M E M O R A N D U M

Universiteit van Stellenbosch
Departement Rekenaarwetenskap

Kursus: RW 314

April 2003 Tydsduur: $2\frac{1}{2}$ uur

Algemene Konsepte *General Concepts*

1. Verduidelik kortliks die verskil tussen semetriese en asemetriese multiverwerking.

   Briefly explain the difference between symmetric and asymmetric multiprocessing.

   > In symmetric multiprocessing (SMP) systems, each processor runs an identical copy of the operating system and also supports communication between the various copies. In asymmetric systems, each processor is assigned a different task. There is usually a master processor that assigns tasks to the other processors and manages the system.

   3

2. Verduidelik wat bedoel word met die term **direkte geheuetoegang** (DMA), met spesiale verwysing na die gebruik daarvan in skyfbeheereenhede. Wat is die voordele daarvan om DMA op hierdie manier te gebruik?

   Explain what is meant by the term ***direct memory access*** *(DMA) with special reference to its use in disk control units. What are the advantages of using DMA in this way?*

   > DMA refers to a mechanism that supports asynchronous I/O. When the operating system receives a request to transfer data to or from an I/O device, it initializes the DMA controller by sending various parameters such as the amount of data and the address of the buffer that must be used during the transfer. Once the DMA controller has been programmed, a command is send to the I/O device to either read or write the data using DMA. Once the command has been send to the device, the operating system can continue by scheduling another process. The I/O device and DMA controller will then exchange the data between themselves without requiring any intervention from the CPU. When the transfer is complete, the I/O device will generate an interrupt, signalling that the operation has been completed. Although DMA allows the CPU to continue with other processing operations, some CPU cycles are lost (cycle stealing) because there will be contention between the CPU and DMA controller to access the system bus. If the I/O requested access to a disk, the operating system will begin by initializing the DMA controller, before sending either a read or write command to the disk controller. Once the disk controller receives the command, it will begin executing the required operation and exchange the data with the DMA controller until the operation has been completed afterwhich the disk controller will generate an interrupt and control is transfered back to the operating system.

   6

3. Noem en bespreek kortliks twee tipes dienste wat deur 'n bedryfstelsel verskaf word.

   Name and briefly discuss two types of services provided by an operating system.

   - **Communication** Operating systems must provide the means for different processes to communicate with each other, for example shared memory and message passing.
   - **File System** A file system is usually provided to allow users to store and retrieve their data or to store programs that can be loaded at a later stage for execution.

4

Prosesbestuur *Process Management*

4. Bespreek kortliks wat bedoel word met die term *race condition*. Lig u bespreking toe met 'n voorbeeld.

   Briefly describe what is meant by a race condition. Give an example to illustrate your answer.

   A race condition will occur if the outcome of the concurrent execution of two or more processes sharing data depend on a particular order of execution. This can be illustrated by examining the interleaving of machine instructions. The interleaving occurs as a result from the instructions generated by a compiler and the order in which processes are scheduled by an operating system. Consider the following two assignments: `i = i+1` and `i = i-1` and assume that variable `i` is shared between two processes P_0 and P_1. A typical processor might require three instructions for each assignement, resulting in the following possible interleaving (assuming that the initial value of `i` is 5):

Time	**Process**	**Instruction**	`i`	**Register 1** (P_0)	**Register 1** (P_1)
T_0	P_0	`LOAD R1, i`	5	5	?
T_1	P_0	`ADD R1, 1`	5	6	?
T_2	P_1	`LOAD R1, i`	5	6	5
T_3	P_1	`SUB R1, 1`	5	6	4
T_4	P_0	`STORE i, R1`	6	6	4
T_5	P_1	`STORE i, R1`	4	6	4

6

5. Bespreek kortliks onder watter omstandighede voorspringskedulering sal plaasvind.

   Briefly describe the necessary conditions for preemptive scheduling.

