مجلة جامعة طرطوس للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (٧) العدد(٤) ٢٠٢٣
Tartous University Journal for Research and Scientific Studies - engineering Sciences Series Vol. (7) No. (4) 2023

تحسين توزيع الموارد في الحوسبة السحابية
باستخدام نظام الاستدلال الضبابي و خوارزمية ذكاء الأسراب

أ .د يعرب ديوب *

حسام محفوض **

(تاريخ الإيداع ٢٠٢٣/١/٢٩ . قُبِل للنشر في ٢٠٢٣/٥/٢٢)

□ ملخّص□

يعتبر مفهوم الحوسبة السحابية Cloud Computing أحد المفاهيم واسعة الانتشار في السنوات الأخيرة، نظراً لدعمها مبدأ المعالجة المتوازية مما يوفّر إمكانية تنفيذ العديد من المهام في الوقت نفسه.

دخلت تطبيقات الحوسبة السحابية مختلف مجالات الحياة، ومكّنت المستخدمين من استخدام قدرات مراكز البيانات التي توفرها السحابة في تنفيذ مهامهم اليومية بشكل فعّال.

يتسبب الاستخدام المشترك للموارد دون وجود استراتيجية عادلة لتوزيعها في العديد من المشكلات والتحديات في البنى السحابية مثل: قابلية التوسع، التسامح مع الأخطاء، الموثوقية، التوافر كذلك كفاءة استخدام الطاقة..إلخ.

تم تصنيف مشكلة توزيع الموارد في النظام السحابي على أنها من النوع NP–Complete، وهي فئة من مشاكل القرار المعقدة التي تستوجب استخدام الخوارزميات الذكية في حلها.

انطلاقاً من ذلك يقترح هذا البحث تحسيناً لعملية توزيع الموارد السحابية اللازمة لتنفيذ طلبات المستخدمين، بالاستناد على خوارزمية ذكاء الأسراب Particle Swarm Optimization، ونظام الاستدلال الضبابي Fuzzy Inference System.

الكلمات المفتاحية: توزيع الموارد، الآلات الافتراضية، الحوسبة السحابية، النظام الضبابي، ذكاء الأسراب.

* أستاذ في قسم هندسة تكنولوجيا المعلومات –كلية هندسة تكنولوجيا المعلومات والاتصالات–جامعة طرطوس–سوريا

** طالب ماجستير في قسم هندسة تكنولوجيا المعلومات– كلية هندسة تكنولوجيا المعلومات والاتصالات–جامعة طرطوس–سوريا

مجلة جامعة طرطوس للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (٧) العدد(٤) ٢٠٢٣
Tartous University Journal for Research and Scientific Studies - engineering Sciences Series Vol. (7) No. (4) 2023

Improving Resources Distribution in Cloud Computing Using Particle Swarm Optimization and Fuzzy Inference System

Prof. Yaroub Dayoub *
Hussam Mahfoud **

(Received 29/1/2023 . Accepted 22/5/2023)

□ ABSTRACT

The concept of cloud computing is one of the widespread concepts in recent years, due to its support of parallel processing principle, which provides the possibility of executing many tasks at the same time.

Cloud computing applications have entered various fields of life, and enabled users to use the capabilities of data centers provided by the cloud in executing their daily tasks effectively.

Shared use of resources without a fair strategy for distributing them causes many problems and challenges in cloud architectures such as: scalability, fault tolerance, reliability, availability as well as energy efficiencydue to..etc.

The resource allocation problem in the cloud system is categorized as NP-Complete, which is a class of complex decision problems that require using of intelligent algorithms to solve them.

Based on this, this research proposes an improvement in the process of distributing cloud resources necessary to implement user requests, based on the Particle Swarm Optimization algorithm and the Fuzzy Inference System.

Key Words: Resources Distribution, Virtual Machines, Cloud Computing, Fuzzy System, Particle Swarm Optimization.

*Professor in Information and Communication Technology Engineering Faculty,Tartous University, Syria.
**Master student, Information Technology Engineering Department, Information and communication Technology Engineering, Tartous University, Syria.

1. المقدمة:

تعدُّ الحوسبة السحابية نموذجاً لتسهيل الوصول إلى الشبكة عند الطلب بهدف مشاركة موارد الحوسبة المختلفة مثل المخدمات Infrastructure as a Service(IaaS)، أجهزة التخزين data Storage as a Service (dSaaS)، أنظمة التشغيل Platform as a Service (PaaS)، البرمجيات والخدمات Software as a Service (SaaS) وغيرها.

تعتبر مراكز البيانات بمثابة النواة للحوسبة السحابية، حيث تتضمن الآلاف من الحواسيب المرتبطة ببعضها البعض بطريقة تمكن من تزويد الخدمات السحابية.

تتضمن مراكز البيانات الموجودة ضمن بيئات الحوسبة السحابية عدد هائل من الموارد السحابية، تقابلها قائمة من مهام تطبيقات المستخدمين المختلفة من حيث الخصائص، مما يجعل عملية توزيع الموارد الموجودة تواجه تعقيدات أكبر بكثير عما هو عليه الحال في البيئات الموزعة التقليدية

مع الزيادة الكبيرة في الآونة الأخيرة لأعداد مستخدمي البيئات السحابية، برزت الحاجة إلى تطوير منهجيات توزيع الموارد فيها بما يتوافق مع معايير جودة الخدمة المتعلّقة بكل من مزودي ومستخدمي الخدمات السحابية على حد سواء، حيث تعددت المنهجيات التي طرحها الباحثين في هذا المجال وتنوَعت أهدافها.

تم استخدام الخوارزميات التقليدية في البداية حيث تم اقتراح توزيع الموارد بالاستناد على خوارزمية القادم أولاً يخدَم أولاً FCFS[١]، والتي تقوم فكرتها على عملية تجميع المهام الواردة إلى النظام في رتل حتى تصبح الموارد متاحة، وعندها يتم تخصيص الموارد لها تبعاً لزمن وصول المهمة إلى النظام، كانت المشكلة في ارتفاع زمن التنفيذ بشكل كبير عند زيادة عدد المهام الواردة إلى النظام.

ثم تم استخدام خوارزمية Optimized FCFS [2]، والتي عملت على تلافي السلبيات التي ظهرت عند تطبيق الخوارزمية الأصلية(FCFS)، من خلال المباشرة بعملية تقسيم الموارد المتاحة على مجموعة جزئية من المهام في حالة عدم كفاية هذه الموارد لتنفيذ الطلب الأقدم في الرتل، بدلاً من الانتظار كما كان يحصل في الخوارزمية السابقة، حيث نجحت في التقليل من زمن التنفيذ ولكنها لم تتطرق لمعايير أخرى.

بالإضافة إلى الدراسة التي اقترحت تخصيص الموارد لكل طلب من طلبات المستخدمين بالتناوب لمدة زمنية محددة بدون أولوية لأي منها باستخدام خوارزمية Round Robin[٣]، تميزت هذه الخوارزمية بسهولة التطبيق، لكن على حساب ارتفاع زمن التنفيذ الكلي نسبياً عن الخوارزميات السابقة.

وهناك دراسة أخرى تطرقت لحل المشكلة عن طريق تخصيص الموارد بالترتيب بدءاً من المهمّة التي تتطلب أقل قدر من الموارد نحو المهمّة التي تتطلب أكبر قدر منها باستخدام خوارزمية Shortest Job First [4]، نجحت هذه الدراسة في تقليل الكلفة، ولكن ارتفع زمن التأخير في بعض الحالات.

