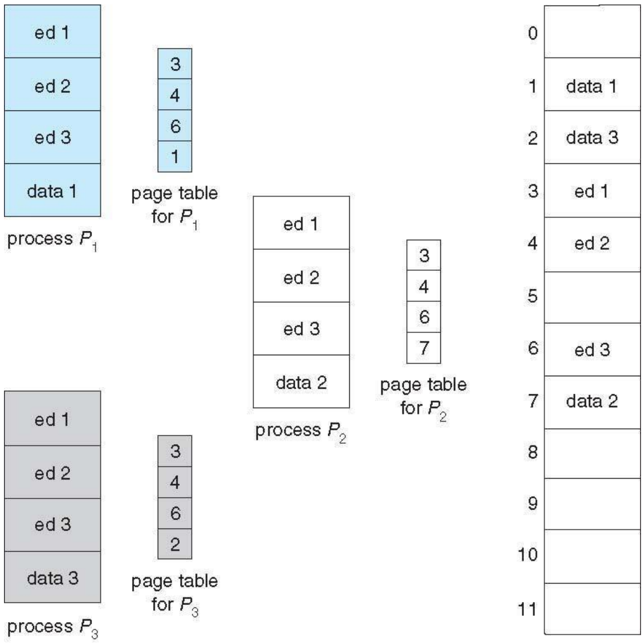


Figure:Sharingofcodeinapagingenvironment

Structureof thePage Table

Themostcommon techniquesforstructuringthepagetable:

1. HierarchicalPaging
2. HashedPage-tables
3. InvertedPage-tables

1. HierarchicalPaging

- Problem:Mostcomputerssupportalargelogical-addressspace(232to264).Inthesesystems,thepage-tableitselfbecomesexcessivelylarge.
- Solution:Dividethepage-tableintosmallerpieces.

TwoLevelPagingAlgorithm:

- Thepage-tableitselfis alsopaged.
- Thisisalsoknownasaforward-mappedpage-tablebecauseaddresstranslationworksfromtheouterpage-tableinwards.

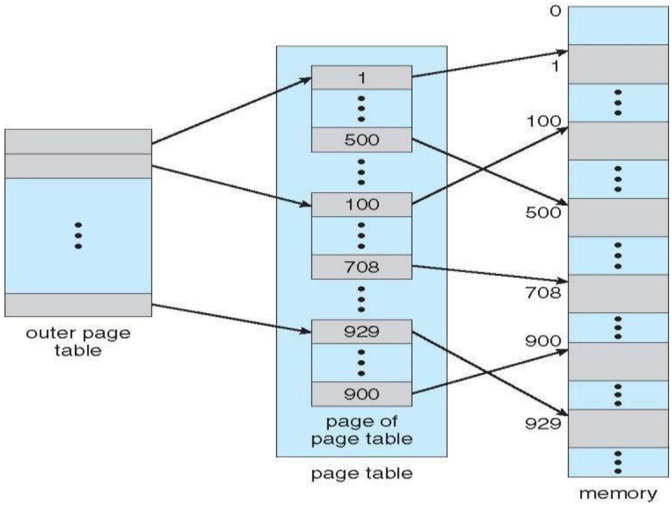


Figure:Atwo-levelpage-tablescheme

Forexample:

Consider the system with a 32-bit logical-address space and a page-size of 4 KB.Alogical-addressisdividedinto

→20-bitpage-numberand

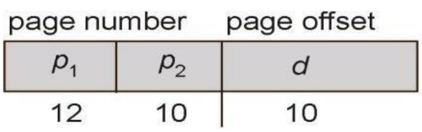
→12-bitpage-offset.

Sincethepage-tableispaged,thepage-numberisfurtherdividedinto

→10-bitpage-numberand

→10-bitpage-offset.

Thus,alogical-address isasfollows:



- where$p_1$isanindexintotheouterpagetable,and$p_2$isthedisplacementwithinthepageoftheinnerpagetable

The address-translation method for this architecture is shown in below figure. Because addresstranslation works from the outer page table inward, this scheme is also known as a forward-mappedpagetable.

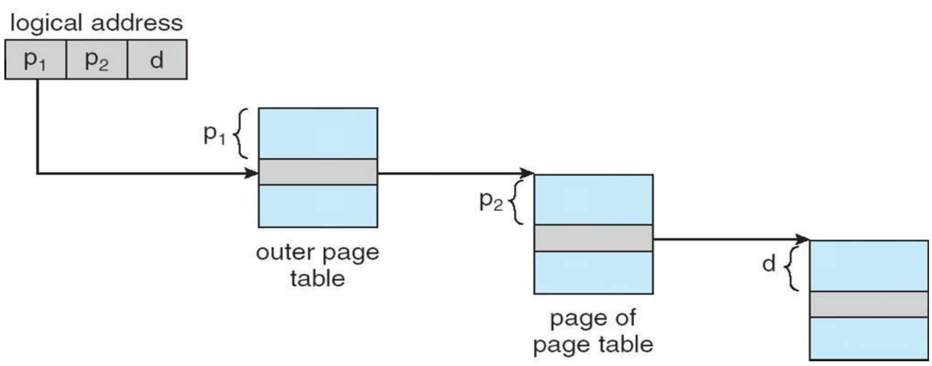


Figure:Addresstranslationforatwo-level32-bitpagingarchitecture

2. HashedPage Tables

- Thisapproachisusedforhandlingaddressspaceslargerthan32bits.
- Thehash-valueisthevirtualpage-number.
- Eachentryinthehash-tablecontainsalinked-listofelementsthathashtothesamelocation(tohandlecollisions).
- Eachelementconsists of 3fields:
    1. Virtualpage-number
    2. Valueof the mappedpage-frameand
    3. Pointerto thenextelementinthe linked-list.

Thealgorithmworksasfollows:

- Thevirtual page-numberishashed intothehash-table.
- Thevirtual page-numberiscomparedwith thefirstelementinthelinked-list.
- Ifthereisamatch,thecorrespondingpage-frame(field2)isusedtoformthedesiredphysical-address.
- Ifthereisnomatch,subsequententriesinthelinked-listaresearchedforamatchingvirtualpage-number.

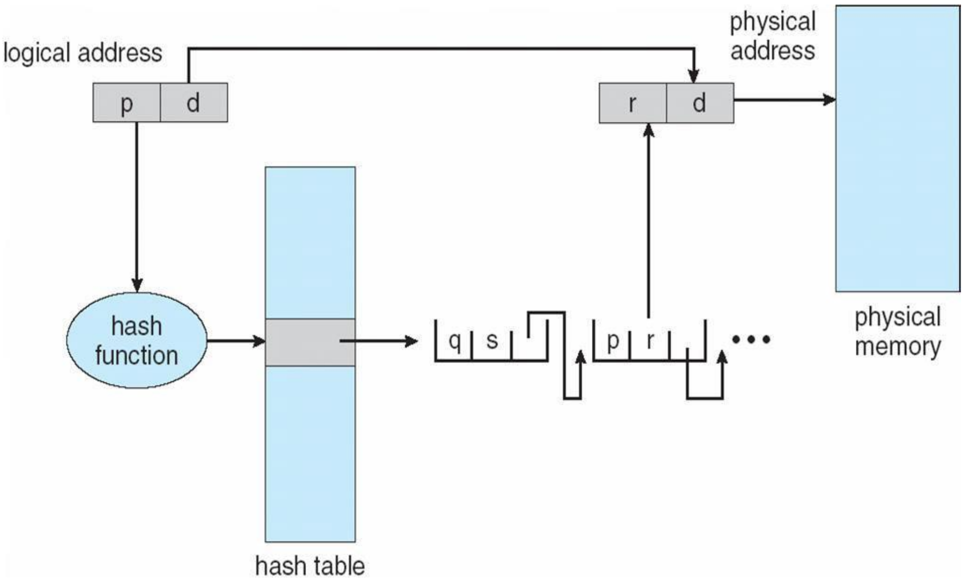


Figure:Hashedpage-table

3. InvertedPage Tables

- Hasoneentryforeach realpageofmemory.
- Eachentryconsistsofvirtual-addressofthepagestoredinthatrealmemory-locationandinformationabouttheprocessthatownsthepage.
- Eachvirtual-addressconsistsof atriplet<process-id,page-number,offset>.
- Eachinvertedpage-tableentryisapair<process-id,page-number>

