Analisis Trade-Off Quantum terhadap Performa Penjadwalan Round Robin

Lely Priska Dameria Tampubolon[1*], *Muhammad Syaiful Fajri*[2]
[1,2]Program Studi Teknik Informatika, Fakultas Teknologi Informasi, Perbanas Institute, Jakarta, Indonesia 12940
* E-mail korespondensi : *lely.priska@perbanas.id*

ABSTRAK

Algoritma Round Robin (RR) adalah salah satu metode penjadwalan CPU tertua dan paling banyak digunakan dalam sistem operasi modern. RR dirancang untuk memberikan pembagian waktu eksekusi yang adil di antara proses-proses dengan menggunakan kuantum waktu sebagai interval waktu tetap untuk setiap proses. Penelitian ini menyajikan hasil simulasi algoritma RR yang dibangun dengan menggunakan bahasa pemrograman Java, didukung oleh visualisasi diagram Gantt dan perhitungan metrik kinerja seperti waktu tunggu rata-rata (AWT), waktu putar balik rata-rata (ATAT), dan jumlah pergantian konteks (CS). Eksperimen dilakukan pada 10 proses yang disimulasikan dengan nilai waktu kuantum yang bervariasi antara 1 dan 10 satuan waktu. Hasil simulasi menunjukkan bahwa kuantum yang terlalu kecil akan meningkatkan frekuensi peralihan konteks, yang menyebabkan beban sistem dan efisiensi berkurang. Sebaliknya, kuantum yang terlalu besar cenderung menyerupai perilaku algoritma FCFS, yang dapat memperburuk responsivitas terhadap proses penjadwalan berdurasi pendek. Nilai kuantum optimal berada dalam rentang 5–6, yang memberikan keseimbangan antara interaktivitas dan efisiensi pemrosesan. Penelitian ini menyoroti pentingnya memilih parameter kuantum yang tepat dalam implementasi RR untuk memaksimalkan kinerja sistem dan menghindari kompromi yang merugikan, terutama dalam sistem dengan banyak proses dan persyaratan waktu nyata yang tinggi.

Kata kunci:
Round Robin
Time Quantum
context switches

Diterima: 12 November 2025
Disetujui: 17 November 2025
Diterbitkan: 1 Desember 2025

Penerbit:
Perbanas Institute

I. PENDAHULUAN

Penjadwalan CPU adalah bagian penting dari sistem operasi modern yang secara langsung memengaruhi kinerja, efisiensi, dan pengalaman pengguna. Sistem multitasking, yang melibatkan berbagai proses yang berjalan secara bersamaan, membutuhkan metode penjadwalan yang dapat mendistribusikan sumber daya secara adil dan responsif (Omotehinwa, 2022). Algoritma Round Robin (RR) adalah salah satu algoritma penjadwalan

yang banyak digunakan hingga hari ini.

Algoritma RR dirancang untuk memberikan waktu eksekusi yang sama kepada setiap proses secara bergiliran melalui satuan waktu tetap yang disebut *time quantum (Zohora et al., 2024)*. Setiap proses memperoleh jatah eksekusi dalam durasi quantum, dan jika proses belum selesai dalam satu giliran, maka ia dipindahkan ke akhir antrian untuk dijadwalkan ulang pada siklus berikutnya. Pendekatan ini menjamin adanya keadilan (*fairness*) dalam sistem, mencegah kelaparan proses (*starvation*), serta menjaga interaktivitas sistem, terutama pada lingkungan berbasis time-sharing (Silberschatz et al., 2018).

Meskipun prinsip dasar algoritma RR terbilang sederhana, performa algoritma ini sangat dipengaruhi oleh pemilihan nilai quantum. Quantum yang terlalu kecil akan menyebabkan frekuensi *context switch* yang tinggi, meningkatkan overhead sistem. Sebaliknya, quantum yang terlalu besar menyebabkan proses-proses pendek menunggu terlalu lama, sehingga algoritma cenderung menyerupai algoritma *First Come First Serve* (FCFS), yang tidak ideal untuk sistem interaktif. Oleh karena itu, pemilihan quantum yang tepat sangat penting untuk mengoptimalkan performa sistem (González-Rodríguez et al., 2024).

