PENYELARASAN PADA MASALAH DINING PHILOSOPHERS MENGGUNAKAN ALGORITMA LOCK & RELEASE

Andysah Putera Utama Siahaan

Universitas Sumatra Utara
Jl. Dr. Mansur No. 9, Medan, Sumatra Utara, Indonesia
andiesiahaan@gmail.com

Abstrak

Pembahasan dari makalah bagaimana mencegah deadlock dalam masalah filsuf makan. Deadlock adalah keadaan yang tidak diinginkan dari sistem konkuren, ditandai dengan serangkaian proses dalam keadaan menunggu melingkar, di mana setiap proses diblokir mencoba untuk mendapatkan akses ke sumber daya yang dimiliki oleh yang berikutnya dalam rantai. Pencegahan deadlock umumnya digunakan dalam didistribusikan real-time dan sistem operasi, tetapi, karena konkurensi sangat terbatas, sebuah sistem didistribusikan untuk menghindari deadlock secara efisien. Makalah ini menyajikan sebuah teknik untuk menghindari deadlock menggunakan algoritma lock dan release yang dapat mencegah thread lain dalam rantai untuk membuat race condition.

Kata Kunci — Dining Philosophers Problem, Race Condition, Concurrent, Deadlock, Starvation

I. PENDAHULUAN

Istilah pemrograman konkuren biasanya mengacu pada jenis program yang dibutuhkan untuk mendukung kegiatan simultan dalam aplikasi perangkat lunak. Definisi seperti itu agak luas, dan memang pemrograman konkuren dapat merujuk ke berbagai model

pemrograman. Sebagai contoh, sistem multi-prosesor, sistem paralel, sistem terdistribusi serta sistem uni-prosesor dengan konkurensi simulasi. Semua platform ini mendukung pemrograman konkuren. Pada artikel ini, kita mengambil penekanan khusus pada sistem prosesor uni atau multi dengan konkurensi aktual atau simulasi. Banyak masalah yang akan ditangani di sini yang bersifat umum dan dapat diterapkan pada sistem lain juga. Perhatikan bahwa berbagi informasi dinamis merupakan kriteria penting. Pemrograman dengan dua program yang berbeda berjalan pada dua mesin yang berbeda adalah sebagai pemrograman konkuren.

II. DINING PHILOSOPHERS

Lima filosof duduk mengelilingi meja bundar dan ada semangkuk besar spaghetti di tengah meja. Ada lima garpu ditempatkan di sekitar meja di antara para filsuf. Ketika seorang filsuf, yang sebagian besar dalam bisnis berpikir mendapat lapar, ia meraih dua garpu untuk nya langsung kiri dan kanan dan mati-set untuk mendapatkan makanan. Skenario biasanya diproyeksikan adalah bahwa jika semua filsuf ambil garpu mereka di sebelah kiri mereka secara bersamaan dan tidak satupun dari mereka akan dapat meraih garpu di sebelah kanan mereka. Ide dasar dibalik skenario adalah bahwa jika aktivitas bersamaan selalu melakukan apa yang tampaknya terbaik untuk dirinya sendiri atau apa yang tampaknya menjadi hal yang benar untuk dirinya sendiri dalam skenario sumber daya bersama, hasilnya bisa menjadi kekacauan. Apakah ada solusi untuk masalah Dining filsuf?

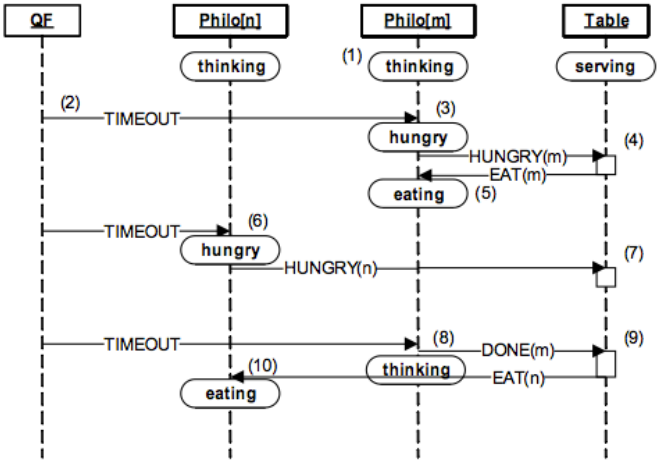
Asumsikan bahwa kita memiliki tugas sederhana menulis beberapa informasi penting menjadi dua file pada disk. Tetapi file-file ini juga dimiliki oleh program lain juga. Oleh karena itu Anda menggunakan strategi berikut untuk memperbarui file:

```
Kunci A
Kunci B
Menulis Informasi ke A dan B
Lepaskan Kunci
```

Coding yang tampaknya sederhana dan jelas hal ini dapat mengakibatkan deadlock jika tugas-tugas lain juga menulis ke file ini. Sebagai contoh, jika yang lain kunci tugas B pertama, kemudian mengunci A, dan jika kedua tugas mencoba untuk melakukan pekerjaan mereka pada saat yang sama deadlock terjadi. Tugas pertama akan mengunci A, tugas lainnya akan mengunci B, maka tugas pertama akan menunggu tanpa batas waktu untuk mengunci B sementara menunggu tugas lain tanpa batas untuk mengunci A.

