Reducing Waiting and Turnaround time in Round Robin Scheduling Algorithm

Dr. Ali Esmaeel*

(Received 9 / 6 / 2022. Accepted 22 / 1 / 2023)

□ ABSTRACT □

The operating system is an intermediary between the user and the computer hardware. It provides an interface that hides from the user the complexity of dealing with hardware, and allocates the resources represented by the processor and main memory to processes in a way that improves the performance of the system.

To execute any program, it must be moved to main memory so that it becomes a process ready to be executed on the processor. In a multitasking operating system, it is allowed to run multiple jobs simultaneously. The ready processes for execution are stored in a special queue called the ready queue, where the scheduler chooses the next process to be executed. The scheduler selects process based on the scheduling algorithms. These algorithms aim to arrange the execution of processes in an optimal manner, as there are several criteria for optimizing the performance of scheduling algorithms, namely: improving CPU Utilization, reducing waiting time, reducing turnaround time, in addition to reducing the number of context switch times.

This paper provides an improvement on the performance of the round Robin scheduling algorithm, which is one of the most important scheduling algorithms. It reduces waiting time and turnaround time by suggesting a dynamic quantum rather than using a fixed quantum. The time allotted for each process is calculated by calculating the median value of the burst time values for all the processes within the ready queue.

The proposed algorithm is compared with a group of other scheduling algorithms in terms of turnaround time, waiting time and number of context switches. The results show the superiority of the proposed algorithm over the other round Robin scheduling algorithms.

Keywords: CPU Scheduling; Round Robin algorithm ;waiting time; turnaround time.

* **Assistant Professor, Department of Systems and Computer Networks, Faculty of Information Engineering, Tishreen University, Lattakia, Syria.**

تخفيض زمن الانتظار وزمن التنفيذ في خوارزمية الجدولة الدائرية (Round Robin)

د. علي اسماعيل*

(تاريخ الإيداع 9 / 6 / 2022. قُبِل للنشر في 22/ 1 / 2023)

□ ملخّص □

يُعدّ نظام التشغيل وسيطاً بين المستخدم وعتاديات الحاسوب. فهو يزوّد واجهة تخفي عن المستخدم تعقيدات التعامل مع العتاديات، ويحصّص الموارد المتمثّلة بالمعالج والذاكرة الرئيسية إلى الإجرائيات بطريقة تحسّن من أداء النظام.

لتنفيذ أي برنامج، يجب نقله إلى الذاكرة الرئيسية ليصبح إجرائية جاهزة للتنفيذ على المعالج. في نظام التشغيل متعدّد المهام (Multitasking)، يُسمح بتنفيذ عدّة إجرائيات بشكل متزامن. تخزّن الإجرائيات الجاهزة للتنفيذ ضمن رتل خاصّ يسمّى رتل الجاهزيّة، إذ يقوم المجدول باختيار الإجرائية التالية للتنفيذ. يستند المجدول في اختياره على خوارزميات الجدولة. تهدف هذه الخوارزميات إلى ترتيب تنفيذ الإجرائيات بطريقة أمثلية، إذ توجد عدّة معايير لتحقيق الأمثليّة في أداء خوارزميات الجدولة، وهي: تحسين استخدام المعالج (CPU Utilization)، وتقليل زمن الانتظار، وتقليل زمن التنفيذ، بالإضافة إلى تقليل عدد مرّات تبديل السياق (context switch).

يزوّد هذا البحث تحسيناً على أداء خوارزمية الجدولة الدائرية التي تعدّ من أهمّ خوارزميات الجدولة. إذ يخفّض زمن الانتظار وزمن التنفيذ من خلال تخصيص شريحة زمنية (quantum time) ديناميكيّة بدلاً من استخدام شريحة ثابتة لكلّ إجرائية طيلة عمل النظام. تُحسب الشريحة الزمنيّة المخصّصة لكلّ إجرائيّة بإيجاد قيمة الوسيط (median) لقيم الرشقات الزمنيةّ (Burst Time) لجميع الإجرائيات الموجودة ضمن رتل الجاهزيّة.

تُقارَن الخوارزمية المقترحة مع مجموعة خوارزميات جدولة أخرى من حيث زمن التنفيذ وزمن الانتظار وعدد مرات تبديل السياق. إذ تُظهر النتائج تفوّق الخوارزمية المقترحة على خوارزميات الجدولة الدائرية الأخرى.

الكلمات المفتاحية: جدولة وحدة المعالجة المركزية؛ خوارزمية الجدولة الدائرية؛ زمن الانتظار؛ زمن التنفيذ.

