Operating systems   
Lab 3 - Report  
Replacement Policies

Nikolay Stoitsov

Adriana Bostandzhieva

Table of Contents

[Replacement policies 2](#_Toc452112015)

[Description 2](#_Toc452112016)

[Solution and Results 2](#_Toc452112017)

# Replacement policies

## Description

The aim of the task is to implement and analyse the performance of 4 different replacement policy algorithms in a paging system – Random, FIFO, LRU and Clock.

## Solution and Results

The different algorithms are implemented in four subclasses of class VMManager.

Class RandomVMManager implements a random replacement policy. If replacement is needed a random page in memory is overwritten. Output is shown in Figure1.

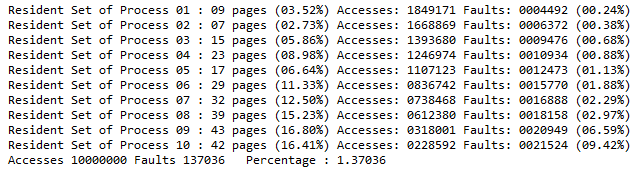
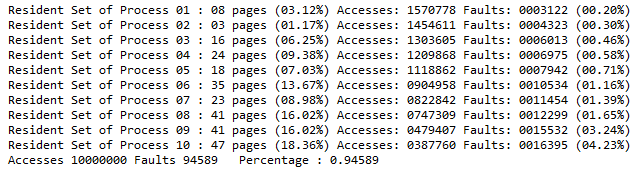
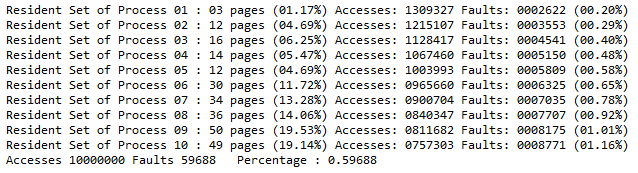


Figure . Random replacement policy

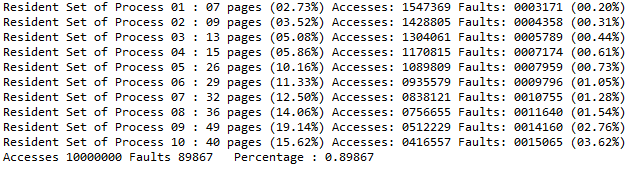
Class FIFOVMManager implements a replacement policy based on First-in-first-out principle, i.e. the oldest page in memory. This method is easy to implement but the performance can suffer because the page age in memory is not always an indicator of its future use. Output is shown in Figure2.

 Figure 2. FIFO replacement policy

Class LRUVMManager implements a replacement policy based on Least Recently Used page in memory. Because of the principle of locality the page fault rate tends to be the minimum but there is a big overhead in the implementation due to keeping track on the usage and computation of the LRU page. Output is shown in Figure3.

 Figure 3. LRU replacement policy

Class ClockVMManager implements a replacement policy based on Clock algorithm. A use bit is set to 1 every time a frame is accessed. When replacement is needed we search for the next 0 use bit, zeroing all 1 use bits on the way. The first page with 0 use bit met is replaced and that bit is set to 1. Output is shown in Figure4.

 Figure 4. Clock replacement policy

We can see from the obtained results that the different algorithms give different page fault rate performances. As expected, the LRU algorithm has the lowest page fault rate. Not surprisingly the random is with highest fault rate since it has the poorest algorithm (if it can be called algorithm at all).

