MODUL VII
CCS 210 SISTEM OPERASI

Judul	STRATEGI PENJADWALAN PROSES	
Penyusun	**Distribusi**	**Perkuliahan**
Nixon Erzed	**FASILKOM** **UNIVERSITAS ESA UNGGUL**	Pertemuan – VII Tatap Muka

Tujuan :

Mahasiswa mengenal model-model algoritma penjadwalan dan dapat menganalisa secara manual bekerja model algoritma tersebut.

Materi:

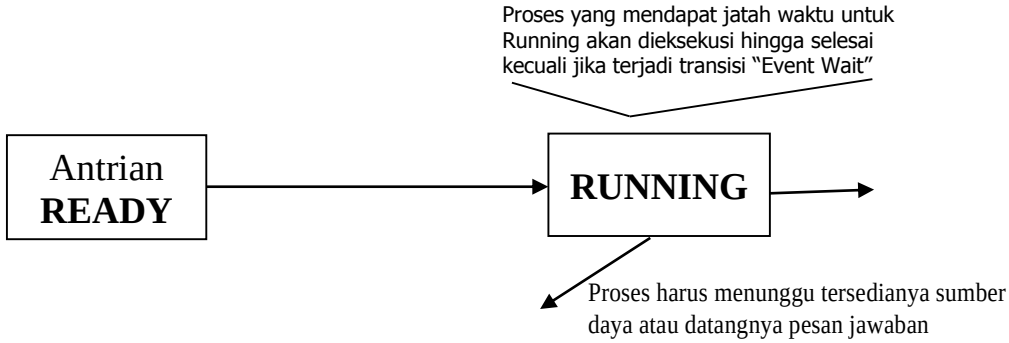
- Model alagoritma non preemptif
  - Algoritma FIFO
  - Algoritma SJF
  - Algoritma HRN
  - Algoritma MFQ
- Model algoritma preemptive
  - Algoritma Round Robin
  - Algoritma SRF
  - Algoritma Varian HRN
  - Algoritma Berprioritas

Referensi :

- Modern Operating System 3th Edition Andrew S Tanembaun 2009
- Operating System, Internals and design Principles, William Stallings 7th Ed. 2012
- Operating System Concepts, Abraham Silberschatz, 9th Ed, 2012
- Sistem Operasi, Bambang Haryanto, Rev.5 2012
- Arsitektur dan Organisasi Komputer, William Stalling, Prehalindo

ALGORITMA PENJADWALAN NON PREEMPTIVE

Pada penjadwalan non preemptive ketika sebuah proses diberi jatah waktu pemroses maka proses tersebut akan dieksekusi hingga menyelesaikan eksekusi jobnya yang terakhir. Secara normal pemroses tidak dapat diambil alih proses lain.



MODEL ALGORITMA NON PREEMPTIVE

Algoritma non preemptive adalah algoritma penjadwalan yang menerapkan/ mengimplementasikan strategi non preemptive.

Terdapat banyak pendekatan logika yang dapat diterapkan untuk mengimplementasikan strategi non preemptive. Berikut ini adalah beberapa model dasar algoritma non preemptive, yaitu :

a. FIFO (First-in, First-out) atau FCFS (First-come, First-serve).

Penjadwalan proses didasarkan pada waktu kedatangan proses atau proses dengan waktu menunggu terlama.

b. SJF (Shortest Job First).

Penjadwalan proses didasarkan pada perkiraan kebutuhan waktu eksekusi proses yang ekivalen dengan ukuran proses (file program).

c. HRN (Highest-Ratio Next).

Penjadwalan proses didasarkan pada suatu rasio, proses dengan nilai rasio tertinggi akan running .

d. MFQ (Multiple Feedback Queues).

Model ini menerapkan multi queue (lebih dari 1 antrian), dan penjadwalan didasarkan pada suatu pola perbadingan kesempatan yang tertentu.

1. Algoritma FIFO (First-in, First-out) atau FCFS (First-come, First-serve).

FIFO adalah algoritma penjadwalan paling sederhana. Mengikuti pola antrian dasar yang mengacu pada waktu tunggu aktual.

Penjadwalan ini merupakan :

- Penjadwalan non-preemptive (run-to-completion).
  Begitu proses mendapat jatah waktu pemroses, proses dijalankan sampai selesai.
- Pemilihan proses yang akan running berdasarkan waktu tunggu aktual
  → berdasarkan waktu kedatangan.
  → tabel antrian akan diurutkan berdasarkan waktu kedatangan proses
- Persoalan yang mungkin muncul : proses; misalnya berukuran kecil, yang datang belakangan harus menunggu proses yang lebih dahulu yang mungkin berukuran sangat besar.