Dalam perkembangan sistem operasi modern, prinsip kerja Round Robin masih menjadi inspirasi bagi penjadwal yang lebih kompleks, seperti *Completely Fair Scheduler* (CFS) pada kernel Linux, serta pada mekanisme penjadwalan paket di jaringan komputer dan sistem tertanam. Dengan demikian, memahami perilaku algoritma RR melalui simulasi dan eksperimen menjadi relevan tidak hanya secara teoretis, tetapi juga praktis.

Penelitian ini bertujuan untuk mengimplementasikan simulasi algoritma Round Robin menggunakan bahasa pemrograman Java, serta mengevaluasi pengaruh variasi nilai quantum terhadap metrik performa penjadwalan. Metrik yang dianalisis mencakup *average waiting time (AWT)*, *average turnaround time (ATAT)*, dan jumlah *context switch (CS)*. Melalui pendekatan eksperimental, studi ini diharapkan dapat memberikan gambaran yang lebih mendalam mengenai trade-off antara efisiensi dan responsivitas dalam penggunaan algoritma RR.

II. TINJAUAN TEORI

2.1. Algoritma Round Robin (RR)

Algoritma Round Robin (RR) pertama kali diperkenalkan pada era komputasi time-sharing pada awal 1960-an, saat komputer mulai digunakan secara bersama oleh banyak pengguna. Tujuan utama pada masa itu adalah menciptakan sistem operasi yang mampu memberikan kesan interaktif dan membagi waktu CPU secara adil di antara sejumlah proses. Algoritma ini bersifat preemptive, memberikan jatah waktu (quantum) yang sama kepada setiap proses, sehingga tidak ada proses yang mendominasi CPU. Prinsip ini menjadi dasar penjadwalan di sistem multiuser, termasuk terminal jarak jauh yang terhubung ke komputer mainframe. Hingga kini, RR tetap digunakan dalam sistem operasi modern karena keunggulannya dalam menjaga fairness dan konsistensi respons terhadap banyak proses (Tanenbaum & Bos, 2015), (Silberschatz et al., 2018).

Round Robin bekerja dengan cara mengeksekusi proses dalam urutan antrian dan memberikan setiap proses alokasi waktu CPU sebesar satu quantum secara bergantian. Jika proses selesai sebelum *quantum* habis, maka CPU langsung beralih ke proses berikutnya. Sebaliknya, jika proses belum selesai, maka proses tersebut ditempatkan kembali di akhir antrian. Kinerja algoritma ini sangat dipengaruhi oleh besar kecilnya nilai quantum. *Quantum* yang terlalu kecil menyebabkan frekuensi context switch meningkat, sehingga membebani sistem (Fiad et al., 2020a).

Quantum (q) adalah waktu tetap yang diberikan CPU untuk satu sesi eksekusi proses. Nilai q sebaiknya lebih besar dari waktu *context switch* dan lebih kecil dari sebagian besar burst time agar efisien. *Context switch* terjadi ketika CPU berpindah dari satu proses ke proses lain, dan membutuhkan overhead karena perlu menyimpan dan memulihkan state proses (Sakshi et al., 2022).

Pemilihan nilai *quantum* yang optimal sangat krusial. *Quantum* terlalu kecil menyebabkan overhead tinggi, sementara quantum terlalu besar mengakibatkan respons time lambat. Model analitik sederhana untuk menentukan nilai optimal q adalah:

$$q_{\text{optimal}} \approx \text{Average}(Burst\ Time) \quad (1)$$

Dengan nilai tersebut, sistem diharapkan mampu menyeimbangkan antara responsivitas dan efisiensi CPU. Namun, dalam praktiknya, faktor lain seperti karakteristik workload dan waktu context switch juga perlu dipertimbangkan (Tanenbaum & Bos, 2015), (Silberschatz et al., 2018), (M. T. D. Putra et al., 2021).

Jika dibandingkan dengan algoritma lain, seperti First-Come First-Served (FCFS) yang non-preemptive dan cenderung menyebabkan starvation, atau Shortest Job First (SJF) yang membutuhkan estimasi burst time, maka RR menawarkan pendekatan yang lebih adil. Sementara itu, Priority Scheduling bisa lebih efisien untuk proses penting, namun berpotensi menunda proses dengan prioritas rendah (González-Rodríguez et al., 2024).

