Politechnika Wrocławska

# Laboratorium Systemów Operacyjnych

Prowadzący: mgr inż. Jakub Klikowski

| „Symulacja działania algorytmów planowania czasu procesora dla zamkniętej puli zadań” | Miłosz Jarząb<br>Wydział Elektroniki<br>Cyberbezpieczeństwo |
|---|---|

| Wtorek. godz.15.15 | Termin oddania sprawozdania: | Ocena: |
|---|---|---|

Zaimplementowane algorytmy:

- FCFS – First Come First Serve
- LCFS – Last Come First Serve
- SJF – Shortest Job Firts
- RR – Round-Robin

1. Opis procedury testowania algorytmów.
2. Opracowanie wyników eksperymentów dla procesów o różnym czasie przybycia.
3. Opracowanie wyników eksperymentów dla procesów o tym samym czasie przybycia równym 0.
4. Wykresy dla różnego czasu przybycia.
5. Wykresy dla tego samego czasu przyjścia.
6. Wnioski.

# 1. Opis procedury testowania.

Cały program opiera się na obiektach typu „Process” o parametrach: nazwa, czas przyjścia i czas wykonywania.

```
class Process:

    def __init__(self, name, a, s):
        self.name = name
        self.a = a
        self.s=s
```

Plik tekstowy „Processes1.txt”, zawierający kolejno: nazwę, czas przyjścia oraz czas wykonywania procesu dla procesów o różnym czasie przybycia (każdy oddzielony przecinkiem oraz każdy proces w oddzielnej linii), wczytuje funkcja „creation” i tworzy 3 listy typu „Process”. Każda z list ma różne długości (50, 100, 500).

Plik tekstowy „Processes0.txt”, zawierający kolejno: nazwę, czas przyjścia oraz czas wykonywania procesu dla procesów o czasie przybycia równym 0 (każdy oddzielony przecinkiem oraz każdy proces w oddzielnej linii), wczytuje funkcja „creation” i tworzy 3 listy typu „Process”. Każda z list ma różne długości (50, 100, 500).

Czasy przybycia i wykonywania losowałem na stronie: RANDOM.ORG - ListRandomizer.

Następnie dla każdej z list wykonuje się funkcja „alhorithms” zawierająca podfunkcje, które tworzą kopie list i szeregują wedle algorytmów planowania:

- FCFS – sortuje listę po parametrze „process.a”, czyli po czasie przyjścia procesu.

- LCFS – sprawdza, który proces ma największy czas przybycia w aktualnym symulowanym czasie i dodaje go do nowej listy. Wtedy ponownie sprawdza, po dodaniu do aktualnego symulowanego czasu czasu wykonywania procesu, który proces ma największy czas przybycia. Powtarza dopóki nie sprawdzi wszystkich procesów z listy.

- SJF – sortuje listę po czasie przybycia rosnąco i sprawdza czy w momencie process[0].a jest w kolejce jeszcze jakiś proces, który jest krótszy i ma czas przyjscia mniejszy lub równy process[0].a. Jeśli nie, dodaje go do nowej listy. Następnie po czasie jego wykonania sprawdza to samo dla procesów spełniających te same warunki co poprzednio były wymagane.

- RR – sortuje listę czasami przybycia procesów i sprawdza czy pierwszy w kolejce proces ma czas wykonywania dłuższy lub równy/mniejszy kwantowi. W pierwszym przypadku do nowej listy dodaje nowy proces o parametrach nazwa i czas przybycia nie zmienionych i czasie wykonywania równemu długości kwantu, a procesowi z pierwszej listy zmniejsza czas wykonywania o kwant. W przypadku gdy czas wykonywania jest mniejszy od kwantu przepisuje jego parametry do nowej listy i zeruje czas wykonywania. Pętla ta wykonuje się dopóki suma wszystkich czasów wykonywania procesów listy pierwszej jest większa od zera.

Po stworzeniu nowych i poszeregowanych list wyliczane są następująco: średni czas przetwarzania, średni czas oczekiwania oraz odchylenie standardowe (dot. czasu oczekiwania).

Na koniec program zwraca pliki dotyczące powyższych wartości oraz wykresy dla wszystkich czterech algorytmów planowania. Wszystko to podzielone jest w foldery (według długości i rodzajów czasu przyjścia) oraz posiada nazwę wskazującą na dany algorytm.

Do wykonywania obliczeń wykorzystałem bibliotekę numpy, rysowania wykresów pylab, a stworzenia nowego folderu zaimportowałem bibliotekę OS.

## 2. Opracowanie wyników eksperymentów dla procesów o różnym czasie przybycia.

Średni czas przetwarzania procesów [ms]

| Liczba procesów | FCFS | LCFS | SJF | RR |
|---|---|---|---|---|
| 50 | 235.7 | 209.82 | 150.08 | 276.42 |
| 100 | 495.71 | 495.25 | 329.11 | 630.58 |
| 500 | 2416.33 | 2549.51 | 1644.56 | 3140.81 |

Średni czas oczekiwania procesów [ms]

| Liczba procesów | FCFS | LCFS | SJF | RR |
|---|---|---|---|---|
| 50 | 226.78 | 200.9 | 141.16 | 267.5 |
| 100 | 485.73 | 485.27 | 319.13 | 620.6 |
| 500 | 2406.39 | 2539.57 | 1634.62 | 3130.87 |

Odchylenie standardowe czasu oczekiwania procesów:

| Liczba procesów | FCFS | LCFS | SJF | RR |
|---|---|---|---|---|
| 50 | 117.7 | 128.93 | 123.99 | 123.10 |
| 100 | 278.61 | 293.06 | 291.7 | 294.01 |
| 500 | 1473.77 | 1487.70 | 1469.42 | 1530.69 |

## 3. Opracowanie wyników eksperymentów dla procesów o tym samym czasie przybycia = 0:

Średni czas przetwarzania procesów [ms]

| Liczba procesów | FCFS | LCFS | SJF | RR |
|---|---|---|---|---|
| 50 | 236.94 | 236.94 | 149.26 | 285.12 |
| 100 | 471.51 | 471.51 | 338.25 | 632.34 |
| 500 | 2460.21 | 2460.21 | 1654.68 | 3160.85 |

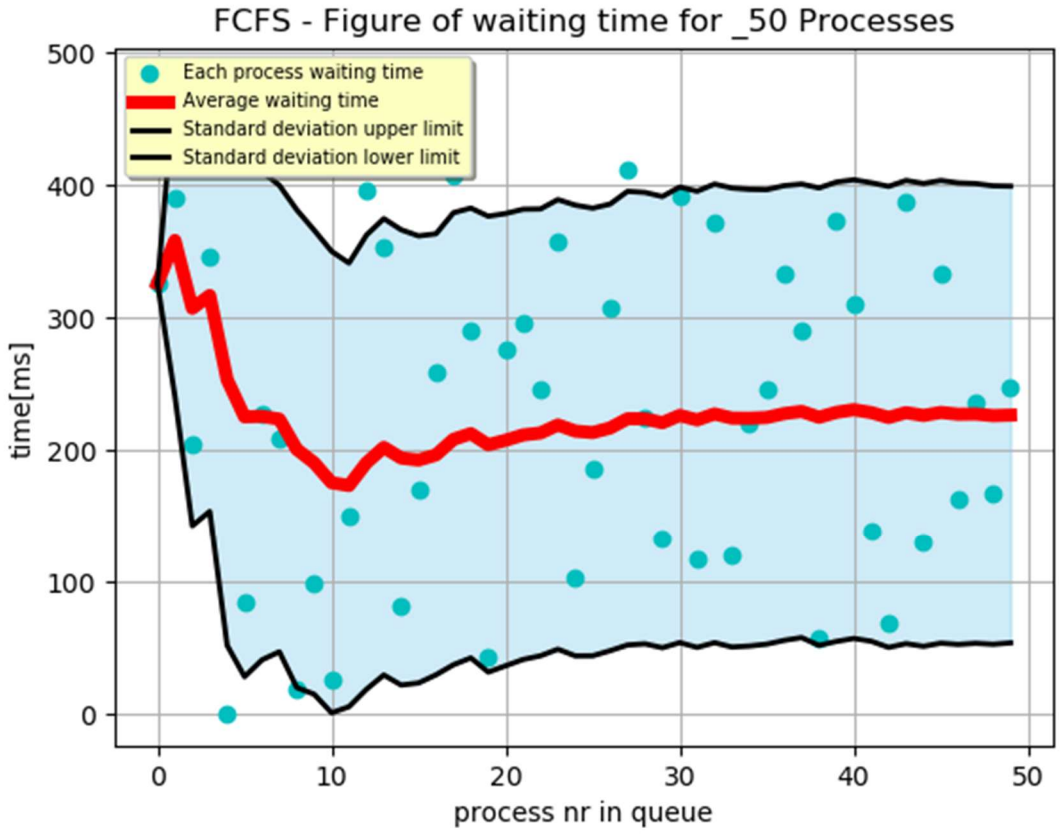
Średni czas oczekiwania procesów [ms]