   - When a process switches from a running state to a ready state, for example, when an interrupt occurs.
   - When a process switches from a waiting state to a ready state, for example, when an I/O operation has been completed.

2

6. Bestudeer die onderstaande implementasie van die "Dining Philosopher's" probleem en beantwoord dan die vrae wat volg.

   Examine the implementation of the Dining Philosopher's problem below and answer the questions that follow.

```
VAR
  Forks: ARRAY 5 OF Semaphore;
...
(* Implementation of a single philosopher *)
PROCESS Philosopher(i: INTEGER);
BEGIN
  WHILE TRUE DO
    IF i MOD 2 = 0 THEN
      Wait(Forks[i]);
      Wait(Forks[(i+1) MOD 5]);
      (* Eat... *)
      Signal(Forks[i]);
      Signal(Forks[(i+1) MOD 5])
      (* Think... *)
    ELSE
      Wait(Forks[(i+1) MOD 5]);
      Wait(Forks[i]);
      (* Eat... *)
      Signal(Forks[(i+1) MOD 5]);
      Signal(Forks[i])
      (* Think... *)
    END
  END
END Philosopher

BEGIN
  Create(Philosopher(0), Philosopher(1), Philosopher(2),
         Philosopher(3), Philosopher(4))
END.
```

   (a) Kan vergrendeling voorkom? Motiveer u antwoord.

   Does the implementation contain a deadlock? Motivate your answer.

   > No. This is an asymmetrical solution that guarantees that the circular wait condition never prevails. Either even numbered (`i MOD 2 = 0`) or odd numbered (`i MOD 2 # 0`) philosophers may eat. For example, if philosopher 1 obtains his/her fork (`Forks[(i+1) MOD 5)`, then philosophers 2 will not be able to obtain the same fork because it will attempt execute `Wait(Forks[i])`.

   5

   (b) Is dit moontlik dat die verhongering van prosesse sal voorkom? Motiveer u antwoord.

   Can the implementation result in the starvation of a philosopher? Motivate your answer.

   > Yes. The solution does not provide for a philosopher to postpone eating again once he/she has done so already (there is no fairness). It is possible that the same philosopher can eat, put down his/her forks and then eat again before the adjacent philosphers get a chance to eat.

   3

7. Bespreek kortliks die verskil tussen gebruikers- en stelsel liggewigprosesse

   Briefly describe the difference between user and kernel threads

   > User level threads are implemented above the kernel level, usually through a thread library that supports thread creation, scheduling and termination. Although thread creation is faster at the user level, there are some drawbacks. If a thread blocks, then the entire process will typically become blocked. Kernel level threads are slower to create, but because the kernel schedules the threads, if a thread performs a blocking system call, other threads in the process can still be scheduled for execution.

   4

8. Verduidelik kortliks wat bedoel word met prioriteitsomkering ten opsigte van intydse skedulering van prosesse.

   Briefly explain what is meant by priority inversion with regards to scheduling real-time processes.

   > Priority inversion occurs when a low level process holds a resource required by a higher priority process. The higher priority process must now wait for the lower priority process before it can be scheduled. A priority inheritance protocol can be used to solve this problem by temporarily assigning a higher priority to the low priority process.

   4

9. Daar bestaan 'n hele aantal faktore wat as kriteria dien ten einde 'n potensiële skeduleringsalgoritme te ontleed. Noem twee faktore wat as kriteria gebruik kan word en bespreek kortliks die belang daarvan.

   A number of factors exist that define the criteria used to analyze potential scheduling algorithms. Name two factors that can be used as criteria and briefly discuess their importance.

