Project Ver_C

Title: Page Placement & Comparative Study between Dynamic Allocation

تحذير هام: علي الطالب عدم كتابة اسمه أو كتابة اي شيء يدل علي شخصيته

Answer Structure

1. Methods

Both of the **First Fit** and **Best Fit** strategies is **placement algorithms** that is a **Dynamic Partitioning** algorithm in which each process is allocated exactly as much memory as required.

Note: in file **uheap.c** there is a List of struct page that represents the user heap memory.

```
struct Page {
int state;       //the state of the page (Empty or not).
int sizeInPages; //the number of pages allocated starting from the current page.
};
struct Page MEMORY_LIST[(USER_HEAP_MAX-USER_HEAP_START)/PAGE_SIZE];

//index of the last allocated start page in the user heap.
int LAST_ALLOCATED_INDEX = 0;
```

1.1 First Fit

- Description:

in this strategy scan the memory from the beginning and allocate the first empty space that is suitable for the given process.

- The way of keep tracking with the free and allocated spaces in memory from the user side:

Each place in MEMORY_LIST there is a state, that state equals 0 if the page is empty and equals 1 if the page is allocated, and there is a number of allocated pages (allocated size) starting from the chosen page.

- How you find the suitable allocation space?

1. Loop on MEMORY_LIST starting from its beginning.
2. If the current page is not empty, then exceed its allocated size and go to the next start page.
3. If the current page is empty, then start looping from it until reaching the given size to allocate.
4. If the current space is not enough for the given size, then continue looping from the next start page and repeat from step 2.
5. If found enough space for the given size, then set the state of starting page with 1, and its number of pages with the allocated size.
6. If reached the end of MEMORY_LIST and no enough space is found return NULL.

1.2 Best Fit

- Description:

in this strategy scan the whole memory from the beginning to the end and allocate the best suitable empty space with the least external fragmentation between pages.

- The way of keep tracking with the free and allocated spaces in memory from the user side:

In addition to the state of each page and the allocated size, there is a global integer variable that represents index of the end of the last allocated block.

This index is used to know if the heap is empty or if there are next allocated space, to reduce the time of searching for the best suitable space.

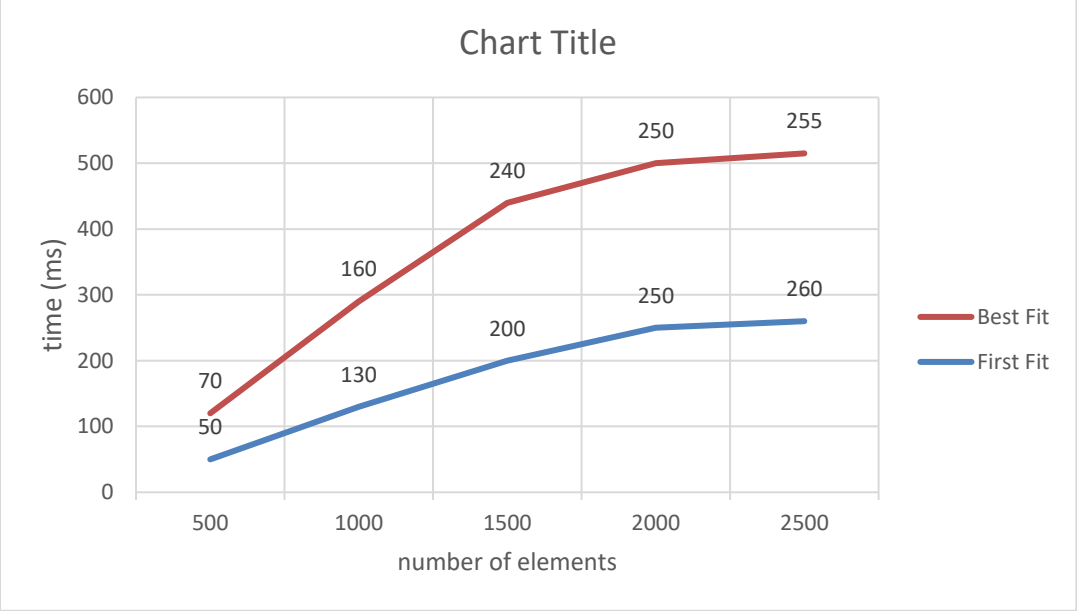
- How you find the suitable allocation space?

1. Loop on MEMORY_LIST starting from its beginning until reaching an empty page.
2. If the current page is not empty exceed its size and continue looping.
3. Else, Start looping from this page until reaching the given size.
4. If the current space is not enough then continue looping starting from the next starting page, then repeat from step 2.
5. If reached an enough space for the given size, then calculate the remaining space (number of remaining pages) from the found enough space to the next allocated space.
6. If the remaining space < the minimum calculated space, then set the current remaining space as minimum space.
7. If reached the end of MEMORY_LIST and no suitable space found, then return NULL.
8. If reached the end of MEMORY_LIST and there is suitable space found, then allocate the block with minimum remaining space (minimum fragmentation).

9. Else, continue looping from the next starting page, then repeat from step 2.

2. Results

2.1 Graph 1:



3. Discussion

3.1 Graph 1:

3.1.1 Which is best & why

The **First Fit** strategy is the best because:

- It is faster than the other as it usually requires scanning just a part of the memory.
- Allocation of the blocks is near of the beginning of the memory.

3.1.2 Which is worst & why

The **Best Fit** strategy is the worst because:

- It takes longer time because it requires to scan all of the memory.
- It causes a large number of small fragmentations.

3.2 Mention whether there is a large difference in time between the two strategies or not? Why?

Yes, there is a large difference in time between both of the 2 strategies, and this is due to that the **Best Fit** strategy requires to scan all of the memory to find the best suitable available space with least fragmentation, unlike the **First Fit** strategy which requires to scan the memory just to find the first suitable available space.

3.3 Suggestions (if exist)