Programming project 2 (final project description) — due on Thursday 30 November 2000

The aim is to examine virtual memory management algorithms employed by the operating system.

Reference traces

Obtain traces of memory references and corresponding page reference traces of benchmark programs, as described in earlier handouts.

Page replacement algorithms

Implement the following page replacement algorithms:

- OPT (also known as MIN), described in Stallings, and in: Belady Laszlo, "A Study of Replacement Algorithms for a Virtual Storage Computer", IBM Systems Journal, 1966.
- FIFO (first-in, first-out), described in Stallings
- LRU (least-recently used), described in Stallings
- MRU (most-recently used)
- Clock, as described in Stallings (1 bit)
- Clock, with preference for clean pages, as described in Stallings

For extra credit, implement:

- SEQ, as described in: Gideon Glass and Pei Cao, "Adaptive Page Replacement Based on Memory Reference Behavior", Sigmetrics 1997.
- EELRU, as described in: Yannis Smaragdakis, Scott F. Kaplan, and Paul R. Wilson, "EELRU: Simple and Effective Adaptive Page Replacement", Sigmetrics 1999, available at http://www.cs.amherst.edu/ sfkaplan/papers/eelru.ps.gz
- Your own page replacement algorithm

Experiments

Run each page replacement algorithm on each page reference trace, simulating a local replacement policy with fixed allocation (according to Stallings' nomenclature).

Vary the number of page frames allocated N. (For each trace, first determine and report the interesting range for N, then run with some suitable number of discrete values of N covering that range.)

For algorithms that have tunable parameters, vary those as well.

Document

To be complete, your document must detail:

- (3) the benchmark suite: which programs; what do they do; which inputs; how long does the program run and how much space does it use, etc.

- (3) the tracing tools used; how do they work; how are they used; what is being traced
- (3) characteristics of the page reference traces (including summary statistics, page usage, and a figure like Figure 8.1 in Stallings)
- (6) how each page replacement algorithm is implemented (describe data structures used and discuss their efficiency)
- (9) statistics of running various page replacement algorithms
  - as tables showing how performance parameters depend on the choice of algorithm and the trace (the primary performance parameter is the number of page faults, but you must also consider the total number of disk page reads and writes)
  - as figures, one for each trace, similar to Figure 8.15 in Stallings
  - as a summary figure for all traces
- (6) discussion of the results
  - a quantitative comparison of different page replacement algorithms
  - a comparison, or categorization, of the several benchmark programs, with reference to the nature of the programs

The document must not exceed 30 pages. It should be printed double-sided and stapled.

Handing in your project

You must hand in hard copy. You can hand in the document in class on Thursday 30 November 2000, or during office hours later that day, or to the TA thereafter. Email submissions will not be accepted. No deadline extensions will be granted. The grade for late submissions will be reduced by $3n^2\%$ (except for the milestones component).

Grading

Regular credit:

- 24 points: 8 points for each of the three milestones
- 6 points for correctly produced memory reference traces
- 6 points for correctly produced corresponding page traces
- 18 points for correct implementation of page replacement algorithms: 3 points each
- 46 points for the document, of which 30 as indicated in parentheses above, and 16 for structure and readability

Extra credit:

- up to 20 points: 10 points for each of SEQ and EELRU
- up to 30 points: for your own page replacement algorithm, depending on the quality of the algorithm (does it achieve low page fault rates; is it feasible in a real system)