FIFO jarang digunakan secara mandiri tapi dikombinasikan dengan skema lain, misalnya:

- Keputusan dasar adalah berdasarkan FIFO.
- Untuk proses-proses yang datang pada waktu yang sama → diputuskan berdasarkan kriteria tambahan, misalnya berdasarkan skala prioritas

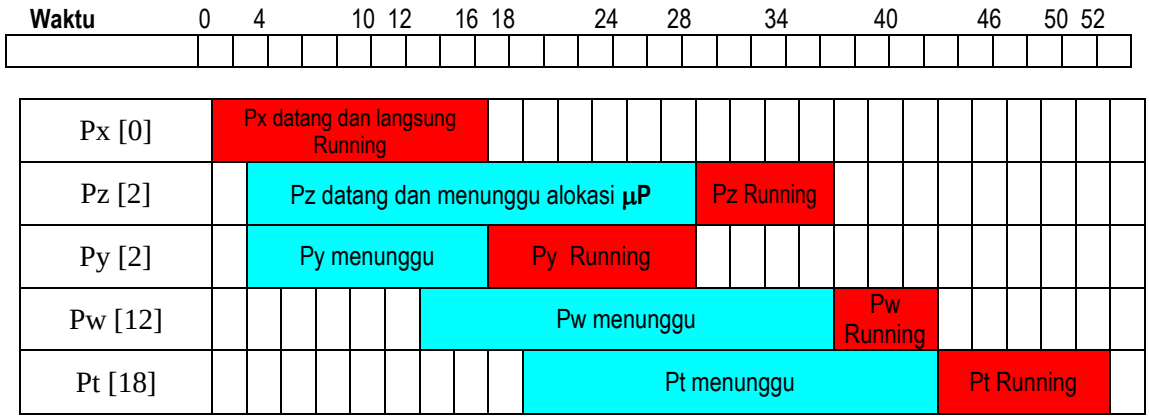
Perhatikan contoh berikut ini

Id proses	Waktu Kedatangan (logon time)	Skala prioritas	Estimasi Waktu Proses (RTE)
P_x	0	4	16
P_z	2	3	8
P_y	2	2	12
P_w	12	1	6
P_t	18	3	10

Pz dan Py datang pada waktu yang bersamaan

Pada data contoh diatas diketahui Proses Pz dan Py datang pada waktu yang bersamaan, sehingga diperlukan kriteria tambahan untuk memutuskan proses yang akan running. Misalnya setiap proses dilengkapi dengan skala prioritas, sehingga kalau diasumsikan 1 lebih baik dari 2, maka pada proses diatas Py mendapat prioritas lebih dahulu dibandingkan Pz.

Untuk memahami bagaimana proses-proses tersebut dijadwalkan oleh prosesor dapat digambarkan dengan Gant Chart berikut ini.



Cara lain penyajian gant chart tersebut adalah sebagai berikut :

0	Px	16	P_y	28	P_z	36	P_w	42	Pt	52

Antrian aktual	***Pz Py Pw***	***Pz Pw Pt***	***Pw Pt***	***Pt***	***-***

Perhitungan kinerja penjadwalan

Id proses	Waktu Kedatangan (logon time)	Skala prioritas	Estimasi Waktu Proses (RTE)	Kinerja	
				Response Time	Turn Arround Time
P_x	0	4	16	0 – 0 = 0	16 – 0 = 16
P_z	2	3	8	28 – 2 = 26	36 – 2 = 36
P_y	2	2	12	16 – 2 = 14	28 – 2 = 26
P_w	12	1	6	36 – 12 = 24	42 – 12 = 30
P_t	18	3	10	42 – 18 = 24	52 – 18 = 34

2. Algoritma SJF (Shortest Job First).

SJF ini merupakan penjadwalan non preemptive (run-to-completion), dan menerapkan penjadwalan tanpa prioritas.
SJF mengasumsikan waktu jalan proses (sampai selesai) diketahui sebelumnya. Mekanisme penjadwalan adalah menjadwalkan proses dengan waktu jalan terpendek lebih dulu sampai selesai. Penjadwalan mempunyai efisien tinggi dan turn arround time rendah.

Walaupun mempunyai turn arround yang bagus, SJF mempunyai masalah yaitu:

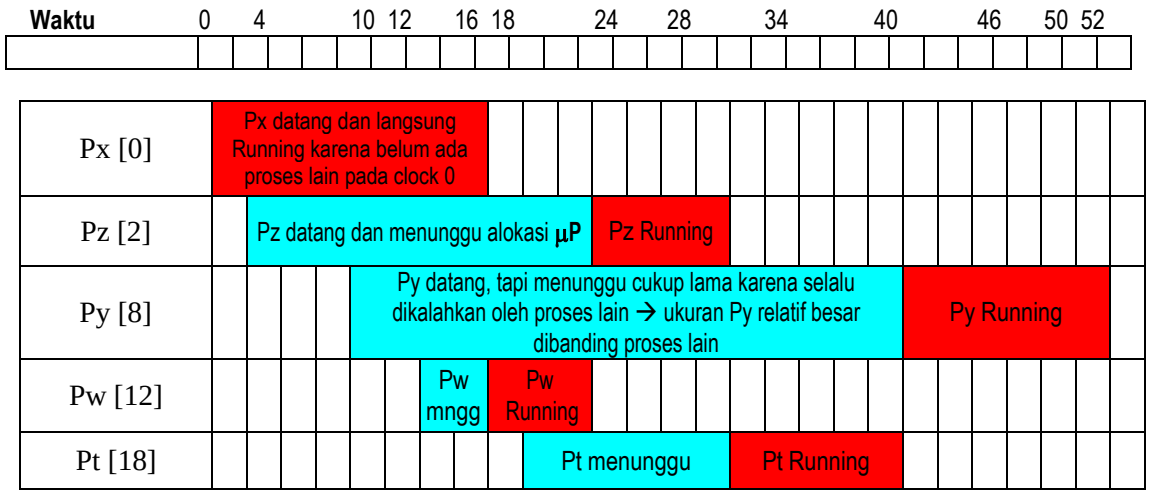
- Tidak dapat mengetahui ukuran job saat job masuk.
- Proses yang tidak datang bersamaan, sehingga penetapannya harus dinamis.
  Untuk mengetahui ukuran job agar dapat ditetapkan yang terpendek biasanya dilakukan pendekatan. Pendekatan yang biasa dilakukan adalah membuat estimasi berdasar kelakuan sebelumnya, atau estimasi berdasarkan size sistem.
  → tabel antrian akan diurutkan berdasarkan waktu kedatangan proses
  → muncul resiko proses yang berukuran besar selalu dikalahkan (terkucil)

Perhatikan contoh berikut ini

Id proses	Waktu Kedatangan (logon time)	Estimasi Waktu Proses (RTE)
P_x	0	16
P_z	2	8
P_y	8	12
P_w	12	6
P_t	18	10

Pemutusan proses yang akan running adalah berdasarkan estimasi waktu proses yang sudah ada didalam antrian Ready, dalam hal ini antrian aktual sesuai dengan waktu kedatangan proses.

Perhatikan Gant Chart berikut ini.



Proses Pw walaupun datang relatif lebih akhir dibanding Pz dan Py tapi mendapat alokasi pemroses lebih dahulu karena ukurannya lebih kecil.

Cara lain penyajian gant chart tersebut adalah sebagai berikut :

Antrian aktual	***Pz [8]*** ***Py [12]*** ***Pw [6]***	***Pz [8]*** ***Py [12]*** ***Pt [10]***	***Py [12]*** ***Pt [10]***	***Py[12]***	**-**

Perhitungan kinerja penjadwalan SJF

Id proses	Waktu Kedatangan (logon time)	Estimasi Waktu Proses (RTE)	Kinerja	
			Response Time	Turn Arround Time
P_x	0	16	0 – 0 = 0	16 – 0 = 16
P_z	2	8	22 – 2 = 20	30 – 2 = 28
P_y	8	12	40 – 8 = 32	52 – 8 = 44
P_w	12	6	16 – 12 = 4	22 – 12 = 10
P_t	18	10	30 – 18 = 12	40 – 18 = 22

3. Algoritma HRN (Highest-Ratio Next).

Pada algoritma FIFO kita dihadapkan pada masalah proses yang misalnya berukuran kecil, yang datang belakangan sehingga harus menunggu proses yang lebih dahulu yang mungkin berukuran sangat besar. Sebaliknya pada penjadwalan SJF proses berukuran lebih kecil diberikan alokasi lebih dahulu, tapi muncul resiko proses yang berukuran besar akan selalu dikalahkan sehingga mengalami pengucilan (starvasion).

Sebagai solusi dari masalah tersebut, dikembangkan algoritma HRN yang mendasarkan pemilihan proses yang akan running pada suatu nilai/ratio yang membandingkan waktu menunggu aktual terhadap kebutuhan waktu proses.
Dengan pola ini didapatkan dasar pemilihan proses yang memperhatikan ukuran proses sekaligus juga lama waktu menunggu. Dalam hal ini proses yang berukuran lebih kecil dan menunggu lebih lama mendapat peluang lebih baik.

Rasio dirumuskan sebagai berikut :

$$\textit{Prioritas (rasio)} = \frac{\textit{waktu tunggu aktual + waktu layanan}}{\textit{waktu layanan}}$$

Untuk lebih jelasnya perhatikan contoh berikut :

1) 2 proses datang pada waktu yang sama (pada waktu ke-4) yaitu P1 berukuran 10 dan P2 berukuran 15.