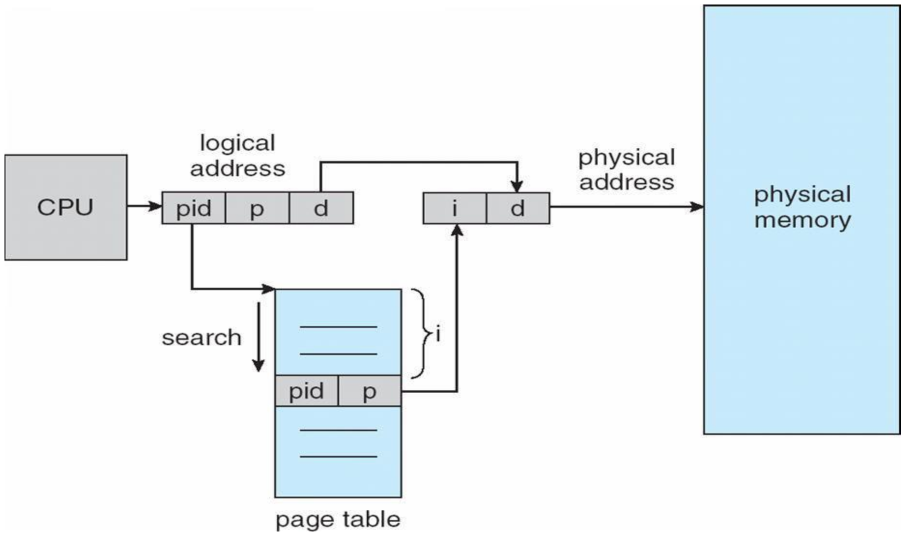


Figure:Invertedpage-table

Thealgorithmworksasfollows:

1. Whenamemory-referenceoccurs,partofthevirtual-address,consistingof<process-id,page-number>,ispresentedtothememorysubsystem.
2. Theinverted page-tableisthensearchedforamatch.
3. Ifamatchisfound,at entryi-then thephysical-address<i, offset> isgenerated.
4. If nomatchisfound,thenanillegal addressaccesshasbeenattempted.

Advantage:

1.Decreasesmemoryneededtostore eachpage-table

Disadvantages:

1. Increasesamountof time neededtosearchtablewhenapagereferenceoccurs.
2. Difficultyimplementing shared-memory

Segmentation

BasicMethodofSegmentation

- Thisisa memory-managementschemethatsupportsuser-viewofmemory(Figure1).
- Alogical-addressspaceisacollectionof segments.
- Eachsegmenthasa name andalength.
- Theaddressesspecifybothsegment-nameandoffsetwithinthesegment.
- Normally,theuser-programiscompiled,andthecompilerautomaticallyconstructssegmentsreflectingtheinputprogram.
- Forex:Thecode,Globalvariables,Theheap,fromwhichmemoryisallocated,Thestacksusedby eachthread,ThestandardClibrary

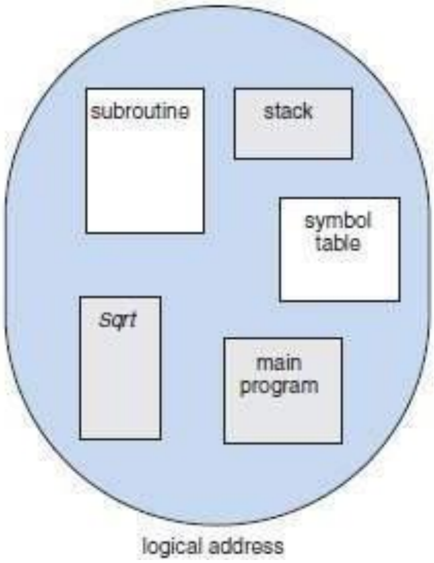


Figure:Programmer’s viewofaprogram

HardwaresupportforSegmentation

- Segment-tablemaps 2dimensionaluser-defined addresses intoone-dimensionalphysicaladdresses.
- Inthesegment-table,eachentryhasfollowing2fields:
  1. Segment-basecontainsstartingphysical-addresswherethesegmentresidesinmemory.
  2. Segment-limitspecifiesthelengthofthesegment(Figure2).
- Alogical-addressconsistsof 2parts:
  1. Segment-number(s) is usedasanindexto the segment-table
  2. Offset(d)mustbebetween0andthesegment-limit.
- Ifoffsetisnotbetween0&segment-limit,thenwetraptotheOS(logical-addressingattemptbeyondendofsegment).
- Ifoffsetislegal,thenitisaddedtothesegment-basetoproducethephysical-memoryaddress.

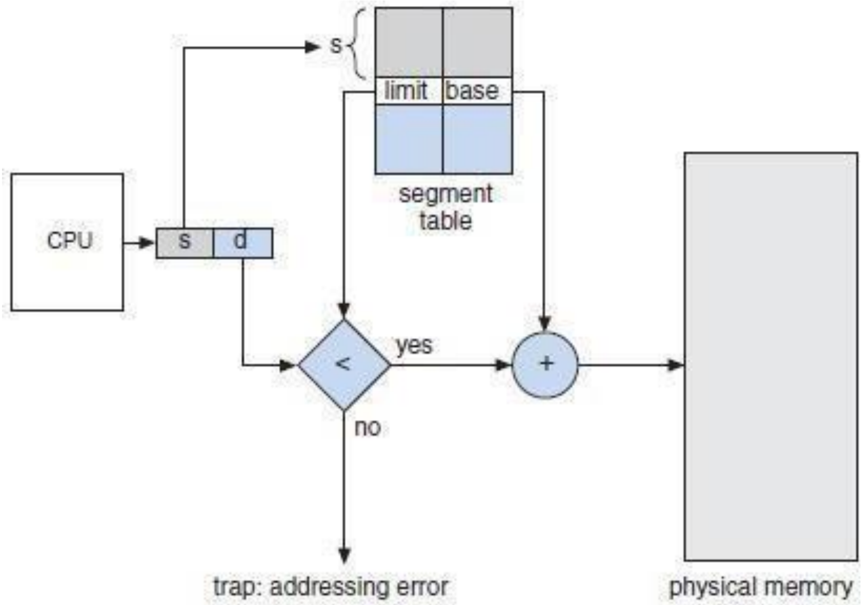


Figure:Segmentation hardware

VIRTUALMEMORYMANAGEMENT

Background

- Virtual memoryis atechniquethatallowsfortheexecution of partiallyloadedprocess.
- Advantages:
  - A program will not be limited by the amount of physical memory that is availableusercanabletowriteintolargevirtualspace.
  - Sinceeachprogramtakeslessamountofphysicalmemory,morethanoneprogram couldbe runat thesametimewhich canincreasethe throughputandCPUutilization.
  - Less i/o operation is needed to swap or load user program in to memory. So eachuserprogramcouldrunfaster.

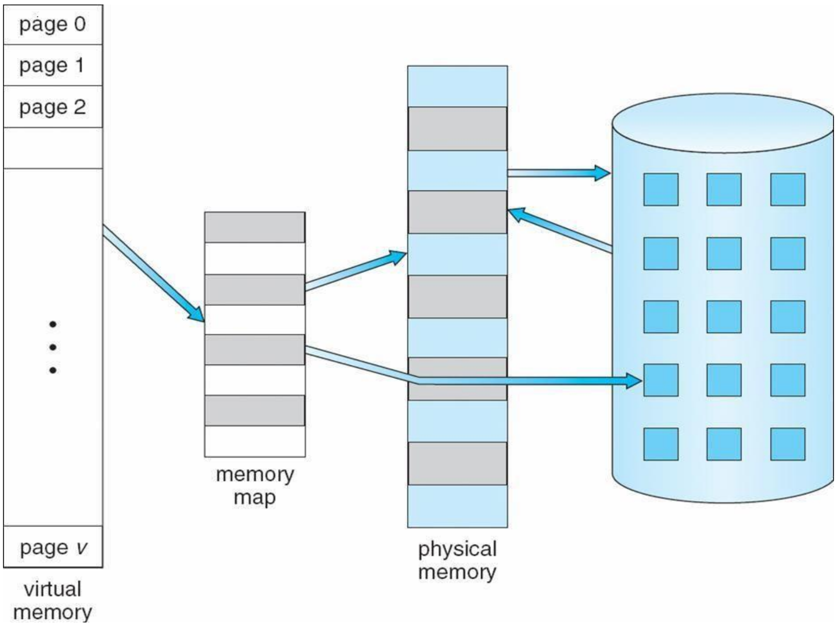


Fig:Virtualmemorythatislargerthanphysicalmemory.