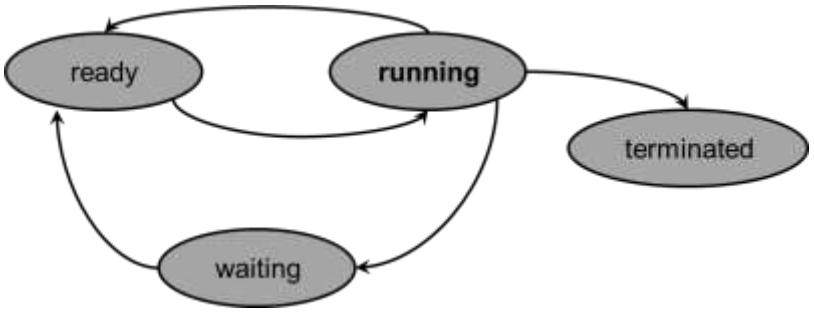
* **مدرّس– قسم النظم والشبكات الحاسوبية– كلية الهندسة المعلوماتية – جامعة تشرين– اللاذقية – سورية.**

مقدمـة:

يُعدّ نظام التشغيل جزءً مهمّاً في أيّ نظام حاسوبي، إذ يمثّل برنامجاً يدير عتاديات الحاسوب، ويؤدي دور الوسيط/الواجهة بين المستخدم وهذه العتاديات [8]. يقوم نظام التشغيل بتحصيص الموارد إلى الإجرائيات، إذ تمثّل وحدة المعالجة المركزيّة (CPU) والذاكرة الرئيسة (RAM) أهمّ هذه الموارد.

تعدّ جدولة CPU أساس نظام التشغيل متعدّد البرمجة (يسمح هذا النظام بتنفيذ عدّة مهام jobs في الوقت نفسه على معالج وحيد). وتهدف هذه الجدولة إلى الاستفادة بشكل أعظمي من استخدام المعالج (تحسين Utilization). تحتاج كلّ إجرائية في نظام التشغيل إلى زمن رشقة (CPU Burst) التي تمثّل مقدار الوقت الذي تستغرقه الإجرائية في استخدام المعالج قبل أن تصبح غير جاهزة (ينتظر المعالج حدث دخل/خرج I/O ليكمل تنفيذها). تخزّن الإجرائيات الجاهزة للتنفيذ في رتل الجاهزية (ready queue)، إذ يقوم المجدول (Scheduler) باختيار إحدى الإجرائيات الموجودة ضمن هذا الرتل ويخصّص المعالج لها لينفّذها.

يعتمد اختيار المجدول للإجرائية التالية على خوارزمية الجدولة المستخدمة في النظام. من الأمثلة على هذه الخوارزميات، خوارزمية FCFS (القادم أولاً يُخدّم أولاً)، وخوارزمية العمل الأقصر أولاً (SFJ)، وخوارزمية الجدولة ذات الأولوية (Priority)، بالإضافة إلى خوارزمية الجدولة الدائرية (Round Robin). يمكن للإجرائية أن تنتقل بين الحالات الموضّحة في الشكل(1).



الشكل(1): حالات الإجرائية في نظام التشغيل.

عندما تصبح الإجرائية جاهزة للتنفيذ (ready) تُوضع ضمن رتل الجاهزية، وعندما يختارها المجدول للتنفيذ تصبح حالتها (running). عندما تنتظر الإجرائية حدث دخل/خرج، يتوقّف تنفيذها مؤقتاً وتصبح في حالة (waiting). بعد إكمال حدث الدخل/الخرج تعود إلى حالة الجاهزية.

يمكن أن تُتّخذ قرارات الجدولة في إحدى الحالات التالية:

- ✓ التبديل من حالة التشغيل (running) إلى حالة الانتظار (waiting).
- ✓ التبديل من حالة التشغيل (running) إلى حالة الجاهزية (ready).
- ✓ التبديل من حالة الانتظار إلى حالة الجاهزية.
- ✓ انتهاء تنفيذ إجرائية.

وُضعت عدّة معايير (Criteria) لأداء خوارزميات الجدولة [8]، وهي:

- استخدام وحدة المعالجة المركزية: يجب أن تكون وحدة المعالجة المركزية CPU مشغولة قدر الإمكان (ليست في حالة idle).
- الإنتاجية (Throughput): وهي عدد الإجرائيات التي تكمل تنفيذها في كلّ وحدة زمنية.
- زمن التنفيذ (Turnaround Time): وهو مقدار الوقت اللازم لتنفيذ إجرائية (المدّة من لحظة وصولها وحتى انتهاء التنفيذ).
- زمن الانتظار(Waiting Time): وهو مقدار الوقت الذي تنتظره الإجرائية في رتل الانتظار (مجموع أزمنة انتظار الإجرائية خلال تنفيذها).
- زمن الاستجابة (Response Time): الزمن المستغرق من لحظة تقديم الطلب حتى إنتاج الاستجابة الأولى

توجد معايير متعدّدة لتحقيق أمثلية خوارزمية الجدولة، وهي:

- ✓ Max CPU Utilization (زيادة استخداميّة المعالج)
- ✓ Max Throughput (زيادة الإنتاجية)
- ✓ Min turnaround Time (تخفيض زمن التنفيذ)
- ✓ Min Response Time (تخفيض زمن الاستجابة)
- ✓ Min waiting time (تخفيض زمن الانتظار)

يجب أيضاً التقليل من عدد مرّات تبديل السياق (context switch) الذي يمثّل عملية إخراج إجرائية من CPU وحفظ حالتها وتحميل حالة الإجرائية التالية لتُنفّذ على CPU.