   Jika dilakukan pengujian pada waktu ke-19, nilai ratio masing-masing proses adalah sebagai berikut :

   Ratio P_1 = [(19-4) + 10] /10 = 2,5

   Ratio P_2 = [(19-4) + 15] / 15 = 2

   Ratio P_1 lebih bagus dari P_2 → karena P_1 ukuran lebih kecil

2) Pengaruh waktu tunggu terhadap ratio, dapat dijelaskan dengan contoh berikut:
   Ingin diketahui Ratio P_1 pada waktu ke-19 dan waktu ke-30 (proses yang sama)

   Waktu ke- 19 → Ratio P_1 = [(19-4) + 10] /10 = 2,5

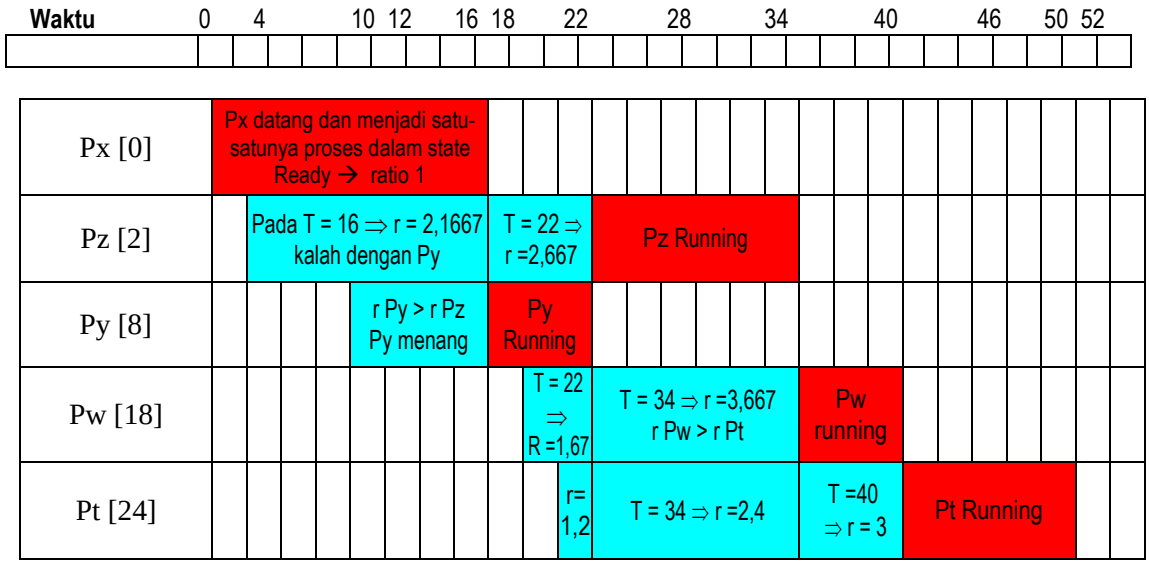
   Waktu ke- 30 → Ratio P_1 = [(30-4) + 10] /10 = 3,6

   Ratio P_1 *akan naik* jika makin lama menunggu

Perhatikan contoh berikut ini

Id proses	Waktu Kedatangan (logon time)	Estimasi Waktu Proses (RTE)
P_x	0	16
P_z	2	12
P_y	8	6
P_w	18	6
P_t	20	10

Perhitungan ratio didasarkan pada status aktual proses dalam antrian.
Untuk memahami bagaimana proses-proses tersebut dijadwalkan oleh prosesor dapat digambarkan dengan Gant Chart berikut ini.



Pada waktu ke-0

- Px datang ke dalam antrian, dan menjadi satu-satunya proses dalam antrian
  - ¤ Waktu datang = 0 dan Waktu pengujian = 0
  - → waktu tunggu aktual : 0 – 0 = 0
  - ¤ Kebutuhan waktu layanan = 16
  - → **ratio Px = (0 + 16)/16 = 1**

Pada waktu ke-16

→ *Px menyelesaikan eksekusi prosesnya dan meninggalkan state Running*
→ *secara aktual pada antrian terdapat proses Pz dan Py*

- **Pz** : waktu datang Pz = 2 dan Waktu pengujian = 16
  - → waktu tunggu aktual : 16 – 2 = 14
  - ¤ Kebutuhan waktu layanan = 12
  - → **ratio Pz = (14 + 12) / 12 = 2,1667**

- **Py :** waktu datang Py = 8 dan Waktu pengujian = 16
  → waktu tunggu aktual : 16 – 8 = 8
  ¤ Kebutuhan waktu layanan = 6
  → **ratio Py = (8 + 6) / 6 = 2,3333**

  → **Py** memiliki rasio lebih baik dibanding **Pz** sehingga
  → **Py mendapat alokasi pemroses**

Pada waktu ke-22

→ Py menyelesaikan eksekusi prosesnya dan meninggalkan state Running
→ secara aktual pada antrian terdapat proses Pz dan Pw

- **Pz** : waktu datang Pz = 2 dan Waktu pengujian = 22
  → waktu tunggu aktual : 22 – 2 = 20
  ¤ Kebutuhan waktu layanan = 12
  → **ratio Pz = (20 + 12) / 12 = 2,667** *→ terjadi kenaikan ratio*