2.2. Kelebihan dan Kelemahan Round Robin (RR)

Keunggulan utama algoritma RR terletak pada kesederhanaannya, sifatnya yang adil, serta kemampuannya menangani banyak proses interaktif. Semua proses mendapatkan jatah eksekusi yang sama, sehingga starvation dapat dihindari.

Namun demikian, performa RR sangat tergantung pada nilai quantum. Quantum yang terlalu kecil menyebabkan frekuensi context switch meningkat, menghasilkan overhead yang besar dan menurunkan efisiensi sistem (Sakshi et al., 2022). Sebaliknya, quantum yang terlalu besar membuat algoritma berperilaku seperti FCFS, yang merugikan proses-proses pendek karena waktu tunggu mereka meningkat.

Selain itu, RR tidak mempertimbangkan durasi burst time proses. Proses pendek bisa mengalami penundaan jika berada di belakang proses yang lebih panjang dalam antrian (Praditha et al., 2023). RR juga tidak bersifat adaptif terhadap beban sistem; ia tidak menyesuaikan quantum secara dinamis. Tanpa strategi tambahan seperti aging atau dynamic quantum, proses prioritas rendah bisa mengalami starvation dalam sistem yang padat.

Karenanya, implementasi RR modern seringkali disertai modifikasi atau integrasi dengan metode lain, misalnya penjadwalan berbasis prioritas, dynamic RR, atau model hybrid untuk meningkatkan efisiensi dan fairness (Chitaliya et al., 2023).

2.3. Penerapan Round Robin dalam Sistem Modern

Round Robin tetap relevan di era modern karena sifatnya yang adil dan deterministik. Contoh penerapan nyatanya adalah pada Completely Fair Scheduler (CFS) di kernel Linux, yang meskipun berbasis algoritma berbeda (red-black tree dan nilai prioritas), tetap mempertahankan semangat berbagi waktu secara adil.

Dalam bidang jaringan komputer, RR digunakan dalam penjadwalan paket pada router untuk memastikan semua sumber memiliki kesempatan transmisi yang seimbang. Demikian pula, dalam sistem load balancing dan cloud computing, RR sering diterapkan untuk mendistribusikan beban kerja ke server-server secara merata, terutama saat tidak ada informasi detail tentang karakteristik tugas (Zohora et al., 2024) .

2.4. Studi Terkait Pengembangan RR

Berbagai penelitian telah mengembangkan variasi dari RR klasik. Inovasi seperti Dynamic Round Robin (DRR), Median-Average Round Robin (MARR), dan Hybrid RR bertujuan memperbaiki kelemahan RR klasik, terutama dalam aspek pemilihan quantum. Algoritma-algoritma ini mencoba menyesuaikan quantum secara dinamis berdasarkan parameter sistem, seperti waktu eksekusi rata-rata, prioritas, atau panjang antrian (Sakshi et al., 2022), (Zohora et al., 2024).

III. METODE

3.1. Desain Penelitian

Penelitian ini bersifat eksperimental kuantitatif dengan pendekatan simulasi berbasis perangkat lunak untuk mengevaluasi performa algoritma Round Robin (RR) dalam skenario penjadwalan proses. Simulasi dilakukan dengan membangun program menggunakan bahasa pemrograman Java, yang dirancang untuk meniru perilaku penjadwalan CPU secara bergiliran dengan input proses dan variasi quantum.

3.2. Skenario Eksperimen

Eksperimen dilakukan dengan mensimulasikan 10 proses yang memiliki waktu kedatangan (arrival time) dan waktu eksekusi (burst time) berbeda. Quantum time divariasikan dari nilai 1 hingga 10 satuan waktu untuk melihat dampaknya terhadap performa penjadwalan. Setiap proses akan dimasukkan ke dalam antrian (ready queue) sesuai waktu kedatangan dan dieksekusi oleh CPU sesuai kebijakan RR.

3.3. Parameter Evaluasi

Kinerja algoritma RR diukur menggunakan tiga metrik utama, yaitu:

1) Average Waiting Time (AWT): waktu rata-rata proses menunggu di antrian sebelum mendapatkan CPU.
2) Average Turnaround Time (ATAT): rata-rata total waktu dari saat proses tiba hingga selesai.