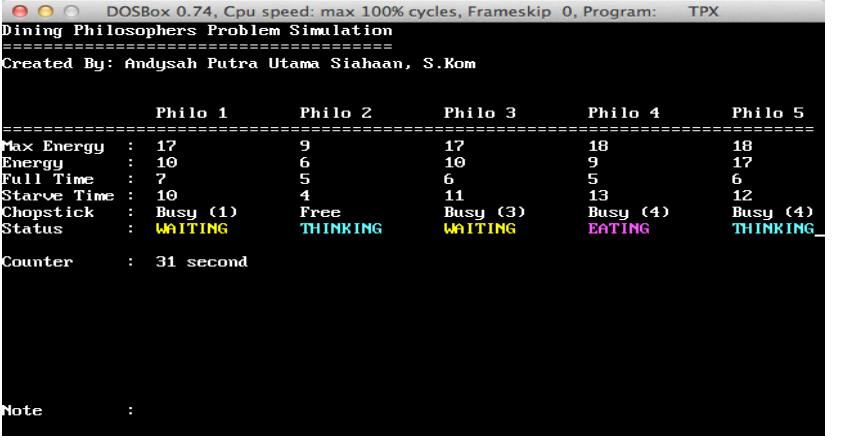
III. PEMBAHASAN

Sebuah titik awal yang baik dalam merancang sistem adalah menggambar diagram urutan untuk skenario utama yang diidentifikasi dari spesifikasi masalah. Untuk menggambar diagram tersebut, kita perlu untuk memecah masalah ke objek aktif dengan tujuan utama meminimalkan duplikasi antara objek-objek yang aktif. Masalah Dining Philosophers telah secara khusus disusun untuk membuat filsuf mengambil garpu, yang merupakan sumber daya bersama dalam kasus ini. Dalam sistem objek yang aktif, strategi desain generik untuk menangani sumber daya tersebut bersama adalah untuk melibatkan mereka di dalam objek aktif berdedikasi dan membiarkan objek yang mengelola sumber daya bersama.

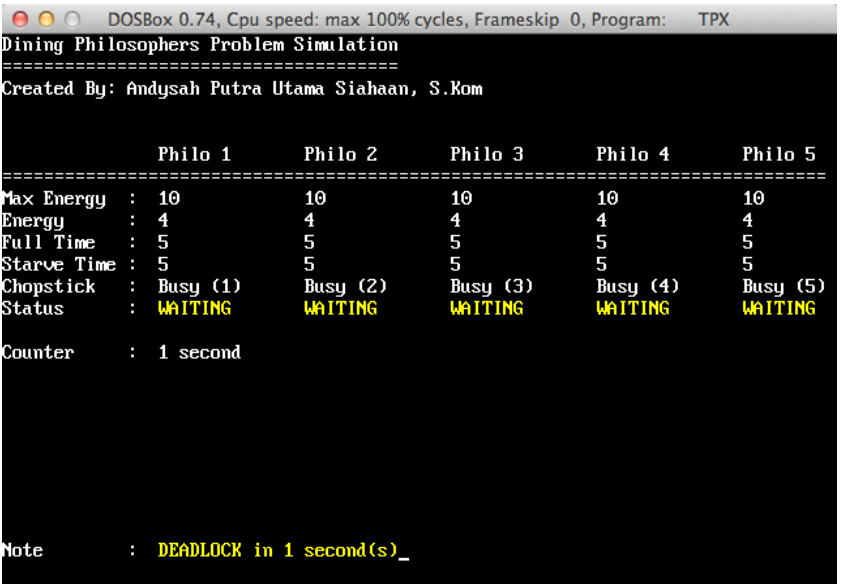


Gambar 1: Urutan dari Dining Philosophers

Gambar 1 menunjukkan keadaan yang berhubungan dengan objek yang aktif. Keadaan ini melacak garpu dan filsuf yang sedang lapar dengan cara menghitung waktu time-out. Antara filsuf pertama dengan filsuf yang berikutnya tidak boleh saling berbenturan sehingga deadlock dapat dihindari.



Gambar 2: Deadlock Tidak Terjadi



Gambar 3: Deadlock Terjadi

Gambar 2 menunjukkan proses antrian pada filsuf dalam menggunakan garpu, sementara filsuf lainnya melakukan makan dan berfikir. Pada gambar 3 terjadi deadlock dimana seluruh filsuf menunggu garpu yang sedang dipakai.

IV. KESIMPULAN

Pada pemrograman konkuren diperlukan beberapa teknik yang dapat menghindari keadaan deadlock pada saat melakukan pekerjaan yang bersamaan. Jika terjadi deadlock komputer akan mengalami waktu tunggu yang lama dan kemungkinan dapat menyebabkan komputer tidak bekerja sama sekali. Teknik lock dan release adalah salah satu cara yang dapat menghindari deadlock dari sistem pemrograman konkuren.

DAFTAR PUSTAKA

W. S. Davis and T. M. Rajkumar, Operating Systems A Systematic View, 5th Edition, Addison Wesley, 2001.

S. Tanenbaum, Modern Operating Systems, 2nd Edition, Prentice Hall, 2001.

Gary Nutt, Kernel Projects for Linux, Addison Wesley, 2000.

H.M. Deitel, An Introduction to Operating Systems, 2nd Edition, Addison-Wesley, Reading, MA 1990.

Steven V. Earhart (Editor), UNIX Programmer's Manual, Holt, Rinehart, and Winston, New York, NY 1986.