semanTIK, Vol.7, No.1, Jan-Jun 2021, pp. 91-98
ISSN: 2502-8928 (Online)
DOI: 10.5281/zenodo.5036494
Terakreditasi "Peringkat 4 (Sinta 4)" oleh Kemenristekdikti
Website: http://ojs.uho.ac.id/index.php/semantik

ANALISIS PERBANDINGAN ALGORITMA PENJADWALAN *ROUND ROBIN* DAN *SHORTEST JOB FIRST* UNTUK MANAJEMEN PROSES DALAM *SINGLE PROCESSING*

La Ode Muh. Taufiq[*1], L.M. Fid Aksara[2], Muh. Yamin[3]
[1,2,3]Jurusan Teknik Informatika, Fakultas Teknik, Universitas Halu Oleo, Kendari
e-mail: [*1]**taufiklaodemuhamad@gmail.com**, [2]fid.aksara@gmail.com,
[3]putra0683@gmail.com

Abstrak

Seiring dengan perkembangan ilmu pengetahuan teknologi, informasi, dan komunikasi, tuntutan dalam mengelola suatu proses dengan baik dan optimal dalam sistem operasi adalah suatu hal yang sangat dibutuhkan pada zaman ini. Ada dua algoritma penjadwalan proses dalam mengelola proses yang akan dibahas yaitu algoritma *Round Robin* (RR) dan algoritma *Shortest Job First* (SJF). Berdasarkan penelitian yang dilakukan selama perancangan, analisis dan perbandingan algoritma penjadwalan SJF dan RR dengan pengujian dilakukan sebanyak tiga kali, maka didapatkan bahwa *Average Waiting Time* (AWT) dan *Average Turnaround Time* (ATAT) dari algoritma *preemptive* SJF lebih kecil dibandingkan *non-preemptive* SJF dan algoritma *Round Robin*. Akan tetapi pada pengujian ketiga didapatkan nilai AWT dan ATAT sama untuk *preemptive* SJF dan SJF *non-preemptive*, disebabkan dari nilai *arrival time* dan *burst time* proses yang diberikan pada saat dieksekusi. Sedangkan algoritma *Round Robin*, pada beberapa nilai *quantum* yang diuji menunjuhkan nilai AWT dan ATAT rendah, akan tetapi lebih tinggi dibandingkan AWT dan ATAT dari algoritma *Shortest Job First.* Sehingga dapat disimpulkan bahwa algoritma penjadwalan *Shortest Job First* lebih optimal dibandingkan dengan algoritma penjadwalan *Round Robin*. Dalam menjalankan suatu proses didalam CPU sampai proses tersebut melepaskan CPU.

Kata Kunci; Sistem Operasi, CPU, Manajemen Proses, *Round Robin*, *Shortest Job First*

Abstract

Along with the development of technological science, information, and communication, the demand to manage a process properly and optimally in the operating system is something that is very needed at this time. There are two of process scheduling algorithms in managing the process are Round Robin algorithm (RR) and the Shortest Job First algorithm (SJF). Based on research conducted during the design, analysis and comparison of SJF and RR scheduling algorithms with testing done three times, it was found that the Average Waiting Time (AWT) and Turnaround Time (TAT) of the SJF preemptive algorithm is smaller than the non-preemptive SJF and algorithm Round Robin. However, in the third test the same AWT and ATAT values were obtained for the preemptive SJF and non-preemptive SJF, due to the arrival time and process burst time values given at the time of execution. While the Round Robin algorithm, the quantum values tested show low AWT and ATAT values, but higher than AWT and ATAT from the Shortest Job First algorithm. So, it can be concluded that the first shortest job scheduling algorithm is more optimal than the Round Robin scheduling algorithm. In running a process inside the CPU until the process releases the CPU.

Keywords; *Operating System*, CPU, *Process Management*, *Round Robin*, *Shortest Job First*

1. PENDAHULUAN

Komputer bisa membantu manusia menyelesaikan sebuah perintah pada tingkat yang rumit sekalipun. Untuk bisa bekerja dengan baik, komputer membutukan banyak perangkat di dalamnya. Komputer memerlukan perangkat lunak dan perangkat keras yang berkolaborasi dengan tepat untuk bisa berjalan. Sistem operasi atau biasa disingkat dengan OS *(Operation System)* adalah salah satu perangkat yang harus ada pada komputer. Tanpa sistem operasi, komputer tidak akan dapat bekerja. Adanya sistem operasi membantu komputer untuk mengatur seluruh sumber daya yang ada pada komputer. Sistem operasi merupakan nyawa dari komputer, jika tidak ada pengoperasian, maka komputer tidak dapat berjalan lancar.