وتطرقت دراسة أخرى لحل يقوم على بناء نموذج يقوم بتوقّع المدة الزمنية التي يستغرقها تنفيذ المهام الموجودة في رتل الانتظار، ثم القيام بتخصيص المهمّة التي من المتوقع أن تستغرق أقل مدة زمنية بموارد الآلة الافتراضية ذات معدل التنفيذ الأسرع وفق خوارزمية (Min–Min)[5]، نجحت هذه الدراسة في تخفيض الكلفة، لكن ظهرت مشكلة التحميل الزائد (Overloading) نظراً لاختيار بعض الآلات الافتراضية بشكل متكرر لتنفيذ عدد كبير من المهام.

ثم ظهر اتجاه جديد في هذا المجال تمثّل في استخدام خوارزميات التحسين المستوحاة من الطبيعة وهي مجموعة من الخوارزميات المستنبطة من الظواهر الطبيعية مثل التطور الطبيعي، والأنظمة البيولوجية والفيزيائية والكيميائية، وغيرها، والتي

شكلت نقلة نوعية في السنوات الأخيرة ، و تم تطبيقها على نطاق واسع في حل المشكلات الرياضية ضمن مختلف المجالات الهندسية بتكلفة حسابية بسيطة مقارنة بالخوارزميات التقليدية [6,7].

حيث تم استخدام خوارزمية الخفافيش Bat-algorithm [8] المستوحاة من آلية البحث بالصدى لدى الخفافيش والتي تعد من أسهل الخوارزميات تنفيذاً لقلة عدد البارامترات فيها، و أظهرت النتائج نجاحها في تقليل الكلفة بشكل فعّال، ولكنها لم تتمكن من تحسين زمن التنفيذ نظراً لبطء هذه الخوارزمية في التقارب إلى الحلول المثلى مقارنةً بالخوارزميات الذكية الأخرى.

كذلك تم استخدام خوارزمية (ABC) Artificial Bee Colony التي تحاكي السلوك الطبيعي لمستعمرات النحل في نمذجة مشكلة توزيع الموارد و البحث عن الحل الأفضل بين مجموعة كبيرة من الحلول الممكنة بالاعتماد على حجم المهمة المطلوبة من قبل المستخدم، و أقرب مسافة تفصله عن أحد المخدّمات ضمن المنظومة السحابية[9]، أظهرت نتائج هذه الدراسة تحسيناً كبيراً في زمن التنفيذ، لكنها لم تنجح في إيجاد الحلول الأفضل في بعض الحالات نظراً لسهولة وقوعها في مشكلة الأمثلة المحلية(Local Optima)[10].

2. أهمية البحث و أهدافه:

يعتبر مجال توزيع الموارد حجر الأساس ضمن طبقة البنية التحتية للأنظمة السحابية و من أهم التحديات التي لم يتم إيجاد حل أمثلي لها حتى الآن، بالتالي إيجاد تحسين في هذا المجال سينعكس على البنية السحابية بشكل كامل.

يهدف هذا البحث إلى تصميم منهجية مطورة ترتكز على مفاهيم المنطق الضبابي، و خوارزمية ذكاء الأسراب التي تسمح باتخاذ قرار سريع في عملية الجدولة دون مسح كامل الحلول كما في الخوارزميات التقليدية، وذلك لتحقيق جملة من الأهداف المتمثلة بالآتي:

- ✓ تحقيق توزيع عادل للموارد مما يرفع من مستوى رضى مستخدمي النظام.
- ✓ حل مشكلة التعارض بين رغبة المستخدمين في تقليل الكلفة المادية، ورغبة مزودي الخدمات السحابية في تحقيق أعلى ربح مادي من خلال تقليل زمن إتمام تنفيذ عملية التوزيع، مما يرفع مستوى الإنتاجية و يسمح للمزودين بإعادة استخدام الموارد التي تم تحريرها في تنفيذ مهام جديدة.
- ✓ التحقق من التحسينات المقترحة من خلال مقارنة قيم بارامترات الأداء مع الدراسات السابقة.

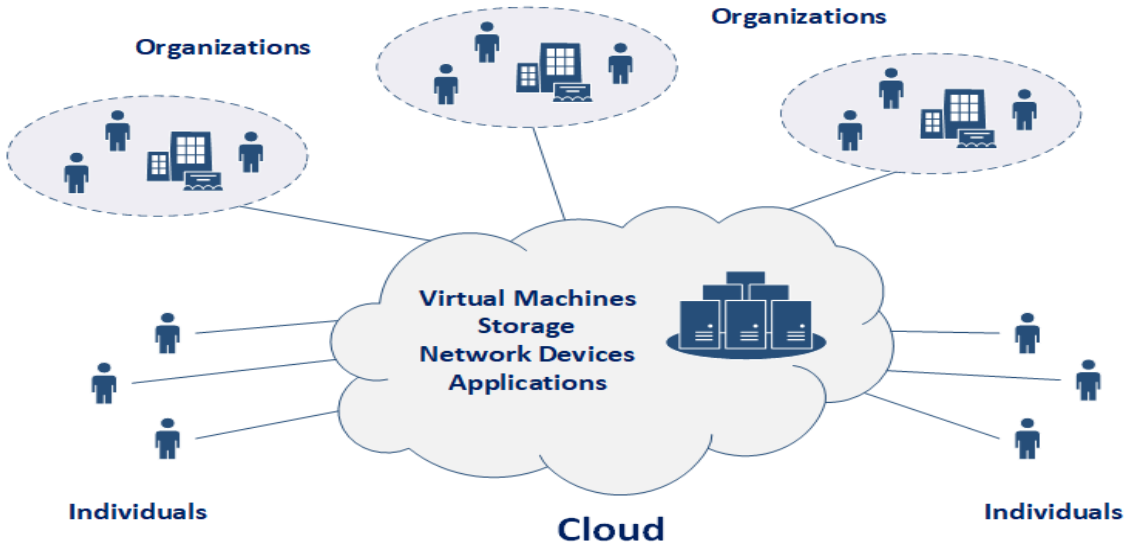
3. طرائق البحث ومواده:

يرتكز البحث على مفهومين أساسيين هما المنطق الضبابي Fuzzy Logic، وخوارزمية ذكاء الأسراب PSO، حيث تم في المرحلة الأولى نمذجة النظام الضبابي الذي يمكن من إسناد الأولويات لمهام المستخدمين وفق معايير محددة باستخدام بيئة ماتلاب Matlab، ثم تصميم خوارزمية توزيع الموارد على هذه المهام باستخدام لغة البرمجة JAVA ضمن بيئة عمل Netbeans، وتم التنفيذ باستخدام حاسوب شخصي يمتلك وحدة معالجة مركزية Intel Core I5-3337U,1.8GHz، وذاكرة 4GB.

3-1 تقنية المحاكاة الافتراضية Virtualization:

تستخدم بيئة الحوسبة السحابية مفهوم المحاكاة الافتراضية التي تنطوي على تقسيم منطقي للموارد الموجودة ضمن المخدمّات (الذاكرة، المعالجة) إلى عدد كبير من الآلات الافتراضية Virtual Machines من خلال برنامج