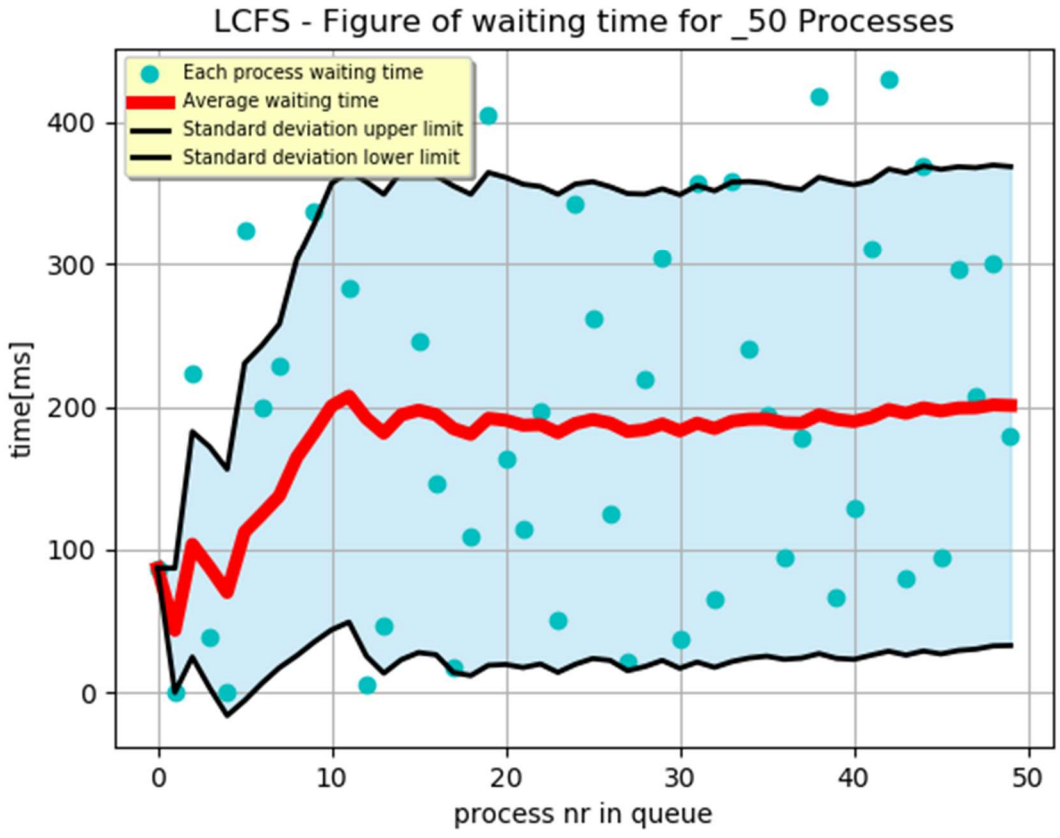
| Liczba procesów | FCFS | LCFS | SJF | RR |
|---|---|---|---|---|
| 50 | 228.02 | 228.02 | 140.34 | 276.2 |
| 100 | 461.53 | 461.53 | 328.27 | 622.36 |
| 500 | 2450.27 | 2450.27 | 1644.74 | 3150.91 |

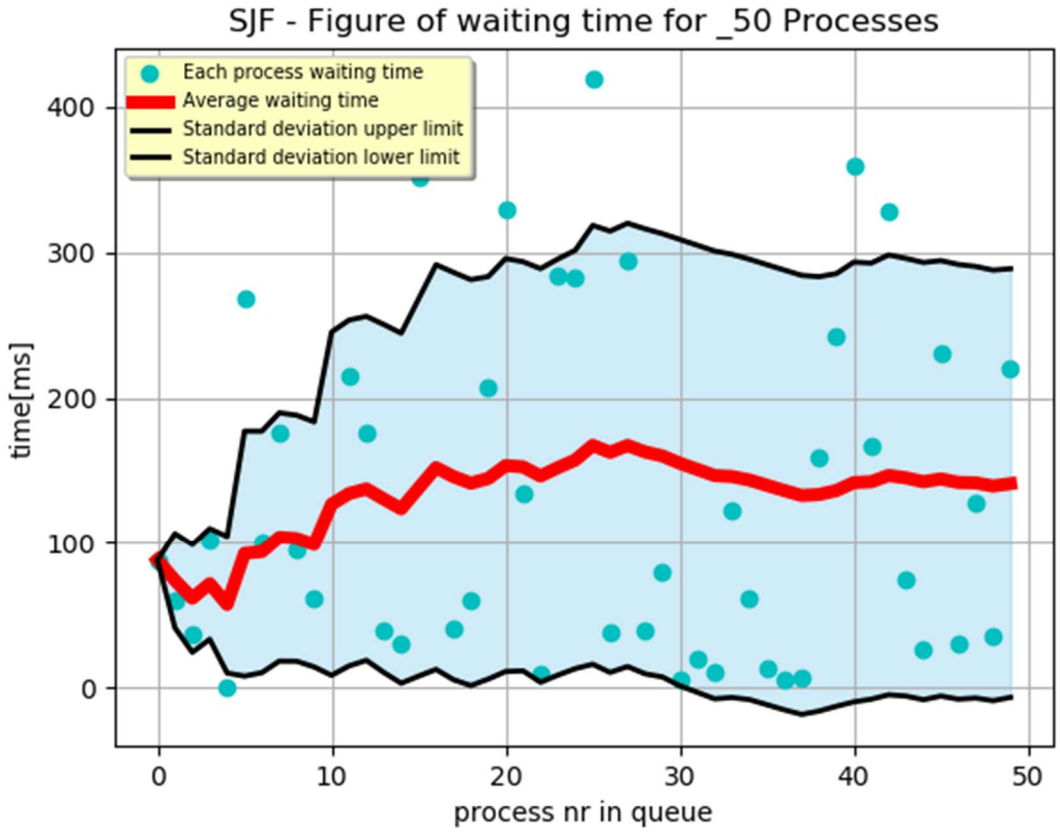
Odchylenie standardowe czasu oczekiwania procesów:

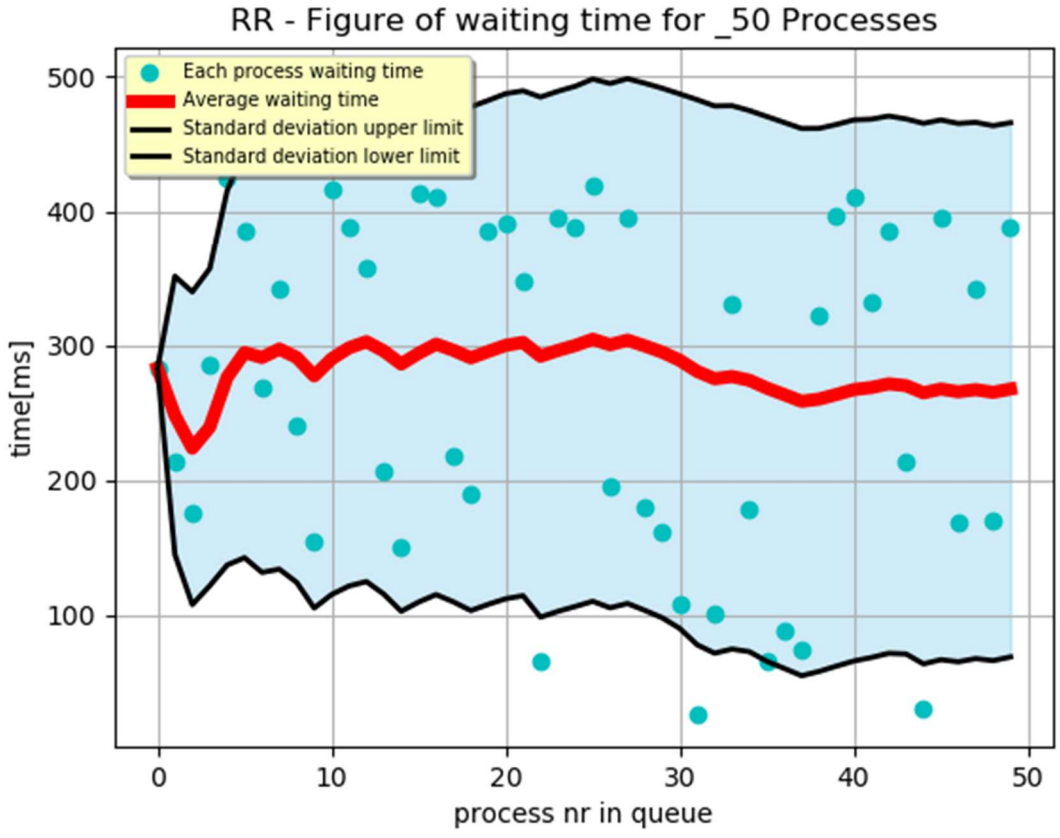
| Liczba procesów | FCFS | LCFS | SJF | RR |
|---|---|---|---|---|
| 50 | 129.07 | 129.07 | 128.37 | 120.94 |
| 100 | 276.64 | 276.64 | 293.18 | 294.20 |
| 500 | 1446.04 | 1446.04 | 1470.05 | 1510.94 |

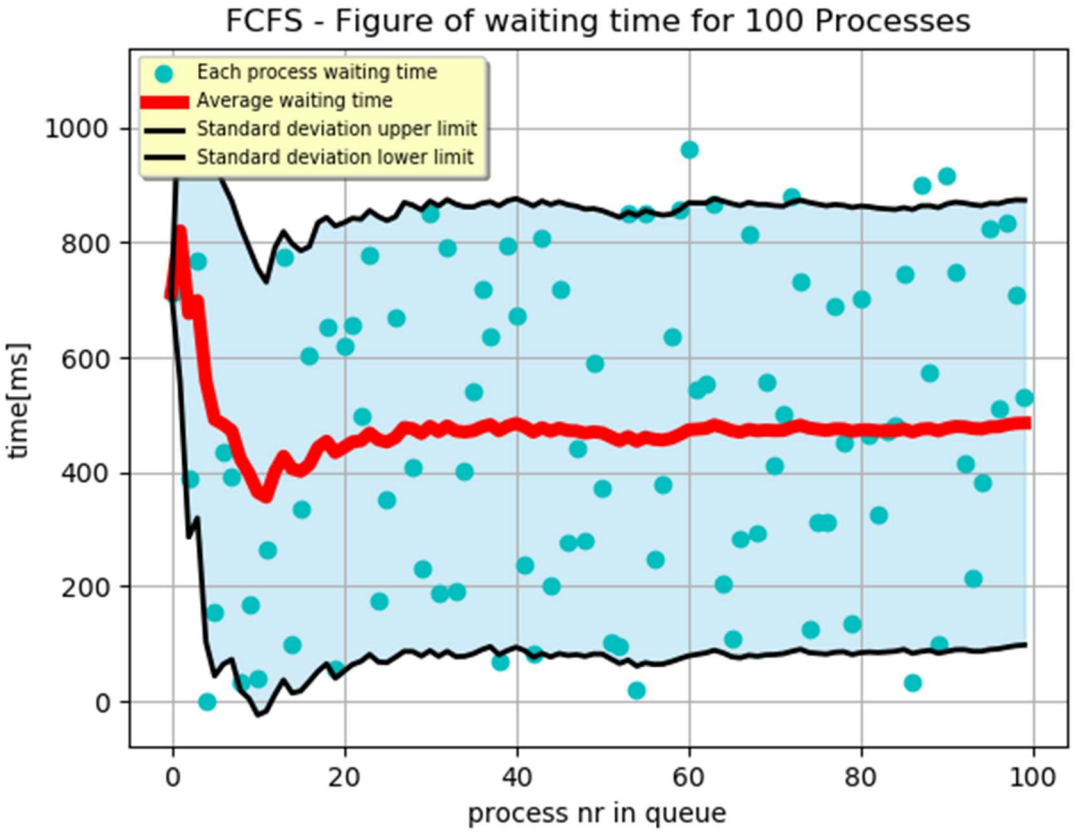
4. Wykresy dla różnego czasu przybycia.