Manajemen proses merupakan konsep pokok dalam sistem operasi, sehingga masalah manajemen proses adalah masalah utama dalam perancangan manajemen proses. Proses adalah sebuah program yang sedang di eksekusi yang mencakup *program counter*, *register*, dan variabel di dalamnya [1]. Sedang program adalah kumpulan intruksi yang di tulis ke dalam bahasa yang dimengerti sistem operasi. Sebuah proses membutuhkan sejumlah sumber daya untuk menyelesaikan tugasnya, berupa CPU *time*, alamat memori, dan berkas-berkas.

Penjadwalan proses merupakan kumpulan kebijaksanaan dan mekanisme sistem operasi yang berkaitan dengan urutan kerja yang dilakukan sistem komputer [1]. Untuk memutuskan proses yang harus berjalan, kapan dan seberapa lama proses itu berjalan di perlukan suatu algoritma penjadwalan proses yang efektif. Penjadwalan proses yang efektif, memiliki kriteria untuk mengukur dan optimasi kinerja diantaranya adil (*fairnes*), efisiensi (*eficiency*), waktu tanggap (*response time*), *turnaround time*, dan *throughput* [2].

$$Average\ WT = \sum \frac{WT}{P} \quad (1)$$

$$Average\ TAT = \sum \frac{TAT}{P} \quad (2)$$

Keterangan:

TAT = *turnaround time*
WT = *waiting time*
P = proses

Persoalan penjadwalan proses dapat di selesaikan dengan beberapa algoritma penjadwalan, dimana setiap algoritma mempunyai kriteria dan keunggulannya masing-masing. Ada berbagai macam algoritma penjadwalan proses di dalam CPU dan dua diantaranya adalah algoritma *Round Robin* dan algoritma *Shortest Job First* yang akan di bahas pada tugas akhir ini. Algoritma *Round Robin* merupakan algoritma yang bergantung pada *quantum time*. Jika *quantum time* terlalu besar, maka *response time* untuk proses yang lain terlalu tinggi. Sebaliknya jika *quantum time* terlalu kecil maka akan mengakibatkan *overhead* pada CPU dimana *context switching* lebih besar. Algoritma *Shortest Job First* adalah algoritma penjadwalan yang optimal dengan rata-rata waktu tunggu yang minimal. Namun pada kenyataannya sulit di implementasikan karena sulit untuk mengetahui panjang CPU *burst* berikutnya.

Sampai saat ini *user* tidak mengetahui berapa lama sebuah job atau proses dikerjakan pada algoritma *Round Robin* dan *Shortest Job First* dalam sebuah komputer untuk manajemen proses dalam *single processing*. Berdasarkan penjelasan latar belakang yang telah diuraikan diperlukan analisis perbandingan algoritma penjadwalan *Round Robin* dan *Shortest Job First* untuk Manajemen Proses dalam *Single Processing*"**.**

2. METODE PENELITIAN

2.1 Algoritma *Round Robin*

Konsep dasar dari algoritma *Round Robin* adalah dengan menggunakan *time-sharing*. Pada dasarnya algoritma ini sama dengan FCFS, hanya saja bersifat *preemptive*. Setiap proses mendapatkan waktu CPU yang disebut dengan waktu kuantum (*quantum time*) untuk membatasi waktu proses, biasanya 1-100 milidetik [3]. Setelah waktu habis, proses ditunda dan ditambahkan pada *ready queue.* Jika suatu proses memiliki CPU *burst* lebih kecil dibandingkan dengan waktu kuantum, maka proses tersebut akan melepaskan CPU jika telah selesai bekerja, sehingga CPU dapat segera digunakan oleh proses selanjutnya.

Sebaliknya, jika suatu proses memilki CPU *burst* yang lebih besar dibandingkan dengan waktu kuantum, maka proses tersebut akan dihentikan sementara jika sudah mencapai

waktu kuantum, dan selanjutnya mengantri kembali pada posisi ekor dari *ready queue*, CPU kemudian menjalankan proses berikutnya [3]. Jika terdapat *n* proses pada *ready queue* dan waktu kuantum *q*, maka setiap proses mendapatkan 1/*n* dari waktu CPU paling banyak *q* unit waktu pada sekali penjadwalan CPU. Tidak ada proses yang menunggu lebih dari (*n*-1) *q* unit waktu [1].