يهدف البحث التالي إلى تحسين أداء خوارزمية الجدولة الدائرية من خلال تخفيض زمن التنفيذ وزمن الانتظار. إذ يقترح خوارزمية جديدة (Median round Robin) أو اختصاراً MRR، تعتمد على إيجاد الحصّة الزمنية لكلّ إجرائية من خلال حساب الوسيط (median) لجميع قيم الرشقات الزمنية للإجرائيات الموجودة في رتل الجاهزية. أثبتت النتائج فعاليّة الخوارزمية الجديدة في تخفيض زمن الانتظار وزمن التنفيذ مقارنةً مع خوارزميات الجدولة الدائرية الأخرى. تمت الاستفادة من محاكي خاص بخوارزميات جدولة CPU [9] والذي يحوي برمجة العديد من خوارزميات الجدولة الرئيسية بلغة جافا، إذ تم استخدامه لبرمجة الخوارزمية الجديدة ومحاكاتها ومقارنتها مع خوارزمية RR.

أهميّة البحث وأهدافه:

تُعطى كلّ إجرائية شريحة (حصّة) زمنيّة في خوارزمية الجدولة الدائرية، وعند انتهاء هذه الحصّة الزمنية، تُخرج هذه الإجرائية من وحدة المعالجة وتُستبدل بإجرائية أخرى (تبديل السياق). تعتمد خوارزمية الجدولة الدائرية (RR) على تخصيص شريحة ثابتة لكلّ إجرائية، إلّا أنّ تغيير قيمة هذه الشريحة بطريقة ديناميكيّة وفعّالة يقلّل من زمن الانتظار وزمن التنفيذ اللازم للإجرائيات، كما يؤثّر على عدد مرّات تبديل السياق. بدوره يؤثر زمن الانتظار وزمن التنفيذ وعدد مرّات تبديل السياق على أداء النظام ككلّ.

يهدف البحث إلى تقليل زمن الانتظار وزمن التنفيذ في خوارزمية الجدولة الدائرية من خلال تحديد الشريحة الزمنيّة لكلّ إجرائية بطريقة ديناميكيّة. اعتمدت الخوارزمية المقترحة على حساب الوسيط (median) لقيم الرشقات الزمنيّة لجميع

الإجرائيّات الموجودة في رتل الجاهزيّة. أظهرت النتائج فعاليّة الخوارزمية المقترحة في تقليل زمن الانتظار وزمن التنفيذ مقارنة مع خوارزميات الجدولة الأخرى.

طرائق البحث ومواده:

يحسّن البحث من أداء خوارزمية الجدولة الدائرية، إذ يقترح طريقة لتحديد الحصّة الزمنيّة للإجرائيّة ديناميكيّاً، ممّا يخفّض من زمن الانتظار وزمن التنفيذ ويُحسّن بالتالي من أداء النظام.

يقارن البحث بين الخوارزمية المقترحة MRR مع خوارزميات الجدولة الدائرية المحسّنة الأخرى من حيث زمن الانتظار الوسطي وزمن التنفيذ الوسطي وعدد مرّات تبديل السياق. يُقصد بزمن الانتظار الوسطي متوسط (average) أزمنة الانتظار لجميع الإجرائيات، كما يُقصد بزمن التنفيذ الوسطي متوسط (average) أزمنة التنفيذ لجميع الإجرائيات.

لإثبات فعاليّة الخوارزمية الجديدة، يقترح البحث مجموعة من السيناريوهات، يحدّد في كلّ سيناريو مجموعة من الإجرائيات مع زمن الرشقة التي تحتاجها كلّ إجرائية، وهي السيناريوهات نفسها التي طُرحت في المقالة [1]، ويُقارن بين الخوارزمية الجديدة مع مجموعة من خوارزميات الجدولة الأخرى من حيث زمن الانتظار الوسطي وزمن التنفيذ الوسطي.

1- الدراسات المرجعيّة:

في نظام التشغيل متعدّد المهام (multitasking) يمكن تنفيذ عدّة إجرائيات بشكل متزامن [8]. تختلف الأزمنة اللازمة لتنفيذ الإجرائيات في نظام التشغيل، لذلك يجب ترتيب تنفيذ هذه الإجرائيات بطريقة جيّدة لتحسين أداء النظام وتقليل زمن الانتظار وزمن التنفيذ اللازم لهذه الإجرائيات.

دمج البحث [3] خوارزمية الجدولة الدائرية مع خوارزمية الجدولة ذات الأولوية، وتغلّب بذلك على سلبيات كلّ من هاتين الخوارزميتين وساعد على تقليل زمن الانتظار وزمن التنفيذ. بُذلت العديد من الجهود لتحسين أداء خوارزمية الجدولة الدائرية، إذ توضّح الفقرات التالية دراسة لمجموعة من هذه الخوارزميات:

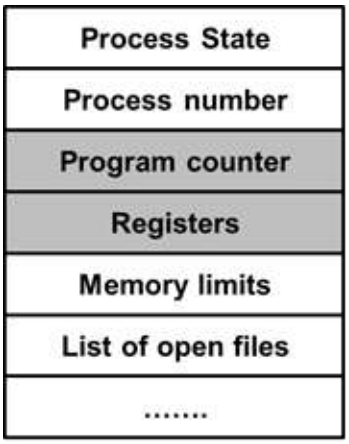
1-1- خوارزمية الجدولة الدائرية (Round Robin)