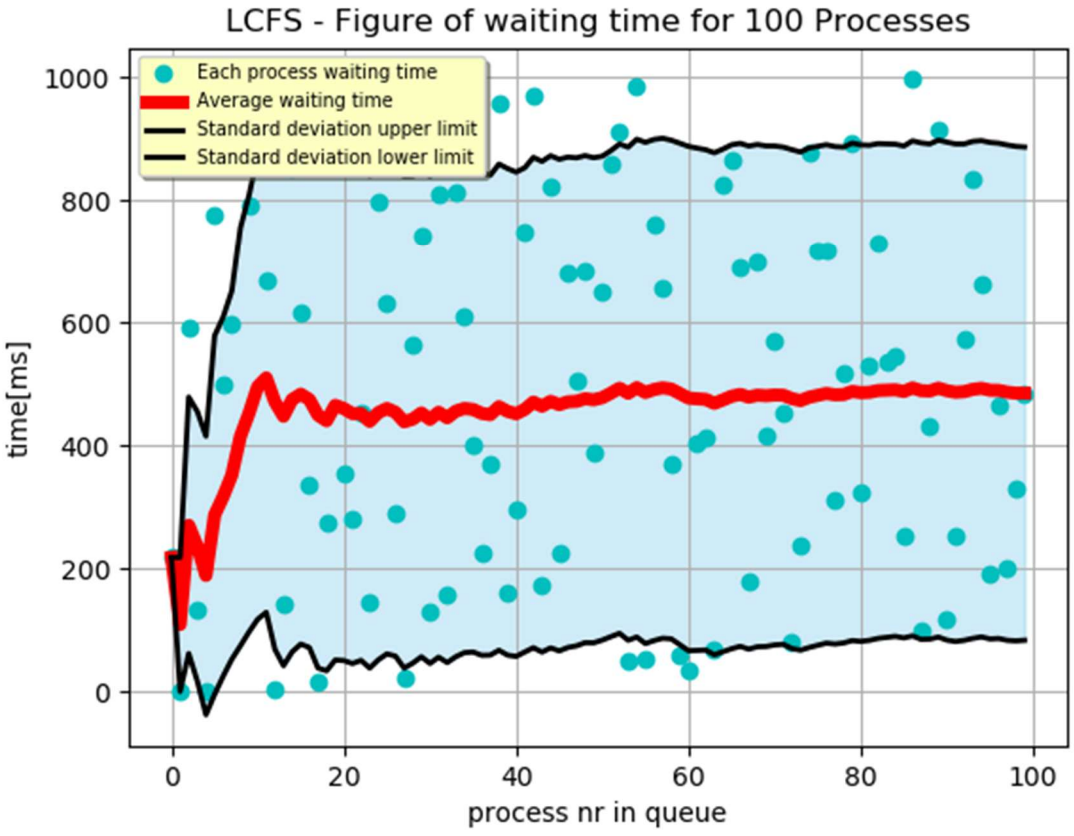


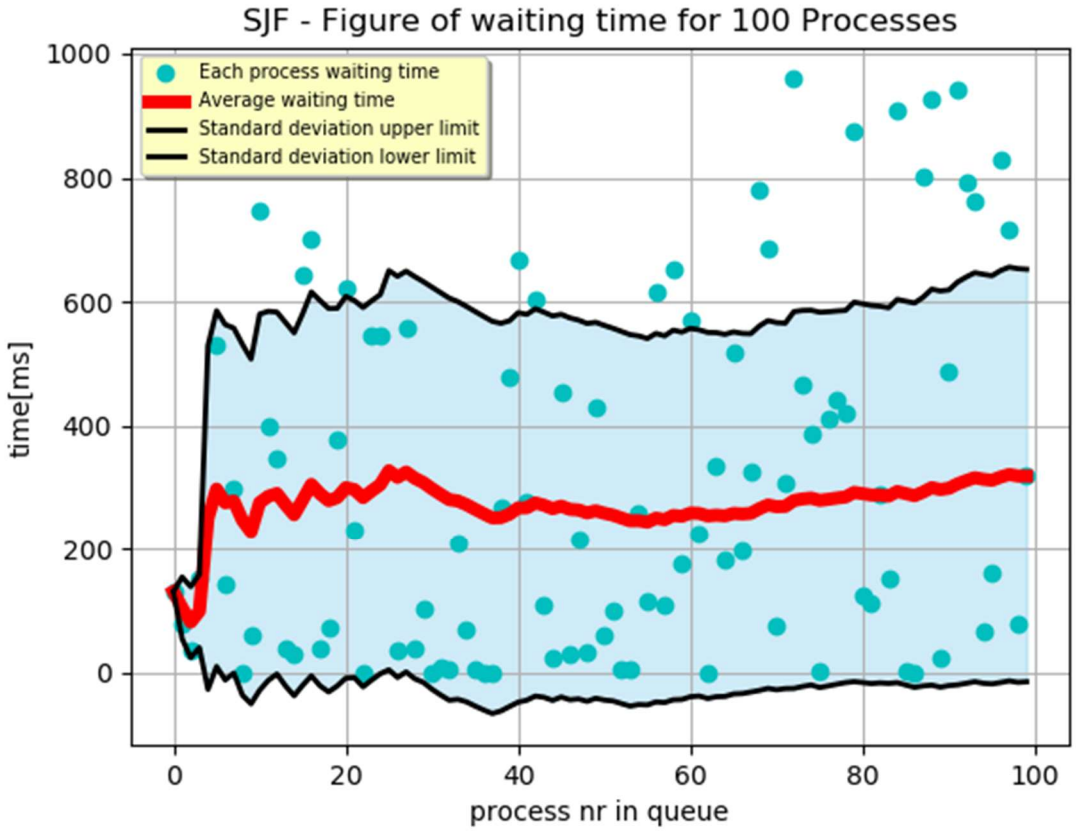



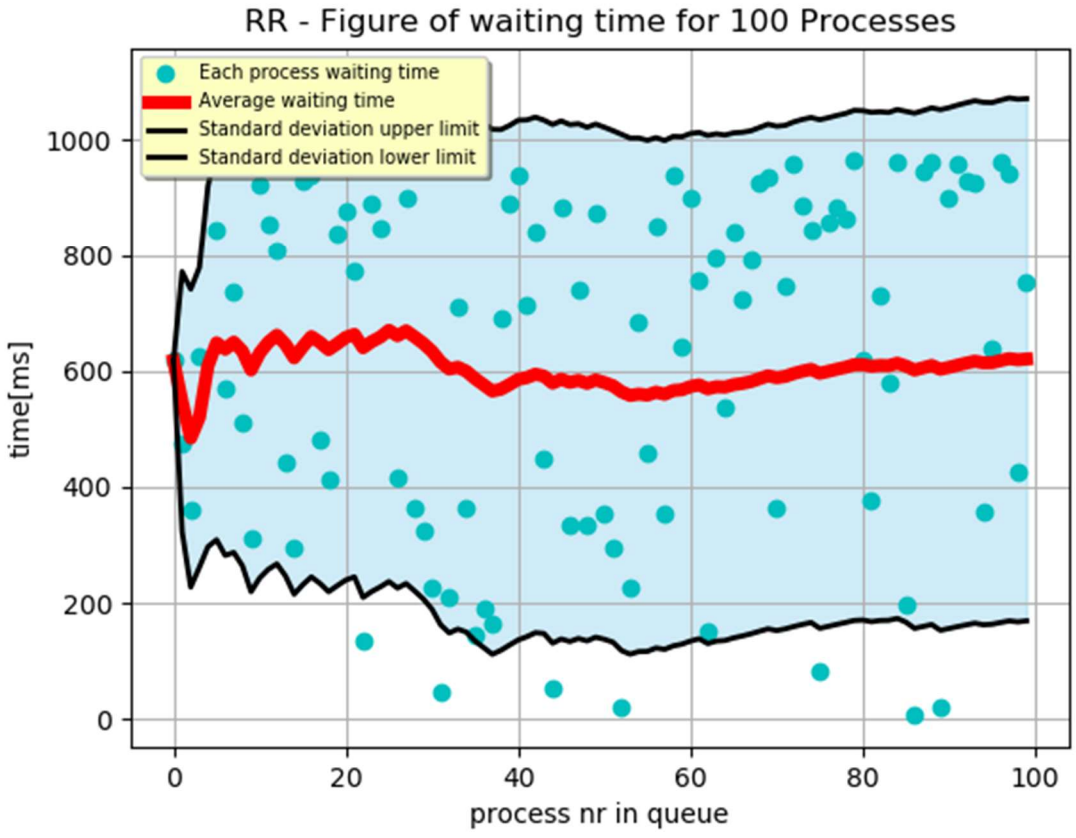

SJF - Figure of waiting time for _50 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
0
100
200
300
400
0
10
20
30
40
50

