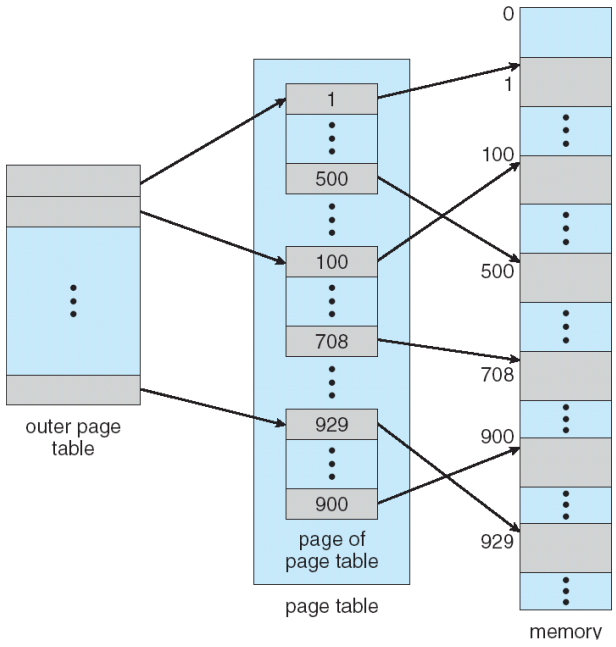
2 (a) With suitable diagrams, explain the different techniques used for structuring the page table.

a. **Hierarchical paging**:-

➔ Recent computer system support a large logical address apace from 2^32 to 2^64. In this system the page table becomes large. So it is very difficult to allocate contiguous main memory for page table. One simple solution to this problem is to divide page table in to smaller pieces. There are several ways to accomplish this division.

➔ One way is to use two-level paging algorithm in which the page table itself is also paged.
Eg:- In a 32 bit machine with page size of 4kb. A logical address is divided in to a page number consisting of 20 bits and a page offset of 12 bit. The page table is further divided



since the page table is paged, the page number is further divided in to 10 bit page number and a 10 bit offset. So the logical address is

b. Hashed page table:-

Hashed page table handles the address space larger than 32 bit. The virtual page number is used as hashed value. Linked list is used in the hash table which contains a list of elements that hash to the same location.
Each element in the hash table contains the following three fields:-
• Virtual page number
• Mapped page frame value

• Pointer to the next element in the linked list

Working:-

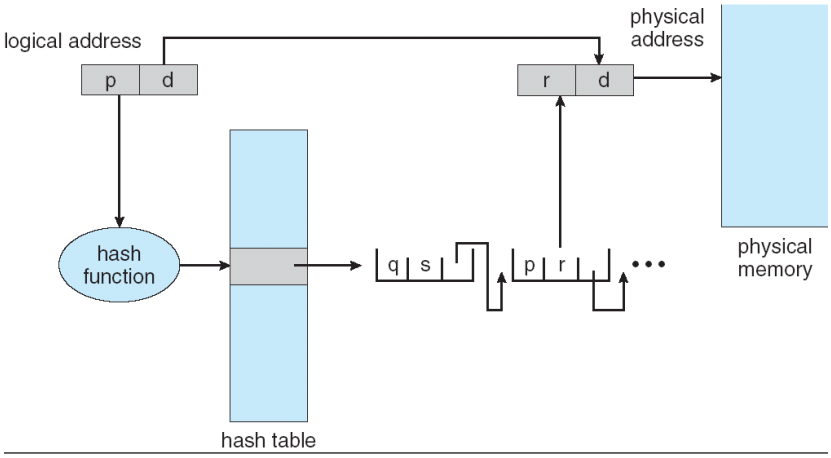
 Virtual page number is taken from virtual address.

 Virtual page number is hashed in to hash table.

 Virtual page number is compared with the first element of linked list.

 Both the values are matched, that value is (page frame) used for calculating the physical address.

 If not match then entire linked list is searched for matching virtual page number.



• Clustered pages are similar to hash table but one difference is that each entity in the hash table refer to several pages.

c. **Inverted Page Tables**:-

 Since the address spaces have grown to 64 bits, the traditional page tables become a problem. Even with two level page tables. The table can be too large to handle.

 An inverted page table has only entry for each page in memory.

 Each entry consisted of virtual address of the page stored in that read-only location with information about the process that owns that page.

