Contoh :

Seorang desainer Complier mencoba untuk memutuskan antara dua urutan kode (code sequence) untuk suatu mesin. Berdasarkan dari implementasi perangkat keras terdapat tiga kelas instruksi: kelas A, kelas B dan kelas C dan masing-masing memerlukan satu, dua dan tiga siklus.

- Urutan kode pertama memiliki 5 instruksi: 2 dari A, 1 dari B dan 2 dari C
- Urutan kode kedua memiliki 6 instruksi : 4 dari A, 1 dari B dan 1 dari C

a. urutan kode mana yang lebih cepat? dan seberapa cepat?

b. Berapa CPI untuk masing-masing urutan kode

Jawab:

a.

CPU clock cycle1 = $\sum_{i=1}^{n} CPI_i x C_i = (2x1) \quad + \quad (1x2) \quad + \quad (2x3) \quad = \quad 10 \quad siklus$

CPU clock cyle2= $\sum_{i=1}^{n} CPI_i x C_i = (4x1) \quad + \quad (1x2) \quad + \quad (1x3) \quad = \quad 9 \quad siklus$

Speedup = 10/9 = 1,11

b.

CPI1 = 10/5 = 2

CPI2 = 9/6 = 1,5

Contoh ini memperlihatkan betap berbahayanya menganalisis komputer hanya dari satu metriks. Saat membandingan dua komputer, harus ditinjau dari 3 metriks yaitu clock rate, jumlah dari instruksi dan CPI.

Contoh 2.

Tinjau penggunaan MIPS ISA dengan 500 MHz clock dan

- Setiap instruksi ALU memerlukan 3 clock cycle
- Setiap instruksi branch/jump memerlukan 2 clock cycle
- setiap intruksi sw memerlukan 4 clock cycle
- setiap instruksi lw memerlukan 5 clock cycle

Juga tinjau sebuah program selama eksekusinya melakukan:

- x = 200 juta instruksi ALU
- y = 55 juta instruksi branc/jump instruksi
- z = 25 juta instruksi sw (store word)
- w = 20 juta instruksi lw (load word)

Carilah CPU Time

a. Pendekatan Pertama:

Clock Cycle untuk program = (x.3 +y.2 + z.4 + w.5)= 910.10^6 clock cycle

CPU Time = Clock cycle untuk program / Clock rate = 910.10^6 / 500.10^6 = 1,82 detik

b. Pendekaktan kedua:

CPI = Clock cycle untuk program / Instruction count

CPI = (x.3 + y.2 + z.4 + w.5) / (x + y + z + w) = 3,03 clock cycle/instruction

CPU Time = Instruction count x CPI / Clock rate = (x + y + z + w) x 3.03 / 500.10^6

CPU Time = (200 + 55 + 25 + 20).10^6 x 3,03 x 500.10^6 = 1,82 detik

Contoh 3:

Tinjau penggunaan MIPS ISA dengan 1 GHz clock dan

- Setiap instruksi ALU memerlukan 4 clock cycle
- Setiap instruksi branch/jump memerlukan 3 clock cycle
- setiap intruksi sw (store word) memerlukan 5 clock cycle
- setiap instruksi lw (load word) memerlukan 6 clock cycle

Untuk program yang sama seperti contoh 2, carilah CPI dan CPU Time

Jawab:

CPI = (x.4 + y.3 + z.5 + w.6) / (x + y + z + w) = 4,03 clock cycle/instruction

CPU Time = Instuction count x CPI / Clock rate = (x + y +z + w) x 4,03/ 1000.10^6

CPU Time = 300 . 10^6 x 4,03 / 100.10^6 = 1,21 detik

Contoh 4: Perhitungan CPI menggunakan frekuensi instruksi

Tabel di bawah ini memperlihatkan frekuensi dari beberapa instruksi yang dieksekusi pada suatu progam, serta jumlah siklus dari masing-masing instruksi