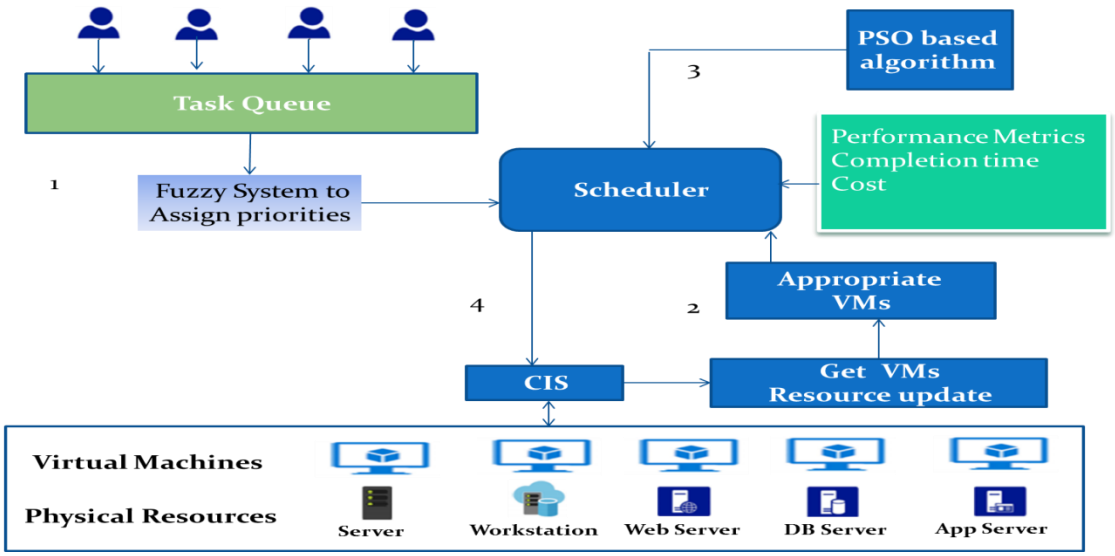
وسيط Hypervisor كمرحلة أولى، ثم يتم تخصيص موارد هذه الآلات الافتراضية لمهام المستخدمين بهدف تحقيق استثمار أفضل للموارد بالمقارنة مع الطريقة التقليدية وذلك كما هو مبين في الشكل(1).



الشكل(1) توضيح مفهوم الآلات الافتراضية في الحوسبة السحابية

3-2 المنهجية المقترحة:

للعمل على تلبية مجموعة من أهم متطلبات مزودي ومستخدمي أنظمة الحوسبة السحابية على حد سواء، المتمثلة بوجود معايير عادلة لتحديد أولوية تخصيص مهام المستخدمين بالموارد في النظام السحابي، وتوزيع الموارد السحابية اللازمة لتنفيذ مهام المستخدمين بشكل يقلل من الكلفة و زمن التنفيذ الكلي، تم اقتراح المنهجية المبينة في الشكل(٢):



الشكل(2): المخطط العام للمنهجية المقترحة في هذا البحث

كما هو موضح في الشكل (٢) تشتمل المنهجية السابقة على أربع مراحل رئيسية:

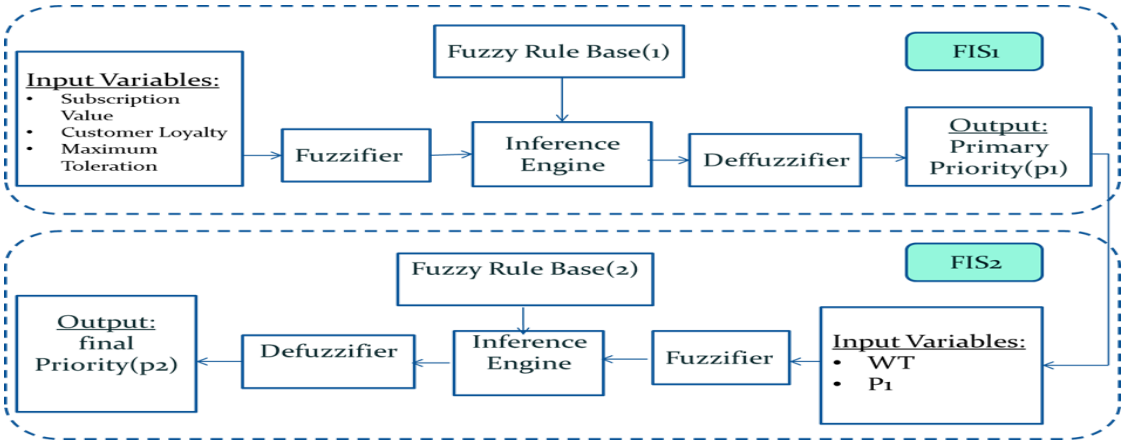
1-إسناد الأولويات لمهام المستخدمين الواردة إلى النظام.

2-تنفيذ استعلام لوحدة نظام المعلومات السحابي Cloud Information System (CIS) حول الموارد الموجودة في كل آلة افتراضية، ثم تحديد مجموعة جزئية منها تتضمن موارد كافية لمعالجة مهام المستخدمين(Appropriate VMs).

3-استخدام خوارزمية التوزيع المستندة على ذكاء الأسراب PSO بهدف توزيع موارد الآلات الافتراضية التي تم تحديدها في المرحلة السابقة على مهام المستخدمين بالاستناد على بارامترات الأداء (زمن التنفيذ الإجمالي Completion Time،الكلفة Cost).

4-إعلام وحدة CIS بنتيجة التوزيع تمهيداً لإعادة تنفيذ المنهجية من أجل مجموعة جديدة من المهام الواردة.

3-2-1 إسناد الأولويات لمهام المستخدمين:



الشكل(3) بنية النظام الضبابي المقترح لإسناد الأولويات

نظراً للعدد الكبير من المهام التي يتعامل معها النظام السحابي فإنّ الاعتماد على الآليات التقليدية التي تقوم على مبدأ تخصيص الموارد للمهام حسب معيار زمن الوصول فقط قد يسبب انتظاراً كبيراً نسبيّاً للبعض منها ويؤدي إلى خسارة مزودي الخدمة السحابية للعديد من المستخدمين.

بالتالي برزت الحاجة الفعلية إلى آلية جديدة ترفع من مستوى رضى المستخدمين، وهذا ما يتم في هذه المرحلة من خلال النظام الضبابي المقترح الذي يمكّن من تحديد أولويات لمهام المستخدمين لجهة تخصيصها بالموارد اللازمة للتنفيذ وفق بعض المعايير القياسية في هذا المجال[11].

كما هو موضح بالشكل (٣) فإن عملية إسناد الأولويات تمر بالمراحل التالية:

١-يتعامل النظام المقترح مع ثلاثة بارامترات دخل هي (قيمة الاشتراك Subscription Value، ولاء المستخدم Customer Loyalty، التسامح الأعظمي Maximum Toleration).

٢-تكون هذه البارمترات بصيغة أرقام صحيحة (Crisp) فتخضع لعملية التضبيب Fuzzification من خلال المكون (Fuzzifier) لتتحول إلى قيمة توافق مقدار انتمائها إلى مجموعة ضبابية معينة محددة وفق توابع عضوية (Membership Functions).

٣-بعدها تحصل عملية الاستدلال Inference في المكون (Inference Engine) و التي تستند على مجموعة من القواعد الضبابية الموجودة في المكون (Fuzzy Rule Base).

٤-تتم عملية فك التضبيب DeFuzzification والتي يتم فيها تحويل القيمة الضبابية (Fuzzy) الناتجة في المرحلة السابقة إلى صحيحة من خلال المكون (DeFuzzifier).