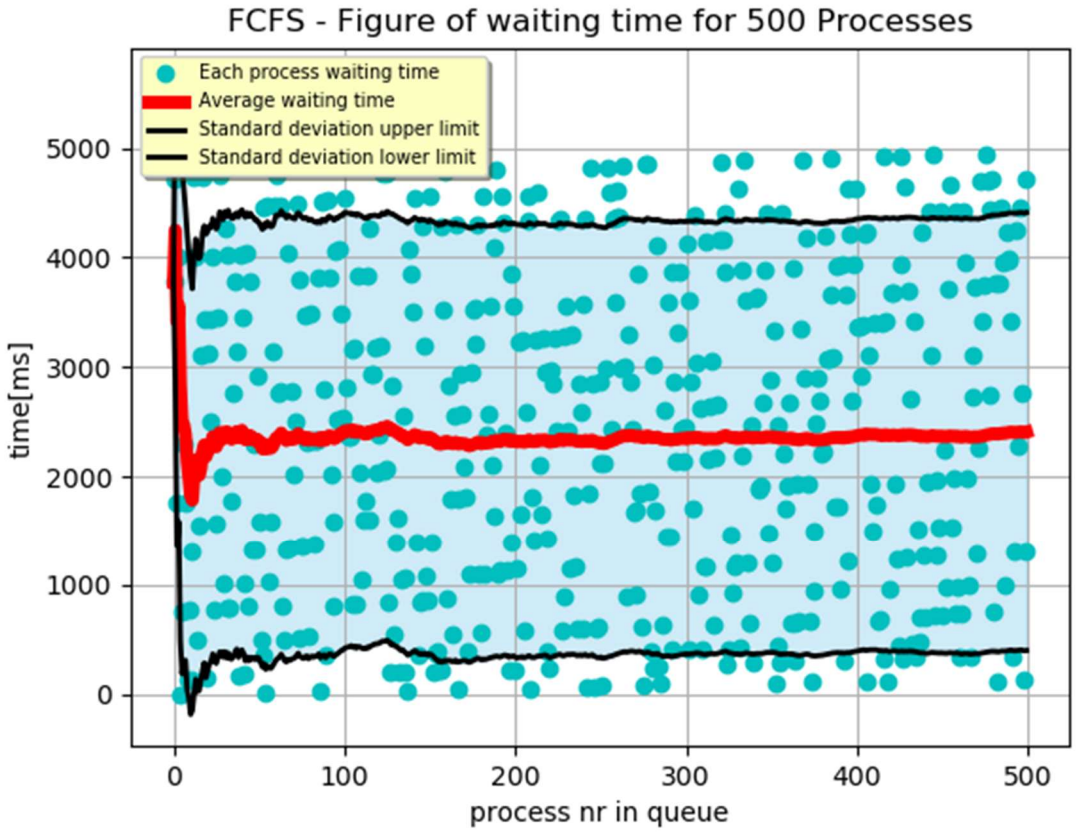

RR - Figure of waiting time for _50 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
100
200
300
400
500
0
10
20
30
40
50

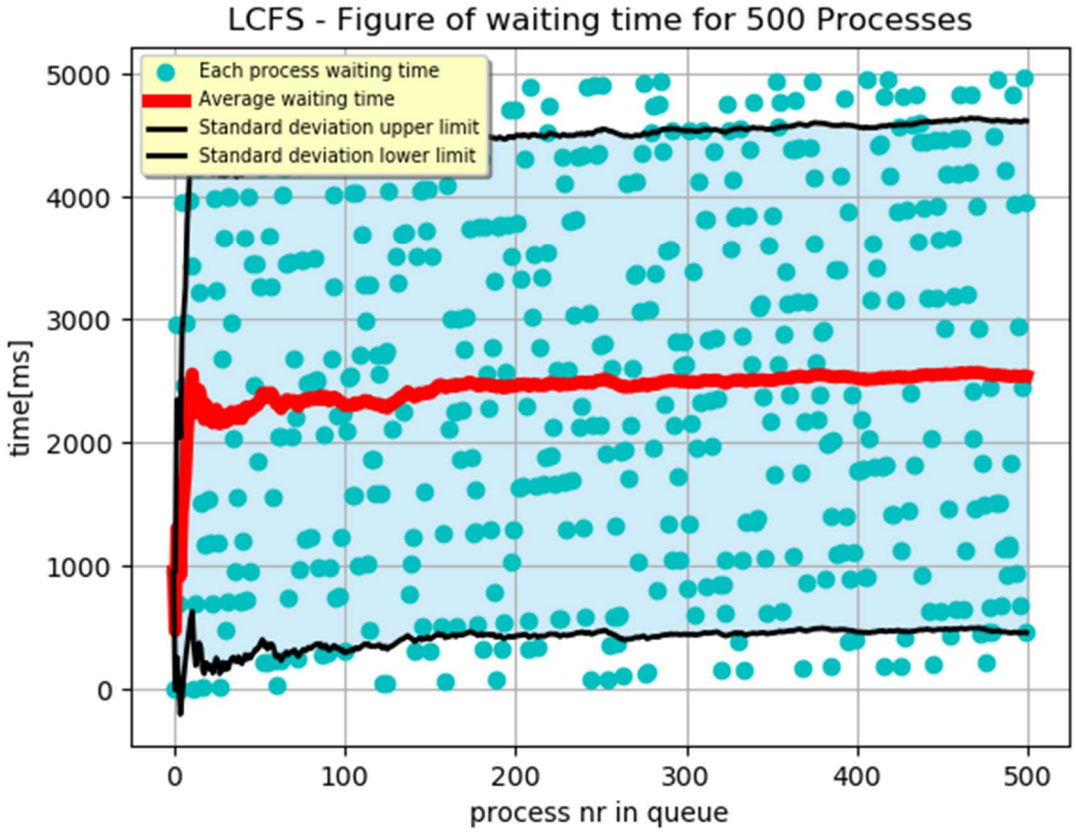

FCFS - Figure of waiting time for 100 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
0
200
400
600
800
1000
0
20
40
60
80
100

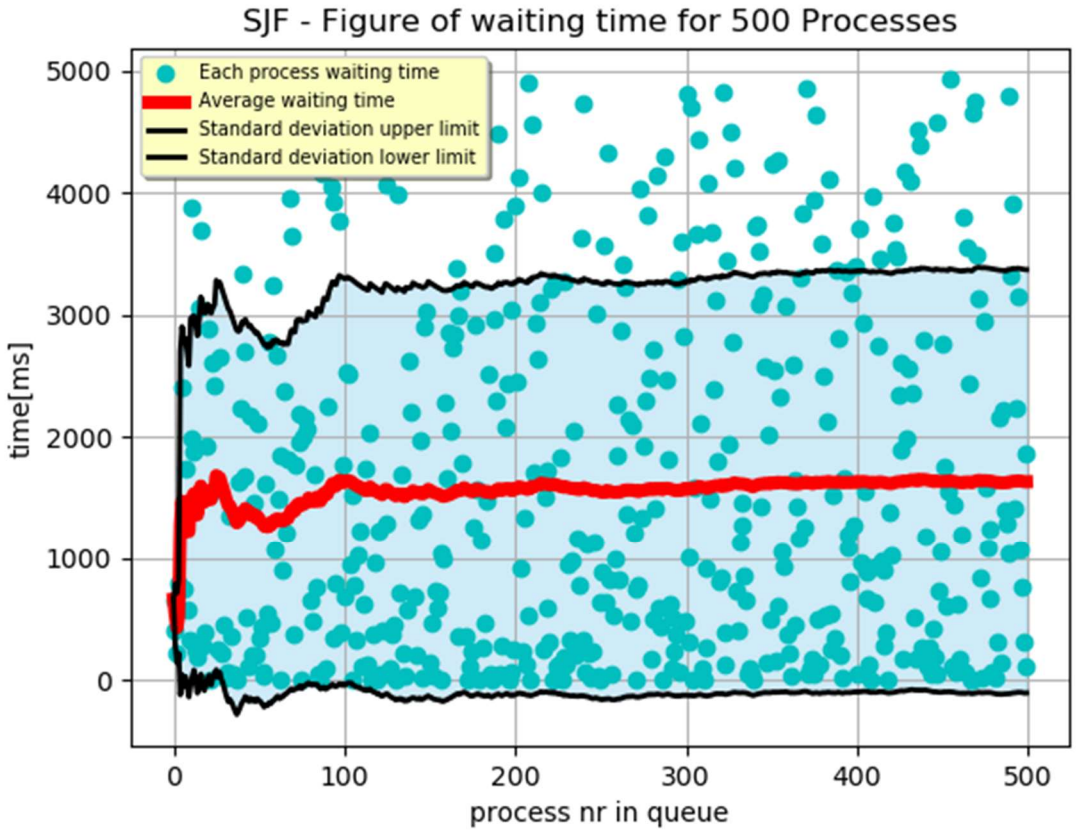

LCFS - Figure of waiting time for 100 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
0
200
400
600
800
1000
0
20
40
60
80
100