Tentukan CPI nya

Jenis Instruksi	Frekuensi	Siklus
Instruksi ALU	50%	4
Instruksi Load	30%	5
Instruksi Store	5%	4
Instruksi Branch	15%	2

Maka

CPI = 0,5*4 + 0,3*5 + 0,05*4 + 0,15*2 = 4 cycle/instruction

Contoh 5:

Dari tabel dibawah ini tentukan

a. seberapa cepat komputer jika penambahan data cachenya mengurangi waktu pembebanan rata-rata (average load time) menjadi 2 siklus

b. bagaiman jika dibandingkan dengan prediksi instruksi branch yang memangkas waktu siklus instruksi branch setengahnya

c. Bagaimana jika dua instruksi ALU dapat dikerjakan dengan sekali eksekusi

d. Seberapa cepat CPU Time sekarang untuk soal a sampai c

Jenis Instruksi	Frekuensi	CPI_i	Frekuensi x CPI_i
ALU	50%	1	
Load	20%	5	
Store	10%	3	
Branch	20%	2	
			$\sum$

Jawab:

Untuk nomor a - c

Jenis Instruksi	Frekuensi	CPI_i	Frekuensi x CPI_i	Kasus a (CPI load =2)	Kasus b (CPI Branch=1)	Kasus c (CPI ALU = 0,5)
ALU	50%	1	0,5	0,5	0,5	**0,25**
Load	20%	5	1	0,4	1	1
Store	10%	3	0,3	0,3	0,3	0,3
Branch	20%	2	0,4	0,4	**0,2**	0,4
		$\sum$	2,2	1,6	2,0	1,95

d.

Kasus	CPU TIME/CPU TIme baru
a	2,2 / 1,6 =1,375
b	2,2 / 2,0 = 1,1
b	2,2 / 1,95 = 1,128

Berarti untuk kasus a lebih cepat 37,5%, kasus b lebih cepat 10% dan untuk kasus c lebih cepat 12,8%

2.2. MIPS (Millions Instruction per Second) Rating

Sebagai catatan jika membicarakan MIPS berarti pada Mikroprosesor tidak terjadi kondisi pipeline terkunci (interlocking pipeline stages). MIPS merupakan alternatif dari metriks yang mengacu pada waktu. Untuk suatu program, MIPS rating atau laju MIPS berarti berapa juta instruksi yang dieksekusi per detik.

JIka :

- waktu yang diperlukan untuk satu instruksi = CPI * CT
- berarti instruksi yang dieksekusi per detik = 1 / (CPI * CT)

maka

MIPS = 1 / (CPI * CT * 10^6) = CR / (CPI x 10^6) = IC (Execution Time x 10^6)

Terdapat tiga masalah saat menggunakan MIPS sebagai metriks dala analisa kinerja sistem komputer

1. MIPS tidak mewakili kapabilitas instruksi, tidak dapat membanding dua komputer dengan kumpulan instruksi yang berbeda menggunakan MIPS karena instruction count-nya akan berbeda
2. MIPS bervariasi nilainya untuk beberapa program pada komputer yang sama, komputer tidak memiliki nilai tunggal MIPS untuk semua program
3. MIPS bisa bervariasi nilanya dan berbanding terbalik dengan kinerja. Nilai MIPS yang besar tidak berarti kinerjanya lebih besar atau waktu eksekusinya lebih baik. Semua tergantung compiler yang digunakan

Namun eksekusi yang cepat berarti MIPS ratingnya besar.

Contoh :

Dua compiler yang berbeda diuji pada mesin 4 GHz dengan tiga kelas instruksi yang berbeda: Kelas A, Kelas B dan Kelas C yang masing-masing memerlukan 1, 2 dan 3 siklus.

Kedua compiler digunakan untuk menghasilkan kode untuk sebuah bagian besar dari perangkat lunak.

- Kode dari Compiler pertama memerlukan 5 juta instruksi kelas A, 1 juta instruksi kelas B dan 1 juta instruksi kelas C
- Kode dari Compiler kedua memerlukan 10 juta instruksi kelas A, 1 juta instruksi kelas B dan 1 juta instruksi kela C

a Berdasarkan waktu eksekusi uruatan manya yang tercepat?

b. Berdasarkan MIPS urutan (sequence) mana yang tercepat?