3) Context Switch (CS): jumlah pergantian proses yang dilakukan selama simulasi berlangsung.

Pemilihan metrik ini didasarkan pada relevansi dalam menilai efisiensi sistem dan pengalaman pengguna, khususnya dalam konteks interaktivitas dan keadilan distribusi waktu CPU (Tanenbaum & Bos, 2015), (Silberschatz et al., 2018), (Sakshi et al., 2022). Analisis pada efisiensi antrian (AWT/ATAT) dan biaya preemption (CS), yang sudah cukup merefleksikan trade-off pemilihan *quantum* pada skenario beban yang dikaji.

Rumus evaluasi performa dapat dilihat pada rumus (2) sampai dengan (7). Adapun notasi untuk tiap proses i = 1…n:

1) A_i = Arrival time proses ke-i
2) B_i = *Burst time* (total CPU time yang dibutuhkan) proses ke-i
3) C_i = *Completion*/Finish time proses ke-i
4) WT_i = *Waiting time* proses ke-i
5) TAT_i = *Turnaround time* proses ke-i
6) n = jumlah proses
7) S = urutan segmen jadwal: $[(p_1,s_1,e_1),\ldots,(p_m,s_m,e_m)]$ dengan $p_k \in \{IDLE,P_1,P_2,\ldots\}$

Turnaround Time (TAT) dan *Average Turnaround Time* (ATAT)

$$TAT_i = C_i - A_i \tag{2}$$

$$\text{ATAT} = \frac{1}{n}\sum_{i=1}^{n} TAT_i \tag{3}$$

Waiting Time (WT) dan *Average Waiting Time* (AWT)

$$WT_i = TAT_i - B_i = (C_i - A_i) - B_i \tag{4}$$

$$\text{AWT} = \frac{1}{n}\sum_{i=1}^{n} WT_i \tag{5}$$

Context switch Count (CS)

$$\text{CS} = \sum_{k=2}^{m} \mathbf{1}\left[p_k \neq p_{k-1} \ \wedge \ p_k \neq \text{IDLE} \ \wedge \ p_{k-1} \neq \text{IDLE}\right] \tag{6}$$

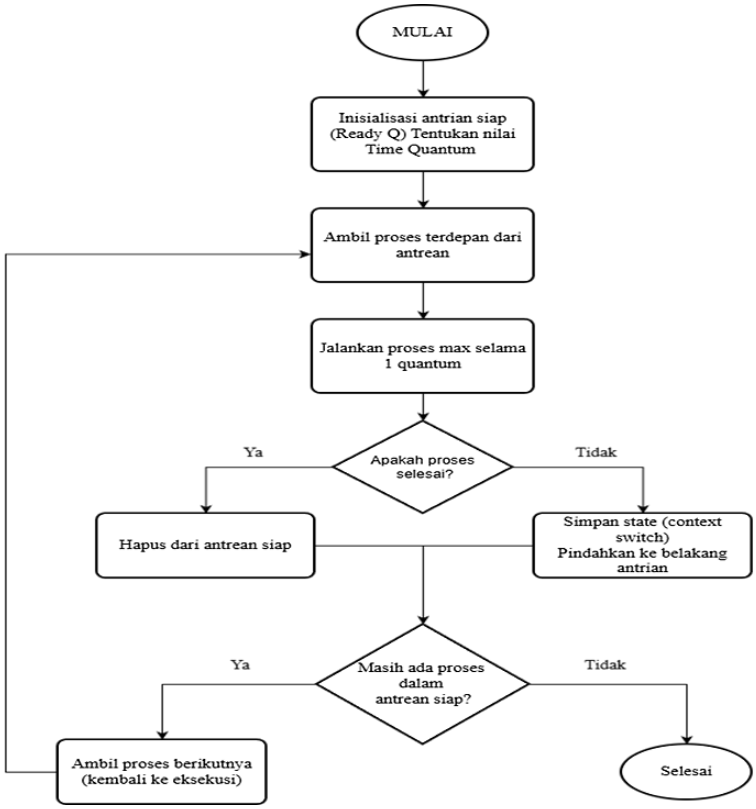
$$\text{CS}_{\text{all}} = \sum_{k=2}^{m} \mathbf{1}\left[p_k \neq p_{k-1}\right] \tag{7}$$

3.4. Alat dan Implementasi

Simulator dikembangkan menggunakan bahasa pemrograman Java dengan antarmuka terminal (console-based). Struktur program mencakup pembuatan kelas Process dan algoritma utama RR, serta pencatatan Gantt Chart dan hasil metrik performa. Visualisasi Gantt Chart dihasilkan dalam bentuk barisan nama proses pada timeline, sedangkan data hasil diekspor untuk divisualisasikan lebih lanjut dalam bentuk grafik menggunakan perangkat lunak spreadsheet.