- Virtual memory is the separation of users logical memory from physical memory. Thisseparation allows an extremely large virtual memory to be provided when these is lessphysicalmemory.
- Separating logical memory from physical memory also allows files and memory to besharedbyseveraldifferentprocessesthroughpagesharing.

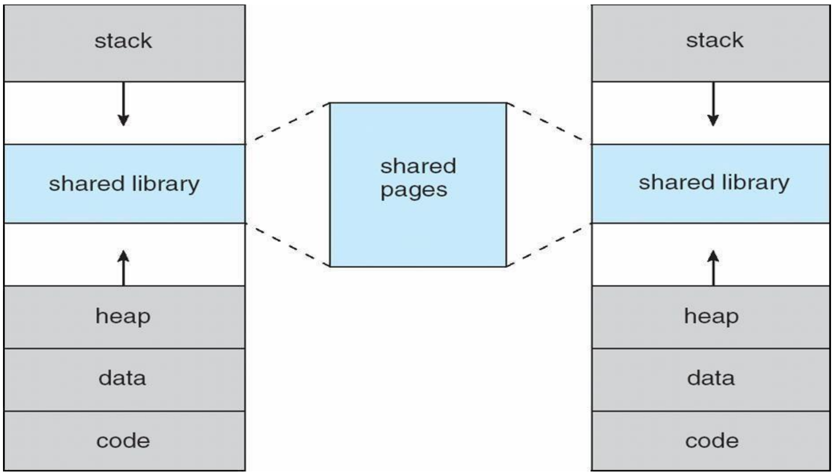


Fig:SharedLibraryusingVirtual Memory

- Virtualmemoryis implementedusingDemandPaging.
- Virtual address space: Every process has a virtual address space i.e used as the stack orheapgrowsin size.

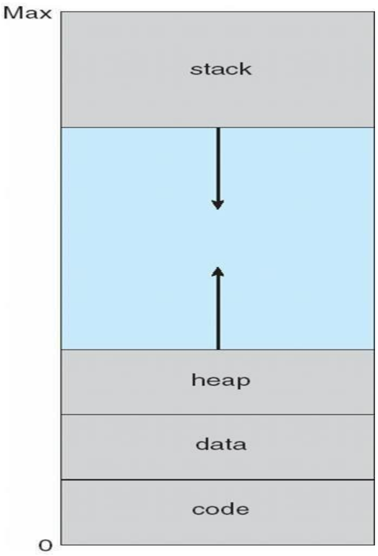


Fig:Virtualaddressspace

DEMANDPAGING

- A demand paging is similar to paging system with swapping when we want to execute aprocess we swap the process the in to memory otherwise it will not be loaded in tomemory.
- A swapper manipulates the entire processes, where as a pager manipulates individualpagesofthe process.
  - Bringapageintomemoryonlywhenit isneeded
  - LessI/Oneeded
  - Lessmemoryneeded
  - Fasterresponse

- Moreusers
- Pageisneeded⇒referenceto it
- invalidreference⇒abort
- not-in-memory⇒bringtomemory
- ***Lazyswapper***–neverswapsapageintomemoryunlesspagewillbeneeded
- Swapperthatdealswithpagesisa**pager.**

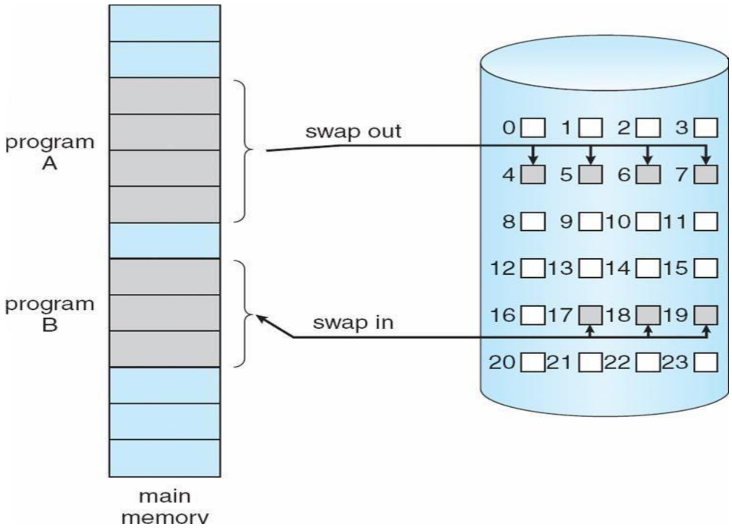
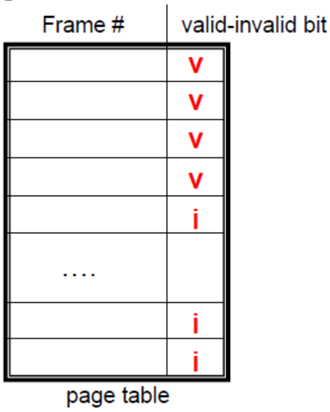


Fig:Transferofapaged memoryintocontinuousdiskspace

- **Basic concept:** Instead of swapping the whole process the pager swaps only the necessarypages in to memory. Thus it avoids reading unused pages and decreases the swap time andamountofphysicalmemoryneeded.
- The valid-invalid bit scheme can be used to distinguish between the pages that are on the diskandthatareinmemory.
  - Witheachpagetableentryavalid–invalidbitisassociated
  - (**v**⇒in-memory,**i**⇒not-in-memory)
  - Initiallyvalid–invalidbitissetto**i**onallentries
  - Exampleof apagetablesnapshot:



- Duringaddresstranslation,ifvalid–invalidbitinpagetableentryis**I**⇒pagefault.
- Ifthebitisvalidthenthepageisbothlegalandisinmemory.
- Ifthebitisinvalidtheneitherpageisnotvalid orisvalidbutiscurrentlyonthedisk. Marking

a page as invalid will have no effect if the processes never access to that page. Suppose if itaccess the page which is marked invalid, causes a page fault trap. This may result in failure ofOStobringthedesiredpageintomemory.

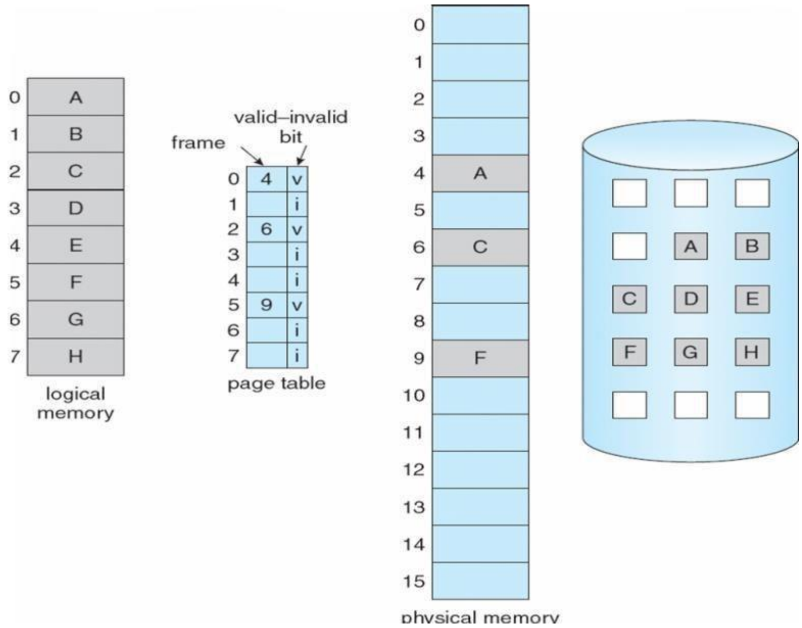


Fig:PageTable whensome pages arenot in mainmemory

PageFault

Ifapageisneededthatwas notoriginallyloaded up,then a***pagefaulttrap***isgenerated.

