University of Glasgow

Wednesday, 27 April 2016
2:00 pm – 4:00 pm
(2 hours)

DEGREES OF MSc, MSci, MEng, BEng, BSc, MA and MA (Social Sciences)

ADVANCED OPERATING SYSTEMS (M)

Answer 3 out of 4 questions

This examination paper is worth a total of 60 marks.

The use of calculators is not permitted in this examination.

INSTRUCTIONS TO INVIGILATORS: Please collect all exam question papers and exam answer scripts and retain for school to collect. Candidates must not remove exam question papers.

1. **(a)** Real-time systems can be implemented using either a clock-driven scheduler or a priority-driven scheduler. When might you use clock-driven scheduling? [2]

(b) Two widely used priority-driven scheduling algorithms are the rate monotonic algorithm, and the earliest deadline first algorithm. The earliest deadline first algorithm is provably optimal, where-as the rate monotonic algorithm is not. Despite this, many applications use the rate monotonic algorithm. Discuss why this might be. [5]

(c) A system comprises a mixture of periodic tasks and sporadic jobs. These are to be scheduled in a pre-emptive manner on a single processor, using the earliest deadline first (EDF) algorithm for the periodic tasks, with a sporadic server to handle execution of the sporadic jobs. Explain the operation of the acceptance test for newly arrived sporadic jobs in such a system. Your answer should clearly state the condition that must be true to guarantee that both the periodic tasks and sporadic jobs can be scheduled, and outline how the acceptance test that checks that condition can be implemented. Comment on the optimality of the acceptance test. [8]

(d) Are sporadic tasks incompatible with hard real-time systems? Discuss. [5]

2. **(a)** We discussed the paper by Bacon et al., on "A real-time garbage collector with low overhead and consistent utilization" (Proc. ACM PoPL, 2003). This paper proposes a real-time garbage collector that executes as a periodic task, scheduled alongside the other tasks in the system. Discuss the trade-off between predictable timing and collection efficiency for this garbage collector. Outline any constraints it imposes on the memory allocation behaviour of the system to ensure correctness. [6]

(b) Some systems use region-based memory management as an alternative to garbage collection. Describe the memory management model offered by a language that supports region-based memory management. Explain why such languages tend to offer some alternative means of memory management, in addition to region-based memory management. [10]

(c) Explain why it is difficult to build a garbage collector for the C programming language, but comparatively straight-forward for Java. [4]

3. **(a)** Over the past decade, or so, processors have moved from emphasising single threaded performance to supporting large numbers of concurrent threads, primarily by moving to multicore processor designs. Explain why this transition has occurred. [5]

(b) Mainstream programming languages, such as C and Java, provide features such as threads, locks, and shared mutable state as their abstractions for concurrency (e.g., via the *pthreads* API for C). With the aid of an example, outline why such mechanisms do not provide a safe concurrency abstraction. [5]

(c) To avoid the problems inherent in languages that use multiple threads with shared mutable state protected by locks, some new languages enable concurrency by isolating tasks and providing explicit message passing features. Explain why this approach does not protect from deadlocks and race conditions. Discuss whether you think deadlocks and race conditions are more or less likely to occur than in systems using message passing compared to those using lock-based synchronisation, justifying your answer. [10]

CONTINUED OVERLEAF

4. **(a)** We discussed the paper by Kamp and Watson on “Jails: Confining the omnipotent root.” (Proc. SANE, 2000). This describes a container-based approach to virtualisation, as was originally developed for FreeBSD, that has since been ported to several other operating systems. Describe what type of virtualisation is provided, and outline the key properties of the *jails* environment. [7]

(b) We also discussed the paper by Barham *et al*. on “Xen and the Art of Virtualization” (Proc. ACM SOSP 2003). Describe what type of virtualisation is provided, and outline the key properties of the Xen system. [7]

(c) Compare and contrast the virtualisation provided by FreeBSD *jails* and by Xen. When would you choose a container-based environment, like *jails*, and when would a Xen-like virtualisation be more suitable? [6]