Computer Systems DV1
Datorsystem DV1

Final Lecture: Feedback and Exam

Arnold Pears

Lecture Outline

- Feedback on postit notes
- Extra material by request
  - Memory Management, virtual memory, thrashing
  - Synchronisation: semaphores, mutex, deadlock, deadlock resolution
  - Scheduling, process management in multi-tasking and realtime systems
- Exam structure and strategy

Postit notes: Difficult/Problematic

- More material today on
  - Synchronisation
  - Memory Management
  - Scheduling
- Exam structure
  - Structure and strategy discussion

Postit notes: Improvements

- Slides available in advance
  - Slides covering all the content are available linked from the course web. I have used some during the course
  - I have a firm belief in the learning benefits of reading and note taking, I am reluctant to change my approach

Postit notes: Improvements

- Laboratory exercises
  - Should be better integrated with the lectures
  - Are too small and artificial in nature
  - Dependency on C knowledge
  - Are not particularly well introduced
- Feedback
  - I will review the lab exercises before next year's course
  - Look at introducing C for OS prog in 2-3 lectures into the course
  - I will attend the introduction lectures for the lab.next year.

Postit notes: Improvements

- Lectures
  - Should be better integrated with the lab
  - Did not always cover all the reading
- Feedback
  - The issue of lab and lecture connection will be looked at before next year.
  - I take up the most vital areas in the lectures, you are expected to do some reading in your own time.