Stepsin HandlingaPageFault

1. Thememory addressrequestedis first checked,tomakesureitwasavalidmemoryrequest.
2. Ifthereferenceistoaninvalidpage,theprocessisterminated.Otherwise,ifthepageisnotpresentin memory,itmustbepagedin.
3. Afreeframeislocated, possiblyfromafree-framelist.
4. Adiskoperationisscheduledtobringinthenecessarypagefromdisk.
5. After the page is loaded tomemory, the process's page table is updated with thenewframe number, and the invalid bit is changed to indicate that this is now a valid pagereference.
6. Theinstructionthatcausedthepagefaultmustnowberestartedfromthebeginning

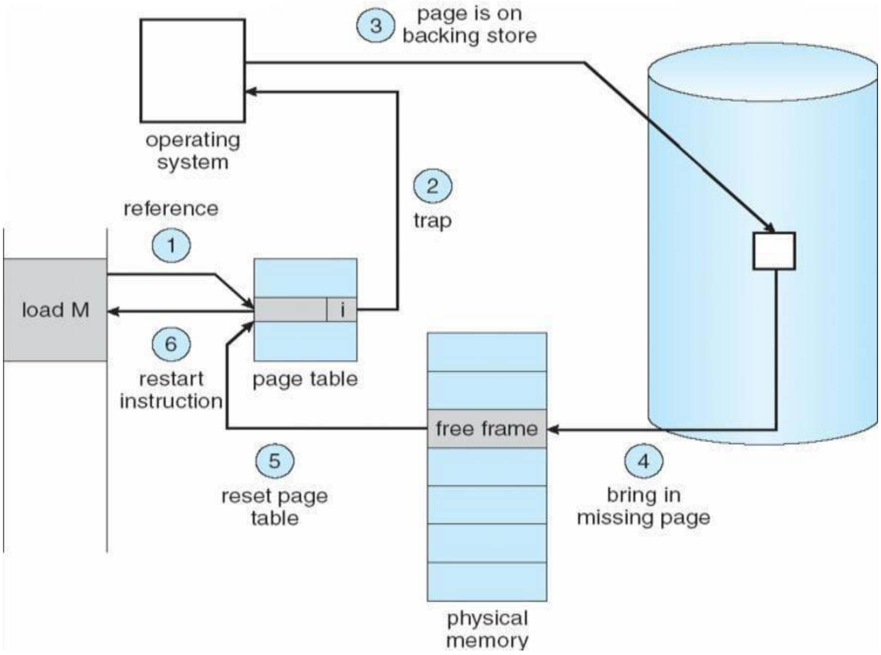


Fig:stepsinhandlingpagefault

PureDemandPaging:Neverbringapageintomainmemoryuntilit isrequired.

- We canstartexecuting aprocesswithoutloadinganyofitspagesinto mainmemory.
- Pagefaultoccursforthenonmemoryresidentpages.
- Afterthepageisbroughtintomemory,processcontinues toexecute.
- Againpagefaultoccursforthenextpage.

Hardwaresupport: Fordemandpagingthesamehardwareisrequiredaspagingandswapping.

1. Pagetable:-Has theabilitytomarkanentryinvalidthroughvalid-invalid bit.
2. Secondarymemory:-Thisholdsthepagesthat arenotpresent in main memory.

PerformanceofDemandPaging: Demandpagingcanhavesignificanteffectontheperformanceofthecomputer system.

- LetPbetheprobabilityofthepagefault(0<=P<=1)
- **Effectiveaccesstime=(1-P)*ma+P *pagefault.**
  - WhereP=pagefaultand ma=memoryaccesstime.
- Effectiveaccesstimeisdirectlyproportionaltopagefaultrate.Itisimportanttokeeppagefaultratelowindemandpaging.

DemandPagingExample

- Memoryaccesstime=200 nanoseconds
- Averagepage-faultservicetime=8milliseconds
- EAT=(1–p)x200+p(8milliseconds)

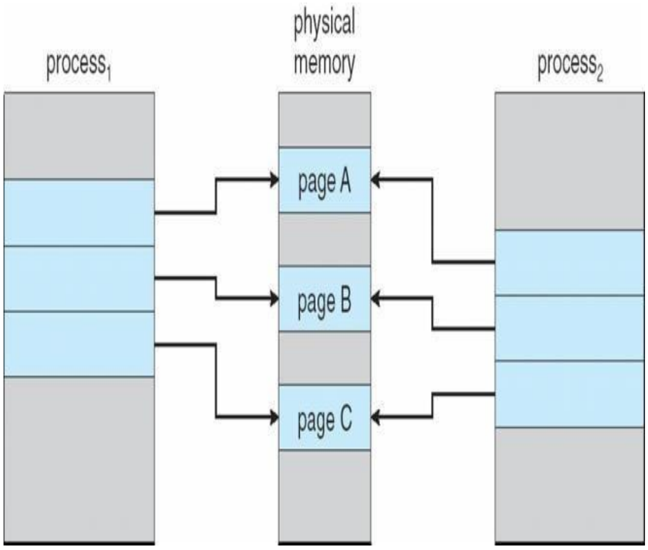
  =(1 –px200+px8,000,000

  =200+ px 7,999,800
- Ifoneaccessoutof1,000causesapagefault,thenEAT=8.2microseconds.Thisisaslowdownby afactorof40.

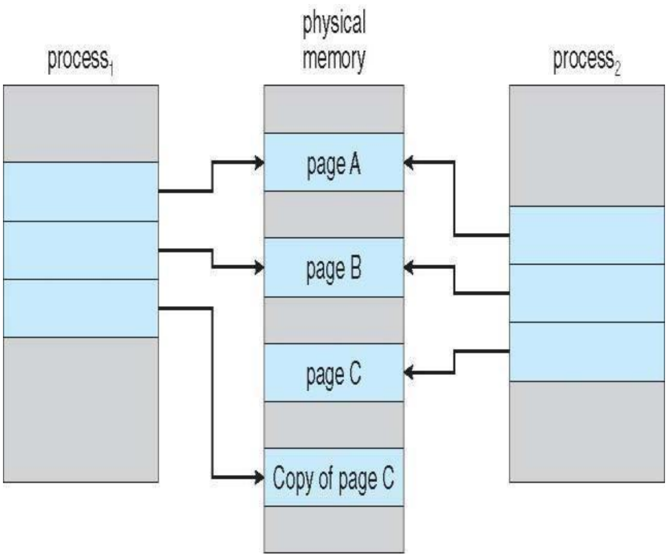
COPY-ON-WRITE

- Technique initially allows the parent and the child to share the samepages.Thesepagesaremarkedascopyon-writepagesi.e.,ifeitherprocesswritestoasharedpage,acopyofsharedpageiscreated.
- *Eg*:-If a child process try to modify a page containing portions of the stack; the OSrecognizes them as a copy-on-write page and create a copy of this page and maps it onto the address space of the child process. So the child process will modify its copiedpageandnotthepagebelongingtoparent.Thenewpagesareobtainedfromthepooloffreepages.
- The previous contents of pages are erased before getting them into main memory. Thisiscalled**Zero –onfill demand**.

a) BeforeProcess1modifiespageC



b) Afterprocess1modifiespageC



PAGEREPLACEMENT

- Page replacement policy deals with thesolution ofpages inmemory to be replacedby anewpagethat must bebroughtin.Whenauserprocessisexecutingapagefaultoccurs.
- Thehardwaretrapstotheoperatingsystem,whichcheckstheinternaltabletoseethatthisisapage faultandnotanillegalmemoryaccess.
- The operating system determines where the derived page is residing on the disk, and thisfindsthattherearenofreeframesonthelistoffreeframes.
- When all the frames are in main memory, it is necessary to bring a new page to satisfy thepage fault, replacement policy is concerned with selecting a page currently in memory to bereplaced.
- Thepagei,etoberemovedshouldbethe pagei,eleastlikelytobe referencedinfuture.