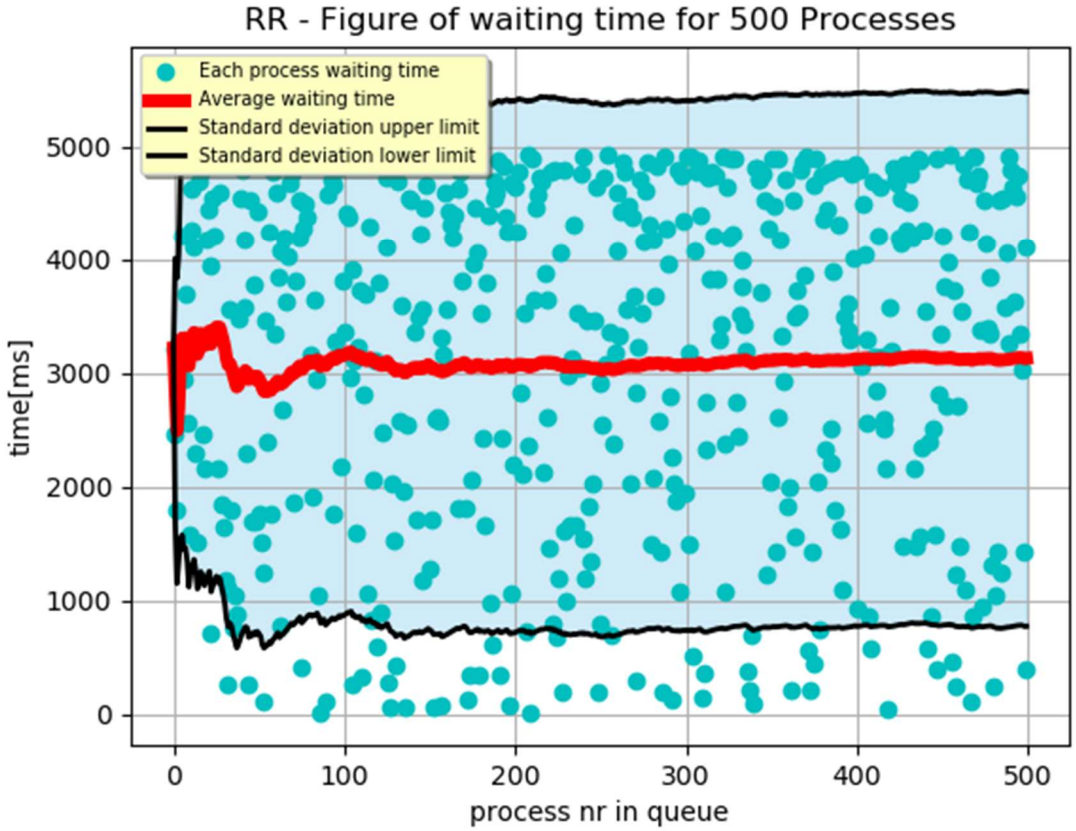

SJF - Figure of waiting time for 100 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
0
200
400
600
800
1000
0
20
40
60
80
100


RR - Figure of waiting time for 100 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
0
200
400
600
800
1000
0
20
40
60
80
100

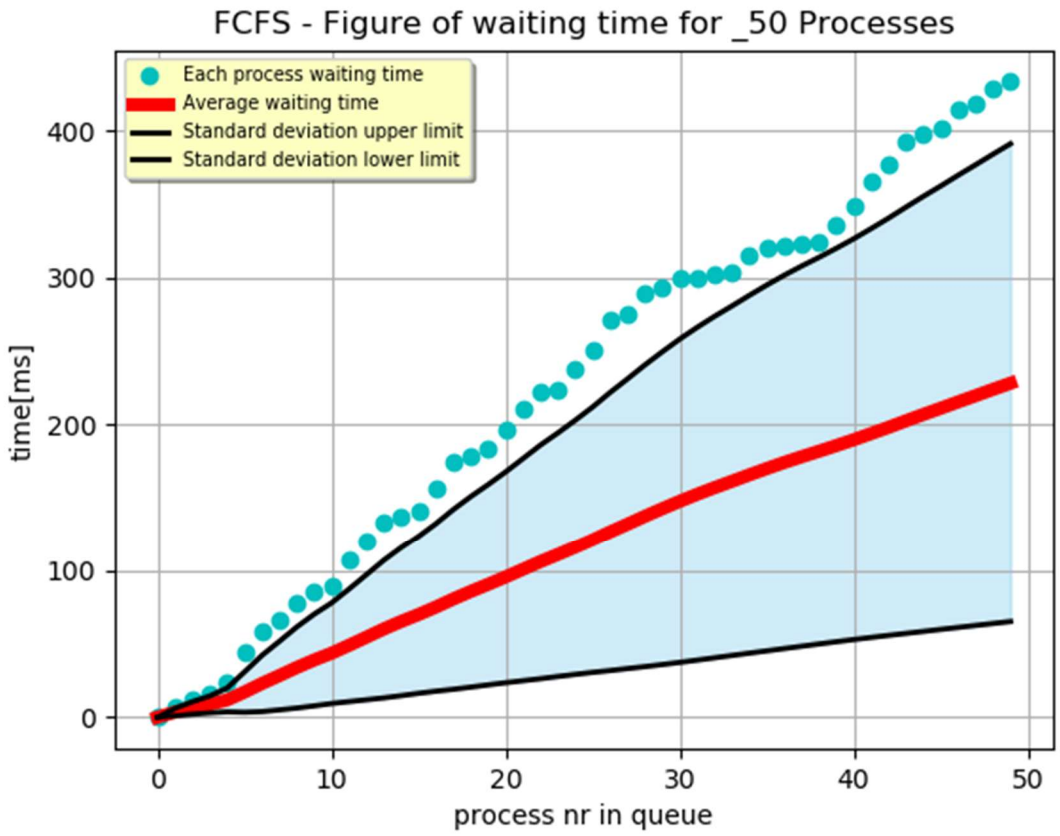

FCFS - Figure of waiting time for 500 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
0
1000
2000
3000
4000
5000
0
100
200
300
400
500

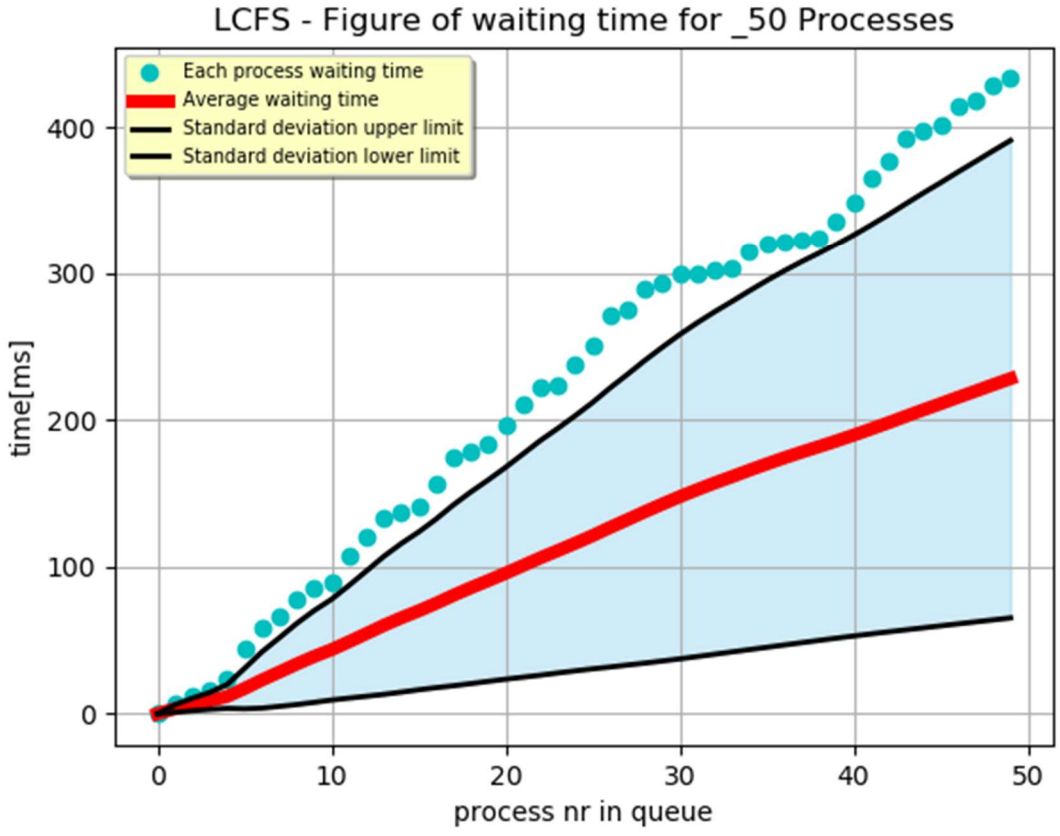

LCFS - Figure of waiting time for 500 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
0
1000
2000
3000
4000
5000
0
100
200
300
400
500

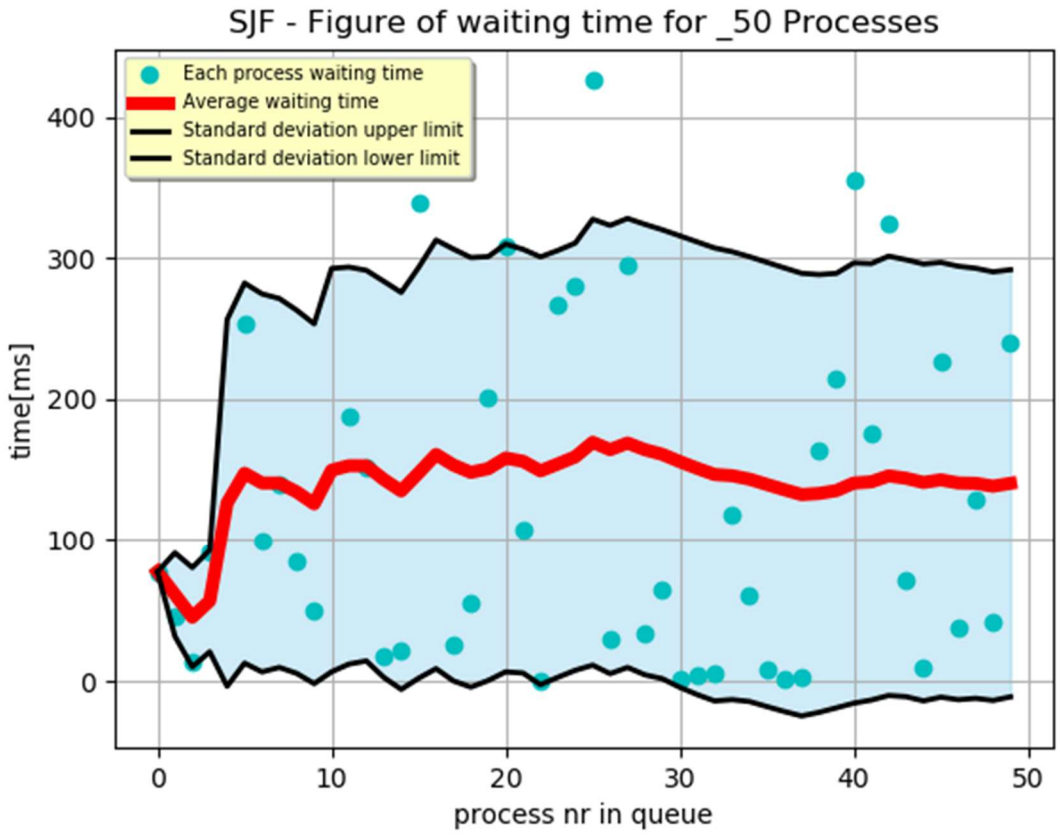

SJF - Figure of waiting time for 500 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
0
1000
2000
3000
4000
5000
0
100
200
300
400
500

