Sistem Waktu Nyata (CS4613)
Tugas 2

Analisis Penjadwalan Proses dengan Cheddar Real-Time Scheduling Simulator

Dimodifikasi dari:
CSE 3141 Prac #4-5
Analysing Scheduling using
the Cheddar Real-Time Simulator
Dr. Carlo Kopp, PEng
Monash University
2004

Pengantar

Dalam tugas ini ada empat hal yang harus dilakukan:

1. Instalasi dan perkenalan dengan Cheddar Real-Time Scheduling Simulator. Untuk tugas ini digunakan Cheddar-1.3p5, release date : November the 1st, 2005
2. Analisis penjadwalan proses periodik dengan algoritma penjadwalan 'highest priority first'
3. Analisis penjadwalan dengan algoritma 'Earliest Deadline First'
4. Analisis penjadwalan dengan algoritma 'Earliest Deadline First' dan 'Least Laxity First' pada sistem yang *overload*.

Bersama dengan spesifikasi tugas ini, disertakan tiga file xml (RTS200601-2-0.xml, RTS200601-3-0.xml, dan RTS200601-4-0.xml) yang secara berurutan akan digunakan pada bagian 2, 3, dan 4 dari tugas ini.

Tugas dikerjakan secara kelompok. Maksimal dua orang tiap kelompok.
Yang dikumpulkan:

- Semua file xml yang dibuat dalam setiap aktifitas (kecuali tiga file xml yang disebutkan di atas)
- Satu file pdf yang berisi identitas anggota kelompok dan jawaban semua pertanyaan dalam tugas ini

Dikumpulkan via email fay@stttelkom.ac.id paling lambat tanggal 27 September 2006 jam 23:59:59 WIB dengan subject: RTS Tugas2. Single file attachment; nama file: RTS_tugas2_<NIM1>_<NIM2>.[zip|rar].
Disarankan saat pengiriman memilih option *request a read receipt*.

Beberapa hal tentang Cheddar:

- Cheddar adalah free real time scheduling framework.
- Cheddar dirancang untuk dapat melakukan pengujian terhadap temporal constraint dari suatu task pada aplikasi atau sistem waktu nyata.
- Cheddar adalah free software, distribusinya dibawah GNU General Public Licence.
- Cheddar dibangun oleh LYSIC Team dari University of Brest, Prancis.
- Cheddar bisa didapat dari situs http://beru.univ-brest.fr/~singhoff/cheddar. Di situs ini, selain binaries (untuk Linux, Windows, dan Solaris) juga tersedia pula source code, dan dokumentasi yang berkaitan dengan Cheddar.
- Cheddar-1.3p5, release date : November the 1st, 2005 telah diuji pada platform:
    - Linux Redhat (6.1, 6.2, 7.1 dan 9.1).
    - Sparc/Solaris 2.7, 2.8 dan 2.9.
    - Windows Me, 95, 98, 2000, NT4 dan XP.
    - Cheddar juga bisa beroperasi pada semua platform yang mendukung GNAT/GtkAda.

Tugas II.1 – Instalasi dan Perkenalan

1. Download Cheddar (berisi binary file, dokumentasi, dan beberapa contoh project) :
   - Untuk Windows: Cheddar-1.3p5-win32-bin.zip
   - Untuk Linux: Cheddar-1.3p5-linux-bin.tar.gz