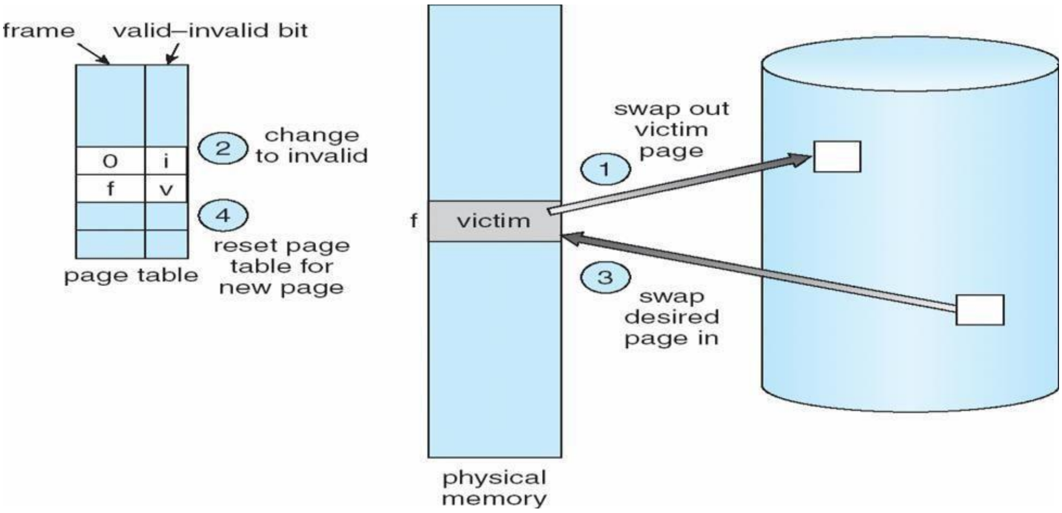


Fig:PageReplacement

WorkingofPage ReplacementAlgorithm

1. Findthelocationof derivedpageonthe disk.
2. FindafreeframexIfthereisafreeframe,useit.xOtherwise,useareplacementalgorithmto selectavictim.
   - Writethevictimpagetothedisk.
   - Changethepageandframetablesaccordingly.
3. Readthedesiredpageintothefreeframe;changethepageandframetables.
4. Restarttheuserprocess.

VictimPage

- Thepagethatissupportedoutof physicalmemoryiscalledvictimpage.
- Ifno framesarefree,thetwopagetransformscome(outandonein)areread.Thiswillseetheeffectiveacc esstime.
- Eachpageorframemayhaveadirty(modify)bitassociatedwiththehardware.Themodifybitforap ageissetbythehardwarewheneveranywordorbyteinthepageis

writteninto,indicating thatthepagehasbeenmodified.

- When we select the page for replacement, we check its modify bit. If the bit is set, then thepageismodifiedsinceitwasreadfromthedisk.
- If the bit was not set, the page has not been modified since it was read into memory.Therefore, if the copy of the page has not been modified we can avoid writing the memorypagetothedisk,if itisalreadythere.Sumpagescannotbemodified.

Modifybit/Dirty bit:

- Eachpage/framehas a modifybitassociated withit.
- Ifthepageisnotmodified(read-only)thenonecandiscardsuchpagewithoutwritingitontothedisk**.** Modifybitofsuchpageissetto0.
- Modifybitissetto1,ifthepagehasbeenmodified.Suchpagesmustbewrittentothedisk.
- Modifybitisusedtoreduceoverheadofpagetransfers–onlymodifiedpagesarewrittentodisk

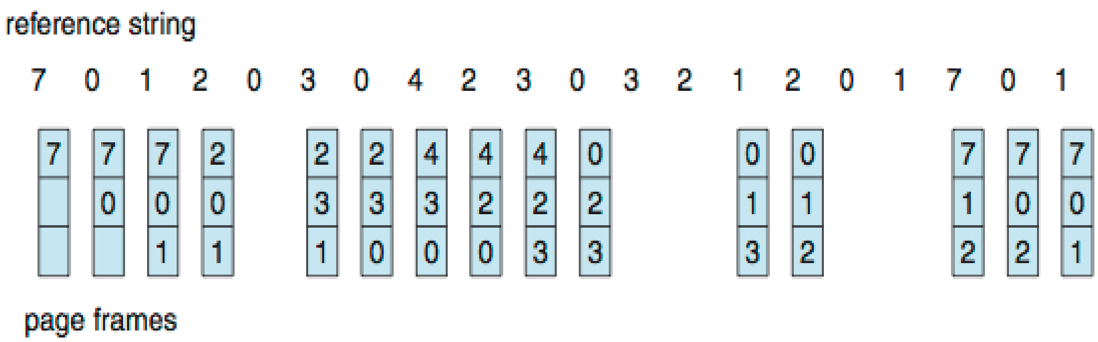
PAGEREPLACEMENTALGORITHMS

- Wantlowestpage-faultrate
- Evaluatealgorithmbyrunningitonaparticularstringofmemoryreferences(referencestring)andcomputingthenumberofpagefaultsonthatstring
- Inallourexamples,thereferencestringis

1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

FIFOAlgorithm:

- Thisisthesimplestpagereplacementalgorithm.AFIFOreplacementalgorithmassociateseachpage thetimewhen thatpagewasbrought intomemory.
- WhenaPageisto bereplacedtheoldestoneis selected.
- Wereplacethequeueattheheadofthequeue.Whenapageisbroughtintomemory,weinsertitatthetailofthequeue.
- Inthefollowingexample,areferencestringisgivenandthereare3freeframes.Thereare20pagerequests,whichresultsin15page faults



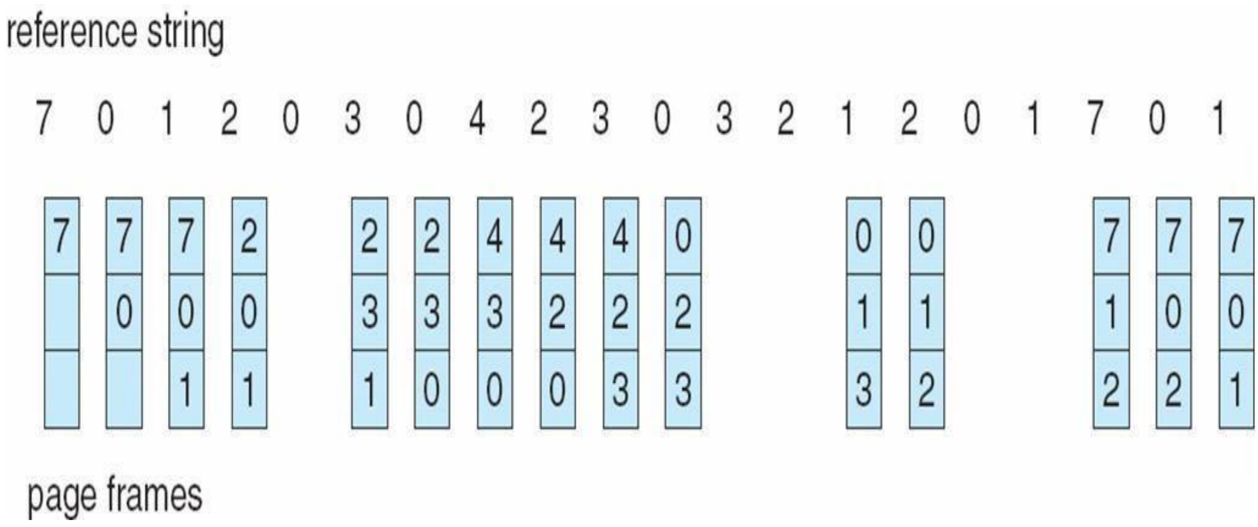
Belady'sAnomaly

- Forsomepagereplacementalgorithm,thepagefaultmayincreaseasthenumberofallocatedframes increases.FIFOreplacement algorithmmayfacethisproblem.

