

OPERATING SYSTEM

IN OUR CLASSROOM


WE RESPECT EACH OTHER.

WE TRY OUR BEST.




WE ARE A TEAM.

WE LEARN FROM MISTAKES.



WE CREATE.

WE CELEBRATE EACH OTHER'S SUCCESS.


KONKURENSI OPERATING SYSTEM BAGIAN I



CAPAIAN PEMBELAJARAN

- Mahasiswa memahami konsep konkurensi sistem operasi
- Mahasiswa memahami kondisi-kondisi sebagai imbas dari konkurensi
- Mahasiswa memahami penyebab dan penanggulangan konkurensi

Agenda.

- Konkurensi
 1. Prinsip Konkurensi
 2. Kesulitan Konkurensi
- Mutual Exclusion
- Deadlock

KONKURENSI

Prinsip-prinsip Konkurensi.

1. Alokasi waktu antar proses
2. Pemakaian bersama dan persaingan sumber daya
3. Komunikasi antar proses
4. Sinkronisasi aktivitas banyak proses

5

KONKURENSI

Permasalahan yang ditimbulkan dari munculnya konkurensi:

1. ***Mutual Exclusion***
2. ***Deadlock***
3. ***Startvation***

6

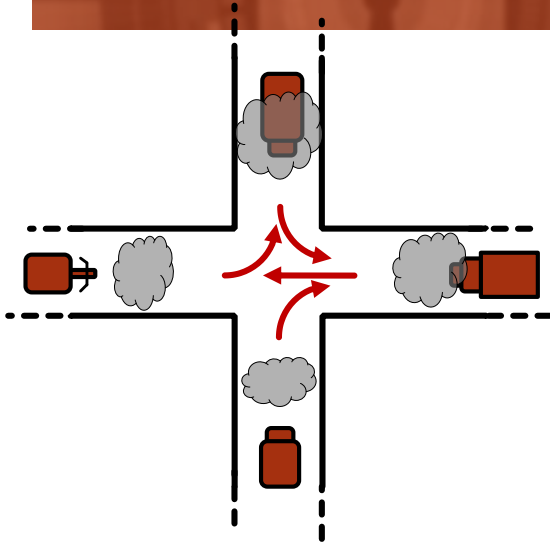
KONKURENSI | MUTUAL EXCLUSION

- Kondisi/kejadian yang muncul ketika lebih dari satu *thread* mengakses *resource* pada waktu yang bersamaan (*critical region*) dan terdapat *thread* yang mendapatkan sumber daya secara khusus (*exclusive*).
- Imbasnya adalah banyak *thread* yang menunggu hingga *thread* yang memiliki hak khusus mengakses sumber daya melepaskan sumberdayanya
- *Bagaimanakah strategi yang harus dilakukan sistem operasi agar tidak terjadi mutual exclusion?*



IT'S A DEADLOCK!

KONKURENSI | DEADLOCK

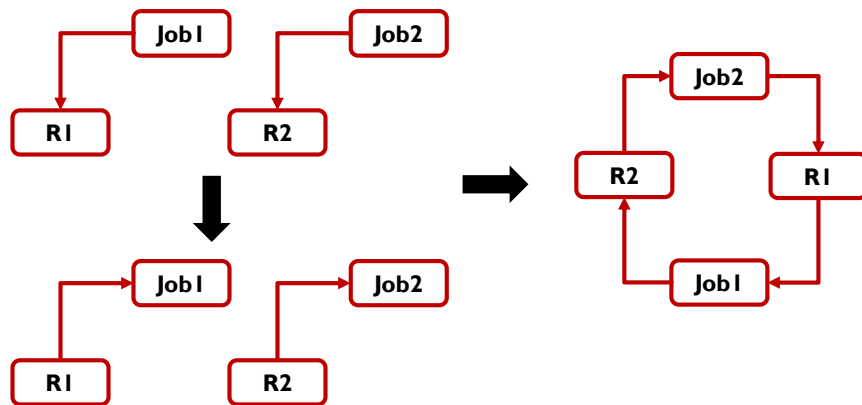


Suatu kondisi dimana setiap thread dalam sebuah *job* yang diproses dalam waktu bersamaan saling tidak melepaskan *resources*

9

KONKURENSI | DEADLOCK

Model Deadlock (Graph).