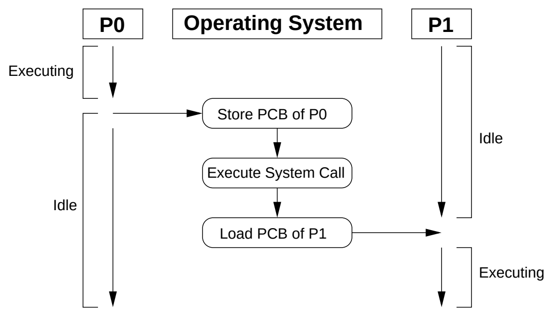
   > - **CPU Utilization** An efficient scheduling algorithm will try to keep the CPU occupied at all times to waste as little as possible processing cycles.
   > - **Waiting Time** Minimizing the amount of time that a process waits before being scheduled for execution will increase the overall processing within the system and can also affect other conditions such as throughput.

   6

10. Daar is tans twee prosesse in 'n stelsel, P_0 en P_1. P_0 is tans besig om uit te voer terwyl P_1 wag om geskeduleer te word. P_0 voer nou 'n stelselroep uit. Verduidelik kortliks wat met die verskillende prosesse gebeur en hoe die bedryfstelsel die stelselroep hanteer. Lig u verduideliking toe met 'n skets.

A system currently contains two processes, P_0 and P_1. P_0 is currently executing while P_1 is waiting to be scheduled. P_0 executes a system call. Briefly explain what will happen with the two processes and how the operating system will handle the system call. Use a diagram to illustrate your answer.

When P_0 executes a system call, control is transfered back to the operating system and P_0 is suspended and its state stored in a process control block. Depending on the type of the system call, P_0 may be placed in a waiting queue. This will most likely occur if the system call required an I/O operation. If possible, the operating system begins executing the requested system call. The scheduler must select another process to schedule, for example P_1. P_1 is removed from the ready queue and its PCB is loaded into memory, changing its state from ready to running. The scheduler will now invoke the dispatcher to transfer control back to P_1 to resume execution.



5

11. Bespreek kortliks die verskil tussen direkte en indirekte proses kommunikasie.

Briefly discuss the difference between direct and indirect process communication.

- **Direct Communication** requires a process that initiates communication to explicityly name the recipient or sender and communication can only take place between two processes and only a single link may be established between two processes.
- **Indirect Communication** takes place through special mailboxes called ports. Any number of processes can communicate with each other on any number of links as long as the various ports are shared between them.

6

12. Vergrendeling word opgespoor in 'n stelsel deur die toekenning van hulpbronne te ontleed. Die toekenning van 'n hulpbron, R_j, aan 'n proses, P_i, word aangedui as $R_j \rightarrow P_i$. Wanneer 'n proses op 'n hulpbron wag word dit aangedui as $P_i \rightarrow R_j$. Beskou die onderstaande toekennings en beantwoord die vrae wat volg. Daar is slegs een instansie van elke bron beskikbaar.

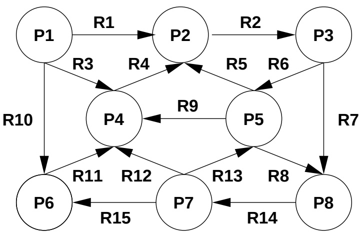
Deadlock can be detected in a system by analyzing the allocation of resources. The allocation of a resource, R_j, to a process, P_i, is expressed by $R_j \rightarrow P_i$. A process waiting to be allocated a resource is expressed by $P_i \rightarrow R_j$. Examine the various allocations below and answer the questions that follow. There is only one instance of every resource available.

$P_1 \rightarrow R_1$, $P_1 \rightarrow R_3$, $P_1 \rightarrow R_{10}$, $P_2 \rightarrow R_2$, $P_3 \rightarrow R_6$, $P_3 \rightarrow R_7$, $P_4 \rightarrow R_4$, $P_5 \rightarrow R_5$, $P_5 \rightarrow R_8$, $P_5 \rightarrow R_9$, $P_6 \rightarrow R_{11}$, $P_7 \rightarrow R_{12}$, $P_7 \rightarrow R_{13}$, $P_7 \rightarrow R_{15}$, $P_8 \rightarrow R_{14}$,