- **Pw :** Waktu datang Pw = 18 dan Waktu pengujian = 22
  → waktu tunggu aktual : 22 – 18 = 4
  ¤ Kebutuhan waktu layanan = 6
  → **ratio Pw = (4 + 6) / 6 = 1,6667**

  → **Py** memiliki rasio lebih baik dibanding **Pz** sehingga
  → **Py mendapat alokasi pemroses**

Demikian seterusnya, setiap proses di state running menyelesaikan eksekusinya, maka akan dihitung ratio aktual proses-proses yang berada dalam antrian

Cara lain penyajian gant chart tersebut adalah sebagai berikut :

0	Px	16	Py	22	Pz	34	Pw	40	Pt	50

Antrian aktual	*Pz=(16-2+12)/12 =2.1667* *Py=(16-8+6)/6 =2,3333*	*Pz=(22-2+12)/12 =2.667* *Pw=(22-18+6)/6 =1,667* *Pt=(22-20+10)/10 =1,2*	*Pw=(34-18+6)/6 =3,667* *Pt =(34-20+10)/10 =2,4*	*Pt =(40-20+10)/10 =3*	-

Perhitungan kinerja penjadwalan HRN

Id proses	Waktu Kedatangan (logon time)	Estimasi Waktu Proses (RTE)	Kinerja	
			Response Time	Turn Arround Time
P_x	0	16	0 – 0 = 0	16 – 0 = 16
P_z	2	12	22 – 2 = 20	34 – 2 = 32
P_y	8	6	16 – 8 = 8	22 – 8 = 14
P_w	18	6	34 – 18 = 16	40 – 18 = 22
P_t	20	10	40 – 20 = 20	50 – 20 = 30

4. Algoritma MFQ (Multiple Feedback Queues).

Algoritma MFQ akan mengelompokkan proses yang datang dalam beberapa antrian. Dasar pengelompokan proses adalah size atau bisa juga berdasarkan prioritas. Setiap antrian akan diberikan kesempatan/peluang, dalam hal ini peluang tersebut dapat sama dan dapat pula didefinisikan berbeda.
Misalnya jika terdapat 3 antrian Q1 Q2 dan Q3 :

- Peluang sama;

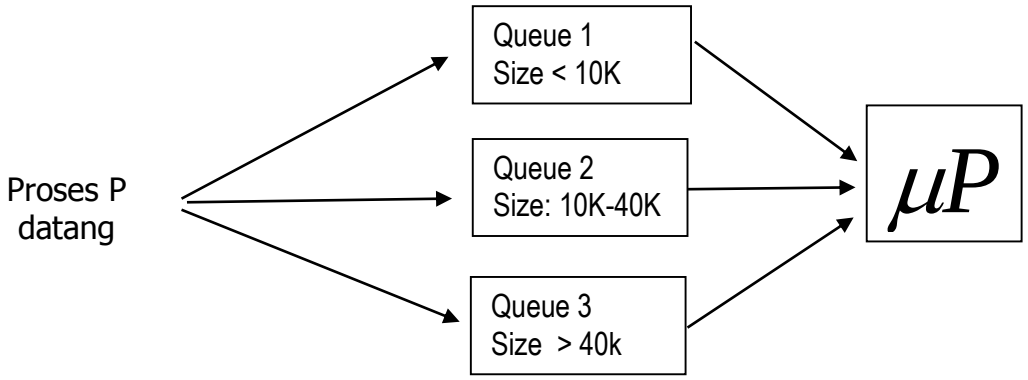
  Q1, Q2 dan Q3 masing-masing mendapat kesempatan 2 kali secara berturut kecuali jika pada antrian tersebut hanya ada satu proses atau tidak ada proses

- Peluang berbeda;

  Q1, Q2 dan Q3 memiliki kesempatan dengan perbandingan;
  → Q1 : Q2 : Q3 = 4 : 2 : 1
  → Setiap Q1 mendapat 2 kali kesempatan, Q2 mendapatkan 1 kali kesempatan
  → Setiap Q2 mendapat 2 kali kesempatan, Q3 mendapatkan 1 kali kesempatan

Contoh

Dimiliki sebuah aturan antrian sebagai berikut :



Dengan perbandingan kesempatan → Q1 : Q2 : Q3 = 4 : 2 : 1
dan akan disimulasikan untuk data proses berikut ini

Id proses	Waktu Kedatangan (AT)	Estimasi Waktu Proses (RTE)	Queue
P_x	0	8	Q1
P_z	1	12	Q2
P_y	3	8	Q1
P_w	7	9	Q1
P_a	12	50	Q3
P_k	20	20	Q2
P_n	30	7	Q1
P_t	32	25	Q2

Menghasilkan simulasi sebagai berikut :