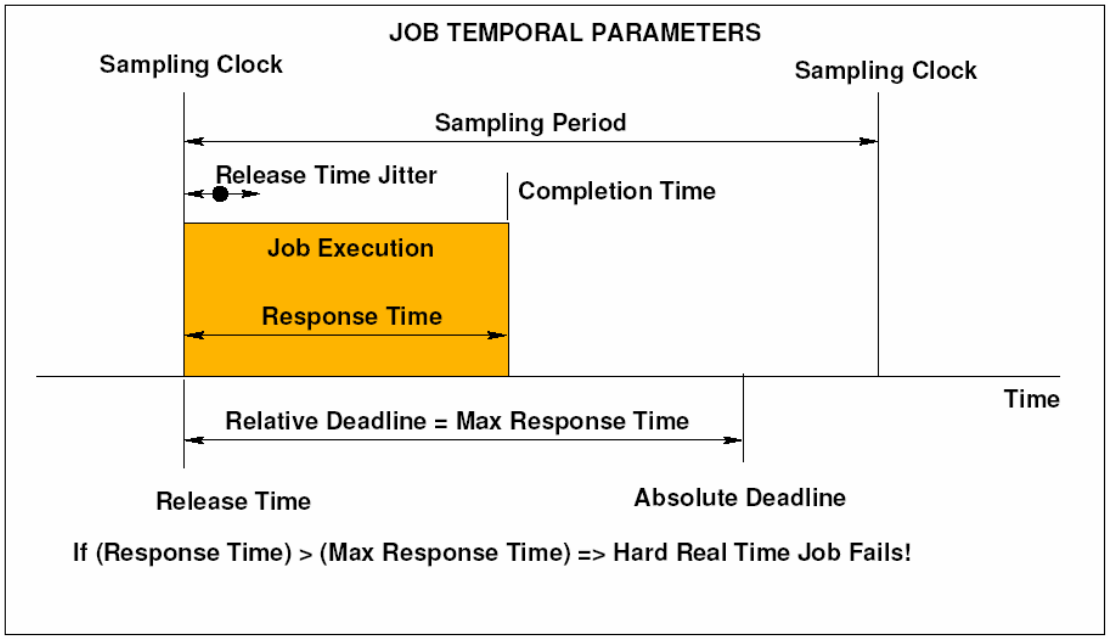
2. Install Cheddar:
   - Untuk Windows: unzip file Cheddar-1.3p5-win32-bin.zip, cheddar.exe sudah bisa langsung dieksekusi
   - Untuk Linux: diperlukan GNU make (gmake) Cheddar-1.3p5-linux-bin.tar.gz. Cara mudah untuk install program ini:
     - Edit Makefile: update variabel CHEDDAR_ROOT dengan nama directory tempat Cheddar akan di-install. Default directory adalah /usr/local
     - ‘gmake’ untuk meng-install package ini
     - Update variable PATH dan LD_LIBRARY_PATH (lihat pada file INSTALL)

3. Baca user guide untuk mengetahui bagaimana cara mengoperasikan Cheddar

4. Jalankan beberapa project yang ada di directory project_examples/tests

Tugas II.2 – Periodik; Highest Priority First

Berikut adalah ilustrasi parameter-parameter task yang bersifat periodik:



Aktifitas 2.0

1. Jalankan Cheddar. Buka file RTS200601-2-0.xml.
2. Gunakan menu Edit > Update > Update tasks untuk melihat definisi tiga task. Semuanya adalah task yang bersifat periodik dengan periode = 10 dan deadline = 10. Dua task memiliki waktu eksekusi (Cheddar menggunakan istilah *capacity*) = 1, satu task lainnya memiliki waktu eksekusi = 2. Task diberi prioritas berturut-turut 1,2,dan 3.
3. Gunakan menu Edit > Update > Update processors untuk melihat definisi sebuah prosesor yang menggunakan algoritma penjadwalan Highest Priority First (POSIX) dan mode non-pre-emptive.
4. Jalankan ‘scheduling simulation’ mulai dari 0 hingga 30. Catat hasil yang tampil pada workspace. Eksekusi task digambarkan dengan batang horisontal warna hitam, sementara garis vertikal warna merah menunjukkan release time dari task.

Aktifitas 2.1

1. Buat salinan file RTS200601-2-0.xml, beri nama RTS200601-2-1.xml.
2. Lakukan percobaan dengan menaikkan waktu eksekusi per satu unit waktu untuk tiga task secara bersamaan, hingga tidak dapat lagi ditemukan jadwal yang feasible. Simpan file RTS200601-2-1.xml yang memuat jadwal feasible terakhir.
3. Jawab pertanyaan berikut:

a. Pada nilai waktu-eksekusi berapakah tidak dapat lagi ditemukan jadwal yang feasible?
b. Berapakah utilisasi prosesor pada jadwal feasible terakhir?
c. Berapakah response time tiap task pada jadwal feasible terakhir?

Aktifitas 2.2

1. Buat salinan file RTS200601-2-0.xml, beri nama RTS200601-2-2.xml.
2. Modifikasi periode Task_1 menjadi 20 dan deadline menjadi 20. Lakukan percobaan dengan menaikkan waktu eksekusi per satu unit waktu untuk tiga task secara bersamaan, hingga tidak dapat lagi ditemukan jadwal yang feasible. Simpan file RTS200601-2-2.xml yang memuat jadwal feasible terakhir.
3. Jawab pertanyaan berikut:
    a. Pada nilai waktu-eksekusi berapakah tidak dapat lagi ditemukan jadwal yang feasible?
    b. Berapakah utilisasi prosesor pada jadwal feasible terakhir?
    c. Berapakah response time tiap task pada jadwal feasible terakhir?

