DEADLOCK

Core Jurusan Teknik Informatika
Kode MK/SKS : TIF 19417/ 4

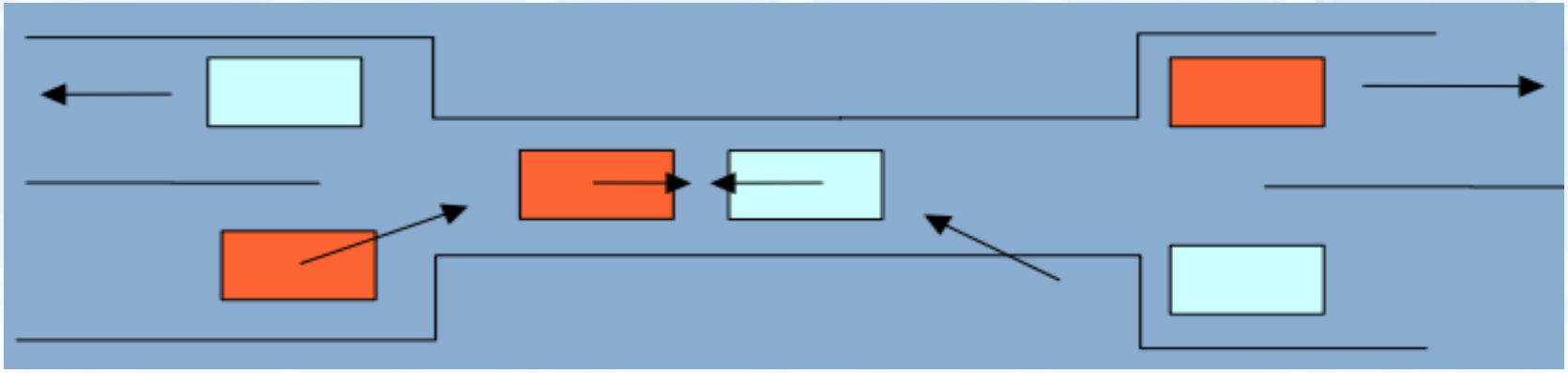
Deadlock

- Jika proses 1 sedang menggunakan sumber daya 1 dan menunggu sumber daya 2 yang ia butuhkan, sedangkan proses 2 sedang menggunakan sumber daya 2 dan menunggu sumber daya 1

Deadlock

- Atau dengan kata lain saat proses masuk dalam status menunggu, ia tidak akan pernah selesai menunggu sebab sumber daya yang dibutuhkan sedang digunakan oleh proses lain yang sedang menunggu pula

Deadlock



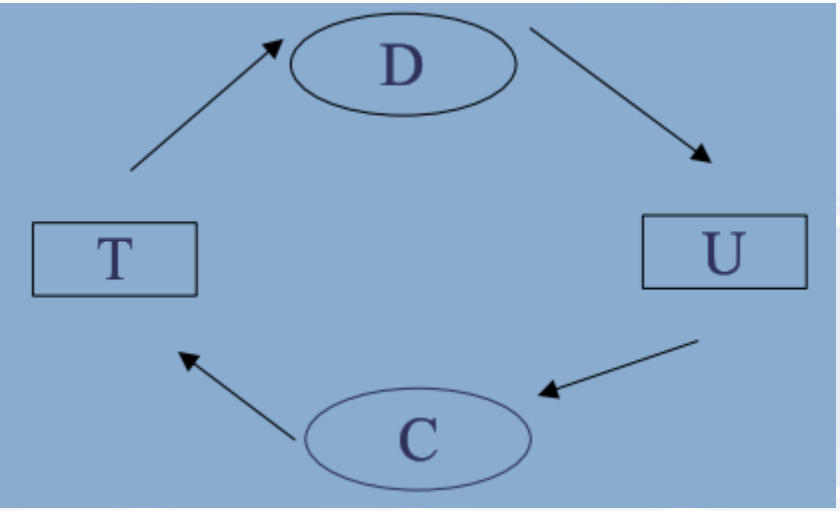
Deadlock

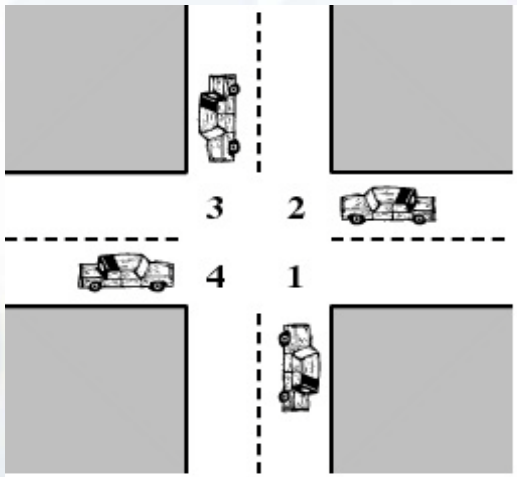
- Pada satu jalan yang memungkinkan hanya satu arah yang berjalan
- Setiap jalan bisa dianggap sebagai sumber daya
- Saat deadlock terjadi hanya bisa diatasi jika salah satu mobil mundur, dalam hal ini butuh sumber daya yang direalokasikan