3.5. Pseudocode dan Mekanisme

Gambar 1 merupakan flowchart dari mekanisme penjadwalan RR, sedangkan pseudocode singkat dari mekanisme penjadwalan RR yang diimplementasikan disajikan pada Algoritma 1.



Gambar 1 Flowchart Algoritma Round Robin

```
Algoritma 1 Round Robin
01  START
02  Buat antrian proses (Ready Queue)
03  Tentukan nilai Time Quantum
04
05  WHILE Ready Queue tidak kosong DO
06      Ambil proses terdepan dari Ready Queue
07      Jalankan proses selama maksimum 1 quantum
08
09      IF proses selesai THEN
10          Hapus proses dari Ready Queue
11      ELSE
12          Simpan state proses (context switch)
13          Pindahkan proses ke belakang Ready Queue
14      END IF
15
16  END WHILE
17  END
```

3.6. Tujuan Simulasi

Tujuan dari simulasi ini adalah untuk mengevaluasi dampak variasi nilai quantum terhadap kinerja sistem berdasarkan tiga metrik utama tersebut, serta mengidentifikasi nilai quantum yang memberikan keseimbangan optimal antara efisiensi (minim *context switch*) dan interaktivitas (AWT dan ATAT rendah). Analisis ini akan menjadi dasar dalam merumuskan rekomendasi konfigurasi quantum pada sistem time-sharing berbasis RR.

IV. HASIL DAN DISKUSI

Hasil simulasi menggunakan tiga indikator, yakni AWT, ATAT, dan CS. Ketiganya dipilih untuk menilai efisiensi antrean dan overhead penjadwalan yang timbul akibat variasi *quantum*. Dengan meniadakan response time, interpretasi aspek keresponsifan direpresentasikan melalui kombinasi AWT (rata-rata keterlambatan eksekusi) dan CS (intensitas preemption) yang relevan pada konteks *time-slicing* Round Robin.

4.1. Hasil Simulasi

1) Dataset pada Tabel 1, merupakan contoh untuk simulasi yang digunakan dalam penelitian ini. Dataset terdiri dari 10 proses dengan arrival time dan burst time, dengan Quantum yang digunakan yaitu 1-10.

Tabel 1 Proses Penjadwalan

Proses	Arrival time	Burst time
P1	0	5
P2	1	3
P3	2	12
P4	3	8
P5	4	9
P6	5	10

P7	5	6
P8	7	11
P9	8	5
P10	12	9

2) Langkah eksekusi (RR, q=3)

Hasil simulasi langkah eksekusi yang diberikan adalah hasil ujicoba dengan Quantum time=3. Potongan waktu (start–end), sisa burst proses setelah potongan, dan catatan saat terjadi kedatangan proses. Hasil simulasi direpresentasikan sebagai gantchartt yang dapat dilihat pada Gambar 2

1. P1 : 0–3, sisa→2 *(datang P2@1, P3@2, P4@3)*
2. P2 : 3–6, sisa→0 *(datang P5@4, P6@5, P7@5)*
3. P3 : 6–9, sisa→9 *(datang P8@7, P9@8)*
4. P4 : 9–12, sisa→5 *(datang P10@12)*
5. P1 : 12–14, sisa→0
6. P5 : 14–17, sisa→6
7. P6 : 17–20, sisa→7
8. P7 : 20–23, sisa→3
9. P8 : 23–26, sisa→8
10. P9 : 26–29, sisa→2
11. P3 : 29–32, sisa→6
12. P10 : 32–35, sisa→6
13. P4 : 35–38, sisa→2
14. P5 : 38–41, sisa→3
15. P6 : 41–44, sisa→4
16. P7 : 44–47, sisa→0 (selesai)
17. P8 : 47–50, sisa→5
18. P9 : 50–52, sisa→0 (selesai)
19. P3 : 52–55, sisa→3
20. P10 : 55–58, sisa→3
21. P4 : 58–60, sisa→0 (selesai)
22. P5 : 60–63, sisa→0 (selesai)
23. P6 : 63–66, sisa→1
24. P8 : 66–69, sisa→2
25. P3 : 69–72, sisa→0 (selesai)
26. P10 : 72–75, sisa→0 (selesai)