Aktifitas 2.3

1. Buat salinan file RTS200601-2-0.xml, beri nama RTS200601-2-3.xml.
2. Modifikasi periode Task_1 menjadi 20 dan deadline menjadi 20. Modifikasi periode Task_2 menjadi 20 dan deadline menjadi 30. Lakukan percobaan dengan menaikkan waktu eksekusi per satu unit waktu untuk tiga task secara bersamaan, hingga tidak dapat lagi ditemukan jadwal yang feasible. Simpan file RTS200601-2-3.xml yang memuat jadwal feasible terakhir.
3. Jawab pertanyaan berikut:
    a. Pada nilai waktu-eksekusi berapakah tidak dapat lagi ditemukan jadwal yang feasible?
    b. Berapakah utilisasi prosesor pada jadwal feasible terakhir?
    c. Berapakah response time tiap task pada jadwal feasible terakhir?

Aktifitas 2.4

1. Buat salinan file RTS200601-2-0.xml, beri nama RTS200601-2-3.xml.
2. Modifikasi periode Task_1 menjadi 20 dan deadline menjadi 20. Modifikasi periode Task_2 menjadi 20 dan deadline menjadi 30. Ubah mode penjadwalan menjadi pre-emptive. Lakukan percobaan dengan menaikkan waktu eksekusi per satu unit waktu untuk tiga task secara bersamaan, hingga tidak dapat lagi ditemukan jadwal yang feasible. Simpan file RTS200601-2-3.xml yang memuat jadwal feasible terakhir.
3. Jawab pertanyaan berikut:
    a. Pada nilai waktu-eksekusi berapakah tidak dapat lagi ditemukan jadwal yang feasible?
    b. Berapakah utilisasi prosesor pada jadwal feasible terakhir?
    c. Berapakah response time tiap task pada jadwal feasible terakhir?
    d. Apa dampak pre-emption terhadap response time dan waktu eksekusi? Berikan kajian tentang hal ini!

Tugas II.3 – Earliest Deadline First (EDF)

Aktifitas 3.1

1. Buka file RTS200601-3-0.xml. Perhatikan deadline pada definisi task dan penggunaan algoritma penjadwalan Earliest Deadline First pada prosesor.
2. Lakukan percobaan dengan menaikkan waktu eksekusi per satu unit waktu untuk tiga task secara bersamaan, hingga tidak dapat lagi ditemukan jadwal yang feasible. Simpan dalam file RTS200601-3-1.xml yang memuat jadwal tidak feasible yang pertama ditemukan.
3. Jawab pertanyaan berikut:
    a. Pada nilai waktu-eksekusi berapakah tidak dapat lagi ditemukan jadwal yang feasible?
    b. Task manakah yang pertama kali gagal dieksekusi? Mengapa?
    c. Berapakah utilisasi prosesor pada jadwal feasible terakhir?
    d. Berapakah response time tiap task pada jadwal feasible terakhir?

Aktifitas 3.2

1. Buat salinan file RTS200601-3-0.xml, beri nama RTS200601-3-2.xml.
2. Ubah periode Task_1 menjadi 20. Deadline task berturut-turut tetap 8, 9, dan 10. Lakukan percobaan dengan menaikkan waktu eksekusi per satu unit waktu untuk tiga task secara bersamaan, hingga tidak dapat lagi ditemukan jadwal yang feasible. Simpan dalam file RTS200601-3-2.xml yang memuat jadwal feasible terakhir.
3. Jawab pertanyaan berikut:
    a. Pada nilai waktu-eksekusi berapakah tidak dapat lagi ditemukan jadwal yang feasible?
    b. Task manakah yang pertama kali gagal dieksekusi? Mengapa?
    c. Berapakah utilisasi prosesor pada jadwal feasible terakhir?
    d. Berapakah response time tiap task pada jadwal feasible terakhir?