10

KONKURENSI | DEADLOCK

Pemicu Deadlock.

- ***Mutual Exclusion Condition***

Kondisi dimana terdapat proses/thread yang memiliki akses khusus terhadap sumber daya

- ***Hold & Wait Condition***

Kondisi yang terjadi akibat adanya beberapa thread yang menunggu suatu keadian tertentu untuk melepaskan sumberdaya yang tengah digunakan

11

KONKURENSI | DEADLOCK

- ***Non-preemption Condition***

Kondisi yang diakibatkan oleh adanya sebuah/beberapa thread yang tidak mengijinkan thread lain mendapatkan kesempatan untuk dilayani oleh CPU

- ***Circular Wait Condition***

Kondisi yang menyebabkan thread tidak mendapatkan giliran untuk penggunaan sumber daya yang diakibatkan oleh sebuah/beberapa thread yang mengintervensi kesempatan thread menggunakan sumber daya.

12

KONKURENSI | DEADLOCK

Penanganan Deadlock.

- Metode penanganan *deadlock* terbagi menjadi 3 kelompok:
 1. *Deadlock prevention*
 2. *Deadlock avoidance*
 3. *Deadlock detection & recovery*

13

KONKURENSI | DEADLOCK

Strategi Pencegahan Deadlock.

1. Full Resources
2. Release Resources
3. Linear Queues
4. Menghilangkan *mutual exclusion*
5. Menghilangkan *hold and wait*
6. Menghilangkan *non-preemption*
7. Menghilangkan *circular wait*

14

KONKURENSI | PENCEGAHAN DEADLOCK

■ Menghilangkan *mutual exclusion*

Cara yang dapat dilakukan adalah dengan mengantrikan antrian *spooler* pada *disk*.

Masalah:

1. Tidak semua sumber daya eksklusif dapat di-*spooling*
2. Persaingan terhadap ruang *disk* dapat mengakibatkan *deadlock*

15

KONKURENSI | PENCEGAHAN DEADLOCK

■ Menghilangkan *hold and wait*

Metode yang dapat dilakukan:

Mengalokasikan atau tidak sama sekali *resources*

Masalah:

Sulit untuk mengetahui keseluruhan kebutuhan *resources* setiap proses di awal dan tidak efisien.

16

KONKURENSI | PENCEGAHAN DEADLOCK

- **Menghilangkan *hold and wait***

Setiap proses melakukan mekanisme *hold and release* .

Masalah:

Karena terdapat beberapa proses yang dapat memegang beberapa *resources* sekaligus, maka teknik ini menjadi tidak mungkin untuk dilakukan.

17

KONKURENSI | PENCEGAHAN DEADLOCK

- **Menghilangkan *non-preemptive***

Mencegah proses-proses lain harus menunggu running process selesai dengan melakukan *preemption*

Masalah:

- Tidak mungkin untuk dilakukan.

18

KONKURENSI | PENCEGAHAN DEADLOCK

■ Menghilangkan *circular wait*

Metode yang dapat dilakukan:

1. Proses hanya diperbolehkan menggenggam satu *resource* pada satu saat

Masalah:

Tidak mungkin untuk dilakukan.

2. Penomoran global semua *resources*

Masalah:

Tidak ada mekanisme untuk mengurutkan nomor secara global *resources*

KONKURENSI | PENGHINDARAN DEADLOCK

Safe State.