27. P6 : 75–76, sisa→0 (selesai)
28. P8 : 76–78, sisa→0 (selesai)

Ganttchart:

P1	P2	P3	P4	P1	P5	P6	P7	P8	P9	P3	P10	P4	P5	P6	P7	P8	P9
P3	P10	P4	P5	P6	P8	P3	P10	P6	P8								

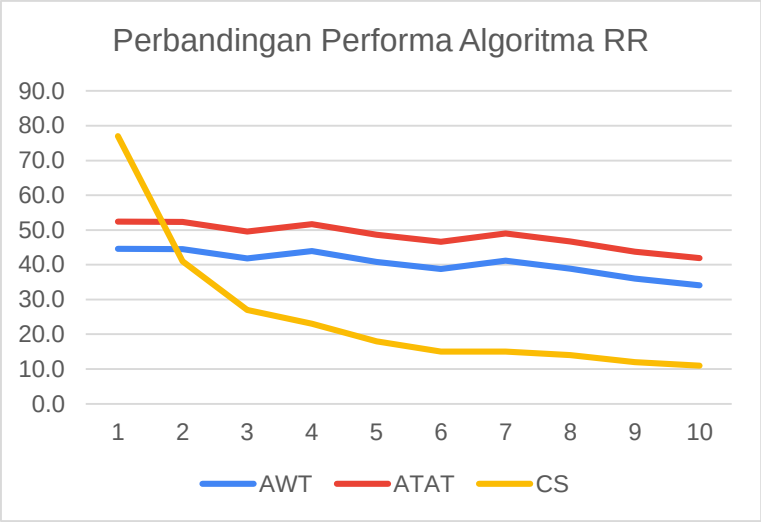
Gambar 2 Ganttchart Proses RR dengan q=3

Berdasarkan hasil simulasi, yang disajikan pada Tabel 2, terlihat bahwa peningkatan nilai quantum secara umum menurunkan nilai Average Waiting Time (AWT) dan Average Turnaround Time (ATAT). Quantum kecil seperti q=1 menghasilkan AWT tertinggi (44.6) dan jumlah *context switch* terbanyak (77), yang mencerminkan tingginya overhead akibat seringnya proses dipreempt. Sebaliknya, pada quantum lebih besar (q=10), AWT dan ATAT minimum tercapai, dengan hanya 11 *context switch*. Namun, penurunan *context switch* ini berpotensi menurunkan responsivitas terhadap proses-proses pendek, sebagaimana tampak pada kecenderungan ATAT menurun lebih lambat setelah q=6.

Tabel 2 Hasil Rata-rata AWT, ATAT, dan CS dengan q= 1 sd 10

Quantum	AWT	ATAT	CS
1	44.6	52.4	77
2	44.5	52.3	41
3	41.8	49.6	27
4	43.9	51.7	23
5	40.8	48.6	18
6	38.8	46.6	15
7	41.2	49.0	15
8	38.9	46.7	14
9	36.0	43.8	12
10	34.1	41.9	11

Gambar 3 menunjukkan hubungan yang tidak linear antara peningkatan quantum dan penurunan metrik performa. Terjadi titik optimal pada q=5–6, di mana efisiensi cukup tinggi tanpa mengorbankan responsivitas secara signifikan.



Gambar 3 Perbandingan Performa Algoritma RR dengan q= 1 - 10 (Sumber: Hasil simulasi Penjadwalan RR, 2025)

4.2. DISKUSI

Hasil simulasi menunjukkan adanya trade-off yang jelas antara ukuran quantum dan performa sistem. Quantum yang terlalu kecil menyebabkan tingginya frekuensi *context switch*, yang memperbesar overhead sistem dan mengurangi efisiensi CPU. Hal ini konsisten dengan temuan dalam literatur sebelumnya (Sakshi et al., 2022), di mana nilai quantum yang terlalu rendah cenderung mengorbankan throughput sistem demi responsivitas tinggi.

