|  |
| --- |
| Name: Phạm Mai Dung  ID: 19520477  Class: IT007.L21.1 |

OPERATING SYSTEM  
LAB 6’S REPORT

**SUMMARY**

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | | **Status** | **Page** |
| Section 1.5 | Task name 1 | Done | 2 – 18 |
| FIFO | Done | 8 – 10 |
| LRU | Done | 11 – 14 |
| OPT | Done | 15 – 18 |
| Task name 2 | Done | 19 |
| Task name 3 | Done | 19 – 22 |

**Self-scrores: 9**

*\*Note: Export file to* ***PDF*** *and name the file by following format:* ***LAB X – <Student ID>.pdf***

# Section 1.5

## Task name 1: Sử dụng ngôn ngữ lập trình C viết chương trình mô phỏng các giải thuật thay thế trang đã nêu trong câu số 3 mục 6.3.3 với các yêu cầu trong [Lab6.pdf](Lab%206.pdf) .

* Thực hiện chương trình:

Diagram

Description automatically generated+ Lưu đồ chung:

Hình 1 \_ Lưu đồ thực hiện chương trình

Text

Description automatically generatedText

Description automatically generated+ Source code:

Text

Description automatically generatedText

Description automatically generatedText

Description automatically generated

Text

Description automatically generatedText

Description automatically generatedText

Description automatically generated

Text

Description automatically generatedText

Description automatically generatedText

Description automatically generatedText

Description automatically generated

Text

Description automatically generated

Hình 2\_ Source code chương trình các giải thuật thay thế trang

1. **Giải thuật thay thế trang FIFO:**

Diagram

Description automatically generated+ Lưu đồ giải thuật:

Hình 3\_ Lưu đồ FIFO Algorithm

+ Giải thích:

* Trước khi duyệt mảng pageRequests để các trang đi vào bộ nhớ chính, ta gán cho các phần tử của mảng pageFrames (có num\_Frames phần tử là số khung trang) giá trị -1.
* Duyệt qua các phần tử của mảng trang yêu cầu, nếu trang chưa có trong khung thì đặt cờ avail1 = 0 để thêm trang đó vào khung, đồng thời đánh dấu lỗi trang, đếm lỗi trang lên 1, “index = (index+1)%num\_Frames” sẽ xét khi khung đã đầy thì trang mới vào tiếp theo sẽ thay cho trang vào trước nhất trong số num\_Frames. (ví dụ có 3 khung trang, duyệt qua i sẽ là 0, 1, 2, sau khi frames[2] = pages[2] thì index = (2 + 1)% 3 = 0 thì sẽ thay trang tại frames [index == 0], vòng lặp tiếp theo sẽ thay tại frames[index == 1]). Sau đó xuất ra khung trang và đánh dấu \* để báo hiệu xảy ra lỗi trang. Nếu trang đó đã có trong khung thì cờ avail = 1, không có lỗi nên chỉ cần xuất ra khung trang.
* Khi đã duyệt hết các phần tử mảng pageRequests, xuất ra tổng số lỗi trang là tổng số faults đã đếm được và kết thúc chương trình.

+ Kết quả:

* A picture containing diagram

  Description automatically generatedText

  Description automatically generatedChuỗi mặc định: [19520477 + 007]:

Hình 4 \_ Kết quả khi thực hiện FIFO với 3 frames cho chuỗi mặc định.

* A picture containing text

  Description automatically generatedText

  Description automatically generatedNhập chuỗi: (chuỗi trong ví dụ của bài lab):

Hình 5 \_ Kết quả thực hiện FIFO với 3 frames cho chuỗi đã nhập.

1. **Giải thuật thay thế trang LRU:**

Diagram

Description automatically generated+ Lưu đồ giải thuật:

Hình 6\_ Lưu đồ LRU Algorithm

+ Giải thích:

* Trước khi duyệt mảng pageRequests để các trang đi vào bộ nhớ chính, ta gán cho các phần tử của mảng pageFrames (có num\_Frames phần tử là số khung trang) giá trị -1; mảng queue (có num\_Frames phần tử) sẽ lưu thời điểm pages đi vào frames. Thời điểm sẽ tăng dần theo số pages vào, được đếm bằng biến count. Đặt các cờ avail1, avail2 = 0.
* Duyệt qua các phần tử của mảng trang yêu cầu, nếu trang chưa có trong khung (avail1 = 0) thì thêm trang đó vào khung, đồng thời đếm lỗi trang và biến lưu thời điểm lên 1, gán thời điểm cho phần tử j của mảng queue. Đặt cờ avail2 = 1.
* Nếu cờ avail2 = 0, mảng frames đầy mà trang mới vào lại không có trong frames, tìm thời điểm nhỏ nhất trong mảng queue, gán phần tử có thời điểm nhỏ nhất đó cho pos. Sau đó tiếp tục tăng thời điểm, đếm lỗi lên 1, frames[pos] = pages[i] và queue[pos] = count (gán trang mới vào vị trí pos trong mảng frames và cập nhật thời điểm mới ở vị trí pos).
* Nếu trang đó đã có trong frames, cập nhật thời điểm và đặt các cờ bằng 1 (không xảy ra lỗi trang).
* Sau mỗi vòng lặp thay trang, in ra số trang đó và khung của nó, nếu có lỗi trang thì đánh dấu \* báo hiệu. Khi đã duyệt hết các phần tử mảng pageRequests, xuất ra tổng số lỗi trang là tổng số faults đã đếm được và kết thúc chương trình.