2.2 Algoritma *Shortest Job First*

Algoritma *Shortest Job First* sangat optimal, karena memberikan rata-rata waktu tunggu lebih kecil dibandingkan algoritma penjadwalan yang lain dengan cara memindahkan *job-job* pendek di depan *job-job* yang panjang, sehingga akan mengurangi waktu tunggu [4]. Pada penjadwalan SJF, proses yang memiliki CPU *burst* paling kecil dilayani terlebih dahulu. Terdapat dua skema penjadwalan SJF yaitu :

1. *Non-preemptive*

Penjadwalan *non-preemptive* ialah salah satu jenis penjadwalan dimana sistem operasi tidak pernah melakukan *context switch* dari proses yang sedang berjalan ke proses yang lain. Dengan kata lain, proses yang berjalan tidak bisa di *interrupt.* CPU tidak memperbolehkan proses yang ada di *ready queue* untuk menggeser proses yang sedang dieksekusi oleh CPU meskipun proses yang baru tersebut mempunyai *burst time* yang lebih kecil.

2. *Preemptive*

Penjadwalan *Preemptive* mempunyai arti kemampuan-kemampuan sistem operasi untuk memberhentikan proses, sementara proses yang sedang berjalan untuk memberi ruang kepada proses yang prioritasnya lebih tinggi. Jika ada proses yang sedang dieksekusi oleh CPU dan terdapat proses *ready queue* dengan *burst time* lebih kecil daripada proses yang sedang dieksekusi tersebut, maka proses yang sedang dieksekusi oleh CPU akan digantikan oleh proses yang berada di *ready queue* tersebut. *Preemptive* SJF sering disebut juga *shortest-remaining-time-first-scheduling*.

3. HASIL DAN PEMBAHASAN

3.1 Implementasi

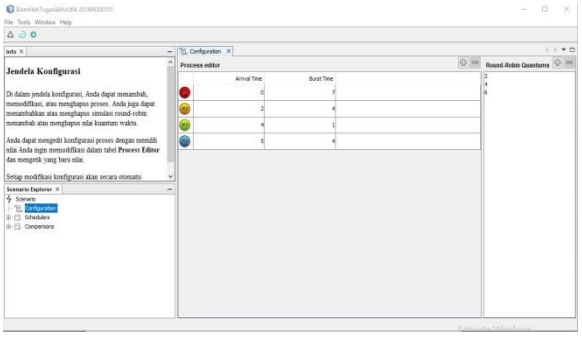
Implementasi rancangan antarmuka, yaitu:

A. *Interface* Aplikasi

Pada *interface* aplikasi di antaranya:

1. Halaman *Configuration*

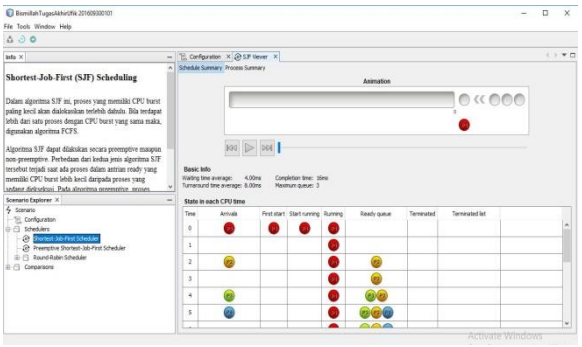
Pada saat pertama membuka menu *configuration* terdapat data proses otomatis yang sudah dimasukkan pada *bar code* aplikasi. Data proses tersebut dapat dihapus dan diedit oleh *user*. Pada jendela *configuration* terdapat tombol ✚ yang mempunyai fungsi menambah proses baru, dan tombol ▬ yang mempunyai fungsi menghapus data proses.



