HÁSKÓLI ÍSLANDS
Iðnaðarverkfræði-, vélaverkfræði- og tölvunarfræðideild

TÖL401G: Stýrikerfi / Operating Systems · Vormisseri 2022

Assignments 19–20 · Due 29.3.2022, 13:00

Assignment 19

In a dynamic memory management system that is responsible for contiguous-memory allocation, differently sized “holes” of free memory are available in the following order (from left to right): 15 KB, 10 KB, 26 KB, 30 KB, 21 KB, 13 KB, and 17 KB. Now, contiguous-memory m_i of size

$$m_1=16\text{ KB},\ m_2=8\text{ KB},\ m_3=20\text{ KB},\ m_4=14\text{ KB and } m_5=5\text{ KB}$$

is requested in consecutive order.

1. Draw a table or chart (like the example on slide 8-30) to represent these memory holes and to show into which of these holes each of the strategies *First Fit*, *Next Fit*, *Best Fit*, and *Worst Fit* would allocate these requests.

   Note: It is easiest to use a spreadsheet for filling in the solutions:

   If you have a Google account, you can work online on a copy of a template: `https://docs.google.com/spreadsheets/d/1jJBDoErbeSwmP928KT62n0jWf06F_1lbRLVemVXPVXo/copy`

   If you have no Google account, you view and download a copy a template from here: `https://docs.google.com/spreadsheets/d/1jJBDoErbeSwmP928KT62n0jWf06F_1lbRLVemVXPVXo/edit?usp=sharing`

2. How many “check” steps (i.e. investigating a hole to check whether it fits) does each of the four allocation strategies need for placing all the requests?

   Assume that a very basic data structure is used: a list of free memory “holes” which uses always the above initial ordering (i.e. no optimisation, such as search trees, made to reduce the number of steps to search for the biggest or best hole). If a hole shrinks, this entry remains at its list location, just the size gets updated.

   Hint: checking whether m_i fits into the first hole is one check step, checking then the next hole is another step, and so on.

   Concerning *Best Fit* and *Worst Fit* think about a way for implementation that keeps track of the hole with best (or worst) fit so far and only updates this information when a better (or worse) fit is found, so that in the end that information can be used to place a memory request into that hole without additional search steps.

Assignment 20

Consider a system with a Paged Memory Management Unit (PMMU) that supports 10 bit physical addresses, however a process is only able to use 8 bit logical addresses.[1] The page size is 64 byte, the word size is 1 byte.

1. Into how many pages is the logical address space divided?
2. Into how many frames is the physical address space divided?
3. What is the maximum *degree of multiprogramming* (i.e. how many programs can be loaded into memory →chapter 3) if no swapping (or demand paging) is used in this system (assuming that all the physical address space is available for the programs that may use their whole logical address space)?
4. Is it possible to make a statement about how many processes can be in the system at the same time if swapping (or demand paging) is used? Justify your answer!
5. How many entries does a page table have in this system?
6. How many bits are needed for each page table entry if each page table entry consists of just the minimal required number of bits for the frame number and the valid bit?

Now, assume that the first three frame numbers stored in the page table are: 0, 5, 10:

7. What is the physical address of logical address 42?
8. What is the physical address of logical address 191?

Note: All numbers are decimal numbers, not hexadecimal!

Preparation

Read chapter 8 as preparation for class this week

Examples from each chapter will in videos that are available in Canvas via the tab "Panopto".

Report via Piazza any questions that you have, so that we can clarify this during class or directly via Piazza.

[1] While it sounds strange to have more physical memory than logical memory, this was actually the case with the Intel 8086 processor used in the first PCs: it had 16 Bit address registers, but a 20 Bit address bus: using a segmenting MMU, the logical address space was then mapped to the (larger) physical address space.