KONKURENSI

BAGUS SUDIRMAN., S.KOM., M.KOM.

KONKURENSI

- Konkurensi merupakan landasan umum perancangan sistem operasi.
- Proses-proses disebut konkuren jika proses-proses berada pada saat yang sama.

KONKURENSI

- Beberapa masalah yang harus diselesaikan:
  - ***Mutual Exclusion***
  - ***Sinkronisasi***
  - ***Deadlock***
  - ***Startvation***

Prinsip-prinsip Konkurensi

Konkurensi meliputi hal-hal sbb:

- Alokasi waktu pemroses untuk proses-proses
- Pemakaian bersama dan persaingan untuk mendapatkan sumber daya
- Komunikasi antarproses
- Sinkronisasi aktivitas banyak proses.

KONKURENSI

Konkurensi dapat muncul pada konteks berbeda, antara lain:

- **Banyak aplikasi (*multiple application*).**
- **Aplikasi terstruktur.**
- **Struktur sistem operasi**
- **Untuk Strukturisasi Satu Proses.**

Banyak aplikasi (*multiple application*)

Multiprogramming memungkinkan banyak proses sekaligus dijalankan.

Aplikasi terstruktur.

- Perluasan prinsip perancangan modular dan pemrograman terstruktur adalah suatu aplikasi dapat secara efektif diimplementasikan sebagai sekumpulan proses.

Struktur sistem operasi.

- Keunggulan strukturisasi dapat juga diterapkan ke pemrograman sistem.
- Sistem operasi bermodelkan client/server menggunakan pendekatan ini.

Untuk Strukturisasi Satu Proses.

- Saat ini untuk peningkatan kinerja maka satu proses dapat memiliki banyak thread yang independen.
- Thread-thread tersebut harus dapat bekerjasama untuk mencapai tujuan proses.

Interaksi Antar Proses.

Pada sistem dengan banyak proses, terdapat 2 katagori interaksi, yaitu:

1. Proses-proses Saling Tidak Peduli (Independen).
2. Proses-proses Saling Mempedulikan Secara Tidak Langsung.

KONKURENSI

Masalah yang dihadapi proses-proses kongkurensi pada *multiprogramming* dan *multiprocessing* serupa, yaitu:

- kecepatan eksekusi proses-proses di sistem tidak dapat diprediksi.

KONKURENSI

Kecepatan proses pada sistem tergantung pada beberapa hal, antara lain:

- Aktivitas proses-proses lain
- Cara sistem operasi menangani interupsi
- Kebijaksanaan penjadwalan yang dilakukan oleh sistem operasi.

Beberapa kesulitan yang dapat muncul, di antaranya adalah:

- Pemakaian bersama sumber daya global.
- Pengelolaan alokasi sumber daya agar optimal
- Pencarian kesalahan pemrograman.

Konkurensi

Proses-proses konkuren mengharuskan beberapa hal yang harus ditangani, antara lain:

- Sistem operasi harus mengetahui proses-proses yang aktif
- Sistem operasi harus mengalokasikan dan mendealokasikan beragam sumber daya untuk tiap proses aktif.

Konkurensi

- Sistem operasi harus memproteksi data dan sumber daya fisik masing-masing proses dari gangguan proses-proses lain.
- Hasil-hasil proses harus independen terhadap kecepatan relatif proses-proses lain dimana eksekusi dilakukan.

Pokok Penyelesaian Masalah Kongkurensi

Pada dasarnya penyelesaian masalah kongkurensi terbagi menjadi 2, yaitu:

- Mengasumsikan adanya memori yang digunakan bersama
- Tidak mengasumsikan adanya memori yang digunakan bersama.

MUTUAL EXCLUSION

- Mutual exclusion adalah jaminan hanya satu proses yang mengakses sumber daya pada satu interval waktu tertentu.
- Sumber daya yang tidak dapat dipakai bersama pada saat bersamaan.

MUTUAL EXCLUSION

- Bagian program yang sedang mengakses *memory* atau sumber daya yang dipakai bersama disebut *critical section*. Jika proses pada *critical section* memblokir proses-proses lain dalam antrian, maka akan terjadi *startvation* dan *deadlock*.

MUTUAL EXCLUSION

- Kesuksesan proses-proses konkurensi memerlukan pendefinisian *critical section* dan memaksakan mutual exclusion di antara proses-proses konkuren yang sedang berjalan.

MUTUAL EXCLUSION

Fasilitas atau kemampuan menyediakan dukungan mutual exclusion harus memenuhi 6 kriteria sbb:

- Mutual exclusion harus terjadi proses tunggal.
- Proses yang berada di noncritical section, dilarang mem-blocked proses-proses lain yang ingin masuk critical section.

MUTUAL EXCLUSION

- Harus dijamin bhwa proses yang ingin masuk *critical section* tidak menunggu selama waktu yang tak terhingga
- Ketika tidak ada proses pada *critical section*, maka proses yang ingin masuk *critical section* harus ijinkan masuk tanpa waktu tunda.