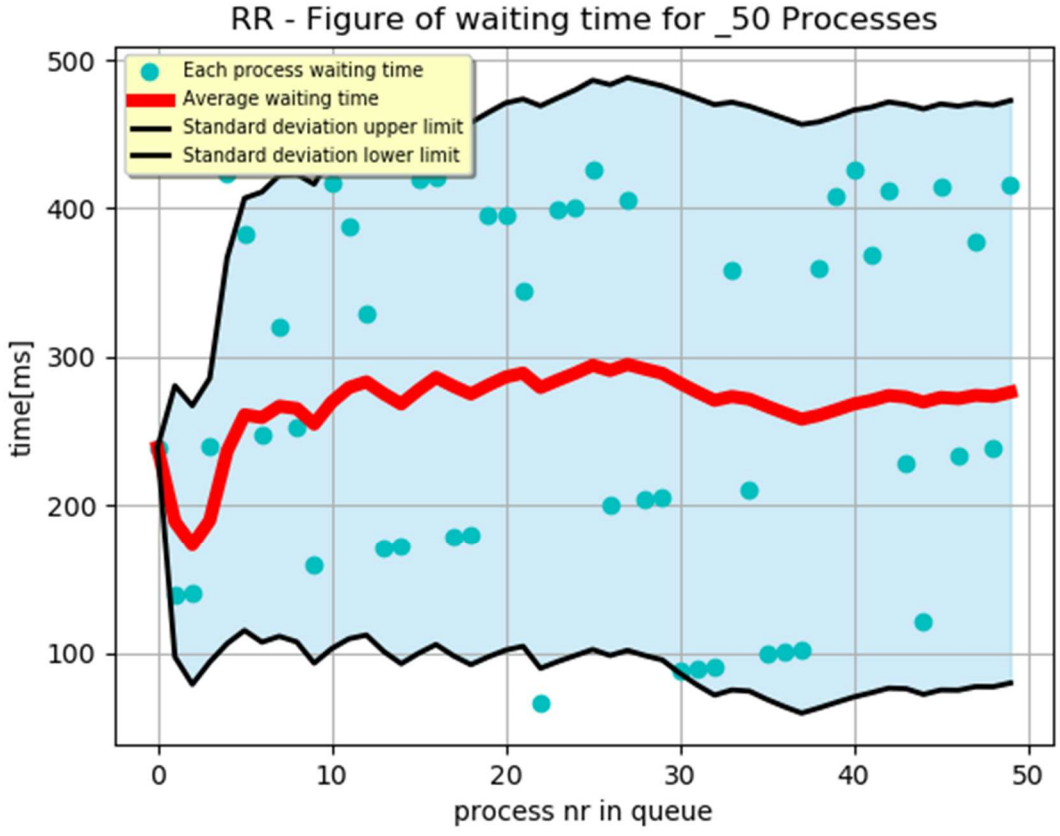

RR - Figure of waiting time for 500 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
0
1000
2000
3000
4000
5000
0
100
200
300
400
500

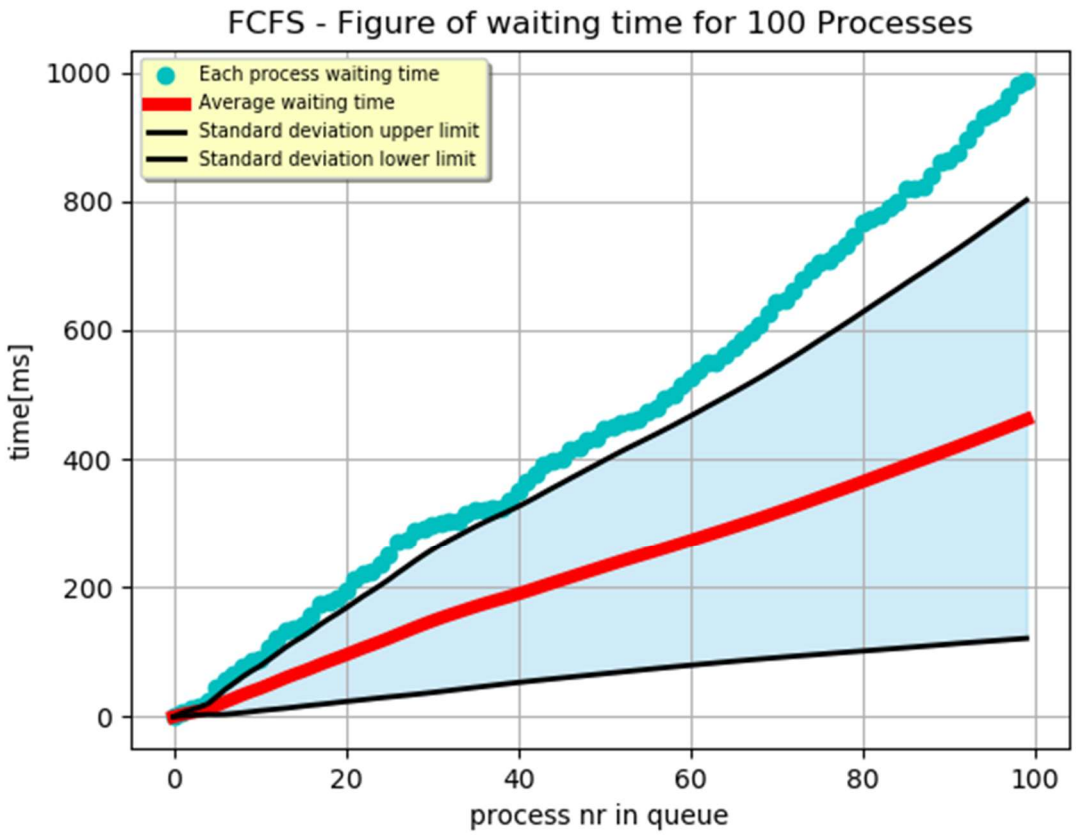
5. . Wykresy dla tego samego czasu przyjścia.