+ Kết quả:

* A picture containing text

  Description automatically generatedText

  Description automatically generatedChuỗi mặc định: [19520477 + 007]:

Hình 7 \_ Kết quả khi thực hiện LRU với 4 frames cho chuỗi mặc định.

* A picture containing diagram

  Description automatically generatedText

  Description automatically generatedNhập chuỗi: (chuỗi trong ví dụ của bài lab):

Hình 8 \_ Kết quả thực hiện LRU với 3 frames cho chuỗi đã nhập.

1. **Giải thuật thay thế trang OPT:**

Diagram

Description automatically generated+ Lưu đồ giải thuật:

Hình 9 \_ Lưu đồ OPT Algorithm

+ Giải thích:

* Trước khi duyệt mảng pageRequests để các trang đi vào bộ nhớ chính, ta gán cho các phần tử của mảng pageFrames (có num\_Frames phần tử là số khung trang) giá trị -1; mảng queue (có num\_Frames phần tử) sẽ lưu vị trí pages sau page hiện tại sẽ đi vào frames. Biến m sẽ cập nhật giá trị cho các phần tử mảng queue. Đặt các cờ avail1, avail2 = 0.
* Duyệt qua các phần tử của mảng trang yêu cầu, nếu trang chưa có trong khung (avail1 = 0) thì thêm trang đó vào khung, đồng thời đếm lỗi trang lên 1, đặt cờ avail2 = 1.
* Nếu cờ avail2 = 0, mảng frames đầy, đặt cờ avail3 = 0. Cập nhật mảng queue bằng cách duyệt qua các trang vào sau trang hiện tại, nếu trang đó sẽ được truy xuất lại thì gán vị trí trang đó(m) cho phần tử queue[ j ]. Duyệt queue, nếu trang không xuất hiện lại thì gán vị trí j cho pos. Nếu trang sẽ xuất hiện lại (avail3 == 0), tìm trang xuất hiện ở xa nhất trong mảng queue, gán vị trí của giá trị đó cho pos. Sau mỗi lần pos được cập nhật, gán trang mới vào vị trí pos trong mảng frames và đếm biến lỗi lên 1.
* Nếu trang đó đã có trong frames, đặt các cờ bằng 1 (không xảy ra lỗi trang).
* Sau mỗi vòng lặp thay trang, in ra số trang đó và khung của nó, nếu có lỗi trang thì đánh dấu \* báo hiệu. Khi đã duyệt hết các phần tử mảng pageRequests, xuất ra tổng số lỗi trang là tổng số faults đã đếm được và kết thúc chương trình.

+ Kết quả:

* A picture containing text, screenshot

  Description automatically generatedText

  Description automatically generatedChuỗi mặc định: [19520477 + 007]:

Hình 10 \_ Kết quả khi thực hiện OPT với 4 frames cho chuỗi mặc định.

* A picture containing diagram

  Description automatically generatedText

  Description automatically generatedNhập chuỗi: (chuỗi trong ví dụ của bài lab):

Hình 11 \_ Kết quả thực hiện OPT với 3 frames cho chuỗi đã nhập.

## Task name 2: Nhận xét về mức độ hiệu quả và tính khả thi của các giải thuật FIFO, OPT, LRU.

* Giải thuật OPT là giải thuật tối ưu nhất, ít xảy ra lỗi trang nhất trong 3 giải thuật trên nhưng OPT cũng là giải thuật bất khả thi nhất. Vì ta không thể biết người dùng sẽ yêu cầu truy xuất trang nào trước trang nào sau ở tương lai, thậm chí người dùng cũng không thể kiểm soát được việc đó.
* Giải thuật FIFO đơn giản, dễ lập trình, ít tốn tài nguyên máy nhưng không tối ưu vì lỗi trang nhiều và 1 vài trường hợp gặp nghịch lý Belady.
* 2 giải thuật LRU và OPT là 2 giải thuật tối ưu hơn nhưng phức tạp hơn vì cần nhiều tài nguyên máy và cài đặt thuật toán cũng phức tạp hơn.

## Task name 3: Nghịch lý Belady:

* Nghịch lý Belady: số page faults tăng mặc dù tiến trình đã được cấp nhiều frames hơn.
* A picture containing text

  Description automatically generatedNghịch lý này xuất hiện trong FIFO algorithm: (sử dụng ví dụ trong bài lab)

Hình 12 \_ Khi cấp 3 frames, giải thuật FIFO xảy ra 9 lỗi trang

A picture containing text

Description automatically generated

Hình 13 \_ Khi được cấp 4 frames, lỗi trang lại nhiều hơn (xảy ra 10 lỗi trang)

* A picture containing diagram

  Description automatically generatedTrong khi đó, OPT khắc phục được nghịch lý này:

Hình 14 \_ Giải thuật OPT chỉ xảy ra 7 lỗi trang khi cùng thực hiện 1 chuỗi trang yêu cầu với 3 frames được cấp.

A picture containing diagram

Description automatically generated

Hình 15\_ Khi được cấp 4 frames, lỗi trang ít hơn (xảy ra 6 lỗi trang) với giải thuật OPT