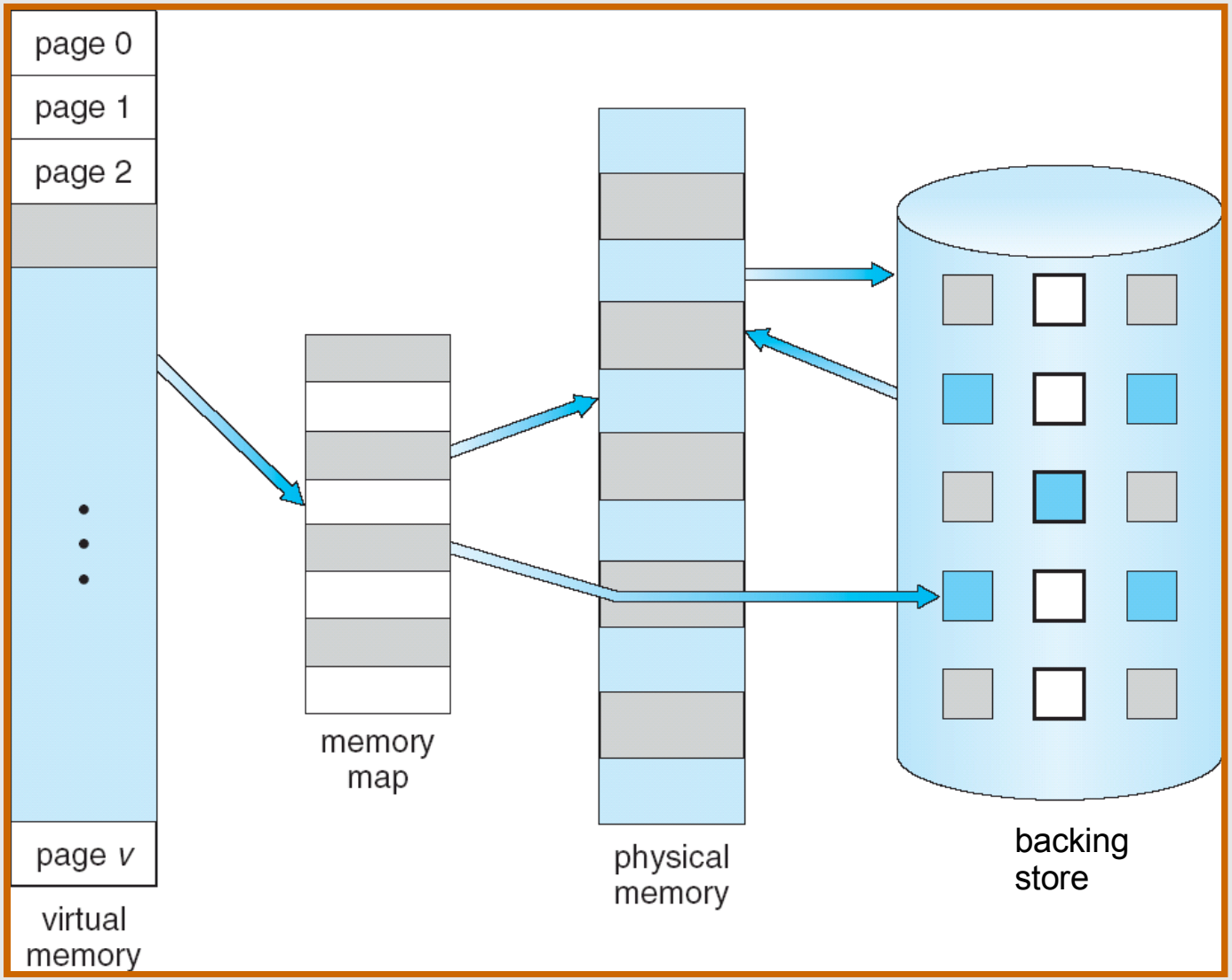
Postit notes: Improvements

- Exam
  - Past exams are needed for study
  - Worried that there will be very specific questions that require very detailed knowledge
- Feedback
  - Since I have not had the course before the solution is that I have put some practice questions on the web. Some of these questions will form the basis of about 50% of the exam points
  - There will be some questions about details, but they will not dominate the exam

