مجلة جامعة طرطوس للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (5) العدد(7) 2021

Tartous University Journal for Research and Scientific Studies - engineering Sciences Series Vol. (5) No. (7) 2021

تحسين أداء الموازنة الديناميكية للحمل في بيئة الحوسبة السحابية باستخدام خوارزمية أَمْثلة سرب العناصر

أ.د. رضوان صالح دندة*

د. قاسم قبلان**

م. حيان رجب***

(تاريخ الإيداع 19 /5 /2021 . قُبِل للنشر في 4 /8 /2021)

□ ملخّص □

تمثل جدولة عدد كبير من المهام غير المتجانسة مع المحافظة على مستويات عالية من موازنة الحمل بين مختلف الآلات الافتراضيةغير المتجانسة إحدى أهم التحديات الموجودة في بيئة الحوسبة السحابية لما لها من تأثير كبير على أداء النظام الكلي.

نقدم في هذا البحث خوارزمية الموازنة الديناميكية للحمل المدركة لعدم التجانس باستخدام أمثلة سرب العناصر (Heterogeneity aware Dynamic Load Balancing using Particle Swarm Optimization (HADLB–PSO) والتي تسعى لتلبية كل من متطلبات مزود الخدمة والمستخدمين السحابيين.

قمنا بإجراء تقييم تجريبي باستخدام أداة المحاكاة Cloudsim لاختبار فعالية الخورازمية المقدمة ومقارنة أدائها مع مجموعة من الأعمال السابقة المقدمة في هذا المجال والتي تشمل الخورازميات التالية: ILBMM,RALBA,IPSO,MCT–PSO وقد أظهرت النتائج بأن الخورازمية المقترحة تتفوق على هذه الخورازميات من حيث زمن التنفيذ الكلي للمهام (makespan)، معدل استخدام الموارد الوسطي، درجة عدم التوازن.

الكلمات المفتاحية: الحوسبة السحابية، أَمْثلة سرب العناصر، موازنة الحمل

---

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

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

***طالب دراسات عليا(دكتوراه)- قسم النظم والشبكات الحاسوبية-كلية الهندسة المعلوماتية-جامعة تشرين-اللاذقية-سورية.

مجلة جامعة طرطوس للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (5) العدد(7) 2021

Tartous University Journal for Research and Scientific Studies - engineering Sciences Series Vol. (5) No. (7) 2021

Enhancing the Performance of Dynamic Load Balancing in Cloud Environment Using Particle Swarm Optimization Algorithm

Prof.Dr.Radwan Saleh Dandah[*]
Dr.Kasem Kabalan[**]
Eng.Hayyan Rajab[***]

(Received 19 /5 / 2021 . Accepted 4 / 8 / 2021)

□ ABSTRACT □

Scheduling a vast number of heterogeneous tasks while maintaining high levels of load balancing among different heterogeneous virtual machines (VMs) is one of the biggest challenges that exists in cloud computing environmentbecause of its significant impact on the performance of the whole system.

In this research, we introduce Heterogeneity aware Dynamic Load Balancing using Particle Swarm Optimization (HADLB-PSO) Algorithm which aimto meetboth cloud users and providers' requirements.

We have performed experimental evaluation on Cloudsim toolkitto validate the effectiveness of the proposed algorithm and compare its performance against a collection of many previous works in this fieldwhich includesthe following algorithms: ILBMM,RALBA,IPSO,MCT-PSO. The simulation results showed that the suggested algorithm outperforms these algorithms regarding makespan, average resource utilization ratio, and Degree of imbalance.

Key words: Cloud Computing, Particle Swarm Optimization, Load Balancing.

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

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

[***] Postgraduate student (Phd),Department of Systems and Computer Networks, Faculty of Information Engineering, Tishreen University, Lattakia, Syria.

1-مقدمة:

ظهرت الحوسبة السحابية كحقل جديد في مجال تكنولوجيا المعلومات كنتيجة للتطور الكبير في تقنيات الاتصال والاستخدام المتزايد للانترنت.ويمكن تعريفها بأنها نموذج حوسبة يعتمد على الانترنت لمشاركة الموارد (الشبكات ، المخدمات، التخزين، التطبيقات) وتقديمها كخدمات حسب الطلب للمستخدمين [1].

يتألف مركز البيانات السحابي من عدد كبير من المخدمات، ويتم استخدام تقنية المحاكاة الافتراضية (virtualization) بهدف تشغيل أكثر من آلة افتراضية (Virtual Machine) على نفس المخدم، تحقق هذه القدرة على التحصيص (allocation) الاستثمار الأمثل للموارد الفيزيائية المتوفرة [2].