Gantchart **0 Px 8 Py 16 Pz 28 Pw 37 Pn 44 Pk 64 Pa 114 Pt 139**

	Q1	Q1	Q2	Q1	Q1	Q2	Q3	Q2
Q1:	Py Pw	Pw	Pw	Pn	-	-	-	-
Q2:	Pz	Pz	Pk	Pk Pt	PkPt	Pt	Pt	-
Q3:	-	Pa	Pa	Pa	Pa	Pa	-	-

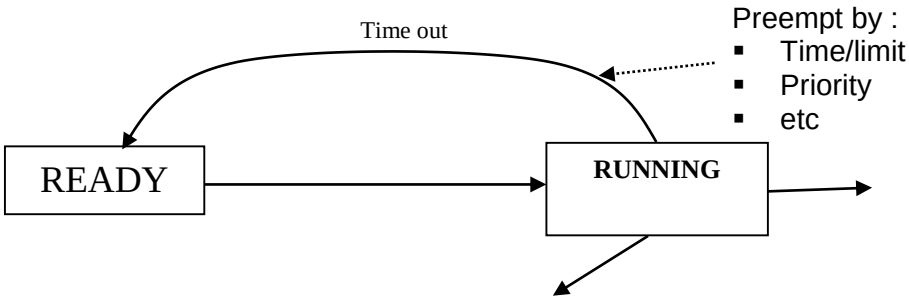
ALGORITMA PENJADWALAN PREEMPTIVE

Pada penjadwalan preemptive ketika sebuah proses diberi jatah waktu pemroses maka pemroses tersebut dapat diambil alih proses lain :

→ proses disela (*preempt)* sebelum selesai

→ harus dilanjutkan/menunggu jatah waktu pemroses tiba kembali pada proses itu

→ proses akan masuk state ready



Penjadwalan preemptive terutama untuk real time system, karena pada Real Time system → proses-proses memerlukan pemroses ***(μP)*** secara tanggapan cepat (dalam limit waktu yang sangat terbatas). Sehingga untuk mencapai tanggapan cepat tersebut, proses yang baru datang harus diizinkan untuk mempreemptif proses yang sedang running.

MODEL ALGORITMA PREEMPTIVE

Algoritma preemptive adalah algoritma penjadwalan yang menerapkan/ mengimplementasikan strategi preemptive. Terdapat banyak pendekatan logika yang dapat diterapkan untuk mengimplementasikan strategi preemptive. Berikut ini adalah beberapa model dasar algoritma preemptive, yaitu :

a. Round Robin.
Merupakan penjadwalan *preemptive by time*, bukan di-preempt oleh proses lain tapi terutama oleh penjadwal berdasarkan lama waktu berjalannya proses.

b. SRF (Shortest Remaining First).
Penjadwalan proses didasarkan pada perkiraan sisa kebutuhan waktu eksekusi proses yang ekivalen dengan ukuran proses (file program).
Sama seperti Round Robin, penjadwalan SRF mengizinkan preemptive berdasarkan lama waktu berjalannya proses, disebut preempt-by-time.

c. Varian HRN (Highest-Ratio Next).
Varian HRN merupakan pengembangan dari Algoritma HRN untuk diimplementasikan sebagai penjadwal preemptive. Dalam hal ini penjadwal akan diberi batas suatu Kwanta (QT) → preempt by time

d. Penjadwalan Berprioritas (PS)
Ide penjadwalan adalah tiap proses diberi prioritas dan proses berprioritas tertinggi running (mendapat jatah waktu pemroses). Preemptif dilakukan berdasarkan batas waktu (preemtive by time). Dalam keadaan tertentu dapat juga by priority.

1. ALGRORITMA PENJADWALAN ROUND-ROBIN (RR)

Penjadwal Round Robin merupakan :

- Penjadwalan preemptive, bukan di-preempt oleh proses lain tapi terutama oleh penjadwal berdasarkan lama waktu berjalannya proses, disebut preempt-by-time.
- Semua proses dianggap penting dan diberi sejumlah waktu pemroses yang disebut kwanta (quantum) atau time-slice dimana proses itu berjalan.
  - Proses mendapat jatah waktu tertentu untuk Running
  - jika batas waktu habis maka proses time out (kembali masuk antrian)
  - Penjadwalan berprioritaskan waktu kedatangan pada antrian (state Ready)

Perhatikan contoh berikut ini :

Id proses	Waktu Kedatangan (AT)	Estimasi Waktu Proses (RTE)
P_x	0	16
P_z	2	8
P_y	3	12
P_w	8	7
P_t	18	10
Pk	25	2

Pemutusan proses yang akan running adalah berdasarkan kedatangan proses didalam antrian Ready, dalam hal ini antrian aktual sesuai dengan waktu kedatangan proses. Proses running dibatasi oleh waktu kwanta (preempt by time). Proses yang kehabisan jatah waktu kembali masuk kedalam antrian.
Untuk memahami bagaimana proses-proses tersebut dijadwalkan oleh prosesor dapat digambarkan dengan Gant Chart berikut ini.