Jawab:

a. Execution Time $= \dfrac{CPU \quad clock \quad cycle}{Clock \quad rate}$

CPU clock cyle1 $= \sum CPI_i x C_i = \left[(5x1) \quad + \quad (1x2) \quad + \quad (1x3)\right] x10^6 = 10x10^6$

CPU clock cyle2 $= \sum CPI_i x C_i = \left[(10x1) \quad + \quad (1x2) \quad + \quad (1x3)\right] x10^6 = 15x10^6$

Execution Time1 = $10.10^6/4.10^9$ = 2,5 ms

Execution Time2 = $15.10^6/4.10^9$ = 3,75 ms

b. $MIPS = \dfrac{Instruction \quad count}{Execution \quad time \quad x \quad 10^6}$

MIPS1 $= \dfrac{7x10^6}{2{,}5x10^{-3} \quad x \quad 10^6} = 2800$

MIPS2 $= \dfrac{12x10^6}{3{,}7x10^{-3} \quad x \quad 10^6} = 3200$

Berarti kode dari compiler 2 memiliki MIPS rating terbesar namun compiler 1 lebih cepat

2.2.1 Kinerja Lup MIPS

Untuk lup berikut:

```
for (i=0; i<1000; i=i+1){
    x[i] = x[i] + s; }
```

maka dalam kode asemblernya:

```
lw $3, 8($1)                # load s in $3
addi $6, $2, 4000           # $6 = address of last element + 4
loop: lw $4, 0($2)          # load x[i] in $4
      add $5, $4, $3        # $5 has x[i] + s
      sw $5, 0($2)          # store computed x[i]
      addi $2, $2, 4        # increment $2 to point to next x[ ] element
      bne $6, $2, loop      # last loop iteration reached?
```

kode MIPS dieksekusi pada CPU dengan spesifikasi 500 MHz (clock cycle rate 2ns) dengan jenis instruksi CPI sebagai berikut:

Jenis Instruksi	CPI
ALU	4
Load	5
Store	7
Branch	3

untuk kode MIPS yang dijalankan pada CPU ini tentukan

a. Fraksi instruksi terhadap total instruksi untuk masing-masing jenis instruksi
b. Total jumlah siklus CPU (CPU cycle)
c. CPI rata-rata
d. fraksi dari waktu eksekusi untuk masing-masing instruksi
e. Waktu Eksekusi.
f. MIPS Rating

Jawab:

a. dari program bahasa mesin dapat dihitung jumlah instruksi

Jenis Instruksi	Jumlah Instruksi	Fraksi instruksi
ALU	1999	1999/4997=40%
Load	1000	1000/4997=20%
Store	999	999/4997=20%
Branch	999	999/4997=20%
Total	4997	

b. Total CPU Clock cycle= [(0,4*4) + (0,2*5) + (0,2*7) + (0,2*3)]x10^6 = 4,6 x 10^6 siklus

c. CPI rata-rata $= \frac{CPU \quad Clock \quad cyle}{instruction \quad count} = \frac{4{,}6x10^6}{4997} = 921 \quad siklus$/instruksi

d. Fraksi dari waktu eksekusi

Jenis Instruksi	Fraksi instruksi	CPI	Fraksi x CPI	Fraksi waktu
ALU	40%	4	1,6	1,6/4,6=35%
Load	20%	5	1	1/4,6=22%
Store	20%	7	1,4	1,4/4,6=30%
Branch	20%	3	0,6	0,6/4,6=13%
Total			4,6	

e. Waktu Eksekusi $= \frac{CPU \quad Clock \quad cycle}{CPU \quad clock \quad rate} = \frac{4{,}6 \quad x10^6}{500x10^6} = 0{,}0092 \quad detik$

f. MIPS rating

$$= \frac{Instruction \quad Count}{Execution \quad time \quad x \quad 10^6} = \frac{4997}{0{,}0092x10^6} = 0{,}54 \quad juta \quad instruksi/detik$$