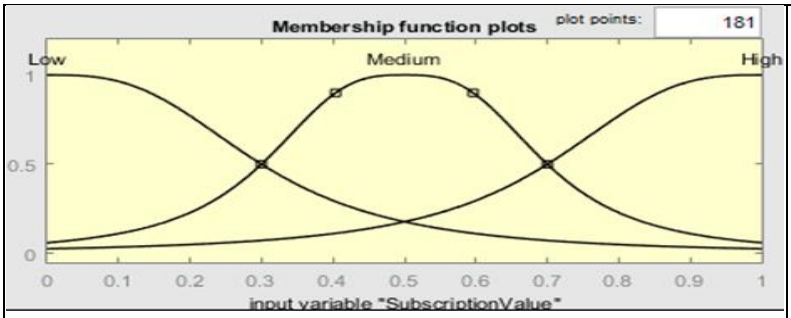
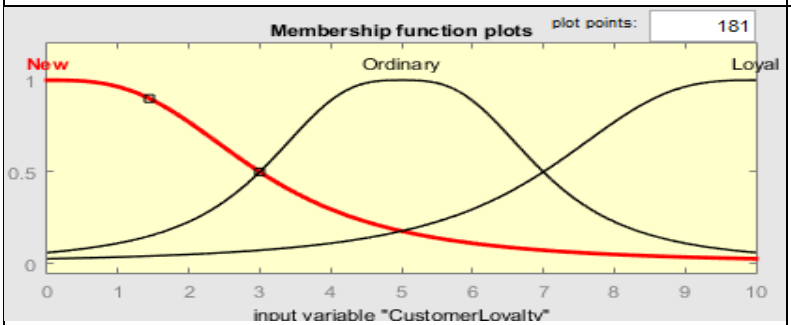
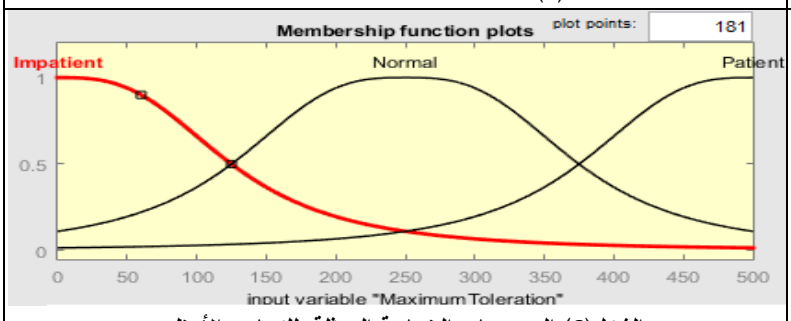
٥-بنتيجة المرحلة الأولى (FIS_1) نحصل على قيمة الأولوية بدائية (Primary Priority).

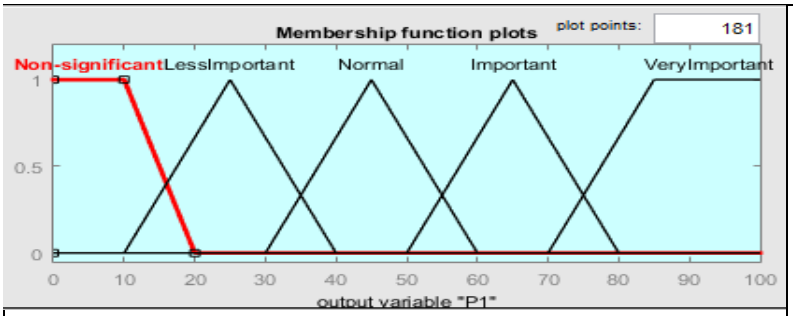
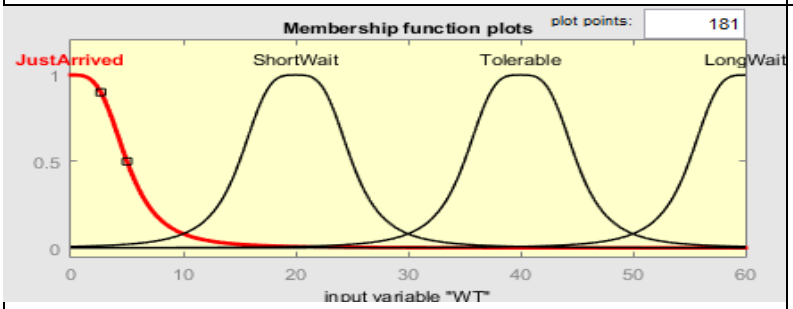
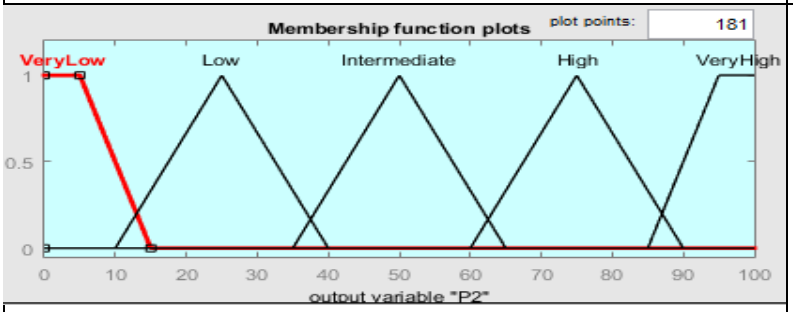
ثم يتم في المرحلة الثانية (FIS_2) بآلية مشابهة رفع مستوى الأولويات المسندة للمهام بشكل ديناميكي كلما ارتفع زمن انتظار المهام في النظام Waiting Time بهدف تجنب ظاهرة عادةً ما تواجه الأنظمة المعتمدة على الأولويات وتعرف باسم المجاعة (Starvation) وتعني عدم تخصيص بعض المهام بموارد في حال استمرار ورود مهام أخرى ذات أولوية أعلى منها بشكل مستمر إلى النظام.

3-2-1-1 تحديد توابع العضوية Membership Functions:

باستخدام الأداة Fuzzy Logic Designer ضمن برنامج Matlab يتم تعريف توابع العضوية(Membership Functions) التي تحدد مقدار انتماء كل نقطة ضمن فضاء القيم إلى عدد من المجموعات الضبابية (Fuzzy Sets) بقيمة بين (0و1)، حيث نستخدم توابع عضوية من النوع (generalized bell function) للتعبير عن قيم متحولات الدخل و توابع عضوية من النوع (triangular, trapezoidal fuzzy membership function) للتعبير عن متحولات الخرج وذلك كما هو مبين في الجدول (١).

الجدول(1):توابع العضوية والمجموعات الضبابية المستخدمة في النظام

1-قيمة الاشتراك(SV): حيث يقدم معظم مزودي الخدمات السحابية عدة أنماط (Low,Medium,High) من نماذج الخدمات بالاعتماد على الاشتراك المدفوع.	 الشكل(4) المجموعات الضبابية الممثلة لقيمة الاشتراك
2-ولاء الزبون (CL): لها ثلاث مستويات (New,Ordinary,Loyal) تحدد وفقاً لعدد المرات التي استخدم فيها العميل النظام السحابي والتي يتم تسجيلها في وحدة CIS.	 الشكل(5) المجموعات الضبابية الممثلة لولاء الزبون
3-التسامح الأعظمي (MT): بالاعتماد على طبيعة مستخدم النظام يتم تحديد ثلاث مستويات (Impatient,Normal,Patient).	 الشكل(6) المجموعات الضبابية الممثلة للتسامح الأعظمي

4-الأولوية البدائية (P1): تتحدد قيمتها في نهاية المرحلة الأولى من النظام الضبابي بالاعتماد على بارامترات الدخل الثلاث السابقة.	 **الشكل(7) المجموعات الضبابية الممثلة للأولوية البدائية**
5-زمن الانتظار (WT): تمثل قيمة انتظار المهمة في النظام بانتظار تخصيصها بالموارد اللازمة لتنفيذها ولها أربع مستويات (JustArrived,ShortWait,Tolerable,LongWait)	 **الشكل(8) المجموعات الضبابية الممثلة لزمن الانتظار**
6-الأولوية النهائية (P2): التي تتحدد قيمتها في نهاية المرحلة الثانية من النظام الضبابي نتيجةً لتطبيق القواعد الضبابية(Fuzzy Rules).	 **الشكل(9) المجموعات الضبابية الممثلة للأولوية النهائية**

3-2-1-2 تحديد القواعد الضبابية Fuzzy Rules:

وهي مجموعة من القواعد الشرطية (IF-THEN) تحدد بناءً على خبرة مصمّم النظام لتمكن النظام الضبابي من استنتاج قيمة الخرج و تأخذ الصيغة التالية:

IF x is A THEN y is B

حيث أن: x تمثل مجموعة متحولات الدخل، بينما y تمثل متحول الخرج و A,B: تمثل قيم هذه المتحولات التي يتم تحديدها من خلال المجموعات الضبابية.