Round Robin (QT = 5)

Gant Chart

0	5	10	15	20	25	28	33	38	43	45	47	49	54	55
Px	Pz	Py	Px	Pw	Pz	Py	Pt	Px	Pw	Pk	Py	Pt	Px	

Queue

Pz (2) Py (3)	Py (3) Px(5) Pw(8)	Px(5) Pw(8) Pz (10)	Pw Pz Py Pt (18)	Pz Py Pt Px(20)	Py(15) Pt(18) Px(20) Pw(25) Pk(25)	Pt Px Pw Pk	Px Pw Pk Py	Pw Pk Py Pt	Pk Py Pt Px	Py Pt Px	Pt Px	Px	-

Perhitungan kinerja penjadwalan

Id proses	Waktu Kedatangan (logon time)	Estimasi Waktu Proses (RTE)	Kinerja	
			Response Time	Turn Arround Time
P_x	0	16	0	55-0=55
P_z	2	8	5-2=3	28-2=26
P_y	3	12	10-3=7	49-3=46
P_w	8	7	20-8=12	45-8=37
P_t	18	10	33-18=15	54-18=36
Pk	25	2	45-25=20	47-25=22

2. ALGORITMA PENJADWALAN SRF (SHORTEST REMAINING FIRST).

- Sama seperti RR → preemptive berdasarkan lama waktu berjalannya proses, disebut preempt-by-time.
- Pemilihan proses yang akan running berdasarkan ukuran proses → estimasi kebutuhan/sisa waktu proses → dasar estimasi : besar/ukuran program
  → tabel antrian akan diurutkan berdasarkan kebutuhan/sisa waktu proses
  → muncul resiko proses yang berukuran besar selalu dikalahkan (terkucil)

Perhatikan contoh berikut ini :

Id proses	Waktu Kedatangan (AT)	Estimasi Waktu Proses (RTE)
P_z	2	8
P_y	3	12
P_w	8	7
P_t	18	10
Pk	25	2

Pemutusan proses yang akan running adalah berdasarkan sisa kebutuhan waktu proses yang sudah ada didalam antrian Ready, dalam hal ini antrian aktual sesuai dengan waktu kedatangan proses. Proses running dibatasi oleh waktu kwanta (preempt by time). Proses yang kehabisan jatah waktu kembali masuk kedalam antrian.

Untuk memahami bagaimana proses-proses tersebut dijadwalkan oleh prosesor dapat digambarkan dengan Gant Chart berikut ini.

SRF (QT = 5)

Gant Chart

0	5	10	15	18	20	25	30	32	37	42	47	48	53	55
Px	Pz	Pw	Pz	Pw	Pt	Px	Pk	Pt	Px	Py	Px	Py	Py	

Queue

Pz (8) Py (12)	Py (12) Px(11) Pw(7)	Py(12) Px(11) Pz(3)	Py(12) Px(11) Pw(2)	Py(12) Px(11) Pt(10)	Py(12) Px(11)	Py(12) Pk(2) Pt(5)	Py(12) Pt(5) Px(6)	Py(12) Px(6)	Py(12)	Px(1)	Py(7)	-	-

Perhitungan kinerja penjadwalan

Id proses	Waktu Kedatangan (logon time)	Estimasi Waktu Proses (RTE)	Kinerja	
			Response Time	Turn Arround Time
P_x	0	16	0	48-0=48
P_z	2	8	5-2=3	18-2=16
P_y	3	12	42-3=39	55-3=52
P_w	8	7	10-8=2	20-8=12
P_t	18	10	20-18=2	37-18=19
Pk	25	2	30-25=5	32-25=7

3. ALGORITMA PENJADWALAN VARIAN HRN

Algoritma HRN yang mendasarkan pemilihan proses yang akan running pada suatu nilai/ratio yang membandingkan waktu menunggu aktual terhadap kebutuhan waktu proses. Dalam implementasi Algoritma HRN sebagai penjadwal preemptive diberikan suatu batas waktu (kwanta) (preempt by quantum/time).
Dengan pola ini didapatkan dasar pemilihan proses yang memperhatikan ukuran proses sekaligus juga lama waktu menunggu. Dalam hal ini proses yang berukuran lebih kecil dan menunggu lebih lama mendapat peluang lebih baik.

Terdapat dua kemungkinan metoda dalam menentukan rasion.

a. Kemungkinan 1 : berdasarkan waktu kedatangan pertama kali di atrian ready

$$\text{Prioritas (rasio)} = \frac{\text{Waktu tunggu dari SUBMIT + sisa waktu layanan}}{\text{Sisa waktu layanan}}$$

b. Kemungkinan 2 : berdasarkan waktu kedatangan terakhir di antrian ready

$$\text{Prioritas (rasio)} = \frac{\text{Waktu tunggu dari TIME OUT + sisa waktu layanan}}{\text{Sisa waktu layanan}}$$

Perhatikan contoh berikut ini :

Id proses	Waktu Kedatangan (AT)	Estimasi Waktu Proses (RTE)
P_x	0	16
P_z	2	12
P_y	3	11
P_w	12	7

Ingin diterapkan algoritma penjadwalan varian HRN (preemptive), untuk memahami bagaimana proses-proses tersebut dijadwalkan oleh prosesor dapat digambarkan dengan Gant Chart berikut ini.