تعتمد خوارزمية الجدولة الدائرية على استراتيجيّة الشرائح الزمنيّة، إذ يخصّص لكلّ إجرائية شريحة زمنية ثابتة للتنفيذ. توضّح الخطوات التالية آلية عمل خوارزمية الجدولة الدائرية [2]:

عند تنفيذ أيّة إجرائية، ينبغي نقلها إلى الذاكرة الرئيسية، إذ توضع ضمن رتل الجاهزية (Ready Queue). يأتي هنا دور المجدول (Scheduler) في اختيار الإجرائية التالية الواجب تنفيذها. يحتفظ المجدول بقائمة بلوكات التحكّم بالإجرائيات (Process Control Blocks) التي يحتوي كلّاً منها على معلومات متعلّقة بكلّ إجرائية، ويحذف المجدول الإجرائيات التي اكتمل تنفيذها من قاعدة بياناته.

يوضّح الشكل(2) المعلومات المضمّنة ضمن PCB.

تُضاف PCBs إلى مكدّس (Stack)، إذ يلتقط المجدول الإجرائية التالية من قمّة المكدّس. تُعطى كلّ إجرائيّة شريحة أو حصّة زمنيّة (quantum time) ثابتة للتنفيذ. عند انتهاء الحصّة الزمنية للإجرائية قبل انتهاء تنفيذها، تُبدل (swap out) مع الإجرائية التالية وتُضاف إلى نهاية رتل الجاهزية للتنفيذ التالي.



الشكل(2): بلوك التحكّم بالإجرائية PCB

1-2- خوارزمية الجدولة الدائرية (Average Max Round Robin AMRR)

ينفّذ المعالج الإجرائيات وفقاً لزمن وصول (arrival time) كلّ منها. تُجدول الإجرائيات للتنفيذ من رتل الجاهزية، الذي يحوي الإجرائيات الجاهزة للتنفيذ. يُضبط زمن الوصول لجميع الإجرائيات على القيمة 0. ثمّ يُحسب عدد الإجرائيات الموجودة حالياً (وليكن n)، يُؤخذ زمن الرشقة (Burst Time BT) كدخل لعملية الجدولة ويستخدم لحساب الحصّة الزمنيّة. يُشير الرمز TQ (Time Quantum) إلى الحصّة الزمنيّة لكلّ إجرائية [4].

ليكن لدينا على سبيل المثال أربع إجرائيات، وزمن الوصول لكلّ منها يساوي 0. وزمن الرشقة BT لكلّ منها: C1=8, C2=40, C3=72, C4=84. يجب ترتيب هذه الإجرائيات ترتيباً تصاعديّاً وفقاً لزمن الرشقات، وتُحسب الحصّة الزمنية TQ وفق العلاقة التالية:

$$\text{Time Quantum} = (\text{Average} + \text{Maximum BT})/2$$

إن قيمة المتوسط (Average) تساوي 51، وأعلى قيمة للرشقات الزمنية يساوي 84، تصبح قيمة الحصّة الزمنية في الدور الأول من التنفيذ TQ=(51+84)/2=67.5.

بعد انتهاء الدور الأوّل من تنفيذ جميع الإجرائيات، تبقى لدينا إجرائيتين فقط (C3=4, C4=16) إذ تكون الإجرائيتان C1 وC2 قد انتهى تنفيذهما وتُحذفان من رتل الجاهزيّة.

يُحسب زمن الحصّة الزمنية في الدور الثاني من التنفيذ بالطريقة السابقة نفسها، وتكون قيمته 13، عندها تنتهي الإجرائية C3 وتحذف من الرتل وتحتاج الإجرائية C3 إلى 3 فقط لتنهي تنفيذها. تُنفّذ الإجرائية C3 في الدور الثالث وتُحذف من الرتل بعدها.

1-3- خوارزمية جدولة دائرية جديدة (A New Round Robin ANRR)

تهدف هذه الخوارزمية إلى إيجاد قيمة أمثليّة للحصّة الزمنيّة لكلّ إجرائيّة [7]. إذا كان رتل الجاهزية فارغاً، فإن الحصّة الزمنيّة تساوي الرشقة الزمنية للإجرائيات التي تعمل حاليّاً، وإلّا فإن الحصّة الزمنيّة تساوي متوسط (average) جميع الرشقات الزمنيّة للإجرائيات الموجودة في رتل الجاهزيّة.

1-4- خوارزمية الجدولة الدائرية (Modified Median round Robin MMRR)

اقترح الباحث في هذه الخوارزميّة قيمة الحصّة الزمنية لتكون مساوية للجذر التربيعي لناتج الجداء التالي [6]:

$$\text{Median}*\text{HB}$$

يمثّل median القيمة الوسطى لجميع قيم الرشقات الزمنيّة بعد ترتيبها بشكل تصاعدي، بينما يمثّل HB قيمة أعلى رشقة زمنية من بين الإجرائيات الموجودة.

1-5- خوارزمية الجدولة الدائرية (Median-Average Round Robin MARR)

اقترح الباحث في هذه الخوارزمية [1] طريقة ديناميكيّة جديدة لإيجاد قيمة الحصّة الزمنيّة التي تُحسب بالعلاقة التالية:

$$QT = (Average + Median)/2$$

يمثّل Average متوسط قيم جميع الرشقات الزمنية للإجرائيات، ويمثّل Median الوسيط لقيم جميع الرشقات الزمنيّة بعد ترتيبها ترتيباً تصاعديّاً.