Gambar 1. Tampilan *Configuration*

2. Halaman *Schedule Summary*

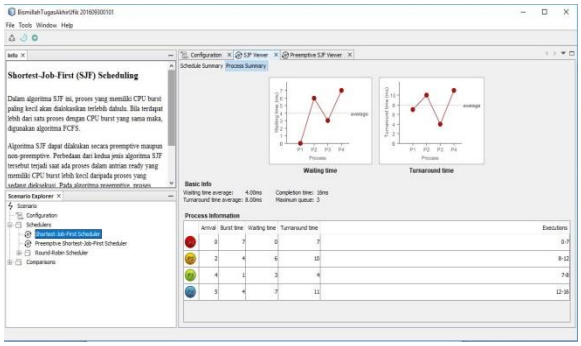
Berisi data umum dari proses sebelumnya yang bekerja sesuai aturan (*rule*) algoritma non-*preemptive Shortest Job First schedule*. Data pengalokasian *CPU* yang divisualisasikan ke dalam bentuk animasi *gantt chart.*



Gambar 2. *Schedule Summary Shortest Job First*

3. Halaman *Process Summary Shortest Job First*

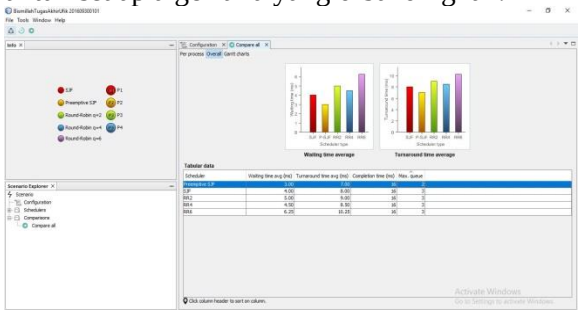
Berisi data umum dari simulasi pada proses sebelumnya yang bekerja sesuai aturan (*rule*) algoritma *non-preemptive Shortest Job First scheduler. Process summary*, pengalokasian CPU untuk setiap proses ditampilkan dalam bentuk grafik untuk *waiting time* dan *turnaround time* dan tabel.



Gambar 3. *Process Summary*

4. Halaman *Comparison Overall*

Pada *tab overall* menampilkan grafik dan tabel perbandingan rata-rata *waiting time* dan *turnaround time* dalam bentuk diagram batang untuk setiap algoritma yang dibandingkan.



Gambar 4. *Comparison Overall*

B. Pengujian dan Analisis

Pada tahap pengujian, penulis melakukan uji coba dengan menjalankan 3 data proses yang akan dijalankan dengan menggunakan masing-masing dari algoritma penjadwalan.

Tabel 1. Pengujian Pertama *Non-Preemptive Shortest Job First*

Proses	AT	BT	ST	CS	WT	TAT
P1	0	7	0	7	0	7
P2	2	4	8	12	6	10
P3	4	1	7	8	3	4
P4	5	4	12	16	7	11
P5	3	6	16	22	13	19

Keterangan:

AT (Arrival Time) : Waktu kedatangan sebuah proses.

BT (Burst Time): Waktu yang diberikan untuk sebuah proses dalam menggunakan CPU.

ST (Start Time) : Waktu mulai proses dalam menggunakan CPU.

CS (Completion Start) : Waktu selesai proses dalam menggunakan CPU.

WT (Waiting Time): Waktu proses untuk menunggu di *ready queue*.

TAT (Turnaround Time): Waktu yang di perlukan untuk mengeksekusi proses.

$$\text{Average WT} = \sum \frac{\text{WT}}{\text{P}} = 29 / 5 = 5.8 \text{ ms}$$

$$\text{Average TAT} = \sum \frac{\text{TAT}}{\text{P}} = 51 / 5 = 10.2 \text{ ms}$$

Tabel 2. Pengujian Pertama *Preemptive Shortest Job First*

Proses	AT	BT	ST	CS	WT	TAT
P1	0	7	0	16	9	16
P2	2	4	2	7	1	5
P3	4	1	4	5	0	1
P4	5	4	7	11	2	6
P5	3	6	16	22	13	19

$$\text{Average WT} = \sum \frac{WT}{P} = 25 / 5 = 5 \text{ ms}$$

$$\text{Average TAT} = \sum \frac{TAT}{P} = 47 / 5 = 9.4 \text{ ms}$$

Tabel 3. Pengujian Pertama RR *Quantum* 2

Proses	AT	BT	ST	CS	WT	TAT
P1	0	7	0	20	13	20
P2	2	4	2	11	5	9
P3	4	1	8	9	4	5
P4	5	4	11	19	10	14
P5	3	6	6	22	13	19