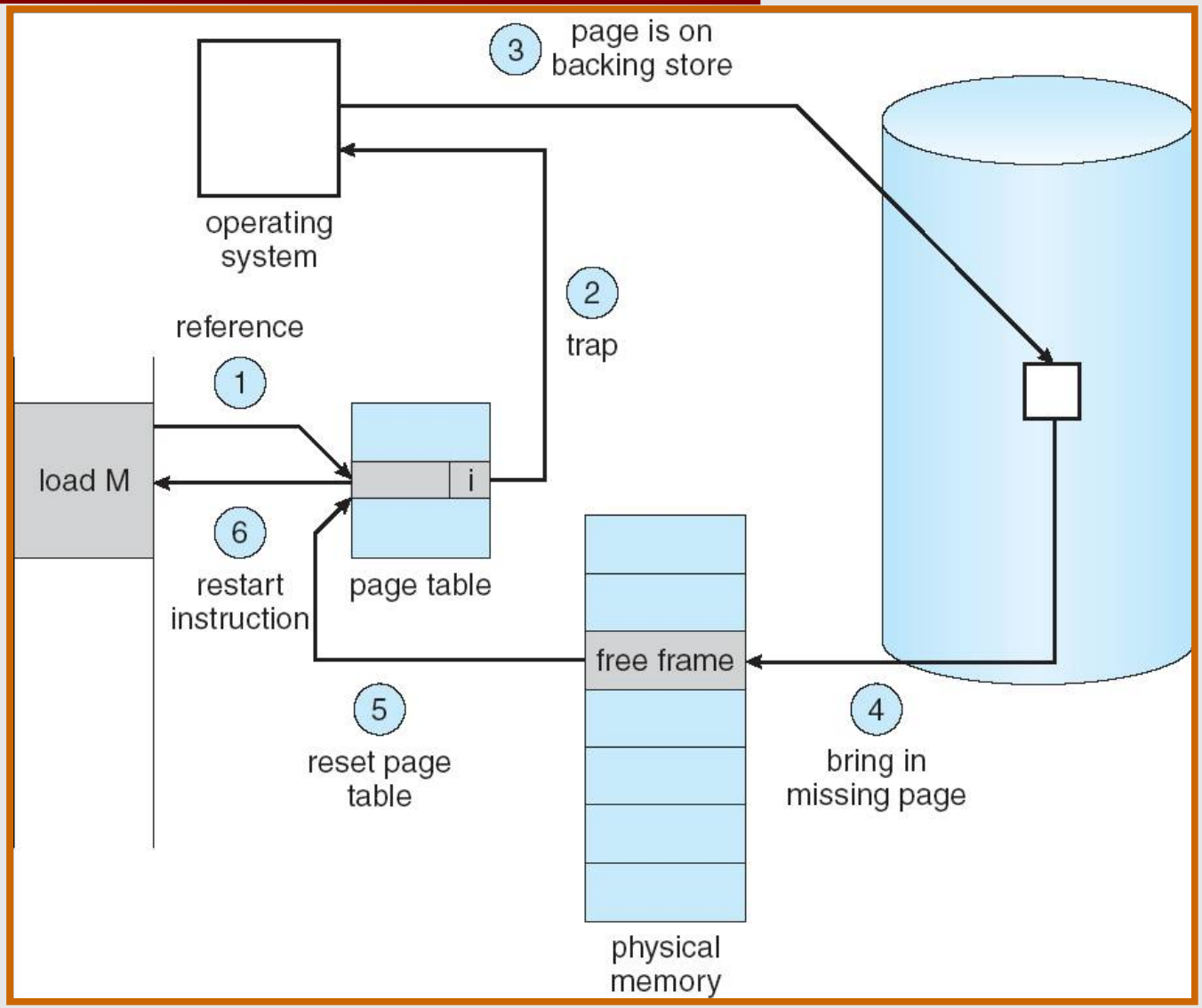
Request Material: Virt. Memory

- **Virtual memory** – separation of user logical memory from physical memory.
  - Only part of the program needs to be in memory for execution
  - Logical address space can therefore be much larger than physical address space
  - Allows address spaces to be shared by several processes
- Virtual memory can be implemented via:
  - Demand paging
  - Demand segmentation

Mapping virtual memory



Handling a page fault



Page replacement alg.

1. Find the location of the desired page on disk

2. Find a free frame:
   - If there is a free frame, use it
   - If there is no free frame, use a page replacement algorithm to select a **victim** frame

3. Bring the desired page into the (newly) free frame; update the page and frame tables
4. Restart the process

How to pick a victim?

- First In First Out (FIFO)
- Least Recently Used (LRU)
- Benchmark against an ideal algorithm where the page that will not be used for the longest time is always replaced.

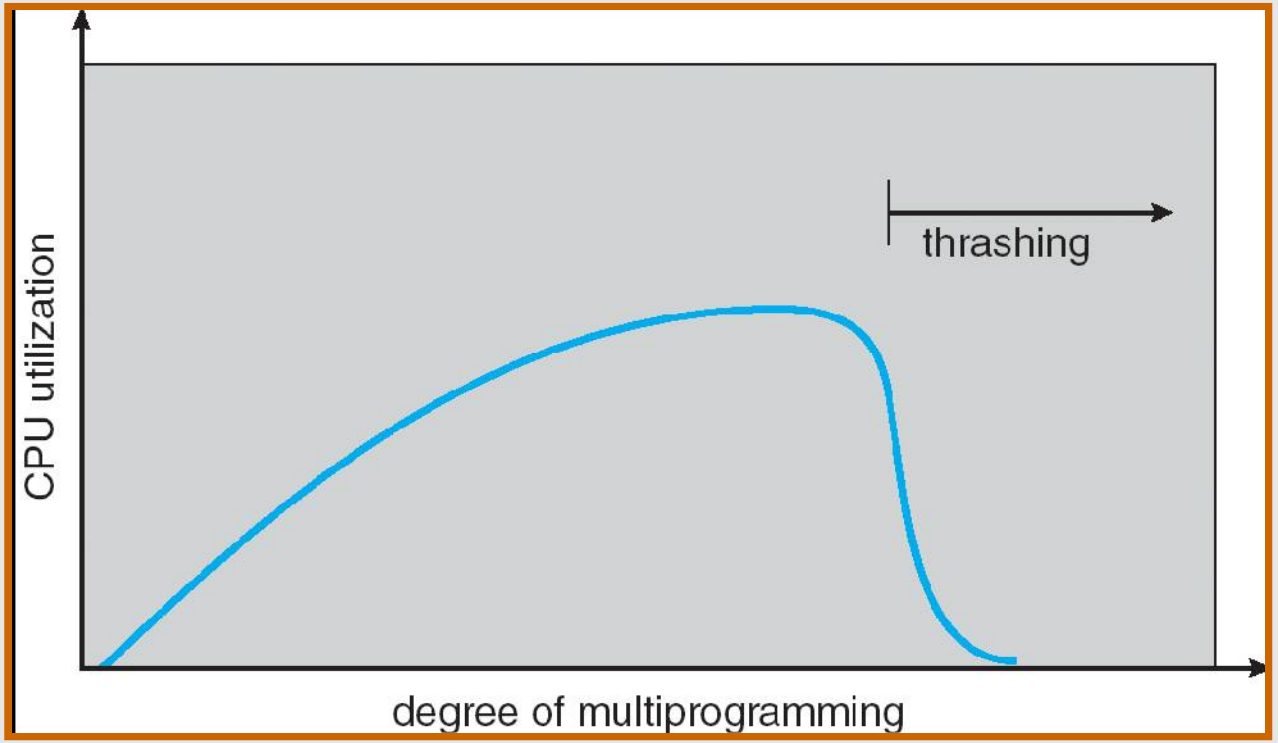
How many frames per process?

