Virtual Memory API

• Most utilization of memory is automatic
  ◦ Program and data allocation.
  ◦ Stack management.

• Some are not
  ◦ malloc or new routines.
  ◦ memory mapped files.
  ◦ shared memory segments.
Virtual memory

virtual address → NMAP → physical address
Segmentation

\[
\begin{array}{|c|c|c|}
\hline
\text{segment} & \text{offset} & \oplus \\
\hline
\text{base address} & \text{length} & \text{access} \\
\text{base address} & \text{length} & \text{access} \\
\text{base address} & \text{length} & \text{access} \\
\text{base address} & \text{length} & \text{access} \\
\text{base address} & \text{length} & \text{access} \\
\hline
\end{array}
\]

if (offset > length) \textit{segment fault}
Paging

if (!valid) page fault
IA-32 architecture (simplified)
SUN UltraSPARC III (simplified)
Demand Paging

When a page fault occurs:

° Find a free page frame.
  • If none, then choose a page to evict.
° Read page from backing store into this frame.
° Assign page to this page frame.
° Resume process.

… which page should be evicted?
Page replacement policies

• FIFO
  Easy to implement, but poor performance.

• Optimal: evict page that will be next referenced *farthest in the future*.
  Need ability to predict the future.

• LRU: using the past to predict the future.
  Too high overhead.

• Clock: LRU approximation.
CLOCK

1. Advance CLOCK hand
2. Test and clear use-bit
3. Use-bit
   - Set
   - Clear
4. Dirty-bit
   - Set
   - Clear
5. Schedule page for cleaning
6. Replace page
Miss ratio for different policies
FIFO miss ratio (detail)
Working set

The only reason that virtual memory works at all in practice is because of the amount of locality there is in address references.

- The page fault penalty is $\sim 10^5$
- If page faulted on only 1 out of ten references, then the program would run 10,000 times slower.
Define the working set of a process's reference string $\omega$ at time $t$ for a window $\tau$ to be the pages $\omega(t)$, $\omega(t-1)$, ... $\omega(t-\tau+1)$.

$$\omega = \langle 1; 2; 3; 4; 5; 4; 5; 4; 5; 4; 5 \rangle$$
Working set, III
Working set policy, I

A process can run only if its working set is resident.

Example:

° UNIX can (temporarily) remove a process from the ready set if the page fault rate is too high.
  • Choose a process that has been sleeping a long time.
  • If none, then choose the longest resident of the four largest processes.
Working set policy, II

One can modify CLOCK to approximate a working set policy.

- Let PT be the current process time.
- Each frame $f$ is examined as usual.
  - If the use bit set, then $LR[f] = PT$.
  - If the use bit isn't set, then $f$ is replaceable if $PT - LR[f] > \tau$. 