2.3. Metriks pada Analisa Kinerja Sistem Jaringan yang memiliki multitasking

Sebagai contoh untuk menganalisis kualitas layanan suatu jaringan (Quality of Services, QoS) dipilih metriks sebagai berikut

a. Waktu tanggap (*response time*)
b. Throughput
c. Availabilitas
d. Reliabilitas
e. Security
f. Scalabilitas
g. Extensibilitas

Penjelasan dari masing-masing metriks dan penggunaanya adalah sebagai berikut:

Waktu Tanggap

Waktu suatu task menanggapi perintah awal penugasan. Pada jaringan misal ingin di amati waktu tanggap sebelum kondisi jaringan sibuk maka dipilih tiga proses utama (thread):

1. Waktu browser (browser time)

2. Waktu Jaringan (Network Time)
3. Waktu server e-commerce

Maka masing-masing task yang diamati waktu tanggapnya sesuai dengan thread ditunjukkan pada tabel 2.1 berikut

Tabel 2.1. Pemilahan Waktu Respon

Waktu Browser		**Waktu jaringan**			**Waktu Server e-commerce**		
Pemrosesan oleh Prosesor	Sumber daya I/O	Waktu Browser ke ISP	Waktu internet	Waktu ISP ke server	Pemrosesan oleh Prosesor	Sumber daya I/O	Jaringan

Ketiga proses utama dapat dijadikan metriks, jika ingin lebih detail masing-masing tugas (task) yang berada di proses utama bisa dijadikan metriks. Ada variabel yang tidak bisa dijadikan metriks untuk kasus kemacetan (congention) jaringan yaitu waktu layanan (service time) karena waktu layanan oleh server tidak ditentukan oleh jumlah beban. Namun semakin banyak browser yang dibuka maka kemacetan jaringan akan semakin cepat terjadi karena itu dapat dikatakan kemacetan jaringan bergantung pada beban (*load dependent*)

Throughput

Throughput dalam sistem komputer seringkali tidak diterjemahkan ke dalam bahasa Indonesia, ada yang menerjemahkan menjadi jumlah keluaran, jumlah produksi atau jumlah capaian. Throughput sendiri merupakan nilai yang merepresentasikan jumlah tugas yang selesai diproses untuk kurun waktu yang dipilih. Karena satuannya per waktu maka throughput secara ilmu fisika merupakan suatu laju (rate). Untuk kasus ini throughput yang dipilih adalah :

a. Jumlah sumberdaya yang digunakan (I/O per sec)
b. Jumlah halaman browser yang di download (Page per sec)
c. Jumlah permintaan http (http request per sec)
d. Jumlah tugas yang selesai (task per sec)
e. Jumlah transaksi yang selesai (transaction per sec, tps

Contoh Soal 2.1.

Sebuah pengerjaan I/O pada harddisk pada saat sistem transaksi OLTP (*Online Transanction Process*) berlangsung rata-rata memerlukan waktu 10 ms. Untuk 1 menit proses hitung:

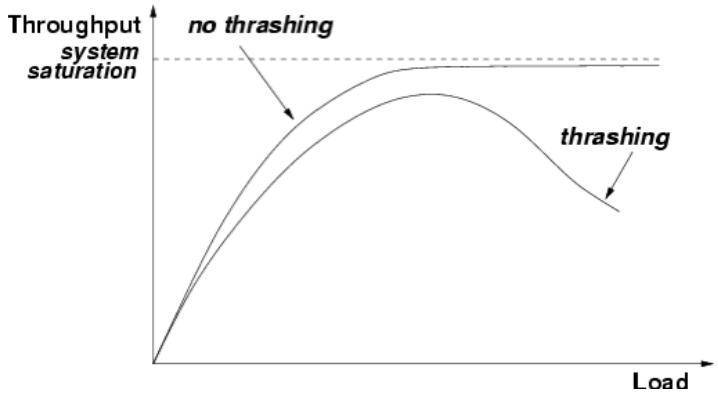
a. Berapa throughput maksimum dari harddisk
b. Jika permintaan I/O pada laju 80 permintaan per detik berapa throughput yang dihasilkan