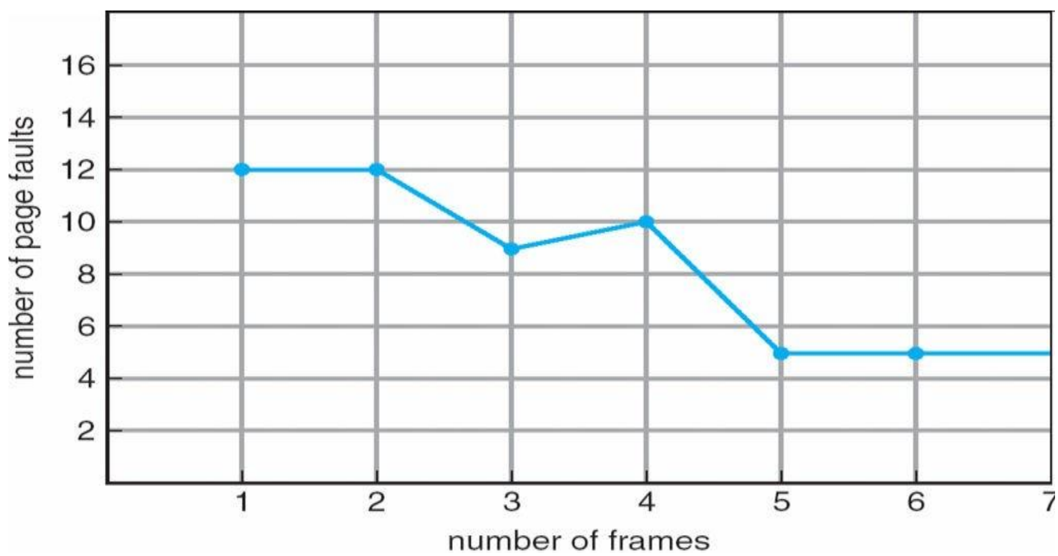
  moreframes⇒ morepagefaults

Example: Consider thefollowingreferencesstringwithframesinitiallyempty.

- Thefirstthreereferences(7,0,1)casespagefaultsandarebroughtintotheemptyframes.
- Thenextreferences2replacespage7becausethepage7wasbroughtinfirst.xSince0isthenext referencesand0isalreadyinmemoryehasnopagefaults.
- Thenextreferences3resultsinpage0beingreplacedsothatthenextreferencesto0causerpagefault. This willcontinuetilltheendof string.Thereare 15faultsalltogether.



FIFOIllustratingBelady'sAnomaly



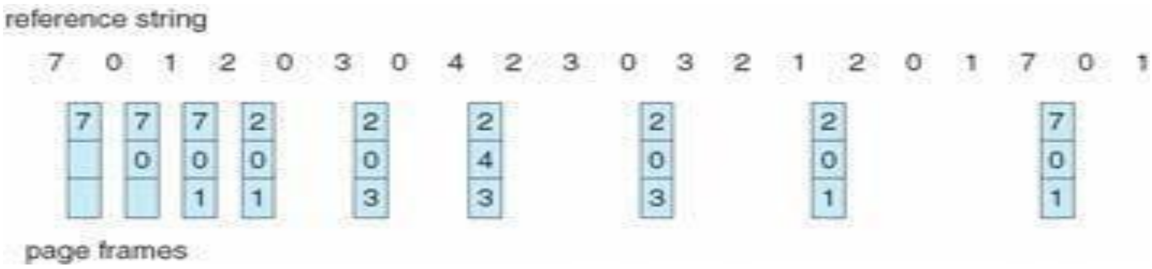
OptimalAlgorithm

- Optimalpagereplacementalgorithmis mainlytosolvetheproblemofBelady's Anomaly.
- Optimalpagereplacementalgorithmhasthelowestpagefaultrateofallalgorithms.
- Anoptimalpage replacementalgorithmexists andhasbeencalledOPT.

The working is simple "Replace the page that will not be used for the longest period of time"Example:considerthefollowingreferencestring

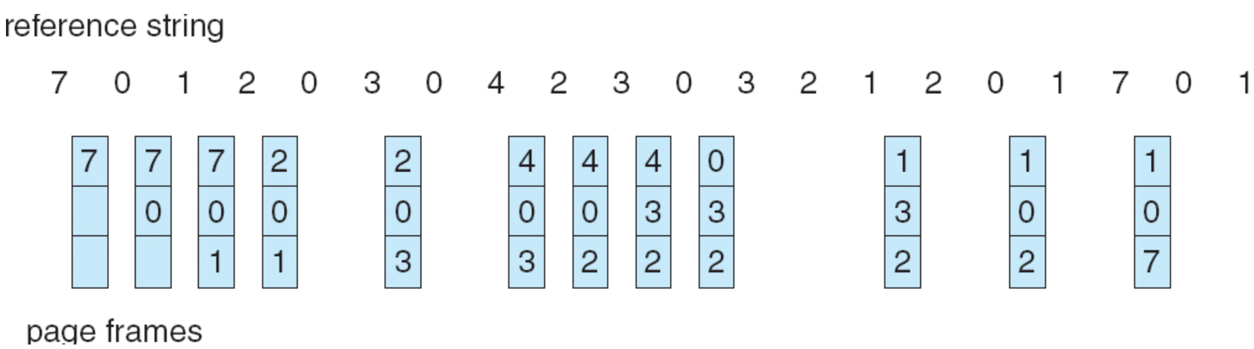
- Thefirstthreereferencescausefaultsthatfillthethreeemptyframes.
- The references to page 2 replaces page 7, because 7 will not be used until reference 18. xThepage0willbeusedat5andpage1at 14.
- With only 9 page faults, optimal replacement is much better than a FIFO, which had 15faults. This algorithm is difficult t implement because it requires future knowledge ofreferencestrings.
- Replacepagethatwill notbeusedforlongest periodoftime

OptimalPageReplacement



LeastRecently Used(LRU)Algorithm

- The***LRU(LeastRecentlyUsed)***algorithm,predictsthatthepagethathasnotbeenusedinthelongesttimeistheonethatwillnot beusedagaininthenearfuture.
- SomeviewLRUasanalogoustoOPT,butherewelookbackwardsintimeinsteadofforwards.



ThemainproblemtohowtoimplementLRU istheLRUrequiresadditionalh/wassistance.

Twoimplementationarepossible:

1. **Counters:** In this we associate each page table entry a time -of -use field, and add tothecpualogicalclockorcounter.Theclockisincrementedforeachmemoryreference. When a reference to a page is made, the contents of the clock register arecopiedtothetime-of-usefieldinthepagetableentryforthatpage.Inthiswaywehave the time of last reference to each page we replace the page with smallest timevalue.Thetimemustalsobemaintainedwhenpagetablesarechanged.
2. **Stack:** Another approach to implement LRU replacement is to keep a stack of pagenumbers when a page is referenced it is removed from the stack and put on to the topof stack. In this way the top of stack is always the most recently used page and thebottom in least recently used page. Since the entries are removed from the stack it isbestimplementbyadoublylinkedlist.Withaheadandtailpointer.

Note: Neither optimal replacement nor LRU replacement suffers from Belady's Anamoly.Thesearecalledstackalgorithms.

LRU-ApproximationPageReplacement

- Manysystemsoffersomedegreeofhardwaresupport,enoughto approximateLRU.
- Inparticular,manysystemsprovidea***referencebit***foreveryentryinapagetable,whichis set anytime that page is accessed. Initially all bits are set to zero, and they can also all becleared at any time. One bit distinguishes pages that have been accessed since the last clearfromthosethathavenotbeenaccessed.