$$\text{Average WT} = \sum \frac{WT}{P} = 45 / 5 = 9 \text{ ms}$$

$$\text{Average TAT} = \sum \frac{TAT}{P} = 67 / 5 = 13.4 \text{ ms}$$

Tabel 4. Pengujian Pertama RR *Quantum* 4

Proses	AT	BT	ST	CS	WT	TAT
P1	0	7	0	16	9	16
P2	2	4	4	8	2	6
P3	4	1	12	13	8	9
P4	5	4	16	20	11	15
P5	3	6	8	22	13	19

$$\text{Average WT} = \sum \frac{WT}{P} = 43 / 5 = 8.6 \text{ ms}$$

$$\text{Average TAT} = \sum \frac{TAT}{P} = 65 / 5 = 13 \text{ ms}$$

Tabel 5. Pengujian Pertama RR *Quantum* 6

Proses	AT	BT	ST	CS	WT	TAT
P1	0	7	0	22	15	22
P2	2	4	6	10	4	8

P3	4	1	16	17	12	13
P4	5	4	17	21	12	16
P5	3	6	10	16	7	13

$$\text{Average WT} = \sum \frac{WT}{P} = 50 / 5 = 10 \text{ ms}$$

$$\text{Average TAT} = \sum \frac{TAT}{P} = 72 / 5 = 14.4 \text{ ms}$$

Tabel 6. Pengujian Kedua *Non-Preemptive* SJF

Proses	AT	BT	ST	CS	WT	TAT
P1	0	7	0	7	0	7
P2	4	9	7	16	3	12
P3	7	9	21	30	14	23
P4	14	10	30	40	16	26
P5	10	5	16	21	6	11

$$\text{Average WT} = \sum \frac{WT}{P} = 39 / 5 = 7.8 \text{ ms}$$

$$\text{Average TAT} = \sum \frac{TAT}{P} = 79 / 5 = 15.8 \text{ ms}$$

Tabel 7. Pengujian Kedua *Preemptive* SJF

Proses	AT	BT	ST	CS	WT	TAT
P1	0	7	0	7	0	7
P2	4	9	21	30	17	26
P3	7	9	7	21	5	14
P4	14	10	30	40	16	26
P5	10	5	10	15	0	5

$$\text{Average WT} = \sum \frac{WT}{P} = 38 / 5 = 7.6 \text{ ms}$$

$$\text{Average TAT} = \sum \frac{TAT}{P} = 78 / 5 = 15.6 \text{ ms}$$

Tabel 8. Pengujian Kedua RR *Quantum* 3

Proses	AT	BT	ST	CS	WT	TAT
P1	0	7	0	10	3	10
P2	4	9	4	28	15	24
P3	7	9	10	33	17	26
P4	14	10	22	40	16	26
P5	10	5	16	30	15	20

$$\text{Average WT} = \sum \frac{WT}{P} = 66 / 5 = 13.2 \text{ ms}$$

$$\text{Average TAT} = \sum \frac{TAT}{P} = 106 / 5 = 21.2 \text{ ms}$$

Tabel 9. Pengujian Kedua RR *Quantum* 6

Proses	AT	BT	ST	CS	WT	TAT
P1	0	7	0	13	6	13
P2	4	9	6	27	14	23
P3	7	9	13	36	20	29
P4	14	10	27	40	16	26
P5	10	5	19	24	9	14

$$\text{Average WT} = \sum \frac{WT}{P} = 65 / 5 = 13 \text{ ms}$$

$$\text{Average TAT} = \sum \frac{TAT}{P} = 105 / 5 = 21 \text{ ms}$$

Tabel 10. Pengujian Ketiga RR *Quantum* 9

Proses	AT	BT	ST	CS	WT	TAT
P1	0	7	0	7	0	7
P2	4	9	7	16	3	12
P3	7	9	16	25	9	18
P4	14	10	30	40	16	26
P5	10	5	25	30	15	20

$$\text{Average WT} = \sum \frac{WT}{P} = 43 / 5 = 8.6 \text{ ms}$$

$$\text{Average TAT} = \sum \frac{TAT}{P} = 83 / 5 = 16.6 \text{ ms}$$

Tabel 11. Pengujian Ketiga *Non-Preemptive* SJF

Proses	AT	BT	ST	CS	WT	TAT
P1	0	7	0	7	0	7
P2	8	18	47	65	39	57
P3	6	16	31	47	25	41
P4	4	14	17	31	13	27
P5	2	10	7	17	5	15