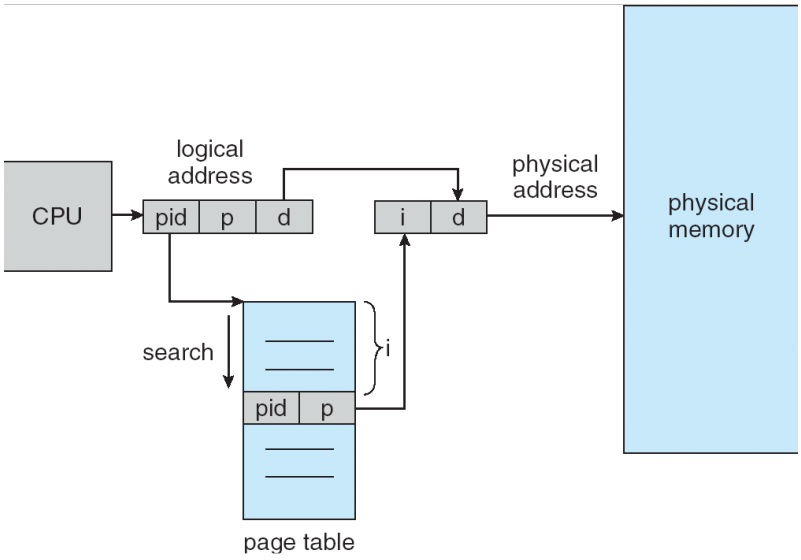
 Each virtual address in the Inverted page table consists of triple <process-id , page number , offset >.

 The inverted page table entry is a pair <process-id , page number>. When a memory reference is made, the part of virtual address i.e., <process-id , page number> is presented in to memory sub-system.

 The inverted page table is searched for a match.

 If a match is found at entry I then the physical address <i , offset> is generated. If no match is found then an illegal address access has been attempted.

 This scheme decreases the amount of memory needed to store each page table, it increases the amount of time needed to search the table when a page reference occurs. If the whole table is to be searched it takes too long.



Advantage:-
 Eliminates fragmentation.

 Support high degree of multiprogramming.

 Increases memory and processor utilization.

 Compaction overhead required for the re-locatable partition scheme is also eliminated.

Disadvantage:-
 Page address mapping hardware increases the cost of the computer.

 Memory must be used to store the various tables like page tables, memory map table etc.

 Some memory will still be unused if the number of available block is not sufficient for the address space of the jobs to be run.

3 (b) Define deadlock and explain the necessary conditions for deadlock situation to occur.

DEADLOCKS:-
- When processes request a resource and if the resources are not available at that time the process enters into waiting state. Waiting process may not change its state because the resources they are requested are held by other process. This situation is called deadlock.
- The situation where the process waiting for the resource i.e., not available is called deadlock.

Necessary Conditions:-

A deadlock situation can occur if the following 4 conditions occur simultaneously in a system:-

1. Mutual Exclusion:-

Only one process must hold the resource at a time. If any other process requests for the resource, the requesting process must be delayed until the resource has been released.

2. Hold and Wait:-

A process must be holding at least one resource and waiting to acquire additional resources that are currently being held by the other process.

3. No Preemption:-

Resources can"t be preempted i.e., only the process holding the resources must release it after the process has completed its task.

4. Circular Wait:-

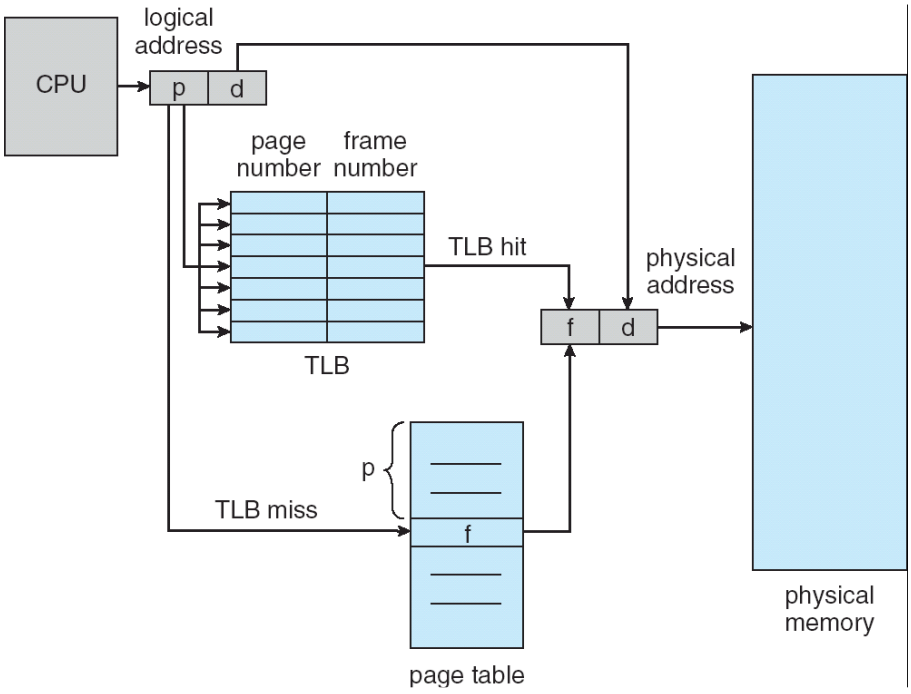
A set {P0,P1……..Pn} of waiting process must exist such that P0 is waiting for a resource i.e., held by P1, P1 is waiting for a resource i.e., held by P2. Pn-1 is waiting for resource held by process Pn and Pn is waiting for the resource i.e., held by P1.