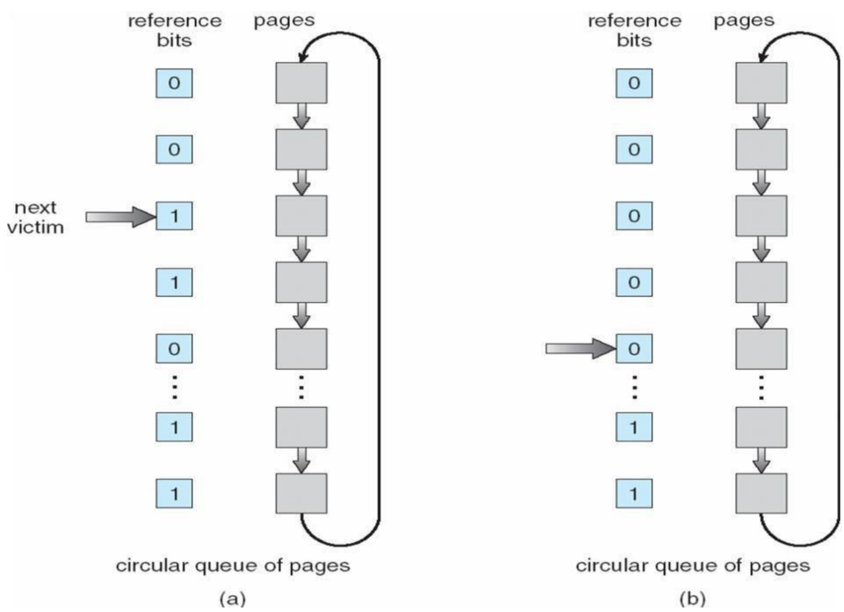
Additional-Reference-BitsAlgorithm

- An8-bitbyte (referencebit)isstoredforeachpageinatableinmemory.
- At regular intervals (say, every 100 milliseconds), a timer interrupt transfers control to theoperating system. The operating system shifts the reference bit for each page into thehigh-order bit of its 8-bit byte, shifting the other bits right by 1 bit and discarding the low-orderbit.
- These8-bitshiftregisterscontainthehistoryof pageuseforthelasteighttimeperiods.
- If the shift register contains 00000000, then the page has not been used for eight timeperiods.
- A page with a history register value of 11000100 has been used more recently than onewithavalueof01110111.

Second-chance(clock)pagereplacement algorithm

- The***secondchancealgorithm***isaFIFOreplacementalgorithm,exceptthereferencebitisusedto givepagesasecondchanceatstayinginthepagetable.
- Whenapagemustbereplaced,thepagetableisscannedinaFIFO(circularqueue)manner.
- Ifapageisfoundwithitsreferencebitas'0',thenthatpageisselectedasthenextvictim.

- If the reference bitvalueis‘1’, then the page is given a second chance and its reference bitvalueiscleared (assigned as‘0’).
- Thus, a page that is given a second chance will not be replaced until all other pages havebeen replaced (or given second chances). In addition, if a page is used often, then it setsitsreferencebitagain.
- Thisalgorithmisalsoknownasthe***clock***algorithm.



EnhancedSecond-ChanceAlgorithm

- The**enhancedsecond chancealgorithm**looksatthereferencebitandthe modifybit(dirtybit)asanorderedpage,andclassifiespagesintooneoffourclasses:
    1. (0,0)-Neitherrecentlyusednormodified.
    2. (0,1)-Notrecentlyused,butmodified.
    3. (1,0)-Recentlyused,butclean.
    4. (1,1)-Recentlyusedandmodified.
- This algorithm searches the page table in a circular fashion, looking for the first page it canfind in the lowest numbered category. i.e. it first makes a pass looking for a (0, 0), and thenifitcan'tfindone, itmakes anotherpasslookingfor a(0,1),etc.
- The main difference between this algorithm and the previous one is the preference forreplacingcleanpagesifpossible.

CountBasedPageReplacement

Thereismanyotheralgorithmsthatcanbeusedforpagereplacement,wecankeepacounterofthenumb erofreferencesthathasmadetoapage.

a) **LFU**(leastfrequentlyused):

This causes the page with the smallest count to be replaced. The reason for this selectionisthatactivelyusedpageshouldhavealargereferencecount.

This algorithm suffers from the situation in which a page is used heavily during theinitial phase of a process but never used again. Since it was used heavily, it has a largecountandremains inmemoryeventhought is no longerneeded.

b)**MFU**Algorithm:

based on the argument that the page with the smallest count was probably just brought inandhasyetto be used

ALLOCATIONOFFRAMES

- The absolute minimum number of frames that a process must be allocated is dependentonsystem architecture.
- Themaximumnumberisdefinedbytheamountof availablephysicalmemory.

AllocationAlgorithms

After loading of OS, there are two ways in which the allocation of frames can be done totheprocesses.

1. **Equal Allocation**- If there are m frames available and n processes to share them, eachprocessgetsm/nframes,andtheleftover'sarekeptinafree-framebufferpool.
2. **Proportional Allocation** - Allocate the frames proportionally depending on the sizeof the process. If the size of process i is S_i, and S is the sum of size of all processes inthe system, then the allocation for process P_i is $a_i = m * S_i / S$. where m is the freeframesavailableinthesystem.

- Consider a system with a 1KB frame size. If a small student process of 10 KB and aninteractive database of 127 KB are the only two processes running in a system with 62freeframes.
- withproportionalallocation,wewouldsplit62framesbetweentwoprocesses,asfollows
  m=62,S=(10+127)=137
  Allocationforprocess1=62X10/137~4Allocationforprocess2=62X127/137 ~57
  Thusallocates4framesand57framestostudentprocessanddatabaserespectively.
- Variationsonproportionalallocationcouldconsiderpriorityofprocessratherthanjusttheirsiz e.

GlobalversusLocalAllocation

- Pagereplacementcanoccurbothatlocalorgloballevel.
- With local replacement, the number of pages allocated to a process is fixed, and pagereplacement occursonlyamongstthepagesallocatedtothisprocess.
- With global replacement, any page may be a potential victim, whetherit currentlybelongstotheprocessseekingafreeframeornot.
- Local page replacement allows processes tobetter control their own page fault rates,and leads to more consistent performance of a given process over different system loadlevels.

- Global page replacement is over all more efficient, and is the more commonly usedapproach.

Non-UniformMemoryAccess (New)

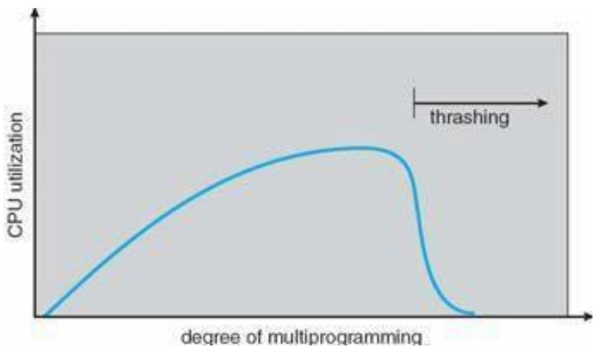
- Usuallythe timerequiredtoaccessall memoryinasystemisequivalent.
- This may not be the case in multiple-processor systems, especially where each CPU isphysically located on a separate circuit board which also holds some portion of theoverallsystemmemory.
- In such systems, CPU s can access memory that is physically located on the same boardmuchfasterthanthememoryontheotherboards.
- Thebasicsolutionis akintoprocessoraffinity -Atthesametimethatwetry toschedule processes on the same CPU to minimize cache misses, we also try to allocatememoryforthoseprocessesonthesameboards,tominimizeaccesstimes.

THRASHING

- If the number of frames allocated to a low-priority process falls below the minimumnumberrequiredbythe computerarchitecturethenwesuspendtheprocessexecution.
- Aprocessisthrashingifitisspending moretimeinpagingthanexecuting.
- If the processes do not have enough number of frames, it will quickly page fault.During this it must replace some page that is not currently in use. Consequently itquicklyfaultsagainandagain.
- The process continuesto fault, replacing pages for whichit thenfaults and bringsback. This high paging activity is called thrashing. The phenomenon of excessivelymovingpagesbackandforthb/w memoryandsecondaryhasbeencalled***thrashing***.

CauseofThrashing

- Thrashingresultsinsevereperformanceproblem.
- Theoperatingsystemmonitorsthecpuutilizationislow.Weincreasethedegreeofmultiprogrammingbyintroducingnewprocesstothesystem.
- Aglobalpagereplacementalgorithmreplacespageswithnoregardstotheprocesstowhichtheybelong.



Thefigureshowsthethrashing

- Asthedegreeofmultiprogrammingincreases,moreslowlyuntilamaximumis reached.Ifthedegreeofmultiprogrammingisincreasedfurtherthrashingsetsinandthecpuutilizationdropssharply.