أثبت الباحث في هذه المقالة [1] تفوّق هذه الخوارزمية على خوارزمية RR، ANRR، MMRR من حيث زمن الانتظار الوسطي وزمن التنفيذ الوسطي.

2- الخوارزمية المقترحة (Median Round Robin MRR)

اعتمدت الدراسات السابقة على إيجاد قيمة الحصّة الزمنيّة بطريقة ديناميكيّة، ويعتمد بعضها على إجراء عمليات حسابيّة على أزمنة الرشقات للإجرائيات الموجودة في رتل الجاهزيّة.

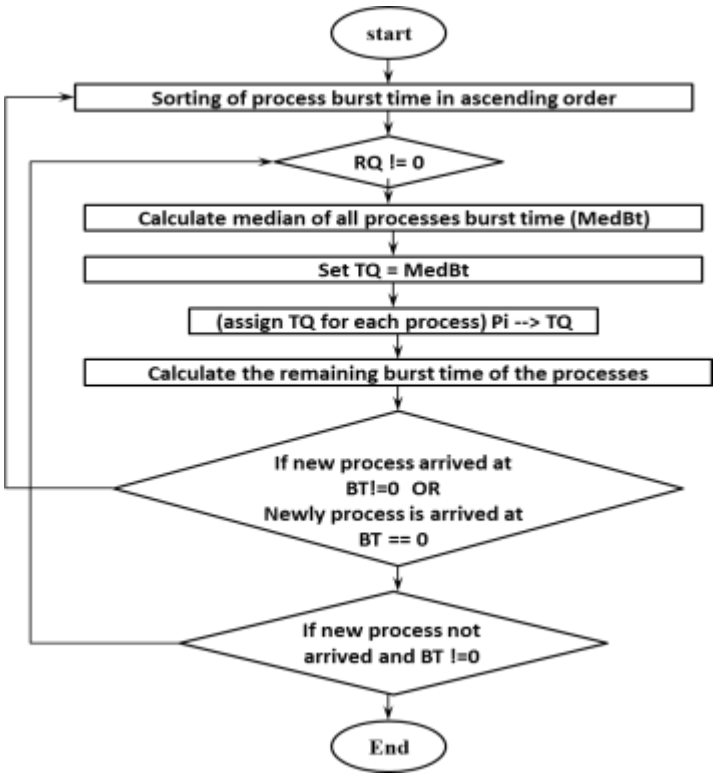
تحسب الخوارزمية الجديدة MRR قيمة الحصّة الزمنيّة من خلال إيجاد قيمة الوسيط (median) فقط لجميع قيم الرشقات الزمنية للإجرائيات الموجودة في رتل الجاهزية.

لتوضيح آلية عمل الخوارزمية نعود للمثال الموجود في الفقرة (4-2). إذ يكون لدينا أربع إجرائيات مع قيم الرشقات الزمنية التالية: (C1=8, C2=40, C3=72, C4=84). في الدور الأوّل من التنفيذ تحسب قيمة الحصّة الزمنيّة لتكون:

$$Q1= median(8,40,72,84) = (40 + 72)/2 = 56$$

بعد انتهاء الدور الأول، ينتهي تنفيذ الإجرائيتين (C1,C2) لأن قيمة الرشقة لكلّ منهما أصغر من الحصّة الزمنيةّ المخصّصة، وتُحذفان من الرتل. يبقى لدينا إجرائيتان مع زمن الرشقة المتبقي التالي لكلّ منهما: (C3=16, C4=28). تُحسب الحصّة الزمنيّة للدور الثاني من التنفيذ (Q2=median(16,28)=22). ينتهي تنفيذ الإجرائية C3 في الدور الثاني، وتحتاج بعدها الإجرائية C4 إلى 6 وتنتهي في الدور الثالث.

يوضّح الشكل(3) المخطّط التدفّقي للخوارزمية الجديدة MRR.



الشكل(3): المخطّط التدفّقي للخوارزمية المقترحة (MRR)

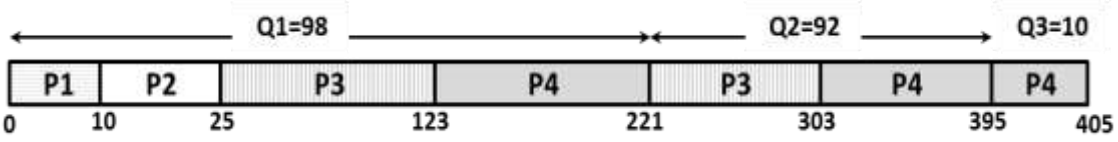
النتائج والمناقشة:

السيناريو الأول: يبيّن الجدول (1) مجموعة من الإجرائيات مع أزمنة الرشقات (Burst Time) الموافقة. إذ أن زمن الوصول لكلّ إجرائية يساوي 0. وتصل الإجرائيات بالترتيب P1-P2-P3-P4. يوضّح العمود الثالث كيفيّة حساب الحصّة الزمنيّة (الوسيط) في الخوارزمية المقترحة (يجب ترتيب أزمنة الرشقات بشكل تصاعدي)، وحساب زمن الانتظار الوسطي (AWT) وزمن التنفيذ الوسطي (ATT).