Sebaliknya, quantum besar memang menurunkan jumlah *context switch* secara signifikan, tetapi meningkatkan waktu tunggu dan turnaround time bagi proses-proses pendek, karena mereka harus menunggu giliran lebih lama. Temuan ini sejalan dengan model analitik optimal quantum yang menyarankan nilai q mendekati *burst time* rata-rata sebagai kompromi terbaik (Fiad et al., 2020b).

Gambar 3 menunjukkan tren performa algoritma Round Robin terhadap variasi nilai quantum, dilihat dari tiga metrik utama: Average Waiting Time (AWT), Average Turnaround Time (ATAT), dan jumlah *Context switch* (CS). Berdasarkan kurva AWT, terlihat bahwa nilai AWT tertinggi terjadi pada quantum = 1, yaitu sekitar 46–47. Nilai ini menurun signifikan hingga sekitar 38 pada quantum = 10. Penurunan ini menunjukkan bahwa proses mendapatkan lebih banyak waktu eksekusi per giliran saat quantum lebih besar, sehingga mengurangi waktu tunggu dalam antrean.

Sementara itu, ATAT juga menunjukkan tren menurun, meskipun tidak secepat AWT. Nilai ATAT cenderung lebih stabil dan mendatar setelah quantum mencapai nilai 6, yang menunjukkan bahwa peningkatan quantum lebih lanjut memberikan dampak yang terbatas terhadap total waktu penyelesaian proses. Hal ini mengindikasikan adanya titik jenuh, di mana efisiensi tidak lagi meningkat secara signifikan walaupun quantum diperbesar.

Secara umum, performa sistem tampak optimal pada kisaran quantum 5–6, di mana waktu tunggu dan *turnaround* cukup rendah, serta *context switch* tidak terlalu sering. Temuan ini sesuai dengan teori bahwa quantum ideal umumnya mendekati nilai rata-rata burst time (q ≈ average (burst time)), guna menghindari overhead berlebih sekaligus menjaga responsivitas (Silberschatz et al., 2018).

Tabel 3 menjabarkan hasil analisis performa algoritma dimana, quantum terlalu kecil (q = 1–2) menyebabkan sistem terlalu sering melakukan preemption, sehingga proses menunggu lebih lama walaupun dieksekusi secara adil. Quantum sedang (q = 5–6) menjadi titik kompromi optimal, di mana waktu tunggu dan turnaround tidak terlalu tinggi, dan jumlah *context switch* tidak berlebihan. Quantum besar (q = 9–10) mengurangi overhead tetapi mulai menyerupai FCFS, menurunkan fairness dan responsivitas terhadap proses pendek.

Tabel 3 Hasil Analisis Tren AWT, ATAT dengan quantum

Quantum	AWT Trend	ATAT Trend	Keterangan
1–2	Sangat tinggi → turun tajam	Tinggi → turun cepat	Preemptif berlebihan
3–5	Fluktuatif ringan	Stabil menurun	Trade-off mulai seimbang
6–10	Turun perlahan	Mendatar	Quantum besar mengurangi CS, namun tak banyak kurangi ATAT

Dengan demikian, pemilihan algoritma dan parameter (seperti quantum pada RR) harus disesuaikan dengan karakteristik sistem target. Dalam sistem interaktif, RR tetap menjadi pilihan utama karena adil dan responsif, namun perlu penyesuaian quantum untuk menjaga efisiensi.

Penghilangan response time tidak mengurangi kekuatan analisis karena AWT dan CS telah memadai untuk menangkap trade-off utama Round Robin antara fairness/keresponsifan (semakin kecil quantum → AWT turun namun CS naik) dan efisiensi (semakin besar quantum → CS turun namun risiko penundaan proses pendek meningkat, tercermin pada ATAT).

V. KESIMPULAN DAN REKOMENDASI

Berdasarkan hasil simulasi algoritma Round Robin (RR) terhadap sepuluh proses dengan variasi quantum dari 1 hingga 10, dapat disimpulkan bahwa nilai quantum memiliki pengaruh signifikan terhadap metrik performa sistem. Quantum yang terlalu kecil menyebabkan jumlah *context switch* meningkat drastis, sehingga membebani sistem dengan overhead tinggi. Sebaliknya, quantum yang terlalu besar menurunkan frekuensi *context switch*, namun mengorbankan waktu tunggu proses-proses pendek, sehingga mengurangi tingkat interaktivitas sistem.