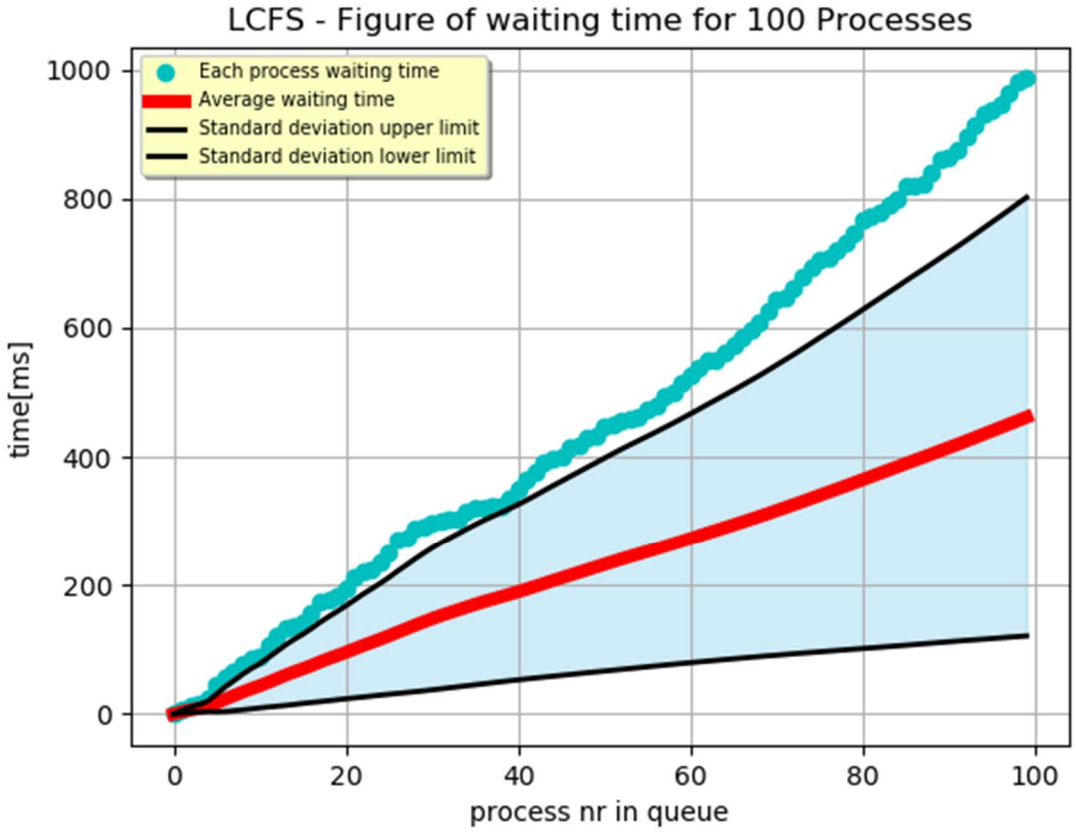


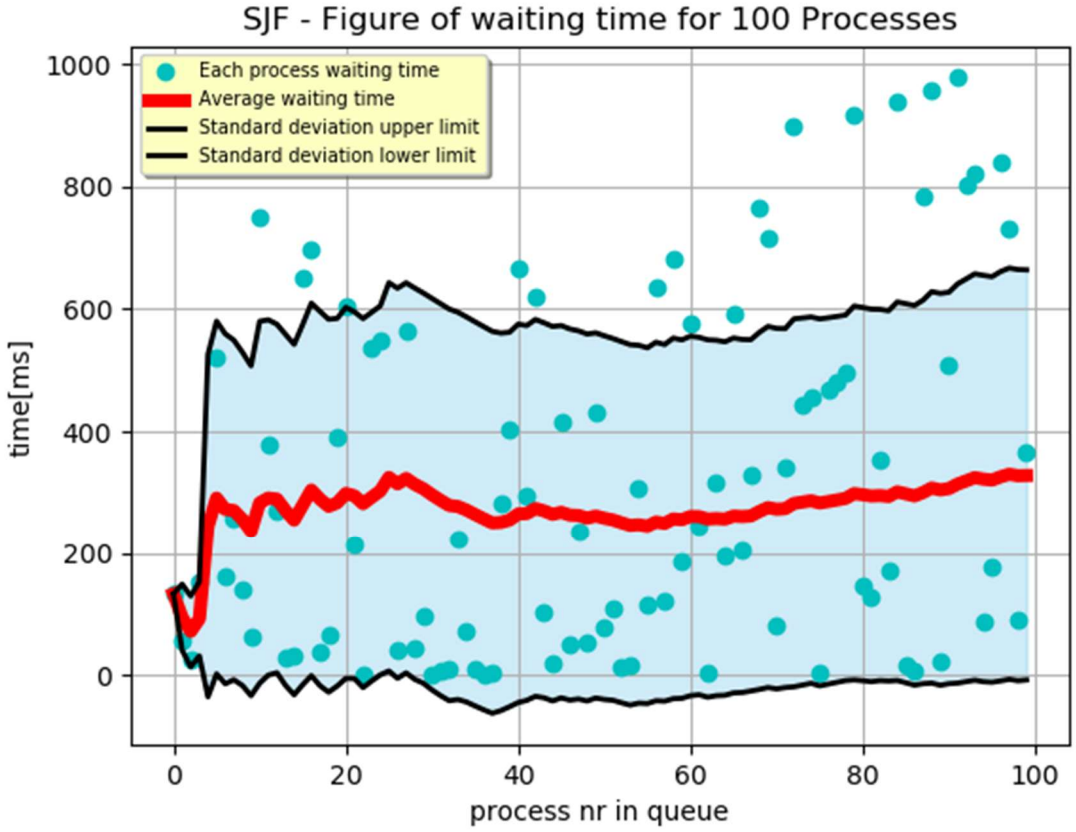



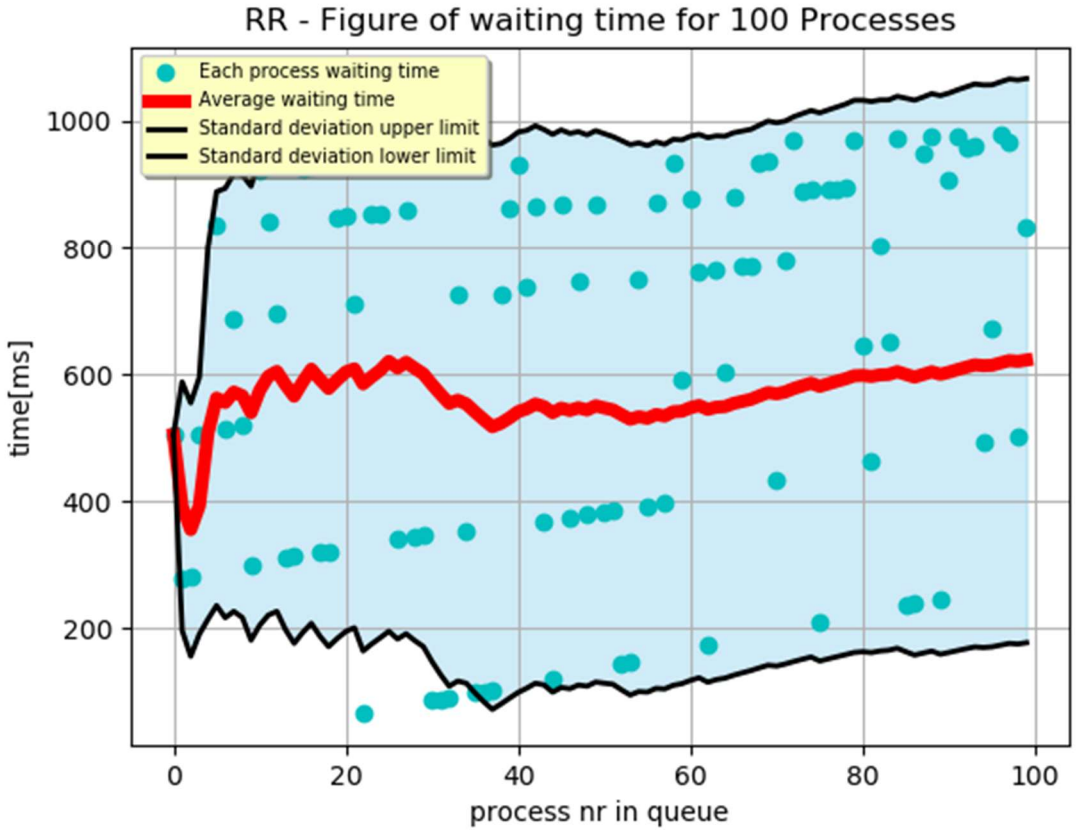

SJF - Figure of waiting time for _50 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
0
100
200
300
400
0
10
20
30
40
50

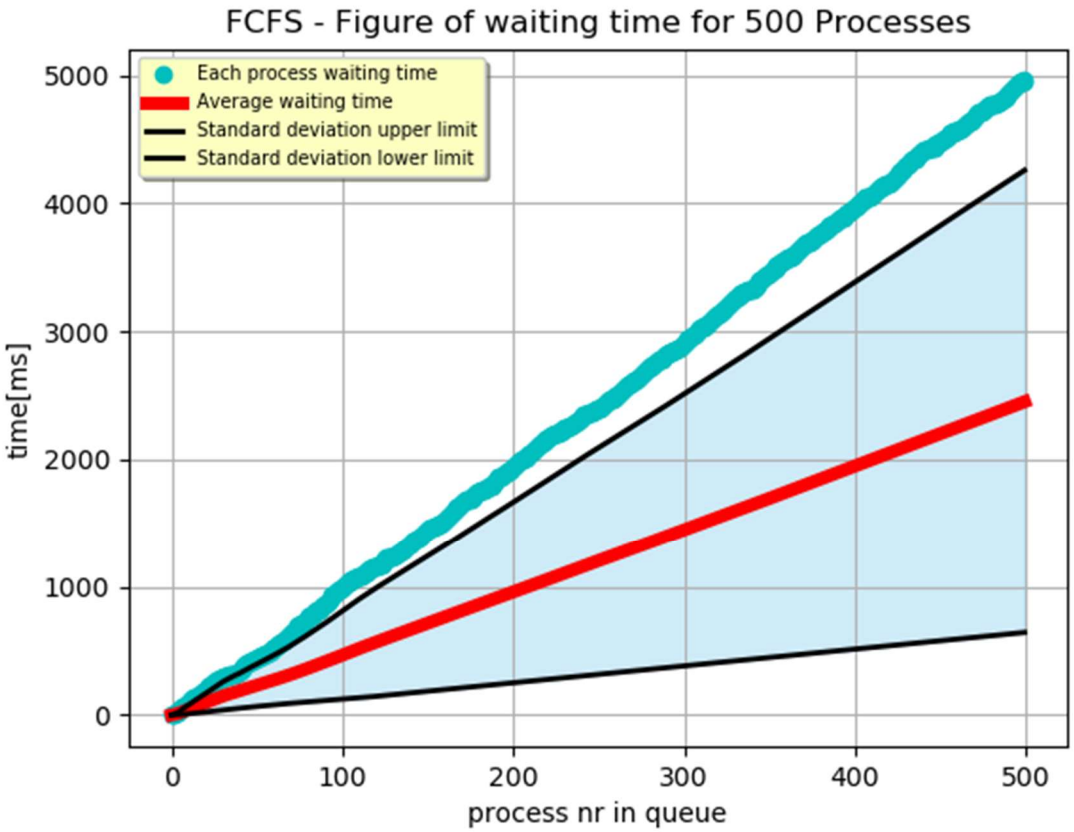

RR - Figure of waiting time for _50 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
100
200
300
400
500
0
10
20
30
40
50