- Atthispoint,toincreasesCPUutilizationandstopthrashing,wemustincreasedegreeofmultiprogramming.
- wecanlimittheeffectofthrashingbyusingalocalreplacementalgorithm.Topreventthrashing,wemustprovideaprocessasmanyframesasitneeds.

LocalityofReference:

- Astheprocessexecutesit movesfromlocalitytolocality.
- Alocalityisasetofpagesthatareactivelyused.
- Aprogrammayconsistof severaldifferentlocalities,whichmayoverlap.
- Localityiscausedbyloopsincodethatfindtoreferencearraysandotherdatastructuresbyindices.
- Theorderedlistofpagenumberaccessedbyaprogramiscalledreferencestring.
- Localityis oftwotypes:
  1.spatiallocality 2.temporallocality

Workingsetmodel

- Working set model algorithm uses the current memory requirements to determine thenumberofpageframestoallocatetotheprocess,aninformaldefinitionis"thecollection of pages that a process is working with and which must be resident if theprocess to avoid thrashing". The idea is to use the recent needs of a process to predictitsfuture reader.
- The working set is an approximation of programs locality. Ex: given a sequence ofmemory reference, if the working set window size to memory references, then workingsetattimet1 is{1,2,5,6,7}andatt2ischangedto{3,4}
- At any given time, all pages referenced by a process in its last 4 seconds of executionareconsidered to compromise itsworking set.
- Aprocess willneverexecuteuntilitsworking setisresidentin main memory.
- Pagesoutsidetheworkingsetcanbediscardedatanymovement.
- Workingsetsarenotenoughandwemustalsointroducebalanceset.
  - If the sum of the working sets of all the run able process is greater than the sizeofmemorytherefusesomeprocessforawhile.
  - Divide the run able process into two groups, active and inactive. The collectionof active set is called the balance set. When a process is made active its workingsetis loaded.
  - Some algorithm must be provided for moving process into and out of the balanceset. As a working set is changed, corresponding change is made to the balanceset.
  - Working set presents thrashing by keeping the degree of multi programming ashigh as possible. Thusif optimizes the CPUutilization. Themain disadvantageofthisiskeepingtrackoftheworkingset.

Page-FaultFrequency

- When page- fault rate is too high, the process needs more frames and when it is too low,theprocessmayhavetoomanyframes.
- Theupperandlowerboundscanbeestablishedonthepage-faultrate.Iftheactualpage-fault rate exceeds the upper limit, allocate the process another frame or suspendtheprocess.
- Ifthe page-fault rate falls below the lower limit, remove a framefrom the process.Thus,wecandirectlymeasureandcontrolthepage-faultratetopreventthrashing.

QUESTIONBANK

DEADLOCKS

1. Whataredeadlocks?Whatareitscharacteristics?Explainthenecessaryconditionsforitsoccurrence.
2. Explaintheprocessofrecoveryfromdeadlock.
3. DescribeRAG:
   i) Withdeadlock
   ii) Withacyclebutnodeadlock
4. WhatisResourceAllocationGraph(RAG)?ExplainhowRAGisveryusefulindescribingdeadlyembrace(deadlock)byconsideringyourownexample.
5. Withthehelpofasystemmodel,explainadeadlockandexplainthenecessaryconditions thatmust holdsimultaneouslyinasystemforadeadlocktooccur.
6. Explain how deadlock can be prevented by considering four necessary conditions cannothold.
7. UsingBanker's algorithmdetermineswhether thesystemisina safestate.
8. Howisasystemrecoveredfromdeadlock?Explainthedifferentmethodsusedtorecoverfromdeadlock.
9. Explaindeadlockdetectionwithalgorithmandexample
10. Definetheterms:safestateandsafesequence.Giveanalgorithmtofindwhetherornotasystemisinasafestate.

MEMORYMANAGEMENT

1. Explainthemultistepprocessingof auserprogramwitha neatblockdiagram.
2. Distinguishbetweeninternalandexternalfragmentation.
3. Explainsegmentation withanexample.
4. Explainwithadiagram,howTLB isusedtosolvetheproblemof simplepagingscheme.
5. With a supporting paging hardware, explain in detail concept of paging with an examplefor a 32-byte memory with 4-type pages with a process being 16-bytes. How many bitsare reserved for page number and page offset in the logical address. Suppose the logicaladdress is 5, calculate the corresponding physical address, after populating memory andpagetable.
6. Whatarethedrawbacksof contiguous memoryallocation?
7. Considerapaging systemwiththepagetable storedinmemory.
   i. if a memory reference takes 200 nano seconds, how long does a paged memoryreferencetake?
   ii. if we add associative register and 75 percentage of all page table references arefound in the associative registers, what is the effective memory access time?(Assumethatfindingapagetableentryintheassociativememory/registerstakes

zerotime,iftheentryisfound).

8. Distinguishbetween:
   i. Logicaladdressspace andphysicaladdressspace.
   ii. Internalfragmentation andexternalfragmentation.
   iii. Pagingand segmentation.
9. ExplainwiththehelpofsupportinghardwarediagramhowtheTLBimprovestheperformanceof ademandpagingsystem.
10. Explainthe concept of forwardmappedpagetable.
11. Whatisfragmentation?Explaintwotypesof memoryfragmentation
12. Whatisswapping?Explainindetail.
13. Whatdoyoumeanbyaddressbinding?Explainwiththenecessarysteps,thebindingofinstructionsanddatatomemoryaddresses.

UniversityQuestions

June/July2018

1. Consider the following snapshot of a system

	Allocation			Max			Available		
	A	B	C	A	B	C	A	B	C
P_0	0	0	2	0	0	4	1	0	2
P_1	1	0	0	2	0	1			
P_2	1	3	5	1	3	7			
P_3	6	3	2	8	4	2			
P_4	1	4	3	1	5	7			

2. Find the need matrix and calculate safe sequence using Banker's algorithm. Mention the above system is safe or not safe. **(08 Marks)**
3. What are the necessary conditions for deadlock? Explain different methods to recover from deadlock. **(08 Marks)**
4. What is paging? Explain paging hardware with translation look-aside buffer. **(06 Marks)**
5. Explain the structure of page table with respect to hierarchical paging. **(06 Marks)**
6. Given the 5 memory partitions 100 KB, 500 KB, 200 KB, 300 KB and 600 KB, how each of the first fit, best fit and worst fit algorithms place processes of 212 KB, 417 KB, 112KB and 426KB size. Which algorithm makes efficient use of memory? **(04 Marks)**

Dec.2018/ Jan2019

1. Define deadlock. Write short notes on 4 necessary conditions that arise deadlocks.
2. Assume that there are 5 processes PO through P4 and 4 types of resources. At time T_0 we have the following state :

Process	Allocation				Max				Available			
	A	B	C	D	A	B	C	D	A	B	C	D
P_0	0	0	1	2	0	0	1	2	1	5	2	0
P_1	1	0	0	0	1	7	5	0				
P_2	1	3	5	4	2	3	5	6				
P_3	0	6	3	2	0	6	5	2				
P_4	0	0	1	4	0	6	5	6				

3. Apply Banker's algorithm to answer the following :
   i) What is the content of need matrix?
   ii) Is the system in a safe state?
   iii) If a request from a process P1(0, 4, 2, 0) arrives, can it be granted?
4. Write short notes on :
   i) External and internal fragmentation
   ii) Dynamic loading and linking.
5. Analyze the problem in simple paging technique and show how TLB is used to solve the problem.

June/July2019

1. Determine whether the following system is in safe state by using Banker's algorithm.

Process	Allocation			Maximum			Available		
	A	B	C	A	B	C	A	B	C
P_0	0	1	0	7	5	3	3	3	2
P_1	2	0	0	3	2	2			
P_2	3	0	2	9	0	2			
P_3	2	1	1	2	2	2			
P_4	0	0	0	4	3	3			

If a request for P_1 arrives for (1 0 2), can the request be granted immediately? **(09 Marks)**

2. Discuss the various approaches used for deadlock recovery. **(07 Marks)**

3. Illustrate with example, the internal and external fragmentation problem encountered in continuous memory allocation. **(07 Marks)**

4. Explain the structure of page table. **(09 Marks)**