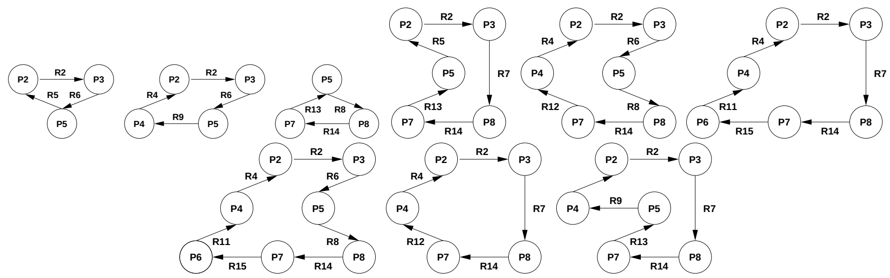
$R_1 \rightarrow P_2$, $R_2 \rightarrow P_3$, $R_3 \rightarrow P_4$, $R_4 \rightarrow P_2$, $R_5 \rightarrow P_2$, $R_6 \rightarrow P_5$, $R_7 \rightarrow P_8$, $R_8 \rightarrow P_8$, $R_9 \rightarrow P_4$, $R_{10} \rightarrow P_6$, $R_{11} \rightarrow P_4$, $R_{12} \rightarrow P_4$, $R_{13} \rightarrow P_5$, $R_{14} \rightarrow P_7$, $R_{15} \rightarrow P_6$, ,

(a) Hoeveel vergrendeling-siklusse bevat die wag-vir grafiek wat die huidige toestand van die stelsel beskryf? Lys telkens die nodusse wat by elk van die bogenoemde siklusse betrokke is.
How many deadlock cycles does the wait for graph depicting the current state of the system contain?. List the nodes in the graph associated with each of the aforementioned cycles.

The wait-for graph looks as follows:



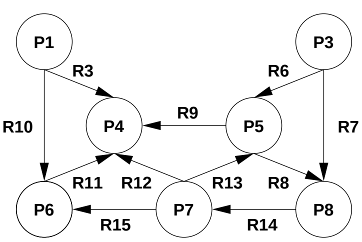
There are nine deadlock cycles in the graph:



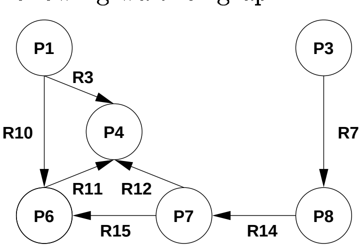
12

(b) Wat is die minimum hoeveelheid prosesse wat vernietig kan word gegewe dat die stelsel prosesse individueel verwyder ten einde van vergrendeling te ontsnap? Verskaf die proses, of volgorde van prosesse indien meer as een proses vernietig moet word.
What is the minimum number of processes that must be terminated to resolve the deadlock in the system, given that the system will terminate one process at a time? Write down the process, or sequence of processes if more than one process must be destroyed.

Two. Removing p_2 results in the following wait-for graph:



Removing p_5 results in the following wait-for graph:



2

(c) Die ontwikkelaars van die stelsel glo dat daar so min as moontlik ontwrigting vir hulle gebruikers moet wees. Veronderstel dat die prosesse reeds 'n sekere aantal tydseenhede aan verwerking afgestaan het: P_1 (5), P_2 (18), P_3 (14), P_4 (3), P_5 (11), P_6 (10), P_7 (9) en P_8 (3). Watter prosesse moet getermineer word ten einde die vergrendeling te breek indien die verlies aan verwerkingstyd 'n minimum moet wees?

The developers of the sytem believe that the user must experience the least possible amount of distruption. Assume that the processes have each spent a certain amount of time executing: P_1 (5), P_2 (18), P_3 (14), P_4 (3), P_5 (11), P_6 (10), P_7 (9) and P_8 (3). Which processes must be terminated to break the deadlock if the loss in processing time is to be kept at a minimum?

p_8 (3 time units) and p_5 (11 time units) or p_4 (3 time units) and p_5 (11 time units).