يتم استخراج قيمة الخرج في الأنظمة الضبابية نتيجة لتفعيل للقاعدة الضبابية المناسبة بناءً على قيمة متحولات الدخل مما يمكن النظام الضبابي من إعطاء النتيجة النهائية حسب المعامل المنطقي (OR,AND) في كل منها ويظهر الجدول (٢) عينة من القواعد التي تم تعيينها.

الجدول(٢):أمثلة عن القواعد الضبابية المستخدمة

ID	Rule
1	IF Primary Priority is Less_Important AND Waiting Time is Just_Arrived THEN Final Priority is Very_Low.
2	IF Primary Priority is Very_Important AND Waiting Time is Long_Wait THEN Final Priority is Very_High.
3	IF Primary Priority is Less_Important AND Waiting Time is Long_Wait THEN Final Priority is .Intermediate

3-2-2 توزيع الموارد على مهام المستخدمين:

يتم في المرحلة الثانية تخصيص المهام التي جرى إسناد الأولويات لها في المرحلة السابقة بموارد الآلات الافتراضية التي تم إنشاؤها في النظام السحابي، وحيث أن فضاء البحث عن الحلول الممكنة في هذه الحالة كبير جداً، كما أن هذه الحلول يجب أن تحقق مجموعة من القيود(constraints) بحيث تضمن مستوى مناسب من الخدمة لعملاء الأنظمة السحابية، فقد تم الاستناد على خوارزمية ذكاء الأسراب (PSO) وهي أحد أهم الخوارزميات الذكية التي أثبتت فعالية كبيرة في مسائل التحسين الرياضية من النوع NP-Complete التي تندرج مشكلة البحث ضمنه[12].

3-2-2-1 آلية العمل في خوارزمية PSO:

- تحاكي هذه الخوارزمية في آلية عملها السلوك الاجتماعي لأسراب الحيوانات التي تعيش في مستعمرات كبيرة (الطيور، الأسماك)، حيث يفترض النموذج الرياضي لهذه الخوارزمية أن الموقع الحالي لكل فرد من أفراد السرب يمثل حلاً لمشكلة التحسين، ويتم التواصل بين هذه الأفراد للعمل على تعديل مواقعها في مجال البحث بما يحسن من الحلول المقترحة، حتى الوصول إلى الحل الأمثل (أفضل موقع) والموافق للقيمة الأفضل لتابع هدف يسمى تابع اللياقة(Fitness function)، وهو عبارة عن تابع رياضي يحدد مدى ملائمة الحل المقترح لمشكلة التحسين.
- يرتبط كل فرد ببارامتر السرعة (Velocity)، يحدد كيفية حركته ضمن فضاء البحث بالاعتماد على المعلومات التي يتبادلها مع باقي عناصر السرب.

3-2-2-2 النموذج الرياضي للخوارزمية:

خلال كل تكرار للخوارزمية يحتفظ كل فرد بالموقع الأفضل الذي قام بزيارته p_i^{best} (الحل الأفضل)، والموافق لأعلى قيمة لتابع اللياقة، ويقوم بمشاركة هذا الحل مع باقي أفراد السرب ليتم تحديد الحل الأفضل العامp^{gbest}، و من ثم تعديل مواقع هذه الأفراد بناءً عليه، وذلك من خلال تعديل قيمة بارامتر السرعة v_i^{t+1} وفقاً للصيغة التالية:

$$v_i^{t+1}=\omega\, v_i^t+c_1r_1(p_i^{best}-p_i^t)+c_2r_2(p^{gbest}-p_i^t) \quad (١)$$

حيث أن:

(v_i^{t+1}): تمثل القيمة الحالية لسرعة الفرد (في التكرار رقم t+1 للخوارزمية).

(v_i^t): تمثل القيمة السابقة لسرعة الفرد (في التكرار رقم t للخوارزمية).

(p_i^{best}): الموقع الأفضل الذي وجد حتى اللحظة للفرد i.

(p^{gbest}): الموقع الأفضل العام الذي تم إيجاده بين كل الأفراد.

(p_i^t): يمثل موقع الفرد i-th في التكرار t.

L: عدد الأفراد . C2،C1: ثوابت للتحكم بتسارع الفرد. r1,r2: أرقام عشوائية في المجال بين ال [1,0].

ω: عامل القصور الذاتي وهو ثابت يستخدم للتحكم بسرعة البحث في الخوارزمية.

كما نلاحظ من الصيغة السابقة فإن القيمة الحالية للسرعة (v_i^{t+1}) يتم حسابها بإضافة مكونين إلى القيمة السابقة للسرعة (v_i^t).

المكون الأول يساوي الفرق بين الموقع الحالي للفرد(p_i^t)، والموقع الأفضل الذي سبق لهذا الفرد أن وجده(p_i^{best}) ، ويمثل هذا المكون التأثير الشخصي(Personal influence).

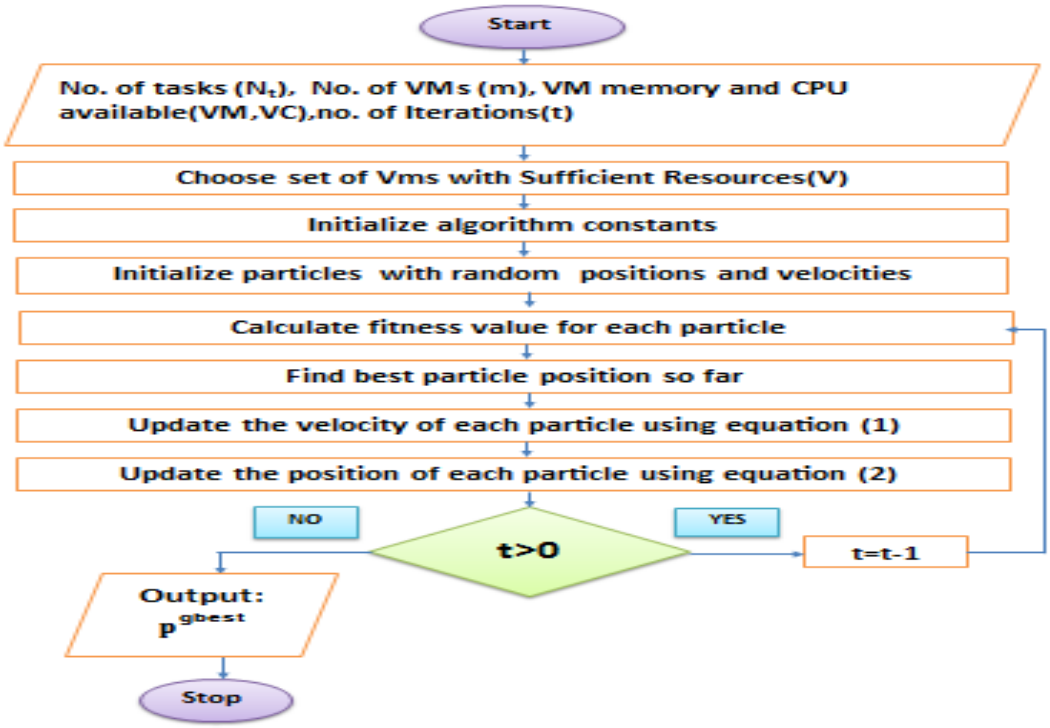
المكون الثاني يساوي الفرق بين الموقع الحالي للفرد (p_i^t)، والموقع الأفضل على مستوى السرب بأكمله (p^{gbest}) ، ويمثل هذا المكون التأثير الاجتماعي(Social influence).