All the four conditions must hold for a deadlock to occur.

4 (a) Explain with the help of supporting hardware diagram, how the TLB improves the performance of a demand paging system.

The only solution is to use special, fast, lookup hardware cache called translation look aside buffer [TLB] or associative register.

 TLB is built with associative register with high speed memory. Each register contains two paths a key and a value.



• When an associative register is presented with an item, it is compared with all the key values, if found the corresponding value field is return and searching is fast.
TLB is used with the page table as follows:-
 TLB contains only few page table entries.
 When a logical address is generated by the CPU, its page number along with the frame number is added to TLB. If the page number is found its frame memory is used to access the actual memory.

 If the page number is not in the TLB (TLB miss) the memory reference to the page table is made. When the frame number is obtained use can use it to access the memory.

 If the TLB is full of entries the OS must select anyone for replacement.

 Each time a new page table is selected the TLB must be flushed [erased] to ensure that next executing process do not use wrong information.

 The percentage of time that a page number is found in the TLB is called HIT ratio.

(b) What do you mean by Belady's anomaly? Which page replacement algorithm suffers from Belady's anomaly?

For some page replacement algorithm, the page fault may increase as the number of allocated frames increases. FIFO replacement algorithm may face this problem.

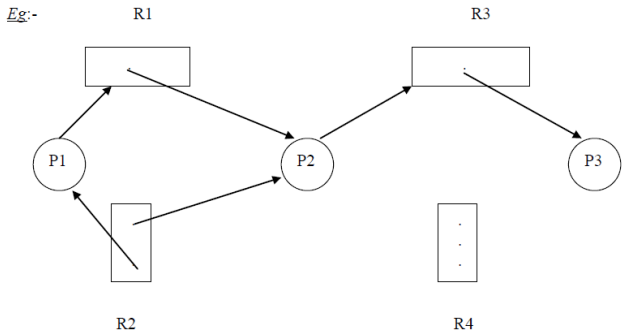
5 (a) Explain with suitable diagram how deadlock can be described using: i) RAG (resource allocation graph) ii) Wait-for-graph.

Resource Allocation Graph:-

➔ Deadlocks are described by using a directed graph called system resource allocation graph. The graph consists of set of vertices (v) and set of edges (e).

➔ The set of vertices (v) can be described into two different types of nodes P={P1,P2……..Pn} i.e., set consisting of all active processes and R={R1,R2…….Rn}i.e., set consisting of all resource types in the system.
➔ A directed edge from process Pi to resource type Rj denoted by Pi→ Ri indicates that Pi requested an instance of resource Rj and is waiting. This edge is called Request edge.

➔ A directed edge Ri→ Pj signifies that resource Rj is held by process Pi. This is called Assignment edge.



➢ If the graph contain no cycle, then no process in the system is deadlock. If the graph contains a cycle then a deadlock may exist.

➢ If each resource type has exactly one instance than a cycle implies that a deadlock has occurred. If each resource has several instances then a cycle do not necessarily implies that a deadlock has occurred.