جدول 1: مجموعة الإجرائيات وزمن الرشقة المطلوب لكلّ منها(السيناريو1)

Process	Burst Time	Quantum Time
P1	10	Q1 = median(10,15,180,200) = (15+180)/2 = 98 Q2 = median(82, 102) =92 Q3 = 10
P2	15	
P3	180	ATT(Average Turnaround Time) = (10+25+303+405)/4 = 185.75 AWT(Average Waiting Time)= (0+10+123+205)/4= 84.5
P4	200	

يوضّح الشكل(4) مخطّط غانت لتسلسل تنفيذ الإجرائيات وفقاً للخوارزمية MRR المقترحة.



الشكل(4): مخطّط غانت لتنفيذ الإجرائيات في السيناريو الأول.

يوضّح الجدول(2) مقارنة بين الخوارزميات الموضّحة ضمن الدراسات المرجعيّة من حيث زمن الانتظار الوسطي وزمن التنفيذ الوسطي، إذ يمثّل الصف الأخير زمن الانتظار وزمن التنفيذ في الخوارزمية المقترحة (MRR).

جدول 2: نتائج (السيناريو1)

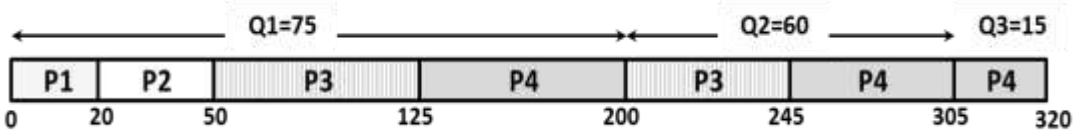
Algorithm	Time Quantum	Turnaround Time	Waiting Time	Context Switch
RR	20	201.25	100	19
AMRR	151, 44, 5	199	97.75	5
AN	102, 88, 10	186.75	85.5	5
MMRRA	140, 55, 5	196.25	95	5
MARR	100, 90, 10	186.25	85	5
MRR	**98, 92, 10**	**185.75**	**84.5**	**5**

السيناريو الثاني: يبيّن الجدول3 مجموعة من الإجرائيات مع أزمنة الرشقات الموافقة. إذ أن زمن الوصول لكلّ إجرائية يساوي 0. وتصل الإجرائيات بالترتيب P1-P2-P3-P4.

جدول 3: مجموعة الإجرائيات وزمن الرشقة المطلوب لكلّ منها(السيناريو2)

Process	Burst Time	Quantum Time
P1	20	Q1 = median(20,30,120,150) = (30+120)/2 = 75 Q2 = median(45,75) = 60 Q3 = 15 ATT = (20+50+245+320)/4 = **158.75** AWT = (0+20+125+170)/4 = **78.75**
P2	30	
P3	120	
P4	150	

يوضّح الشكل(5) مخطّط غانت لتسلسل تنفيذ الإجرائيات السابقة وفق خوارزمية MRR المقترحة.



الشكل(5): مخطّط غانت لتنفيذ الإجرائيات في السيناريو الثاني.

يوضّح الجدول4 مقارنة بين الخوارزميات من حيث زمن الانتظار الوسطي وزمن التنفيذ الوسطي، إذ يمثّل الصف الأخير زمن الانتظار وزمن التنفيذ في الخوارزمية المقترحة (MRR).

جدول 4: نتائج (السيناريو2)

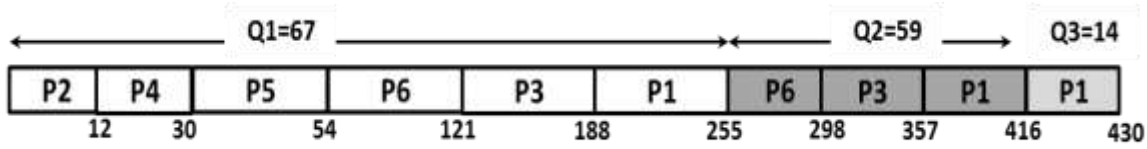
Algorithm	Time Quantum	Turnaround Time	Waiting Time	Context Switch
RR	20	175	95	14
AMRR	115, 28, 7	168.75	88.75	5
AN	80, 55, 15	160	80	5
MMRRA	106, 36, 8	166.5	86.4	5
MARR	78, 57, 15	159.5	79.5	5
MRR	**75, 60, 15**	**158.75**	**78.75**	**5**

السناريو الثالث: يبين الجدول(5) مجموعة من الإجرائيات مع أزمنة الرشقات الموافقة، ولكن الإجرائيات هنا تمتلك أزمنة وصول مختلفة (العمود الثاني).

جدول 5: مجموعة الإجرائيات وزمن الرشقة المطلوب لكلّ منها (السيناريو3)

Process	Arrival Time	Burst Time	Quantum Time
P1	6	140	Q1 = median(12,18,24,110,124,140) =(24+110)/2 = 67 Q2 = median(43,59,73) = 59 Q3 = 14
P2	1	12	
P3	5	126	
P4	2	18	ATT = (424+11+352+28+51+294)/6 = 193.33 AWT = (284+11+226+10+27+184)/6 = 123.67
P5	3	24	
P6	4	110	

يوضّح الشكل(6) مخطّط غانت لتنفيذ الإجرائيات السابقة وفقاً لخوارزمية MRR المقترحة.