$$\text{Average WT} = \sum \frac{WT}{P} = 82 / 5 = 16.4 \text{ ms}$$

$$\text{Average TAT} = \sum \frac{TAT}{P} = 147 / 5 = 29.4 \text{ ms}$$

Tabel 12. Pengujian Ketiga *Preemptive* SJF

Proses	AT	BT	ST	CS	WT	TAT
P1	0	7	0	7	0	7
P2	8	18	47	65	39	57
P3	6	16	31	47	25	41
P4	4	14	17	31	13	27
P5	2	10	7	17	5	15

$$\text{Average WT} = \sum \frac{WT}{P} = 82 / 5 = 16.4 \text{ ms}$$

$$\text{Average TAT} = \sum \frac{TAT}{P} = 147 / 5 = 29.4 \text{ ms}$$

Tabel 13. Pengujian Ketiga RR *Quantum* 4

Proses	AT	BT	ST	CS	WT	TAT
P1	0	7	0	15	8	15
P2	8	18	19	65	39	57
P3	6	16	15	59	37	53
P4	4	14	8	55	37	51
P5	2	10	4	41	29	39

$$\text{Average WT} = \sum \frac{WT}{P} = 150 / 5 = 30 \text{ ms}$$

$$\text{Average TAT} = \sum \frac{TAT}{P} = 215 / 5 = 43 \text{ ms}$$

Tabel 14. Pengujian Ketiga 8 RR *Quantum* 8

Proses	AT	BT	ST	CS	WT	TAT
P1	0	7	0	7	0	7
P2	8	18	31	65	39	57
P3	6	16	23	55	33	49
P4	4	14	15	47	29	43
P5	2	10	7	41	29	39

$$\text{Average WT} = \sum \frac{WT}{P} = 130 / 5 = 26 \text{ ms}$$

$$\text{Average TAT} = \sum \frac{TAT}{P} = 195 / 5 = 39 \text{ ms}$$

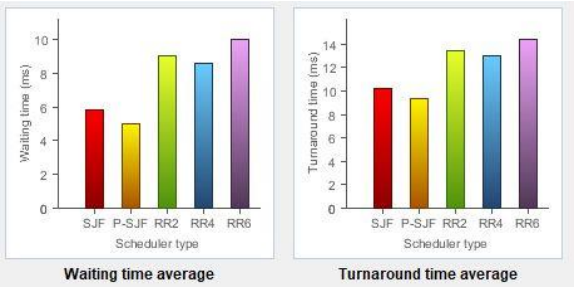
Tabel 15. Pengujian Ketiga RR *Quantum* 12

Proses	AT	BT	ST	CS	WT	TAT
P1	0	7	0	7	0	7
P2	8	18	41	65	39	57
P3	6	16	29	59	37	53
P4	4	14	17	55	37	51
P5	2	10	7	17	5	15

$$\text{Average WT} = \sum \frac{WT}{P} = 118 / 5 = 23.6 \text{ ms}$$

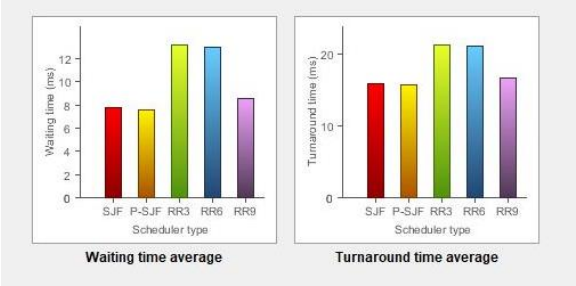
$$\text{Average TAT} = \sum \frac{TAT}{P} = 183 / 5 = 36.6 \text{ ms}$$

Berdasarkan rata-rata *Average Waiting Time* dan *Average Turnaround Time* pada pengujian pertama, algoritma penjadwalan *preemptive Shortest Job First* memiliki AWT dan ATAT paling kecil dalam mengeksekusi dan menyelesaikan proses yaitu 5 *milisecond* untuk AWT dan 9.4 untuk ATAT.