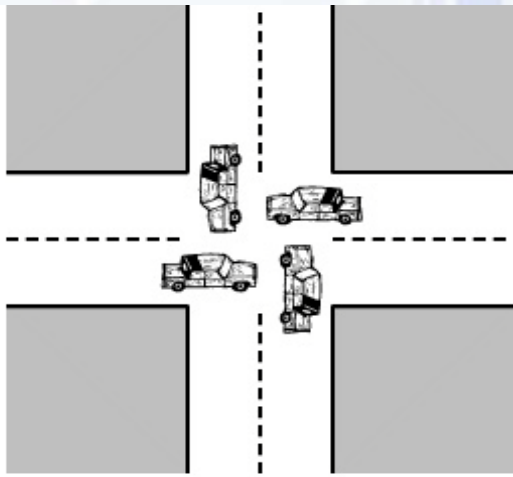
Deadlock

- Bahkan beberapa mobil harus mundur jika deadlock terjadi
- Pada kasus ini juga bisa terjadi "kelaparan", yaitu ada proses yang tidak terlayani

Deadlock







Jenis Resource

- Preemptable: masih boleh diambil dari proses yg sedang memakainya tanpa memberi efek apapun pada proses tersebut
  - Contoh: memory

Jenis Resource

- Non preemptable: tidak boleh diambil dari proses yg sdg memakainya.
  - Contoh: printer
  - Ada potensi menyebabkan deadlock

Penyebab Deadlock

- **Mutual Exclusion**: satu proses satu sumber daya
- **Hold and Wait**: proses yang memegang sumber daya masih bisa meminta sumber daya lain

Penyebab Deadlock

- **No Preemption**: sumber daya yang sedang digunakan oleh suatu proses tidak bisa sembarangan diambil dari proses tersebut, melainkan harus dilepaskan dengan sendirinya oleh proses.

- **Circular Wait**: setiap proses menunggu sumber daya dari proses berikutnya yg sedang dipakai oleh proses lain.

Pencegahan (Deadlock Avoidance)

- Caranya adalah mencegah adanya deadlock dengan memberikan informasi tambahan ttg resource yg dibutuhkan
  - Banyaknya resource yang tersedia
  - Banyaknya resource yang dialokasikan
  - Dan maksimum resource yang dibutuhkan proses

Mengatasi (Deadlock Prevention)

- **Mutual Exclusion:** yaitu dengan cara tidak berbagi data dengan proses lain atau dengan kata lain menyediakan data sendiri

- **Hold and Wait**: penggunaan resource diperbolehkan jika semua sumber daya yang diperlukan tidak digunakan oleh proses lain
  - Sblm suatu proses memakai resource, dia hrs melepaskan resource yg dibawanya

Mengatasi (Deadlock Prevention)

- **No Preemptive:** jika suatu proses meminta resource, dan blm bs dipenuhi, maka proses tersebut hrs membebaskan resourcenya dulu

- **Circular Wait:** Setiap kebutuhan total didata terlebih dahulu
  - Memberi nomor urut pada tiap resource, dan tiap proses hanya boleh menggunakan resource secara berurutan

SINKRONISASI PROSES

Core Jurusan Teknik Informatika
Kode MK/SKS : TIF 19417/ 4

Mengapa Sinkronisasi

- Sinkronisasi diperlukan untuk menghindari terjadinya **ketidakkonsistenan data** akibat adanya akses data secara **konkuren**

Mengapa Sinkronisasi

- Diperlukan adanya suatu mekanisme untuk memastikan **urutan / giliran** pengaksesan suatu data yang saling bekerjasama sehingga terjadi sinkronisasi

- If we don't make process synchronization:
  - **Race Condition**

Race Condition

- Race condition: situasi dimana beberapa proses mengakses dan memanipulasi suatu data secara konkuren.
  - Nilai akhir dari data tersebut tergantung dari proses mana yang terakhir mengubah data

Race Condition

- Untuk menghindari terjadinya situasi tersebut, semua proses yang dapat mengakses suatu data tertentu harus disinkronisasi

Critical Section

- **Lebih dari satu** proses berlomba-lomba pada saat yang sama untuk menggunakan data yang **sama**.
- Setiap proses memiliki segmen kode yang digunakan untuk mengakses data yang digunakan secara bersama-sama. Segmen kode tersebut disebut **critical section**.

Critical Section

- Masalahnya: menjamin bahwa jika suatu proses sedang menjalankan critical section, maka proses lain tidak boleh masuk ke dalam critical section tersebut.

Sinkronisasi Hardware

- Metode dalam sinkronisasi hardware
  - Processor Synchronous (Disable Interrupt)
  - Memory Synchronous (Instruksi Test-And-Set)

- Interrupt used in Round Robin Algorithm, with quantum time

Sinkronisasi Hardware

- Processor synchronous
  - Dengan mendisable interupsi (interrupt)
  - Dalam lingkungan multiprocessor:
    - Hanya satu processor bisa didisable interruptnya

Sinkronisasi Hardware

- Memory synchronous
  - Instruksi Test-And-Set
  - Dalam lingkungan multiprocessor:
    - Bisa dilakukan
    - Semua processor tidak dapat memakai resource karena proteksi dilakukan di memory
  - Instruksi harus bersifat atomik