MUTUAL EXCLUSION

- Tidak ada asumsi mengenai kecepatan relatif proses atau jumlah yang ada.
- Proses hanya tinggal pada *critical section* selama satu waktu yang berhingga

MUTUAL EXCLUSION

Ada 2 metode yang diusulkan untuk menjamin Mutual Exclusion, antara lain:

- **Metode Variable *Lock* :**
  Locking adalah salah satu mekanisasi pengontrol konkuren.

- **Metode bergantian secara ketat**
  Metode ini melakukan refleksi terhadap variabel yang berfungsi untuk memenuhi critical section.

SINKRONISASI

- Akses-akses yang dilakukan secara bersama-sama ke data yang sama, dapat menyebabkan data menjadi tidak konsisten.
- Untuk menjaga agar data tetap konsisten, dibutuhkan mekanisme-mekanisme untuk memastikan pemintaan ekseskusi dari proses yang bekerja.

SINKRONISASI

- Race Condition: Situasi dimana beberapa proses mengakses dan memanipulasi data secara bersamaan. Nilai terakhir dari data bergantung dari proses mana yang selesai terakhir.
- Untuk menghindari Race Condition, proses-proses secara bersamaan harus disinkronisasikan.

Kasus Produsen-Konsumer

- Dua proses berbagi sebuah buffer dengan ukuran yang tetap. Salah satunya produser, meletakkan informasi ke buffer yang lainnya. Konsumen mengambil informasi dari buffer.

Race Condition

- Race Condition adalah situasi di mana beberapa proses mengakses dan memanipulasi data bersama pada saat besamaan.

Race Condition

- Nilai akhir dari data bersama tersebut tergantung pada proses yang terakhir selesai.
- Untuk mencegah race condition, proses-proses yang berjalan besamaan haus di disinkronisasi.

Race Condition

- Dalam beberapa sistem operasi, proses-proses yang berjalan bersamaan mungkin untuk membagi beberapa penyimpanan umum, masing-masing dapat melakukan proses baca *(read)* dan proses tulis *(write)*.

Race Condition

- Penyimpanan bersama (shared storage) mungkin berada di memori utama atau berupa sebuah berkas bersama, lokasi dari memori bersama tidak merubah kealamian dari komunikasi atau masalah yang muncul.

Critical Section

- **Bagaimana menghindari *race conditions*?**

Kunci untuk mencegah masalah ini adalah menemukan beberapa jalan untuk mencegah lebih dari satu proses untuk melakukan proses writing dan reading kepada shared data pada saat yang sama.

Critical Section

- Masalah menghindari *race conditions* dapat juga diformulasikan secara abstrak.
- Walau pun dapat mencegah *race conditions*, tapi tidak cukup untuk melakukan kerjasama antar proses secara pararel dengan baik dan efisien dalam menggunakan shared data.

Critical Section

4 kondisi agar menghasilkan solusi yang baik:

1. Tidak ada dua proses secara bersamaan
2. Tidak ada asumsi mengenai kecepatan
3. Tidak ada proses yang berjalan, yang dapat mengeblok proses lain.
4. Tidak ada proses yang menunggu

Problem Klasik Sinkronisasi

Ada tiga hal Yaitu :

- **Problem *Bounded buffer*.**
- **Problem *Reades and Writer*.**
- **Problem *Dining Philosophers*.**

Problem Readers-Writers

- Problem *readers-writer* adalah problem yang memodelkan proses yang mengakses database.

Problem Readers-Writers

- Sebagai contoh sebuah sistem pemesanan sebuah perusahaan penerbangan, dimana banyak proses berkompetisi berharap untuk membaca (*read*) dan menulis (*write*).

Problem Dining Philosopers

- Program ini menggunakan sebuah array dari semaphore yang dapat ditahan.

DEADLOCK

- Deadlock adalah suatu kondisi dimana dua proses atau lebih tidak dapat meneruskan eksekusinya oleh pemroses.
- Pada umumnya *deadlock* terjadi karena proses mengalami *startvation*.

DEADLOCK

Kondisi yang dapat menimbulkan terjadinya deadlock:

- *Mutual exclusion*.
- *Hold &Wait*.
- *No Preemption*.
- *CircularWait Condition*.

Metode Mengendalikan *Deadlock* :

- Menggunakan suatu protokol
- Mengijinkan sistem mengalami deadlock
- Mengabaikan semua masalah

Deadlock

Penghindaran Deadlock :

- **State Selamat**
- **State Tak Selamat *(unsafe state)***

Deadlock

Mendeteksi dan Memulihkan *Deadlock* :

1. Terminasi Proses
2. *Resources Preemption*

STARTVATION

- Startvation adalah keadaan dimana pemberian akses bergantian terus menerus, dan ada suatu proses yang tidak mendapatkan gilirannya

TERIMAKASIH