Aktifitas 3.3

1. Buat salinan file RTS200601-3-0.xml, beri nama RTS200601-3-3.xml.
2. Ubah periode Task_1 menjadi 20, deadline menjadi 20. Modifikasi periode Task_2 menjadi 30, deadline menjadi 7. Lakukan percobaan dengan menaikkan waktu eksekusi per satu unit waktu untuk tiga task secara bersamaan, hingga tidak dapat lagi ditemukan jadwal yang feasible. Simpan dalam file RTS200601-3-3.xml yang memuat jadwal feasible terakhir.
3. Jawab pertanyaan berikut:
    a. Pada nilai waktu-eksekusi berapakah tidak dapat lagi ditemukan jadwal yang feasible?
    b. Berapakah utilisasi prosesor pada jadwal feasible terakhir?
    c. Berapakah response time tiap task pada jadwal feasible terakhir?
    d. Apa perbedaan antara jadwal ini dengan jadwal pada aktifitas 3.2? Berikan kajian untuk hal ini.

Aktifitas 3.4

1. Buat salinan file RTS200601-3-3.xml, beri nama RTS200601-3-4.xml.
2. Lakukan percobaan dengan menurunkan deadline Task_1 per satu unit waktu, hingga tidak dapat lagi ditemukan jadwal yang feasible. Simpan dalam file RTS200601-3-4.xml yang memuat jadwal feasible terakhir.
3. Jawab pertanyaan berikut:
    a. Pada nilai deadline Task_1 berapakah tidak dapat lagi ditemukan jadwal yang feasible?
    b. Task manakah yang pertama kali gagal? Mengapa?
    c. Berapakah utilisasi prosesor pada jadwal feasible terakhir?
    d. Berapakah response time tiap task pada jadwal feasible terakhir?

Tugas II.4 – EDF & LLF; overload

Aktifitas 4.1

1. Buka file RTS200601-4-0.xml. Perhatikan release time dan deadline pada definisi task, dan penggunaan algoritma penjadwalan Earliest Deadline First pada Processor_1.
2. Tambahkan sebuah prosesor dengan nama Processor_2, yang menggunakan algoritma penjadwalan Least Laxity First (LLF). Tambahkan dua task (Task_2 dan Task_3) yang identik dengan dua task sebelumnya, untuk dijalankan pada Processor_2. Jalankan simulasi dan simpan dalam file RTS200601-4-1.xml.
3. Jawab pertanyaan berikut:
    a. Jika diasumsikan tiap context switch membutuhkan waktu 100 mikro detik, Berapakah waktu yang dihabiskan masing-masing oleh EDF dan LLF?
    b. Jelaskan mengapa LLF menghasilkan jumlah context switch yang berbeda dengan EDF
    c. Bagaimana efek penggunaan algoritma penjadwalan EDF maupun LLF terhadap response time?
    d. Berbekal apa yang sudah ditemukan, berikan rekomendasi dalam kasus apa sebaiknya EDF maupun LLF digunakan.

Aktifitas 4.2

1. Buat salinan file RTS200601-4-1.xml, beri nama RTS200601-4-2.xml.
2. Lakukan percobaan dengan menaikkan release time Task_1 dan/atau Task_3 per unit waktu sampai ditemukan jumlah context switch minimal. Simpan dalam file RTS200601-4-2.xml.
3. Jawab pertanyaan berikut:
    a. Pada nilai release time berapakah tercapai jumlah context switch yang paling sedikit?
    b. Berikan penjelasan dari jawaban pertanyaan (a) di atas

Aktifitas 4.3

1. Buat project baru sebagai berikut:
    a. Dua prosesor: Processor_1 menggunakan EDF, Processor_2 menggunakan LLF
    b. Tiga task (Job_0, Job_1, Job_2) dengan spesifikasi yang identik: execution time = 4, release time = 0, deadline = 8, blocking time = 0, prioritas = 1, policy = sched_fifo. Tiga task ini dieksekusi pada Processor_1.
    c. Tiga task (Job_3, Job_4, Job_5) dengan spesifikasi yang identik: execution time = 4, release time = 0, deadline = 8, blocking time = 0, prioritas = 1, policy = sched_fifo. Tiga task ini dieksekusi pada Processor_2.

2. Simpan dalam file RTS200601-4-3.xml. Jalankan simulasi mulai dari unit waktu 0 hingga 20.
3. Jawab pertanyaan berikut:
    a. Hitung miss rates (jumlah task yang deadline-nya terlanggar dibagi jumlah total task) dari dua algoritma tersebut
    b. Dengan tidak hanya memandang miss rate, algoritma penjadwalan manakah (antara EDF dan LLF) yang lebih baik digunakan pada sistem dengan beban penjadwalan overload?
    c. Berikan kajian tentang hubungan algoritma penjadwalan yang digunakan, overload, miss rate, dan utilisasi prosesor.