باستخدام قيمة السرعة (v_i^{t+1}) التي يتم حسابها وفقا للعلاقة (١) يتحدد الموقع الجديد للفرد i–th في التكرار t+1 للخوارزمية (p_i^{t+1}) وفقاً للمعادلة (٢):

$$p_i^{t+1} = p_i^t + v_i^{t+1} \qquad (٢)$$

3-2-2-3 تطبيق خوارزمية التحسين:

يبين الشكل (١٠) خطوات تطبيق خوارزمية التحسين على مشكلة البحث:



الشكل(10) خطوات الخوارزمية المقترحة

الخطوة الأولى:

تحديد عدد المهام (N_t)، وعدد الآلات الافتراضية الكلي(m)، كذلك موارد الذاكرة (VM) والمعالجة المتاحة (VC)، بالإضافة إلى عدد التكرارات لخطوات الخوارزمية(t).

الخطوة الثانية:

عملية إنشاء الحلول البدائية في خوارزمية (PSO) تتصف بالعشوائية، مما يؤخر من تقارب الخوارزمية إلى الحل الأمثل نسبياً، ويجعل من الحلول النهائية بعيدة عن بعضها (الانحراف المعياري كبير).

لذلك تم اقتراح اختصار فضاء البحث الأولي بما ينعكس على تحسين أداء الخوارزمية بالمجمل من حيث تقليل العبء الحسابي و زمن المعالجة وذلك من خلال الاستبعاد الآني لبعض الآلات الافتراضية التي تمتلك موارد قليلة وقد تكون مواردها مخصصّة لمهام أخرى في لحظة التوزيع وفق الطريقة التالية:

بفرض أنّ T تمثل مجموعة مهام المستخدمين الواردة إلى النظام السحابي و التي عددها N_t خلال الفترة الزمنية D_t، و PV تمثل مجموعة الآلات الافتراضية التي عددها m والمتاحة في نفس الزمن ليتم تخصيص مواردها لمهام المستخدمين:

$$T = \{T_1, T_2, T_3 \ldots\ldots\ldots\ldots T_{N_t}\} \text{ , } PV = \{PV_1, PV_2, PV_3 \ldots\ldots\ldots\ldots\ldots\ldots PV_m\}$$

نقوم بحساب قيمة العتبة (TH) التي تمثل القيمة المتوسطة لموارد المعالجة والذاكرة وفق العلاقة التالية:

$$TH_C = \frac{\sum_{j=1}^{m} VC_j}{m} \text{ , } TH_M = \frac{\sum_{j=1}^{m} VM_j}{m} \quad (٣)$$

حيث أن (VM_j) موارد الذاكرة المتاحة و (VC_j) موارد المعالجة المتاحة في كل آلة افتراضية.

بعدها يتم مقارنة موارد كل آلة افتراضية مع قيمة العتبة واستبعاد الآلات الافتراضية التي تمتلك مواردا أقل من الحد المتوسط، وبالتالي تتشكل لدينا مجموعة جديدة (V) تتضمن عدداً N_v من الآلات الافتراضية:

$$V = \{V_1, V_2, V_3 \ldots\ldots\ldots V_{N_v}\}$$

نتابع تنفيذ ما تبقى من خطوات الخوارزمية باستخدام عناصر هذه المجموعة التي تحتوي مواردا كافية لمعالجة مهام المستخدمين.

الخطوة الثالثة:

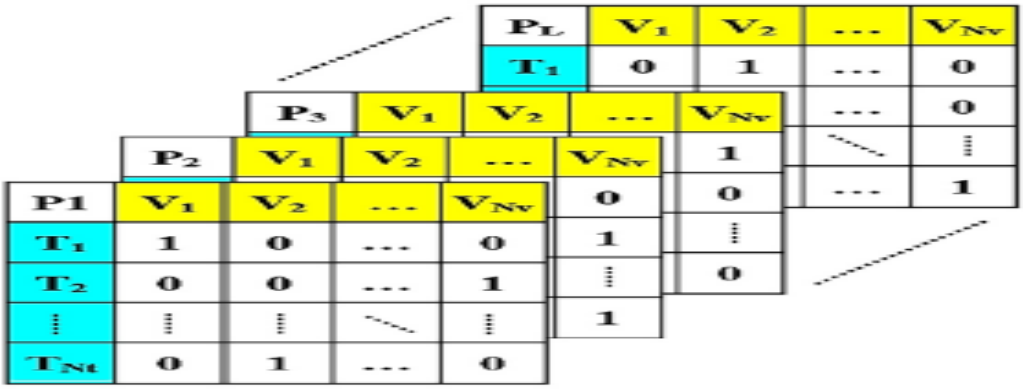
تهيئة قيم ثوابت الخوارزمية $(r_2, r_1, c_2, c_1, \omega)$.

الخطوة الرابعة:

تهيئة الأفراد (الحلول الأولية) بقيم عشوائية لبارامترات السرعة والموقع.

في مشكلة البحث يمثل موقع فرد ما الطريقة التي يتم بها توزيع المهام على الآلات الافتراضية وفق هذا الفرد في كل تكرار للخوارزمية، بينما تمثل السرعة مقدار التغير الذي يجب أن يطرأ على موقع الفرد في كل تكرار للخوارزمية.

نمثّل الموقع الحالي لكل فرد الذي يمثل حلاً لمشكلة التوزيع رياضياً على شكل مصفوفة $(N_t \times N_v)$، حيث أن الأسطر تمثل المهام والأعمدة تمثل الآلات الافتراضية، بالتالي السرب يمثل بمصفوفة ثلاثية الأبعاد $(N_t \times N_v \times L)$، حيث على سبيل المثال وكما هو مبين في الشكل (11) في لحظة معينة كان الحل وفقاً للفرد الأول (P_1) يقتضي أن يتم تخصيص المهمة الأولى (T_1) بموارد الآلة الافتراضية الأولى (V_1)، والمهمة الثانية (T_2) بموارد الآلة الافتراضية رقم N_v (V_{Nv}).



الشكل(11) يبين التمثيل الرياضي للحلول وفق الخوارزمية المقترحة

الخطوة الخامسة :

يتم تقييم مدى ملائمة كل حل مقترح لمشكلة التحسين من خلال مفهوم تابع اللياقة (Fitness Function) الذي يتم تحديده كما يلي:

الزبون يرغب في إنجاز عمله بأقل كلفة ممكنة، بينما يطمح مزودو الخدمات السحابية إلى إنجاز أكبر عدد ممكن من مهام المستخدمين بأقل زمن تنفيذ كلي ممكن.

بفرض أن N_t هو عدد المهام (الطلبات) و N_v هو عدد الآلات الافتراضية(الموارد)، حيث يجب توزيع هذه الموارد إلى المهام حسب متطلباتها تكون الصيغة الرياضية لتابع اللياقة بالشكل التالي:

$$Fitness\ function\ f(Nv) = \alpha * CT + \beta * C$$

$$\alpha + \beta = 1$$

$\boldsymbol{CT}$: زمن إتمام التنفيذ المتوقع للمهام N_t على الموارد N_v (الآلات الافتراضية).

$\boldsymbol{C}$: الكلفة المتوقعة لتنفيذ المهام N_t على الموارد N_v (الآلات الافتراضية).

بينما (α,β) ثوابت تعبر عن أوزان بحيث أن $(0< \alpha <1)$ و $(0< \beta <1)$.

تتم مقارنة قيم تابع اللياقة لكل حل من الحلول المقترحة، ليتم بعدها تحديد الحلول الأفضل على مستوى الفرد، ومن ثم على مستوى السرب ككل.

الخطوة السادسة:

تحديث قيم السرعة من خلال المعادلة (١).

الخطوة السابعة:

تحديث موقع كل فرد حسب المعادلة (٢).

نستمر في تكرار خطوات الخوارزمية من الخطوة الخامسة حتى انتهاء عدد التكرارات، وعندها يتم تحديد الحل الأفضل الذي تم الوصول إليه.

