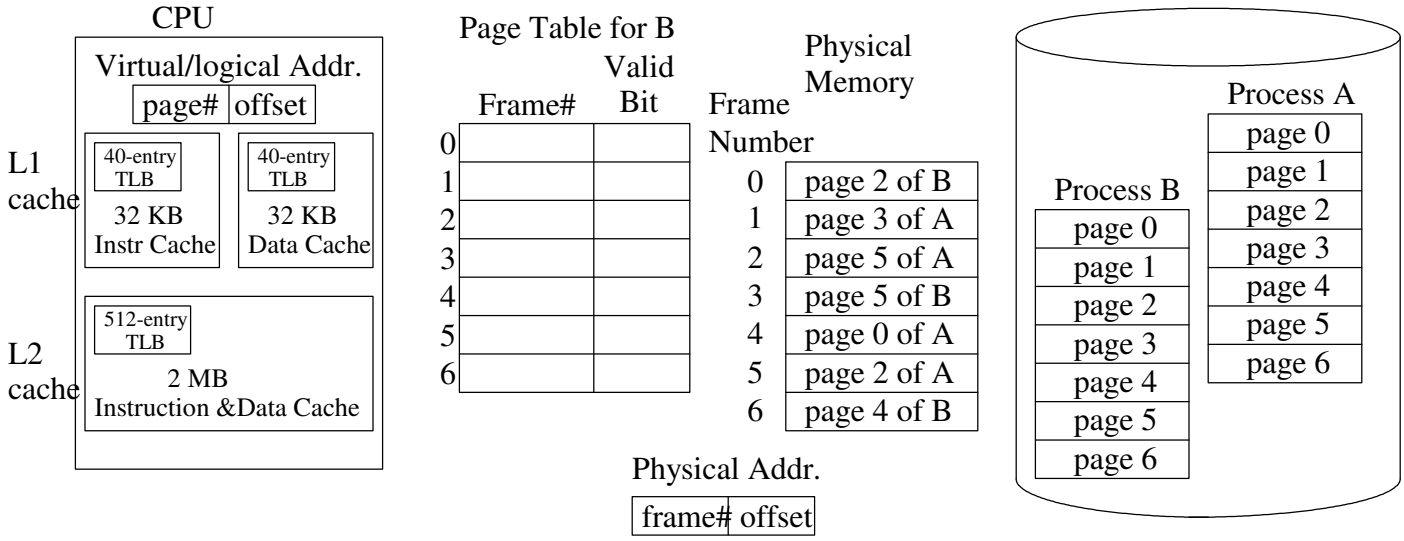
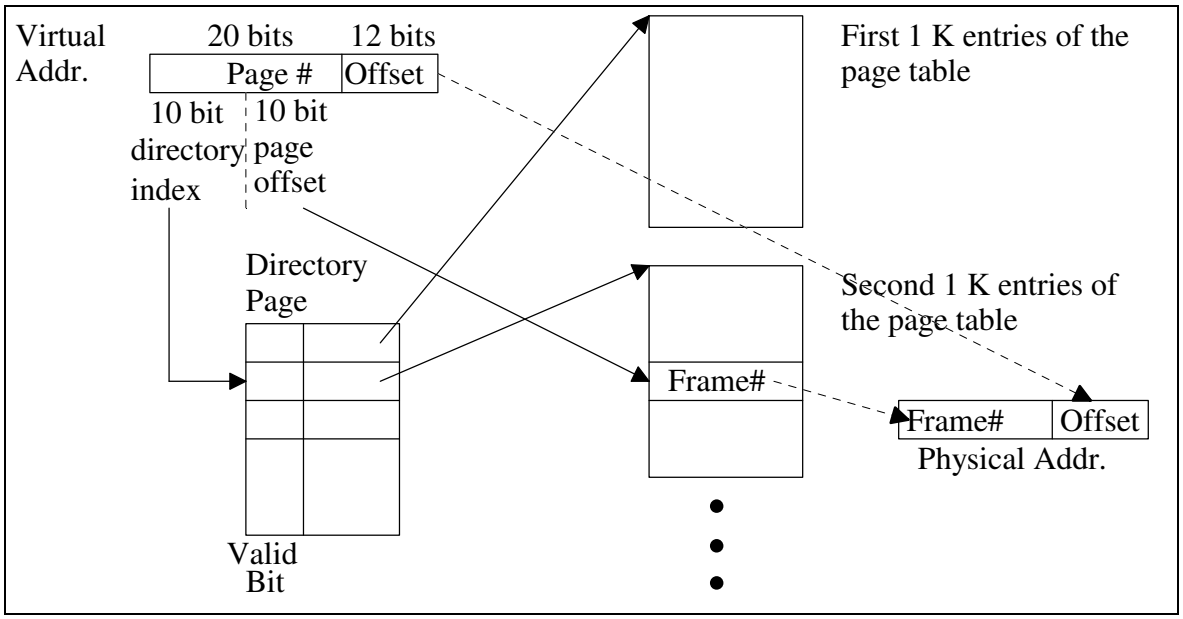
1. Consider the demand paging system with 1024-byte pages.



a) Why have multiple levels of cache?

b) Blocks in the L1 cache can be tagged with physical/actual memory addresses, or by virtual addresses. What would be the advantage of having blocks in the L1 cache be tagged with virtual addresses?

2. 32-bit computers typically had 4KB pages, 4 byte page table entries, and used two-level page tables where the first level (the "directory") acts as an index into the page table which is scattered across several pages.