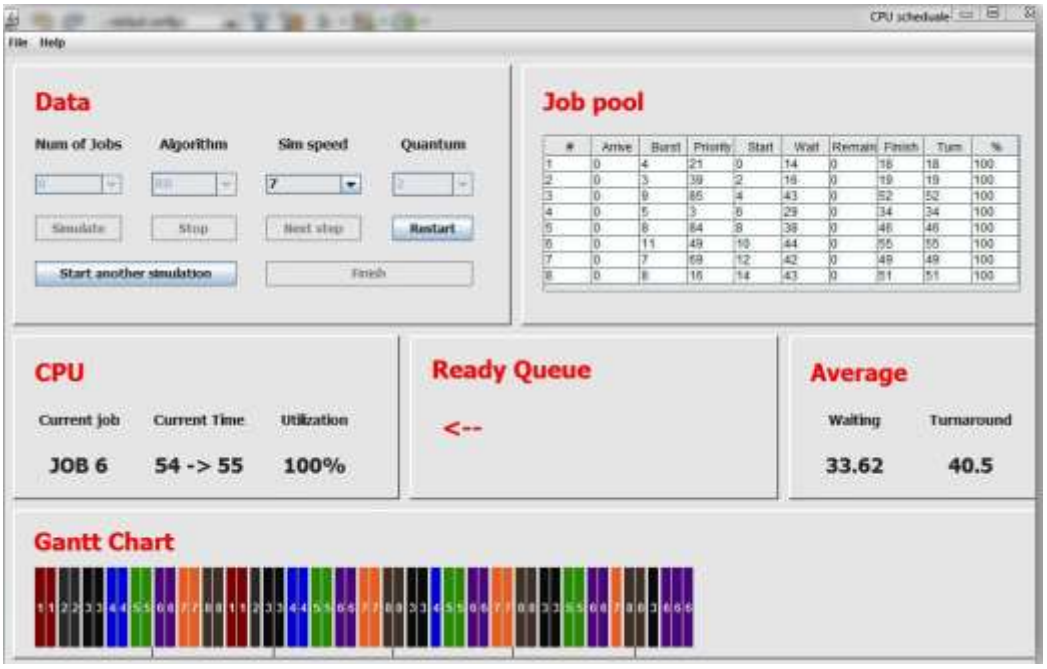
الشكل(6): مخطّط غانت لتنفيذ الإجرائيات في السيناريو الثالث.

يوضّح الجدول6 مقارنة بين الخوارزميات من حيث زمن الانتظار الوسطي وزمن التنفيذ الوسطي، إذ يمثّل الصف الأخير زمن الانتظار وزمن التنفيذ في الخوارزمية المقترحة (MRR).

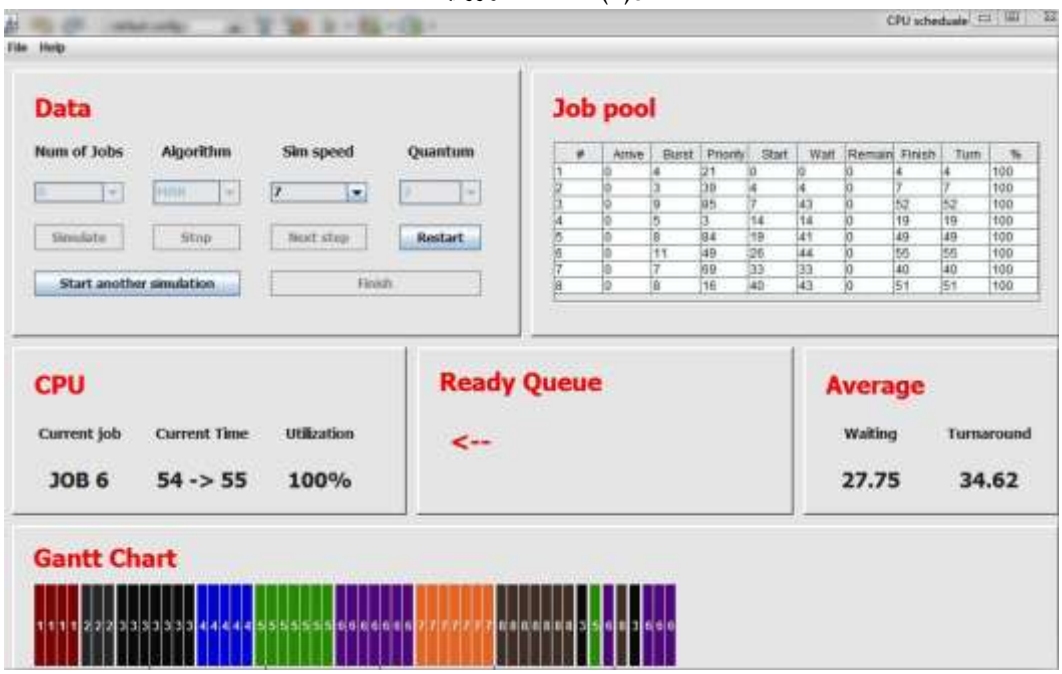
جدول 6: النتائج (السيناريو3)