Varian HRN (QT = 10)

dengan penghitungan rasio didasarkan pada waktu tunggu dari time out.

0 Px 10 Pz 20 Px 26 Pz 28 Pw 35 Py 45 Py 46

Antrian aktual	*Pz=(10-2+12)/12 =1,667* *Py=(10-3+11)/11 =1,6363*	*Py=(20-3+11)/11 =2.5555* *Pw=(20-12+7)/7 =2,142* *Px=(20-10+6)/6 =2,6667*	*Py=(26-3+11)/11 =3.0909* *Pw=(26-12+7)/7 =3,00* *Pz=(26-20+2)/2 =4,00*	*Py=(28-3+11)/11 =3.2727* *Pw=(28-12+7)/7 =3,285*	*Py=(35-3+11)/11 =3.9090*	-	-

Perhitungan kinerja penjadwalan

Id proses	Waktu Kedatangan (logon time)	Estimasi Waktu Proses (RTE)	Kinerja	
			Response Time	Turn Arround Time
P_x	0	16	0	26-0=26
P_z	2	12	10-2=8	28-2=26
P_y	3	11	35-3=32	46-3=43
P_w	12	7	28-12=16	35-12=23

4. PENJADWALAN BERPRIORITAS (PRIORITY SCHEDULE)

- Ide penjadwalan adalah tiap proses diberi prioritas dan proses berprioritas tertinggi running (mendapat jatah waktu pemroses).
  → jika terdapat proses-proses dengan prioritas sama, maka pemilihan proses dapat didasarkan pada:
  - waktu kedatangan dalam antrian (seperti RR)
  - kebutuhan waktu proses (seperti SRF)
- Preempt → umumnya by time
  Dalam keadaan tertentu dapat juga by priority
- Prioritas dapat diberikan secara :
  - Prioritas statis (static priorities).
  - Prioritas dinamis (dynamic priorities)

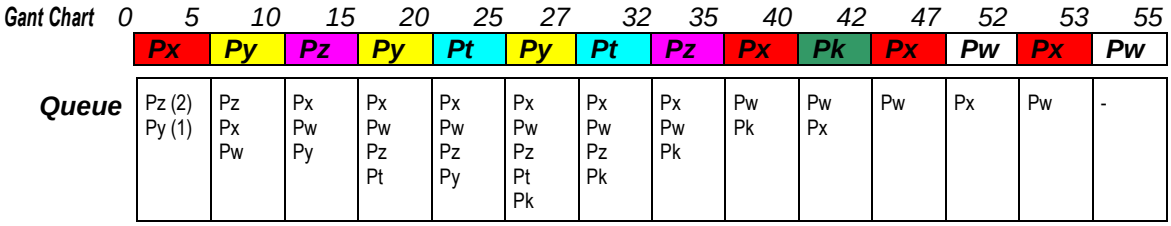
Perhatikan contoh berikut

Id proses	**Waktu Kedatangan (AT)**	**Estimasi Waktu Proses (RTE)**	**Prioritas**				
			Statis	**Dinamis**			
				awal	**1**	**2**	**3**
P_x	0	16	3	0	0	0	0
P_z	2	8	2	10	9	8	7
P_y	3	12	1	10	9	8	8
P_w	8	7	4	10	10	9	8
P_t	22	10	1	0	0	0	0
Pk	25	2	3	10		10	10

Catatan → prioritas 1 lebih baik dari prioritas 2

Untuk Prioritas Statis dapat digambar Gant Chart sebagai berikut :

PS (QT = 5) Statis



Prioritas Statis

Prioritas statis berarti prioritas tak berubah
Keunggulan

- Mudah diimplementasikan.
- Mempunyai overhead relatif kecil.

Kelemahan

- Penjadwalan tak tanggap perubahan lingkungan yang mungkin menghendaki penyesuaian prioritas.

Prioritas Dinamis

Prioritas Dinamis berarti prioritas akan disesuaikan selama waktu aktif proses.
Keunggulan

- Prioritas dinamis merupakan mekanisme menanggapi perubahan lingkungan sistem beroperasi.
- Prioritas awal yang diberikan ke proses mungkin hanya berumur pendek setelah disesuaikan ke nilai yang lebih tepat sesuai lingkungan.

Kelemahan
Implementasi mekanisme prioritas dinamis lebih kompleks dan mempunyai overhead lebih besar. Overhead ini diimbangi dengan peningkatan daya tanggap sistem.