Proses	Resources yang Digunakan (MB)	Maksimum resources yang dibutuhkan (MB)
JOB1	4	8
JOB2	3	10
JOB3	5	9
Jumlah resources tersedia		7

KONKURENSI | PENGHINDARAN DEADLOCK

I: Misalkan *job 1* mendapatkan alokasi *resources* sebanyak 4 MB

Proses	Resources yang Digunakan (MB)	Maksimum resources yang dibutuhkan (MB)
JOB1	8	8
JOB2	3	10
JOB3	5	9
Jumlah resources tersedia		3

KONKURENSI | PENGHINDARAN DEADLOCK

Maka setelah *job 1* selesai, kondisi *resources* menjadi:

Proses	Resources yang Digunakan (MB)	Maksimum resources yang dibutuhkan (MB)
JOB1	0	0
JOB2	3	10
JOB3	5	9
Jumlah resources tersedia		11

KONKURENSI | PENGHINDARAN DEADLOCK

2: Kemudian *job3* mendapatkan *resources* sebanyak 4 MB

Proses	Resources yang Digunakan (MB)	Maksimum resources yang dibutuhkan (MB)
JOB1	0	0
JOB2	3	10
JOB3	9	9
Jumlah resources tersedia		7

KONKURENSI | PENGHINDARAN DEADLOCK

Maka setelah *job3* selesai, kondisi *resources* menjadi:

Proses	Resources yang Digunakan (MB)	Maksimum resources yang dibutuhkan (MB)
JOB1	0	0
JOB2	3	10
JOB3	0	0
Jumlah resources tersedia		16

KONKURENSI | PENGHINDARAN DEADLOCK

3: Berikutnya giliran *job2* yang mendapatkan *resources* sebanyak 7 MB

Proses	Resources yang Digunakan (MB)	Maksimum resources yang dibutuhkan (MB)
JOB1	0	0
JOB2	10	10
JOB3	0	0
Jumlah resources tersedia		9

KONKURENSI | PENGHINDARAN DEADLOCK

Maka setelah *job2* selesai, kondisi *resources* menjadi:

Proses	Resources yang Digunakan (MB)	Maksimum resources yang dibutuhkan (MB)
JOB1	0	0
JOB2	0	0
JOB3	0	0
Jumlah resources tersedia		19

Dengan demikian ketiga proses dapat menyelesaikan prosesnya dengan sempurna.

KONKURENSI | PENGHINDARAN DEADLOCK

Unsafe State.

Proses	Resources yang Digunakan (MB)	Maksimum resources yang dibutuhkan (MB)
JOB1	4	8
JOB2	3	10
JOB3	4	9
Jumlah resources tersedia		7

KONKURENSI | PENGHINDARAN DEADLOCK

I: Misalkan *job1* dan *job2* masing-masing mendapatkan alokasi resources sebanyak 4 MB dan 3 MB

Proses	Resources yang Digunakan (MB)	Maksimum resources yang dibutuhkan (MB)
JOB1	8	8
JOB2	6	10
JOB3	4	9
Jumlah resources tersedia		0

KONKURENSI | PENGHINDARAN DEADLOCK

Maka setelah diproses, kondisi *resource* menjadi:

Proses	Resources yang Digunakan (MB)	Maksimum resources yang dibutuhkan (MB)
JOB1	0	0
JOB2	6	10
JOB3	4	9
Jumlah resources tersedia		8

KONKURENSI | PENGHINDARAN DEADLOCK

2: Berikutnya giliran *job2* dan *job3* yang mendapatkan *resources* masing-masing sebanyak 4 MB dan 5 MB

Proses	Resources yang Digunakan (MB)	Maksimum resources yang dibutuhkan (MB)
JOB1	0	0
JOB2	10	10
JOB3	8 (-1)	9
Jumlah resources tersedia		0 (-1)

Karena terdapat sebuah proses yang

kekurangan resource disaat resource telah habis maka besar kemungkinan proses pada *job3* tidak akan selesai.

KONKURENSI OPERATING SYSTEM BAGIAN I