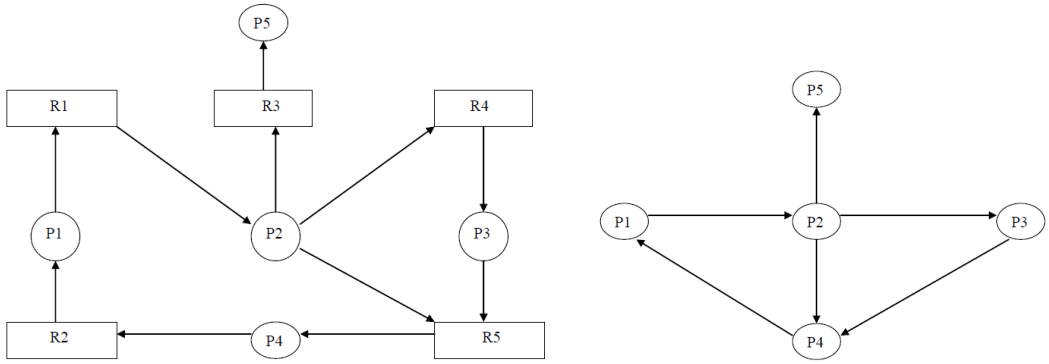
Single Instances of each Resource Type:-

➢ If all the resources have only a single instance then we can define deadlock detection algorithm that uses a variant of resource allocation graph called a wait for graph. This graph is obtained by removing the nodes of type resources and removing appropriate edges.

➢ An edge from Pi to Pj in wait for graph implies that Pi is waiting for Pj to release a resource that Pi needs.

➢ An edge from Pi to Pj exists in wait for graph if and only if the corresponding resource allocation graph contains the edges Pi→Rq and Rq→Pj.

➢ Deadlock exists within the system if and only if there is a cycle. To detect deadlock the system needs an algorithm that searches for cycle in a graph.



(b) Given five memory partitions of 100KB, 500KB, 200KB, 300KB and 600KB. How would the first fit, best fit and worst fit algorithms place processes of 212KB, 417KB, 112KB and 426KB(in order)? Which algorithm makes the most efficient use of memory?

7(a) Write shorts notes on: i) Internal and External fragmentation ii) Segmentation

Fragmentation:-

➢ Memory fragmentation can be of two types:-

- Internal Fragmentation
- External Fragmentation

➢ In Internal Fragmentation there is wasted space internal to a portion due to the fact that block of data loaded is smaller than the partition.

Eg:- If there is a block of 50kb and if the process requests 40kb and if the block is allocated to the process then there will be 10kb of memory left.

➢ External Fragmentation exists when there is enough memory space exists to satisfy the request, but it not contiguous i.e., storage is fragmented in to large number of small holes.

- External Fragmentation may be either minor or a major problem.
- One solution for over-coming external fragmentation is compaction. The goal is to move all the free memory together to form a large block. Compaction is not possible always. If the re-location is static and is done at load time then compaction is not possible. Compaction is possible if the re-location is dynamic and done at execution time.
- Another possible solution to the external fragmentation problem is to permit the logical address space of a process to be non-contiguous, thus allowing the process to be allocated physical memory whenever the latter is available.

Segmentation:-

Basic method:-

 Most users do not think memory as a linear array of bytes rather the users thinks memory as a collection of variable sized segments which are dedicated to a particular use such as code, data, stack, heap etc.

 A logical address is a collection of segments. Each segment has a name and length. The address specifies both the segment name and the offset within the segments.

 The users specifies address by using two quantities: a segment name and an offset.

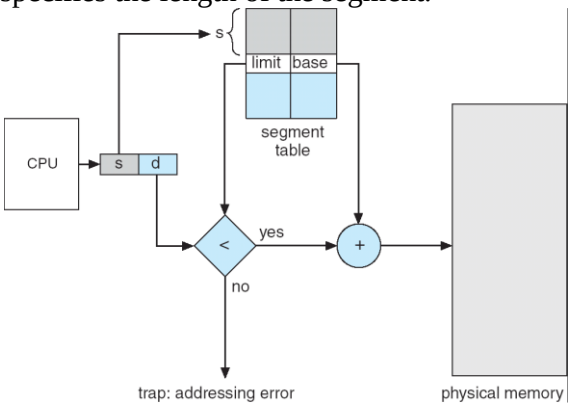
 For simplicity the segments are numbered and referred by a segment number. So the logical address consists of <segment number, offset>.

Hardware support:-

 We must define an implementation to map 2D user defined address in to 1D physical address.

 This mapping is affected by a segment table. Each entry in the segment table has a segment base and segment limit.

 The segment base contains the starting physical address where the segment resides and limit specifies the length of the segment.



The use of segment table is shown in the above figure:-

 Logical address consists of two parts: segment number,„s‟ and an offset,„d‟ to that segment.

 The segment number is used as an index to segment table.

 The offset ‟d‟ must bi in between 0 and limit, if not an error is reported to OS.

 If legal the offset is added to the base to generate the actual physical address.

 The segment table is an array of base limit register pairs.

solution_problem_1

Solution_problem_2