- Each process needs a *minimum* number of pages
- Two major allocation schemes
  - fixed allocation
    - Equal or proportional?
    - Local or global?
  - priority allocation
    - If process P_i generates a page fault,
      - select for replacement one of its frames
      - select for replacement a frame from a process with lower priority number

Thrashing

- If a process does not have “enough” pages, the page-fault rate is very high. This leads to:
  - low CPU utilization
  - operating system thinks that it needs to increase the degree of multiprogramming
  - another process added to the system
- **Thrashing -** a process is so busy swapping pages in and out it cannot do any useful computation

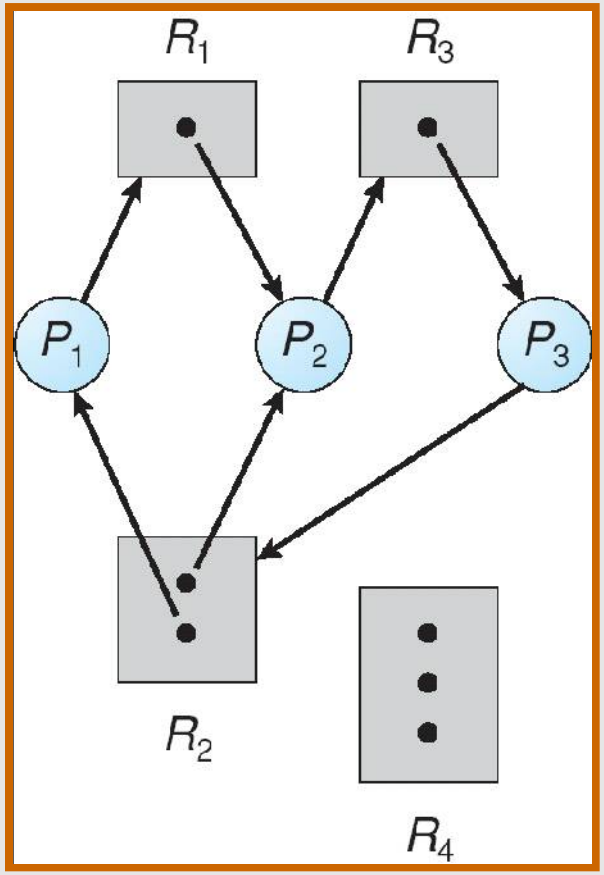
Thrashing



Request Material: Deadlock

- The necessary conditions for deadlock are:
  - Mutual exclusive access to resources
  - Hold and Wait, a process holds a resource is waiting for another resource request to be granted
  - No preemption, resources cannot be acquired preemptively by higher priority processes
  - Circular Wait on a requested resource sub-set.

Deadlock over resources



Banker's Algorithm

- Safety Algorithm

  Determine that there is a sequence of process executions in which all processes area able to acquire the resources they need and execute to termination

- Resource Allocation Algorithm – to determine if a resource request can granted

  1) Provisional allocation
  2) Run the safety algorithm on the new resource state
  3) If safe then

     confirm allocation

     else

     block request, waiting for resources to be released

Bankers Algorithm

- Handle multiple instances of resources
- Relies on
  - Public disclosure of maximum resource requirements
  - Computation of safety before confirming a resource request

Bankers Algorithm

- Data structures

  Available[j]= k, k instances of R_j

  Max[i][j]= n, P_i can request at most n instances of R_j

  Need[i][j]= s, P_i needs s additional instances of R_j to complete

  Alloc[i][j]= q, P_i is currently allocated q instances of R_j

Banker's Algorithm

1. Work = Avail and Finish[i]= false, $\forall i \in [0,n)$
2. if $\exists i$ s.t. finish[i]= false $\wedge$ $Need_i \leq$ Work then