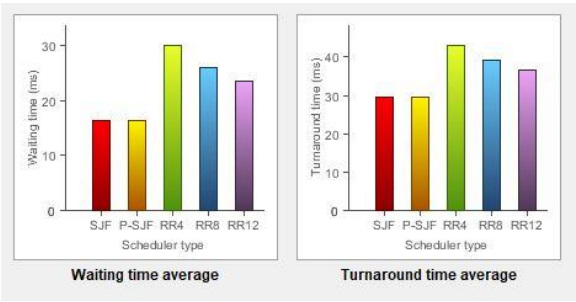
Gambar 5. Grafik *WT Average* dan *TAT Average* Perbandingan NP-SJF, P-SJF, RR2, RR4, RR6 pada Pengujian Pertama

Pada pengujian kedua, berdasarkan rata-rata *average waiting time* dan *average turnaround time*, algoritma penjadwalan *preemptive Shortest Job First* kembali memliki AWT dan ATAT paling kecil dalam mengeksekusi dan menyelesaikan proses yaitu 7.6 ms untuk AWT dan 15,6 ms untuk ATAT.



Gambar 6. Grafik *WT Average* dan *TAT Average* Perbandingan NP-SJF, P-SJF, RR3, RR6, RR9 pada Pengujian Kedua

Pada pengujian ketiga, berdasarkan rata-rata *average waiting time* dan *average turnaround time* algoritma penjadwalan *non-preemptive Shortest Job First* dan *preemptive Shortest Job First* memiliki AWT dan ATAT paling kecil dalam mengeksekusi dan menyelesaikan proses yaitu 16.4 ms untuk AWT dan 29.4 untuk ATAT.



Gambar 7. Grafik *WT Average* dan *TAT Average* Perbandingan NP-SJF, P-SJF, RR3, RR6, RR9 pada Pengujian Ketiga

4. KESIMPULAN

Berdasarkan pengujian yang telah dilakukan diperoleh hasil sebagai berikut:

1. Nilai rata-rata waktu tunggu proses (*average waiting time*) *Shortest Job First preemptive* lebih kecil dibandingkan dengan *Shortest Job First non preemptive* dan *Round Robin*. Akan tetapi pada pengujian ketiga didapatkan nilai yang sama untuk *average waiting time* SJF *preemptive* dan SJF *non-preemptive*.

2. Nilai rata-rata waktu eksekusi proses (*average turnaround time*) lebih kecil dibandingkan dengan *Shortest Job First non preemptive* dan *Round Robin*. Akan tetapi pada pengujian ketiga didapatkan nilai yang sama untuk *average turnaround time* SJF *preemptive* dan SJF *non-preemptive*.
3. Pada algoritma penjadwalan *Round Robin* diberikan nilai *quantum* berbeda, didapatkan nilai *average waiting time* dan *turnaround time* yang rendah, akan tetapi lebih tinggi dibandingkan dengan *average waiting time* dan *turnaround time Shortest Job First.*

5. SARAN

Adapun saran penulis dalam penelitian ini yaitu:

1. Pengujian dapat dilakukan dengan aplikasi simulasi yang lebih baik lagi sehingga mencapai nilai yang maksimal dan mudah dipahami.,
2. Dapat ditambahkan dengan membandingkan algoritma *Shortest Job First* dan *Round Robin* dengan algoritma penjadwalan yang lain.
3. Pengujian dapat ditambahkan dengan menggunkan parameter yang lain, untuk mendapatkan hasil yang lebih baik lagi.
4. Dan algoritma penjadwalan RR dan SJF dapat diimplementasikan, menjalankannya pada sistem nyata dan melihatnya bekerja, untuk keperluan evaluasi pada beban atau kondisi sistem yang nyata.

DAFTAR PUSTAKA

[1] Tanenbaum, Andrew S, dan Albert S Woodhull. 2001. *Operating Systems Design and Implementation (3rd Edition)*..

[2] Sobh, Tarek M, dan Abhilasha Tibrewal. 2006. "Parametric Optimization Of Some Critical Operating System Functions – An Alternative Approach To The Study Of Operating Systems Design," no. May.

[3] Gea, Asaziduhu. 2015. "Optimasi Turn Arround Time Pada Penjadwalan Round Robin Dengan Mencari Quantum Time Optimal Menggunakan" 1 (1): 1–9.

[4] Santika, Monica, dan Seng Hansun. 2014. "Implementasi Algoritma Shortest Job First dan *Round Robin* pada Sistem Penjadwalan Pengiriman Barang" VI (2): 94-99.