Basics of Paging in UNIX and in Windows 2000

UNIX

- UNIX uses *demand paging*, i.e., pages are brought into main memory when they are referenced (no prepaging).
- Paging is implemented by the kernel and the *paging daemon*. The daemon periodically checks the *list of free frames*. If the number of free page frames is below *minfree*, then it initiates action to free up pages so that there would be *lotsfree* available page frames.
- Page replacement is done by the *two-handed clock* algorithm using global page allocation. The first hand clears the reference bit. If the second hand finds the reference bit clear, then the page is evicted, and the frame is included in the list of free frames. The parameters *scanrate* and *handspread* determine the speed of page eviction.
- If the paging rate is too high, then the *swapper* evicts some processes that have not been used recently. It also brings in (page tables of) processes from disk — it prefers small processes that have been waiting for long.

Windows 2000

- Win2000 uses the working set model. Minimal (min) and maximal (max) number of pages are allocated to each process. Initially they are the same for all processes, but they change through time according to the number of page faults caused by the process.
- At page fault a new page is added to the working set — if the working set is bigger than max., then a page of that process is evicted first.
- The *balance manager* periodically checks the number of free pages. If it is too low, the *working set manager* evicts unreferenced pages belonging to processes with working sets above min.
- There is a *swapper thread* as well to swap processes between main memory and disk.