    Work = Work + $Alloc_i$, Finish[i]= true, goto 2

else

    if Finish[i]= true, $\forall i \in [0,n)$ then

        SAFE, allow allocation

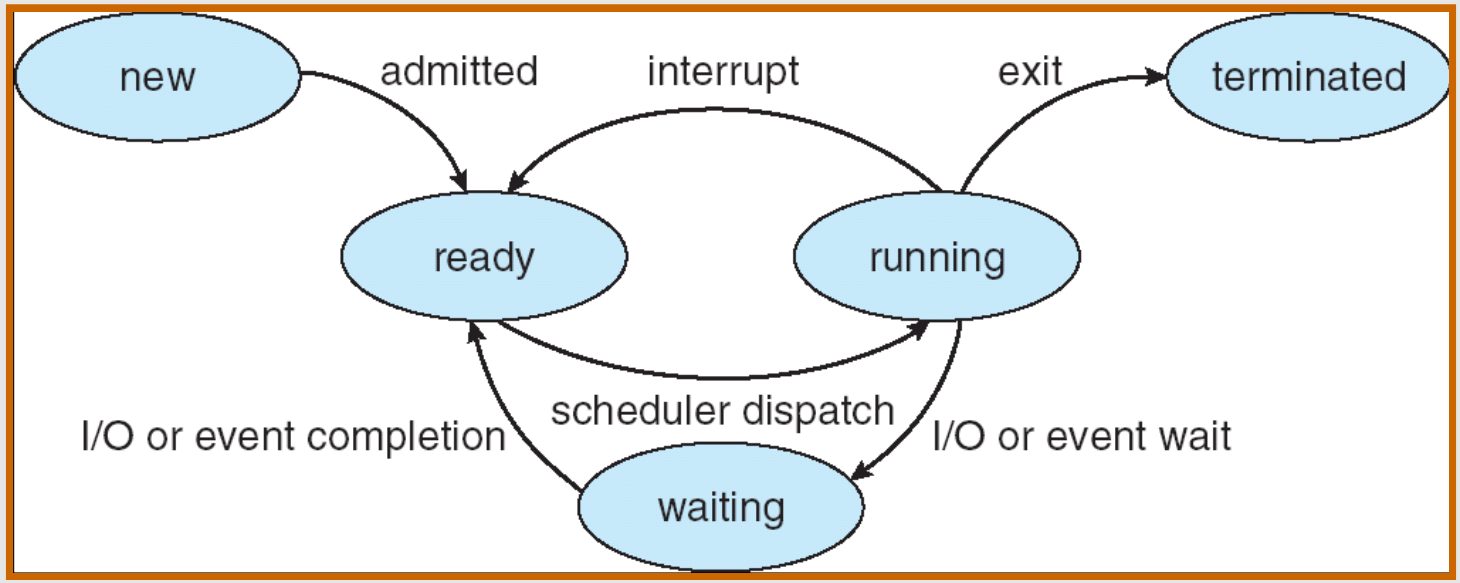
    else

        DENY allocation, rollback provisional allocation of resources to prior state

Request Material: Scheduling

- States of a process
  - Ready
  - Blocked
  - Running
  - Exiting
- Kernel data structures store process related data
- Scheduler
  - moves processes between different queues based on system events.

Process States



Realtime Schedules

- Build on Worst Case Execution Time (WCET) for each process
  - Table driven static schedules
  - Deadline driven scheduling based on a response time calculation for each task in the system based on worst case scheduling behaviour.

Exam structure ....

- Questions from the major course areas
  - Overview and OS structure, kernel designs
  - Process management, creation, termination communication
  - Classical problems, Producer/Consumer, Readers/Writers, Dining Philosophers
  - Semaphores, Mutual Exclusion and Deadlock (nearly 50% of lectures deal with aspects of this).
  - Memory systems, VM, page replacement
  - Files and I/O

Exam structure ....

- 5 hour examination, of 50 points
  - Grade boundaries, 30 G, 40 VG
    - 20 easy points
    - 20 medium hard points
    - 10 hard points
- One long application question [10-15]
- ~4 Medium length questions [4-8]
- ~6 Short answer questions [2-4]

... and strategy

- Mark the questions you can easily answer
- Always clearly state your assumtions
- Be sure that you answer all aspects of the question
  - Analyse what is asked for
  - Check that you have provided all that was asked for.
- Don't write too much, a concise and well reasoned answer gets the most marks