3-2-3 مقاييس الأداء Performance Metrics:

A) زمن اتمام التنفيذ Completion Time:

الزمن الذي يوافق اتمام عملية توزيع الموارد، ويتم حسابه وفق العلاقة التالية:

$$\boldsymbol{CT = \sum CT_{i,j}} \qquad \boldsymbol{\forall i = \{1, 2, \dots, N_t\}, j = \{1, 2, \dots, N_v\}}$$

(4)

$$CTi,j = WT_i + \delta_j(Task_i) \text{ Where } \delta_j(Task_i) = \frac{L(task_i)}{V_j}$$

$CT_{i,j}$:الزمن الكلي اللازم لاتمام تنفيذ المهمة i على الآلة الافتراضية j.

WT_i :زمن الانتظار في الرتل للمهمة i قبل أن تتم عملية تخصيص الموارد.

$\delta_j(Task_i)$:زمن التنفيذ للمهمة iوفق الموارد المخصصة من قبل الآلة الافتراضية j.

$L(task_i)$: حجم المهمة (Task Length) مقاساً بملايين التعليمات البرمجية (MI).

V_j : معدل تنفيذ (سرعة) الآلة الافتراضية مقاساً بملايين التعليمات البرمجية في الثانية(MIPS).

N_t :عدد مهام المستخدمين.

N_v :عدد الآلات الافتراضية(الموارد).

B) الكلفة Cost:

المبلغ الكمي الذي سوف يدفعه المستخدم لمزود خدمة الحوسبة السحابية لقاء استخدامه لموارد الحوسبة السحابية وتعطى وفق العلاقة التالية .

$$C = \sum_{i=1}^{N_t} \sum_{j=1}^{N_v} c_{i,j} * CT_{i,j} \quad (5)$$

$c_{i,j}$: كلفة الموارد التي يتم تخصيصها من الآلة الافتراضية j لتنفيذ المهمة i المسندة إليها.

$CT_{i,j}$: الوقت الذي يستغرقه تخصيص المهمة i بموارد الآلة الافتراضية j.

4. النتائج والمناقشة:

تم القيام بعدة تجارب من أجل تقييم مراحل عمل المنهجية المقترحة حيث تم في البداية اختبار نظام إسناد الأولويات الضبابي باستخدام بيئة Matlab من خلال 15 مهمة ذات قيم بارمترات متباينة، وكانت النتائج بعد تطبيق المرحلة الأولى كما هو مبين في الجدول التالي(٣).

الجدول (٣):الأولويات المسندة لمهام المستخدمين بنتيجة المرحلة الأولى

ID	Subscription Value(SV)	Customer Loyalty(CL)	Maximum Toleration(MT)	P1(%)
1	0.1	5	250	47.2
2	0.4	5	250	47.6
3	0.7	5	250	55.7
4	0.8	5	250	59.7
5	0.9	5	250	61.6
6	0.5	1	250	47.2
7	0.5	3	250	47.6
8	0.5	6	250	52.3
9	0.5	7	250	55.7
10	0.5	9	250	61.6
11	0.5	5	100	56.5
12	0.5	5	200	50.6
13	0.5	5	350	48.4
14	0.5	5	450	48.2
15	0.5	5	500	45.9

نلاحظ ارتفاع قيمة الأولوية عند ارتفاع قيمة اشتراك المستخدم (SV) و ولائه لمزود الخدمة السحابية (CL) كذلك انخفاض قيمة الأولوية كلما زادت قيمة البارامتر (MT) أي كلما كان النظام أكثر تسامحاً مع التأخير الزمني.

لتقييم عمل المرحلة الثانية نلاحظ تغير أولويات بعض مهام المستخدمين ذات الأولوية الدنيا بنتيجة المرحلة الأولى كما هو مبين في الجدول(4)، حيث نلاحظ أن النظام يزيد من مستوى أولوية هذه المهام كلما زاد زمن الانتظار(WT) لها بالتالي يمتلك قدرة أكبر على تجنب حدوث ظاهرة المجاعة Starvation كما ذكرنا سابقاً.

الجدول (٤):الأولويات المسندة لمهام المستخدمين بنتيجة المرحلة الثانية

Task ID	SV	CL	MT	P1(%)	WT	P2(%)
16	0.1	1	450	20.1	15	24.8
					30	37.3
					45	49.8
					60	50
17	0.4	4	250	44	15	48.8
					30	50
					45	51.2
					60	74.6
18	0.3	3	400	32.9	15	33.7
					30	37.8
					45	51
					60	58

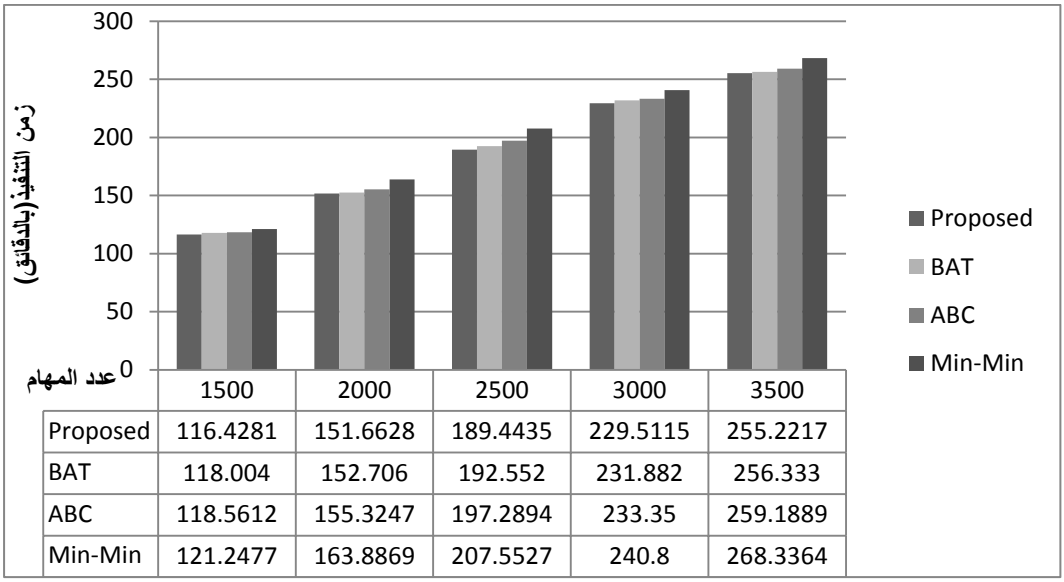
تم مقارنة أداء المنهجية المقترحة من حيث زمن التنفيذ والكلفة المتوقعة باستخدام بيئة NetBeans مع عدد من المنهجيات السابقة التي اعتمدت على الخوارزميات التالية (ABC,BAT,Min–Min) وذلك باستخدام عدد من الآلات الافتراضية ذات خصائص مختلفة من حيث موارد الذاكرة (Memory) والتخزين (Storage) وسرعة المعالجة(Processing Speed)، تقابلها عدد من مهام المستخدمين المتباينة أيضاً وباستخدام بارامترات الخوارزمية المقترحة الموضّحة كما هو مبين في الجدول الآتي:

الجدول (٥):ضبط الإعدادات المستخدمة في عملية المحاكاة

Proposed algorithm parameters		**VM parameters**		**Tasks parameters**	
Number of Particles	50–500	**Number of VMs**	500	**Number of Tasks**	1000–3500
Iterations	100–1000	**Processing Speed (MIPS)**	500–1000	**Task's Length(MI)**	100000–300000
C_1	0.5–2.5				
C_2	0.5–2.5	**Storage(GB)**	10–100		
ω	0.3–0.9	**RAM(GB)**	1–8		

١– معيار زمن إتمام التنفيذ:

كانت نتائج المقارنة كما في الشكل التالي (12):



عدد المهام	1500	2000	2500	3000	3500
Proposed	116.4281	151.6628	189.4435	229.5115	255.2217
BAT	118.004	152.706	192.552	231.882	256.333
ABC	118.5612	155.3247	197.2894	233.35	259.1889
Min-Min	121.2477	163.8869	207.5527	240.8	268.3364

الشكل(12) مخطط بياني يوضح نتائج المقارنة بين المنهجية المقترحة والمنهجيات السابقة من حيث معيار زمن إتمام التنفيذ

من النتائج المبينة في الشكل (١٢) نلاحظ أن تطبيق المنهجية المقترحة قد قلل من الزمن اللازم لإتمام عملية توزيع الموارد في مختلف السيناريوهات التي توافق تغير عدد مهام المستخدمين للنظام السحابي.