2

13. Die onderstaande tabel bevat die toekennnig van vier verskillende hulpbronne (r_1, r_2, r_3 en r_4) aan vyf prosesse ($p_1 \dots p_5$) op 'n gegewe oomblik. Die tabel bevat ook inligting aangaande die maksimum hoeveelheid wat elke proses van 'n spesifieke hulpbron kan aanvra. Daar is tans 2 instansies van r_1 en 1 instansie van r_2 beskikbaar. Beantwoord die onderstaande vrae wat volg.

The tabel below describes the allocation of four different resources (r_1, r_2, r_3 and r_r) to five processes ($p_1 \dots p_5$) at a specific moment in time. The tabel also contains information related to the maxmimum number of resources a process can request. Currently, there are only 2 instances of r_1 and 1 instance of r_2 available. Answer the questions that follow.

Proses *Process*	**Huidige *Current***				**Maks *Max***			
	r_1	r_2	r_3	r_4	r_1	r_2	r_3	r_4
p_1	0	0	1	2	0	0	1	2
p_2	2	0	0	0	2	7	5	0
p_3	0	0	3	4	6	6	5	6
p_4	2	3	5	4	4	3	5	6
p_5	0	3	3	2	0	6	5	2

(a) Verkeer die stelsel tans in 'n veilige of onveilige toestand? Motiveer u antwoord.

Is the system currently in a safe or unsafe state? Motivate your answer.

The system is currently in a safe state. The following execution order results in all processes aquiring their resources and terminating:

Process	r_1	r_2	r_3	r_4
Available	2	1	0	0
p_1 release	0	0	1	2
Available	2	1	1	2
p_4 request	(2)	0	0	(2)
p_4 release	4	3	5	6
Available	4	4	6	6
p_5 request	0	(3)	(2)	0
p_5 release	0	6	5	2
Available	4	7	9	8
p_2 request	0	(7)	(5)	0
p_2 release	2	7	5	0
Available	6	7	9	8
p_3 request	(6)	(6)	(2)	(2)
p_3 release	6	6	5	6
Available	6	7	12	12

3

(b) Watter prosesse, indien enige, is tans of kan moontlik vergrendel raak?

Which processes, if any, are currently or may possibly become deadlocked?

> p_1 does not require any more resources. However, should any other process request the remaining instances of r_1 or r_2 without p_1 terminating and releasing its resources, then no process will be able to terminate making $p_2 \ldots p_5$ possible candidates for a deadlock.

3

(c) 'n Versoek vir bronallokasie word vanaf p_3 (0, 1, 0, 0) ontvang. In watter toestand sal die stelsel verkeer indien hierdie versoek toegestaan word (vergrendel, veilig of onveilig)? Watter prosesse, indien enige, is of kan moontlik vergrendel raak? Motiveer u antwoord.

A request for resource allocation is received from p_3 (0, 1, 0, 0). What will the state of the system be should the request be granted (deadlocked, safe or unsafe)?. Which processes, if any, are or may be deadlocked? Motivate your answer.

> The system would be in an unsafe state and eventually fall into deadlock. If the request is granted, then p_1 can still terminate, releasing its resources and allowing p_4 and p_5 to execute. However, at this point neither p_2 nor p_3 will be able to aquire their outstanding resources. p_3 requires 6 instances of r_1 and there are only 4 instances available. Similarly, p_2 requires 7 instances of r_2 and there are only 6 available. p_2 and p_3 are now deadlocked.

4

14. Gee **twee** goeie redes waarom die "banker's" algoritme (gebruik om vergrendeling te voorkom) nie baie bruikbaar is in die praktyk nie.

Give **two** *good reasons why the banker's algorithm (used to prevent deadlock) is not very useful in practice.*

> (a) The banker's algorithm requires that every process states the maximum number of every resource that it would require. This information is often unknown in practice.
>
> (b) Resources, especially hardware, may break down from time to time. However, the banker's algorithm requires that the number of resources stay constant.

4

Totaal: / *Total:* 80