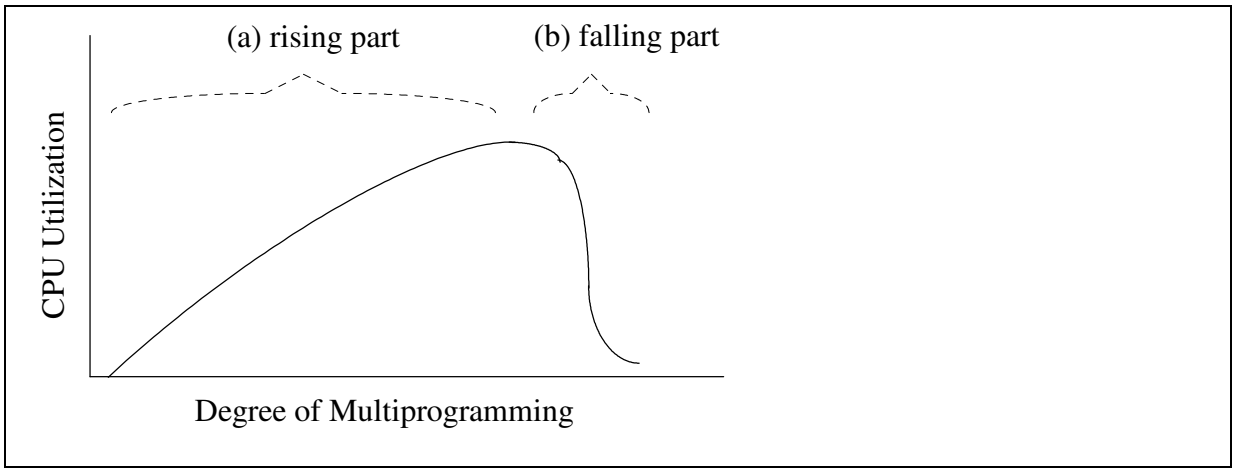


a) A 64-bit computer might not support a full 64-bit address space. How could a 3-level page table support 42-bit address space?

3. Design issues for Paging Systems

Conflicting Goals:

- Want as many (partial) processes in memory (high degree of multiprogramming) as possible so we have better CPU & I/O utilization ⇒ allocate as few page frames as possible to each process
- Want as low of page-fault rate as possible ⇒ allocate enough page frames to hold all of a process' current working set (which is dynamic as a process changes locality)



Explain the shape of each section indicated on the above curve:

a) (rising part of the curve)

b) (falling part of the curve)

4. There are many similarities between the cache-memory level and memory-disk level of the memory hierarchy, but there are also important differences. For example, a cache miss stalls the running program temporarily, but a page fault causes the running program to turnover the CPU to another program. Why are these cases treated differently by the computer system?

5. Complete the following table assuming the Optimal page-replacement algorithm and four page frames allocated to the process.

References String:	**1**	**2**	**3**	**4**	**1**	**2**	**5**	**1**	**2**	**3**	**4**	**5**
Page Frames Allocated = 4												

Number of page faults? Page-fault rate?

6. Complete the following table assuming a LRU page-replacement algorithm and four page frames allocated to the process.

References String:	**1**	**2**	**3**	**4**	**1**	**2**	**5**	**1**	**2**	**3**	**4**	**5**
Page Frames Allocated = 4												

Number of page faults? Page-fault rate?

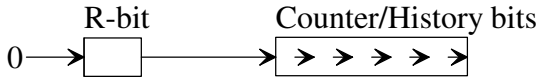
7. Answer the following questions about implementation of the LRU Algorithm:

a) What information would we need to keep track of to implement LRU?

b) When would this information need to be updated?

c) Where would you store this information?

8. To approximate the LRU page-replacement algorithm most hardware supports the updating/setting of a reference bit (R-bit) in the page-table entry corresponding to each memory reference. To get a better approximation of LRU, additional counter/history bits can be maintained. Periodically, say every 20 milliseconds, the process is interrupted so the OS can shift the R-bit into the counter/history bits and clear the R-bit. Such as



a) If the pages in main memory have the below R-bit and counter/history bits, then which page should be selected for replacement on a page fault?

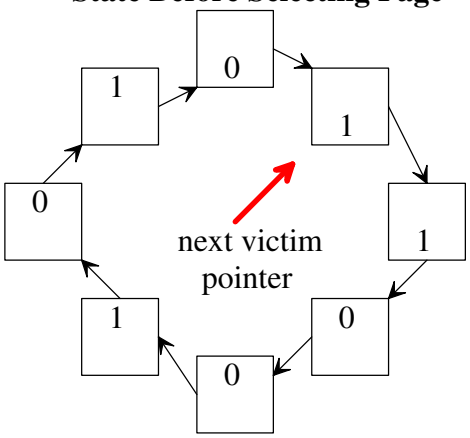
Page	R-bit	Counter/History bits
0	0	0 1 0 1 0 1 0
1	1	0 0 1 0 0 0 0
2	1	0 0 0 1 1 1 1
3	0	0 0 1 1 0 1 0
4	0	1 1 0 1 0 1 0

b) If the R-bits are shifted every 20 milliseconds and R-bits are about to be shifted because the interrupt just occurred, how long (specify a range) has it been since page 0 was referenced?

c) If a file read occurred during the middle of a 20 millisecond interval, the OS will switch to running another process. What should the OS do so the page-replacement algorithm can be resumed when the I/O completes and the process starts executing again?

9. Second-Chance/Clock Replacement - only store one counter/history bit per page-table entry
For the pages in memory, maintain a circular FIFO queue of pages

State Before Selecting Page



On a page fault, the pointer is moved until a page with a reference bit of 0 is found. This page is selected to be replaced. While the pointer is being moved, reference bits are cleared. As the pointer moves it leaves a trail of 0s in its wake. If one of these pages is referenced be for the pointer returns, it gets a "second change" and can remain in memory.

What page should be replaced