Jawab:

Asumsi tidak ada proses pembuangan task (thrashing) pada manajemen jaringan

a. Throughput maksimum adalah 1 menit / 10 ms = 60/0,1 =600 throughput
b. Throughput = 60/80 = 0,75 karena dibawah satu maka belum ada proses yang selesai atau throughputnya adalah nol

Hasil ini akan berbeda jika ada manajemen jaringan (Gambar 2.1)



Gambar 2.1 Throughput yang dihasilkan untuk peningkatkan beban.

Dari gambar 2.1 ada yang menarik yaitu sistem yang sedang dianalisa memiliki batas saturasi. Batas ini menunjukkan untuk beban yang terus meningkat server ada throughput maksimum yang bisa dihasilkan. Sedangkan pada kurva dengan manajemen memori (trashing), saat beban puncak maka sistem jaringan akan mengintruksikan prosesor untuk memilih thread yang mana yang perlu ditunda dulu, dan memberikan prioritas pada thread yang harus didahulukan. Manajemen jaringan ini membuat jumlah antrian thread menjadi lebih sedikit yang perlu diselesaikan dan juga menyebabkan jumlah throughput yang diselesaikan lebih sedikit. Pada kurva terlihat penurunan setelah beban puncak.

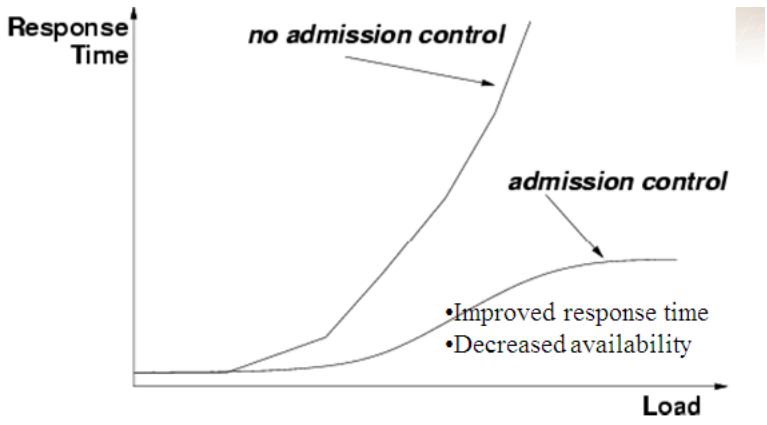
Availabilitas

Availabilitas adalah metriks yang menyatakan kinerja sistem siap untuk digunakan. Nilai Availabilitas dinyatakan dengan 0% sampai 100%. Jika sistem siap sempurna maka availabilitasnya adalah 100%. Adanya interupsi pada layanan membuat beberapa saat sistem tidak siap bekerja. Interupsi yang melampaui batas maksimal dari availabilitas yang ditentukan bisa menyebaban kehilangan suatu ongkos yang sangat besar bahkan jiwa manusia bisa hilang.

Sistem dengan availabilitas tinggi misalnya 99,99% selama 30 hari kerja berarti sistem memiliki kondisi tidak siap sebesar (1-0.9999) x 30 x 24 x 60 = 4,32 menit. Misalnya pernahkah saat membuka suatu browser ada tampilan error yang menyatakan browser sedang sibuk atau erros?

Dalam ilmu jaringan komputer tidak selamanya diperlukan server yang selalu siap, karena pada saat beban bertambah waktu antrian atau dalam hal ini waktu responnya menjadi lebih lama dan umur server menjadi berkurang karena waktu hidupnya dipercepat. Salah satu cara adalah dengan menerapkan algoritma kemacetan yang disebut *admission control*. Task dengan kebutuhan waktu layanan yang lebih cepat akan diprioritaskan sehingga jumlah *throughput* lebih banyak dan waktu tanggap dari suatu task menjadi lebih kecil.