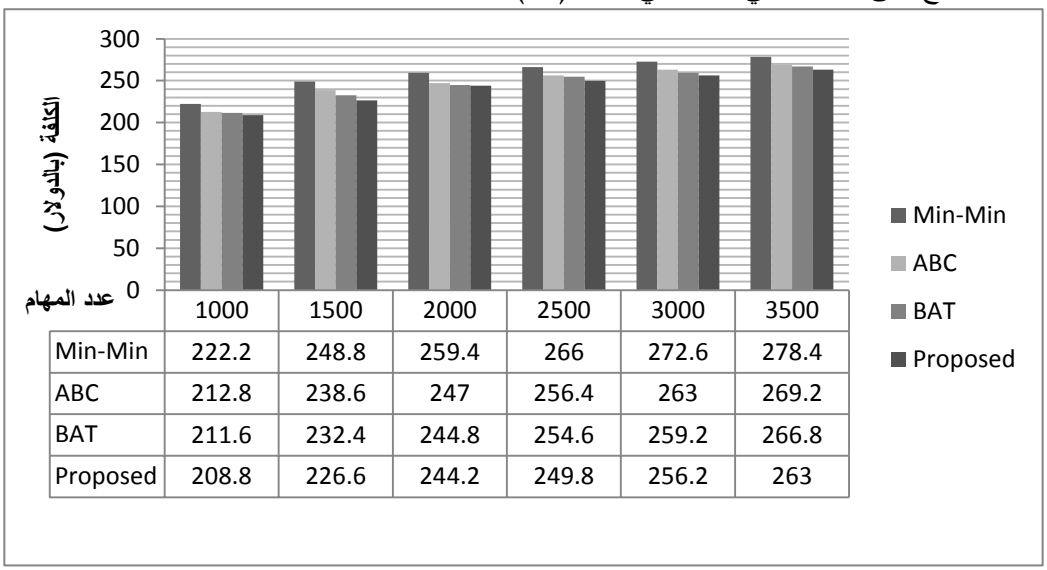
٢- معيار الكلفة:

تمت المقارنة وفقاً لنموذجي تسعير معتمدين لخدمات الحوسبة السحابية [13] مبينين بالجدول التالي(٦):

الجدول(٦):نماذج التسعير المعتمدة في عملية المقارنة

CPU Speed(MIPS)	Memory(GB)	Price in Hour($)	Category
500-750	1-4	0.40	Small
751-1000	5-8	0.60	Medium

وكانت النتائج على النحو التالي المبين في الشكل(13) :



عدد المهام	1000	1500	2000	2500	3000	3500
Min-Min	222.2	248.8	259.4	266	272.6	278.4
ABC	212.8	238.6	247	256.4	263	269.2
BAT	211.6	232.4	244.8	254.6	259.2	266.8
Proposed	208.8	226.6	244.2	249.8	256.2	263

الشكل (13) مخطط بياني يوضح نتائج المقارنة بين المنهجية المقترحة والمنهجيات السابقة من حيث معيار الكلفة

نلاحظ أن الخوارزمية المقترحة قد قللت من كلفة عملية توزيع الموارد في مختلف السيناريوهات، وذلك يتوافق مع التخفيض الحاصل في زمن التنفيذ مما يدل على فعالية التحسين المقترح.

5. الاستنتاجات والتوصيات:

مما سبق نجد أن:

- ✓ أثبتت النتائج التي تم الحصول عليها بتطبيق المنهجية المقترحة قدرتها على تحقيق توزيع عادل لموارد النظام السحابي بالاستناد على تحديد الأولويات بشكل دقيق وفق معايير معتمدة، وذلك ينعكس إيجاباً على مستوى رضى مستخدمي النظام بشكل عام.
- ✓ أظهرت المنهجية المقترحة نتائج جيدة من حيث تقليل زمن التنفيذ والكلفة بالمقارنة مع المنهجيات المستخدمة سابقاً.
- ✓ أثبتت الخوارزمية المقترحة فعالية أكبر من الخوارزميات التقليدية في حل مشكلة توزيع الموارد.

التوصيات:

- دمج تقنيات تعلم الآلة Machine Learning مع المنهجية المقترحة بغية رفع قدرتها على اكتشاف فضاء الحلول، وتحقيق تقارب أسرع إلى الحل الأمثلي وفق عدد أكبر من بارامترات الأداء.
- تطوير مرحلة إسناد الأولويات المقترحة بحيث تشتمل على عدد أكبر من المعايير، بما يناسب طبيعة وصول المهام إلى النظام السحابي بشكل عشوائي [15][14].

6. المراجع:

[1]Marphatia, Aditya. (2013). " Optimization of FCFS Based Resource Provisioning Algorithm for Cloud Computing". IOSR Journal of Computer Engineering.

[2] Zuo L, Shu L, Dong S, Zhu C, Hara T.(2015) "Optimization of FCFS Based Resource Provisioning Algorithm for Cloud Computing". Access, IEEE.

[3] Pandaba Pradhan, Prafulla Ku. Behera, (2016) " Modified Round Robin Algorithm for Resource Allocation in Cloud Computing" . International Conference on Computational Modeling and Security.

[4] Chaturvedi, A., & Rashid, A. (2017). "Analysis of Resource Pooling and Resource Allocation Schemes in Cloud Computing". International Journal of Computer Trends and Technology.

[5] Ali, Syed & Alam, Mansaf. (2018). "Resource-Aware Min-Min (RAMM) Algorithm for Resource Allocation in Cloud Computing Environment". International Journal of Recent Technology and Engineering (IJRTE).

[6] Pooranian, Zahra & Shojafar, Mohammad & Abawajy, Jemal & Abraham, Ajith. (2015). "An efficient meta-heuristic algorithm for grid computing". Journal of Combinatorial Optimization.

[7] Ghribi, C.,(2014). "Energy efficient resource allocation in cloud computing environments, Doctoral dissertation", Institut National des Telecommunications.

[8] Adhikari M et al (2020) "Meta heuristic-based task deployment mechanism for load balancing in IaaS cloud". IOSR Journal of Computer Engineering .

[9] Muthulakshmi, B., Somasundaram, K. (2019) "A hybrid ABC-SA based optimized scheduling and resource allocation for cloud environment". International Journal of Computer Trends and Technology.

[10] Yang, Lei & Yang, Xiaohui & Wu, Yue & Liu, Xiaoping. (2018). "Applied Research on Distributed Generation Optimal Allocation Based on Improved Estimation of Distribution Algorithm". Energies.

[١١] Mahalakshmi, P. , & Ganesan, K. (2015). "Mamdani fuzzy rule based model to classify sites for aquaculture development". Indian Journal of Fisheries.

[١٢] S. Mousavi, A. Mosavi, A. R. Varkonyi-Koczy (2017), "Dynamic resource allocation in cloud computing ". Acta Polytechnica Hungarica.

[١٣] Mashayekhya L, Grosu D (2016) "An online mechanism for resource allocation and pricing in clouds". IEEE Trans Computing.

[١٤] Dayoub, Y. (2015). "Stochastic logical linguistic approach Multi-level automated object's dialogue control (MADC) ". Tishreen University Journal-Engineering Sciences Series.

[15] ديوب، يعرب. 2019، " التوجيه الديناميكي التنبؤي الملائم لحركة الكائن " .مجلة جامعة طرطوس للبحوث والدراسات العلمية في القطر العربي السوري، المجلد الثالث العدد السادس.