Hasil eksperimen menunjukkan bahwa nilai quantum di kisaran 5–6 memberikan *trade-off* terbaik antara efisiensi pemrosesan (dilihat dari penurunan *Average Waiting Time* dan *Average Turnaround Time*) serta beban *context switching*. Dengan demikian, pemilihan quantum yang moderat menjadi kunci untuk menjaga keseimbangan antara performa dan

responsivitas sistem.

Untuk penelitian selanjutnya, disarankan mengeksplorasi algoritma RR dengan pendekatan *dynamic quantum* yang menyesuaikan ukuran quantum berdasarkan karakteristik proses. Selain itu, uji coba dapat diperluas dengan arrival time yang acak, beban proses yang lebih heterogen, serta skenario *real-time* agar hasilnya lebih representatif terhadap kondisi sistem nyata.

DAFTAR REFERENSI

Chitaliya, P., Kulkarni, S., Telang, A., Zaveri, D., Shah, A., & Deulkar, K. (2023). Time Quantum Optimization in Round Robin Algorithm. *2023 International Conference on Network, Multimedia and Information Technology, NMITCON 2023*. https://doi.org/10.1109/NMITCON58196.2023.10276146

Fiad, A., Maaza, Z. M., & Bendoukha, H. (2020a). Improved version of round robin scheduling algorithm based on analytic model. *International Journal of Networked and Distributed Computing*, *8*(4), 195–202. https://doi.org/10.2991/IJNDC.K.200804.001

Fiad, A., Maaza, Z. M., & Bendoukha, H. (2020b). Improved version of round robin scheduling algorithm based on analytic model. *International Journal of Networked and Distributed Computing*, *8*(4), 195–202. https://doi.org/10.2991/IJNDC.K.200804.001

González-Rodríguez, M., Otero-Cerdeira, L., González-Rufino, E., & Rodríguez-Martínez, F. J. (2024). Study and evaluation of CPU scheduling algorithms. *Heliyon*, *10*(9). https://doi.org/10.1016/j.heliyon.2024.e29959

Omotehinwa, T. O. (2022). Examining the developments in scheduling algorithms research: A bibliometric approach. *Heliyon*, *8*(5). https://doi.org/10.1016/j.heliyon.2022.e09510

Praditha, V. S., Hidayat, T. S., Akbar, M. A., Fajri, H., Lubis, M., Lubis, F. S., & Safitra, M. F. (2023). A Systematical Review on Round Robin as Task Scheduling Algorithms in Cloud Computing. *2023 6th International Conference on Information and Communications Technology, ICOIACT 2023*, 516–521. https://doi.org/10.1109/ICOIACT59844.2023.10455832

Putra, M. T. D., Hidayat, H., Septian, N., & Afriani, T. (2021). Analisis Perbandingan Algoritma Penjadwalan CPU First Come First Serve (FCFS) Dan Round Robin. *Building of Informatics, Technology and Science (BITS)*, *3*(3), 207–212. https://doi.org/10.47065/bits.v3i3.1047

Sakshi, Sharma, C., Sharma, S., Kautish, S., A. M. Alsallami, S., Khalil, E. M., & Wagdy Mohamed, A. (2022). A new median-average round Robin scheduling algorithm: An optimal approach for reducing turnaround and waiting time. *Alexandria Engineering Journal*, *61*(12), 10527–10538. https://doi.org/10.1016/j.aej.2022.04.006

Silberschatz, A., Galvin, P. B., & Gagne, G. (2018). *Operating System Concepts* (10th ed.). Wiley.

Tanenbaum, A. S., & Bos, H. (2015). *Modern Operating Systems* (Tracy Johnson, Ed.; 4th ed.). Pearson Education, Inc.

Zohora, M. F., Farhin, F., & Kaiser, M. S. (2024). An enhanced round robin using dynamic time quantum for real-time asymmetric burst length processes in cloud computing environment. *PLoS ONE*, *19*(8 AUGUST). https://doi.org/10.1371/journal.pone.0304517