Sebagai ilustrasi, algoritma First Come First Served (FCFS) pada sistem kemacetan jaringan akan membuat waktu tanggap atau antrian yang lama suatu Task harus menuggu Task lain yang memerlukan sumber daya yang lama. Sedangkan algoritma Short Remaining Time First Schedulling (SRTFS) akan memberikan jumlah Task yang selesai lebih banyak dan waktu tanggap atau antrian menjadi lebih kecil. Karena total waktu tanggap menjadi lebih kecil maka availabilitas secara total juga berkurang (Gambar 2.2.)

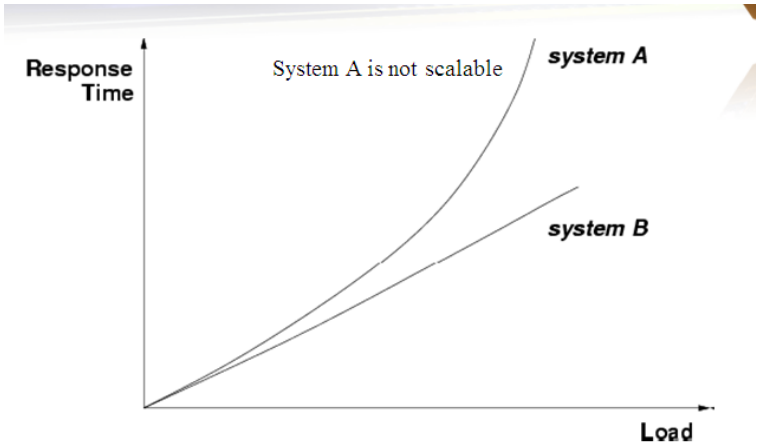


Gambar 2.2. Pengontrolan pembagian beban dengan menambah server (admission control)

Diskusikan : Bagaimana dengan algoritma yang membatasi waktu eksekusi pada nilai tertentu atau yang dikenal dengan algoritma Round Robin

Scalabilitas

Scalabilitas merupakan kemampuan sistem untuk mengubah kualitasnya sesuai dengan beban. Tujuannya untuk memperpanjang umur hidup dan mengefisiensikan ongkos operaional. Biasanya beberapa sistem mengurangi parameter normalnya untuk beban yang kecil dan akan mengubah parameternya seiring dengan penambahan beban. Misalkan pada saat beban jaringan masih normal maka bandwidth jaringannya juga normal dan tidak perlu ada penambahan server pembantu. Sedangkan saat jaringan tersebut bebannya bertambah maka secara bersamaan ada mekanisme yang mengubah bandwidth jaringan menjadi bertambah sesuai dengan penambahan beban tersebut. Jikalau perlu ada penambahan server untuk mempersingkat waktu tanggap. (Gambar 2.3)



Gambar 2.3. Pengaruh Proses Scalabilitas pada waktu tanggap

Pada gambar 2.3. terlihat bahwa dengan adanya scalabilitas pada jaringan, sistem B lebih terkontrol kenaikan waktu tanggapnya dan waktu tanggapnya menjadi linier terhadap beban.

Extensibilitas

Ekstensibilitas adalah kemampuan sistem menambah kualitasnya dengan memperluan cakupan daerah kontrolnya terhadap perubahan beban.

Dalam ekstensibilitas terdapat kemampuan dari sistem untuk memantau kinerja dirinya sendiri dan secara otomatis mengubah parameter di dalam sistemnya untuk membuat kualitas kinerja

tetap normal. Diperlukan kemampuan kecerdasan sistem untuk mengetahui kinerja sistem di luar dari kinerja operasional normal.

Contoh scalabilitas dan *Admission Control* merupakan salah satu ektensibilitas di dalam sistem jaringan. Nama lain ekstensibilitas adalah *Autonomic Computing*, *self-managing system*, *self-recovery and mitigation system*