Algorithm	Time Quantum	Turnaround Time	Waiting Time	Context Switch
RR	20	226.66	155	23
AMRR	106,27,7	216.33	144.66	8
AN	72,53,8,7	208.16	136.5	10
MMRRA	97,35,8	211.83	140.16	8
MARR	70,56,14	198.33	126.66	8
MRR	**67, 59, 14**	**193.33**	**123.67**	**8**

نُفّذت الخوارزمية المقترحة MRR بلغة جافا، وأجريت محاكاة للمقارنة بين الخوارزمية الجديدة مع خوارزمية الجدولة الدائرية من حيث زمن الانتظار وزمن التنفيذ [9]. يوضّح الشكل(7) محاكاة لخوارزمية RR، بينما يوضّح الشكل(8) محاكاة للخوارزمية MRR بعد برمجتها. نُفّذت المحاكاة من أجل ثمان إجرائيات موضّحة في الجدول الموجود على يمين الشكل (7) وهي الإجرائيات نفسها الموضّحة في الشكل(8). يمكن ملاحظة تفوّق الخوارزمية MRR على خوارزمية RR، إذ انخفض زمن الانتظار الوسطي من 33.62 إلى 27.75، كما انخفض زمن التنفيذ الوسطي من 40.5 إلى 34.62.



الشكل(7): محاكاة لخوارزمية RR.



الشكل(8): محاكاة لخوارزمية MRR.

الاستنتاجات والتوصيات:

تؤثّر خوارزميات الجدولة على أداء نظام التشغيل. وتعدّ خوارزمية الجدولة الدائريّة من أهمّ هذه الخوارزميات. تقوم هذه الخوارزمية بتخصيص شريحة زمنية محدّدة لكلّ إجرائية وعند انتهاء هذه الشريحة، تُبدل مع الإجرائية التالية لتنفّذ على المعالج. اقترح هذا البحث طريقة ديناميكيّة لتحديد الشريحة الزمنيّة للإجرائية بدلاً من إعطائها قيمة ثابتة كما في خوارزمية الجدولة الدائرية. تعتمد الخوارزمية الجديدة على حساب الوسيط (median) لقيم الرشقات الزمنية لجميع الإجرائيات الموجودة ضمن رتل الجاهزيّة. أُجريت مقارنة بين الخوارزمية المقترحة مع عدّة خوارزميات جدولة أخرى

(RR، AMRR، AN، ..) من حيث زمن الانتظار الوسطي وزمن التنفيذ الوسطي وعدد مرّات تبديل السياق. تفوّقت الخوارزمية الجديدة على خوارزمية الجدولة الدائرية بشكل كبير من حيث زمن التنفيذ وزمن الانتظار وعدد مرّات تبديل السياق، كما أنّها تفوّقت على خوارزميات الجدولة الدائرية المحسّنة عن خوارزمية الجدولة الدائرية.

يمكن برمجة الخوارزمية المقترحة بلغة برمجة أخرى وإجراء محاكاة للمقارنة بين أداء هذه الخوارزمية وخوارزميات الجدولة الأخرى، وذلك بعد إيجاد محاكي خاصّ بخوارزميات الجدولة في نظام التشغيل.

References:

[1] SAKASHI, et all, *A new median-average round Robin scheduling algorithm: An optimal approach for reducing turnaround and waiting time*, Alexandria Engineering Journal,61,2022.

[2] SINGH, H., SRIVSTAVA, H.M., KUMAR, D. *A reliable numerical algorithm for the fractional vibration equation*, Chaos, Solitons Fractals 103, 2017, 131-138.

[3] ABU-DALBOUH, H.M. *A New Combination Approach to CPU Scheduling based on Priority and Round-Robin Algorithms for Assigning a Priority to a Process and Eliminating Starvation*, International Journal of Advanced Computer Science and Applications, 2022, Vol. 13, No. 4.

[4] BOCHENINA, K. *A comparative study of scheduling algorithms for the multiple deadline-constrained workflows in heterogeneous computing systems with time windows*, Proc. Comput. Sci. 29 ,2014.

[5] AIJAZ, M., TARIQ, R., GHORI, M., RIZVI, S.W., and QAZI, E.F. *Efficient Round Robin Algorithm (ERRA) using the Average Burst Time*. In 2019 International Conference on Information Science and Communication Technology (ICISCT) (pp. 1-5). IEEE, 2019.

[6] MORA, H., ABDULLAHI, S.E, JUNAIDU, S.B. *Modified Median Round Robin Algorithm (MMRRA)*, in: 2017 13th International Conference on Electronics, Computer and Computation, ICECCO 2017, 2018, vol. 2018-Janua, doi: 10.1109/ICECCO.2017.8333325.

[7] NOON, A., KALAKECH, A., and KADRY, S. *A new round robin based scheduling algorithm for operating systems: dynamic quantum using the mean average*, arXiv Prepr. arXiv1111.5348, (2011).

[8] P. B. Galvin, G. Gagne and A. Silberschatz, Operating system concepts. John Wiley & Sons. ,2003.

[9] https://www.mediafire.com/file/8atbafplo17drl1/CPU.rar/file