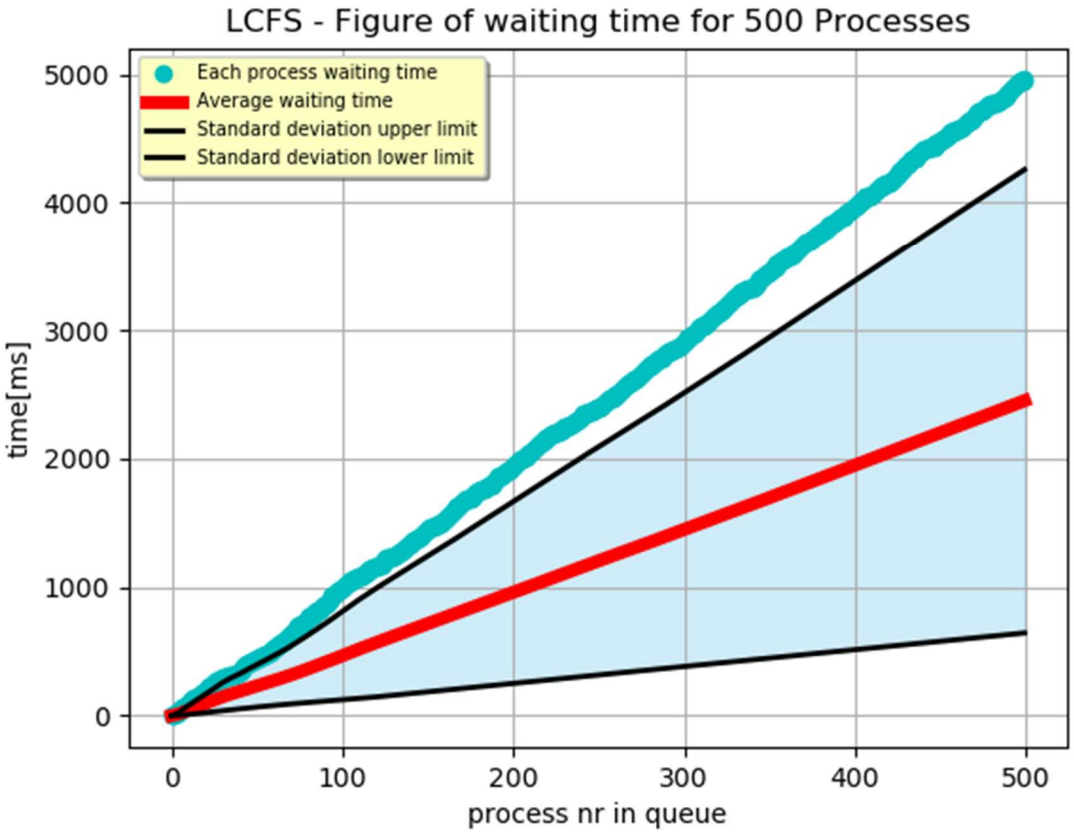

FCFS - Figure of waiting time for 100 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
0
200
400
600
800
1000
0
20
40
60
80
100

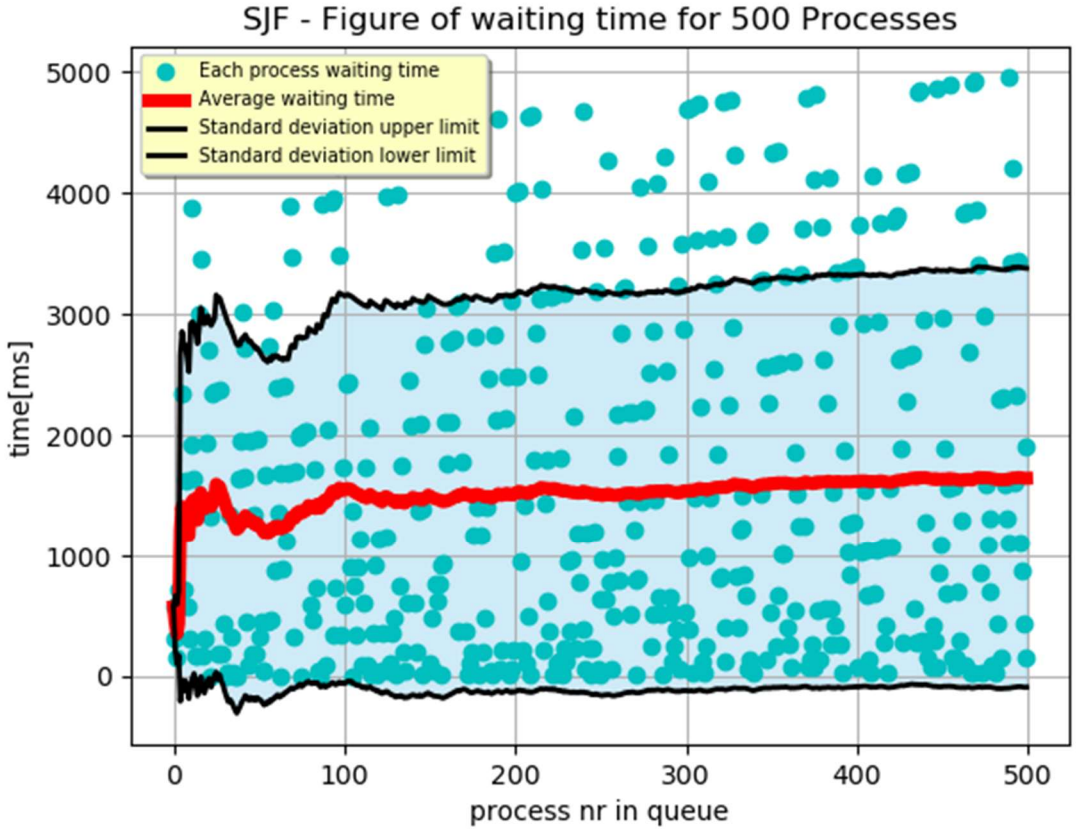

LCFS - Figure of waiting time for 100 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
0
200
400
600
800
1000
0
20
40
60
80
100

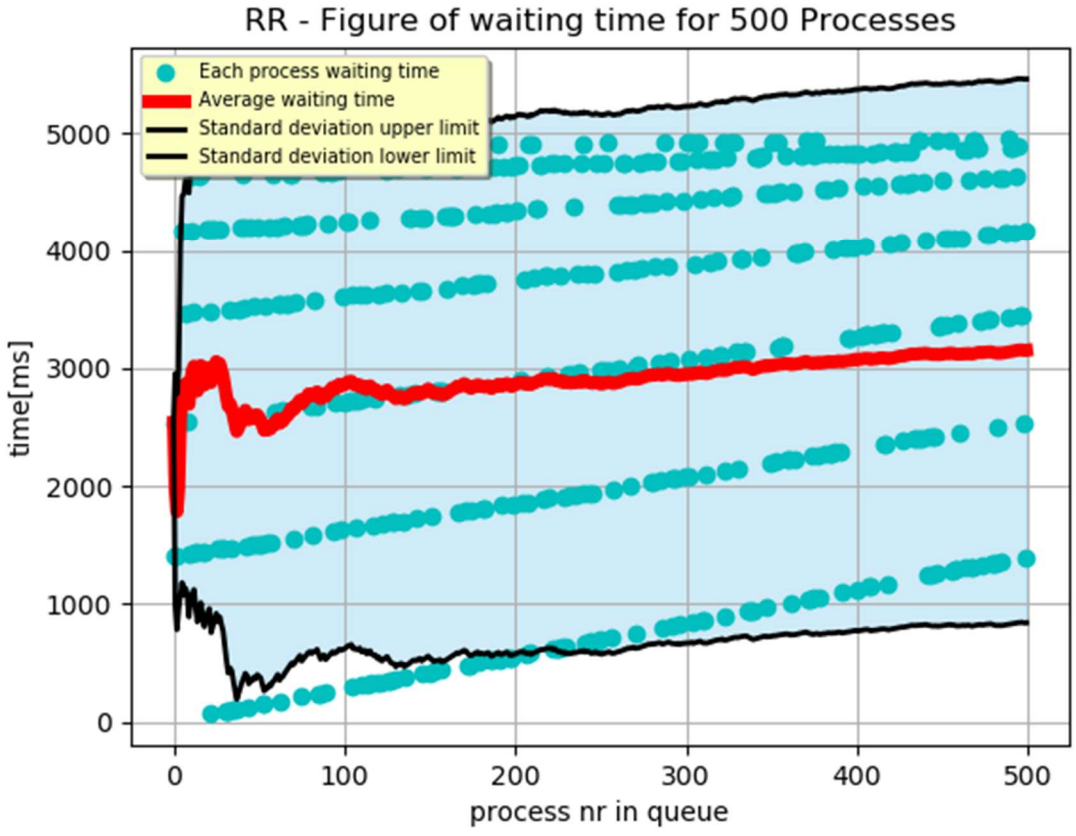

SJF - Figure of waiting time for 100 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
0
200
400
600
800
1000
0
20
40
60
80
100


RR - Figure of waiting time for 100 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
200
400
600
800
1000
0
20
40
60
80
100


FCFS - Figure of waiting time for 500 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
0
1000
2000
3000
4000
5000
0
100
200
300
400
500


LCFS - Figure of waiting time for 500 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
0
1000
2000
3000
4000
5000
0
100
200
300
400
500


SJF - Figure of waiting time for 500 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
0
1000
2000
3000
4000
5000
0
100
200
300
400
500


RR - Figure of waiting time for 500 Processes
Each process waiting time
Average waiting time
Standard deviation upper limit
Standard deviation lower limit
time[ms]
process nr in queue
0
1000
2000
3000
4000
5000
0
100
200
300
400
500

# 6. Wnioski.

Na podstawie wyników eksperymentów łatwo potwierdzić, że najwydajniejszym algorytmem planowania procesów jest algorytm Shortest Job First. Posiada on najmniejsze wyniki dla średniego czasu przetwarzania i oczekiwania procesu dla każdej sprawdzanej ilości procesu oraz dla każdego rodzaju czasu przybycia (różnego i równego zeru dla wszystkich).

W przypadku, gdy procesy przychodzą w tym samym momencie, algorytmy FCFS i LCFS dają te same wyniki, ponieważ one operują na parametrze arrival time, który w tym przypadku jest dla wszystkich równy 0.

W przypadku, gdy procesy przychodzą w różnych momentach, algorytmy FCFS i LCFS dają podobne wyniki, ale nie można stwierdzić, że któryś z nich ma lepsze efekty.

Na wykresach algorytmu FCFS często możemy zauważyć głodzenie procesów, ponieważ wykonuje on „najnowszy” proces.

Algorytm Round-Robin jest najmniej wydajny. Wszystkie zwracane dane są najmniej zadawalające. Jednakże, dla czasów przyjścia równym zeru, można zobaczyć na wykresach dokładne działanie tego algorytmu, ponieważ w przypadku gdy mamy różny czas przyjścia nie widać kolejności.

Gdy spojrzymy obszerniej, czyli porównamy wyniki dla procesów o czasie przyjścia równym zeru i różnym czasie przyjścia, możemy zauważyć, że są one zbliżone. Jednakże pierwsze dwa algorytmy nie spełniają wtedy swojej roli.