يمكن أن تقدم الحوسبة السحابية ثلاثة أنواع من الخدمات وهي:البنية التحتية كخدمة Infrastructure as aservice (IaaS) كمثال عليها Amazon EC2 و Nimbula. المنصة كخدمة Platform as a Service (PaaS) مثل(GAE) Google App Engine و Amazon Web Services (AWS).البرمجيات كخدمة Software as a Service (SaaS) مثل BigCommerce, Google Apps[3].

توجد ثلاثة كيانات تتعاون فيما بينها لتقديم البرمجيات كخدمة في السحابة وهي: المزود السحابي (cloud Provider)، مزود الخدمة (service provider)، والمستخدمين. يقوم المزود السحابي باستخدام تقنية المحاكاة الافتراضية ليقدم الموارد الحاسوبية لمزودي الخدمة على شكل آلات افتراضية ليقوموا باستخدامها لتزويد المستخدمين بخدمات التطبيقات[4].

تعد موازنة الحمل تحدياً كبيراً في بيئة الحوسبة السحابية نتيجة للطبيعة الديناميكية للحمل الذي يمكن أن يتغير بشكل سريع سواء من حيث عدد المهام المطلوب تنفيذها أو أطوال هذه المهام [5]، وهذا ما يجعل تحصيص الآلات الافتراضية لمهام المستخدمين قضية بالغة الأهمية بالنسبة لأداء النظام ككل[6]. وتعرف موازنة الحمل على أنها عملية توزيع الطلبات على الآلات المختلفة باستخدام خوارزمية فعالة لجدولة المهام بحيث يتم تنفيذ المهام بأقل وقت ممكن مع المحافظة على نسبة استخدام عالية للآلات الافتراضية ومراقبة أدائها بشكل يضمن عدم وجود آلات تعاني من حمل زائد (overloaded) في حين أن هناك آلات أخرى ذات حمل منخفض (underloaded)[7].

يمكن تصنيف خوارزميات موازنة الحمل إلى نوعين: ساكنة (Static) وديناميكية (Dynamic). حيث تقوم الخوازرميات الساكنة بتوزيع الحمل من دون الأخذ بعين الاعتبار الحمل الحالي المطبق على كل آلة افتراضية. في حين تعتمد الخوارزميات الديناميكية على هذه المعلومات عند اتخاذ قرار الجدولة كما يمكن أن تقرر خلال مرحلة التنفيذ أن تقوم بتهجير مهمة من آلة افتراضية إلى آلة افتراضية أخرى لتحقيق هدف موازنة الحمل[8].

بالإضافة إلى ذلك يمكن أن تصنف الخوارزميات الديناميكية إلى نوعين : دفعي (Batch) ومباشر (Online). في النمط الدفعي يتم تجميع المهام في مخزن مؤقت ومن ثم القيام بجدولتها خلال فواصل زمنية محددة مسبقاً. في حين تقوم الخورازمية من النمط المباشر أو الفوري بجدولة المهمة لحظة وصولها [9].

من جهة أخرى يمكن تقسيم الخورازميات من حيث طريقة إيجاد الحل إلى نوعين: الخورارزميات التجريبية أو الإرشادية (Heuristic) وهي خوارزميات تعتمد على طبيعة المشكلة المراد حلها مثل خوارزمية

Min–Min و Max–Min. وخوارزميات ذاتية الإرشاد (Meta–heuristic) التي تعتمد على البحث ضمن فضاء حلول عشوائي، وتكون هذه الخورازميات مستقلة عن المشكلة ويمكن استخدامها لحل العديد من المشاكل مثل خوارزمية أمثلة سرب العناصر ((Particle Swarm Optimization (PSO) وخوارزمية أمثلة مملكة النمل(Ant Colony Optimization (ACO))والخوارزميات الجينية [10].

2–أهمية البحث وأهدافه:

إن الموازنة بين متطلبات مزود الخدمة السحابي من حيث الاستثمار الأمثل للموارد، مع المحافظة على جودة الخدمة المقدمة للمستخدمين تتطلب وجود خوارزمية موازنة حمل فعالة تقوم بتوزيع الحمل بشكل عادل بين الآلات الافتراضية. تكمن أهمية هذا البحث في اقتراحه لخوارزمية جدولة مهام باستخدام أمثلة سرب العناصر PSO تأخذ بعين الاعتبار عدم التجانس الموجود على مستوى كل من الآلات الافتراضية والمهام لتحقيق هدف موازنة الحمل.

يهدف هذا البحث إلى دراسة الخوارزمية المقترحة، ومقارنة أدائها مع العديد من الخوارزميات المقدمة سابقاً، وذلك وفقاً لمجموعة من المعايير المعتمدة في هذا المجال.

3–منهجية البحث:

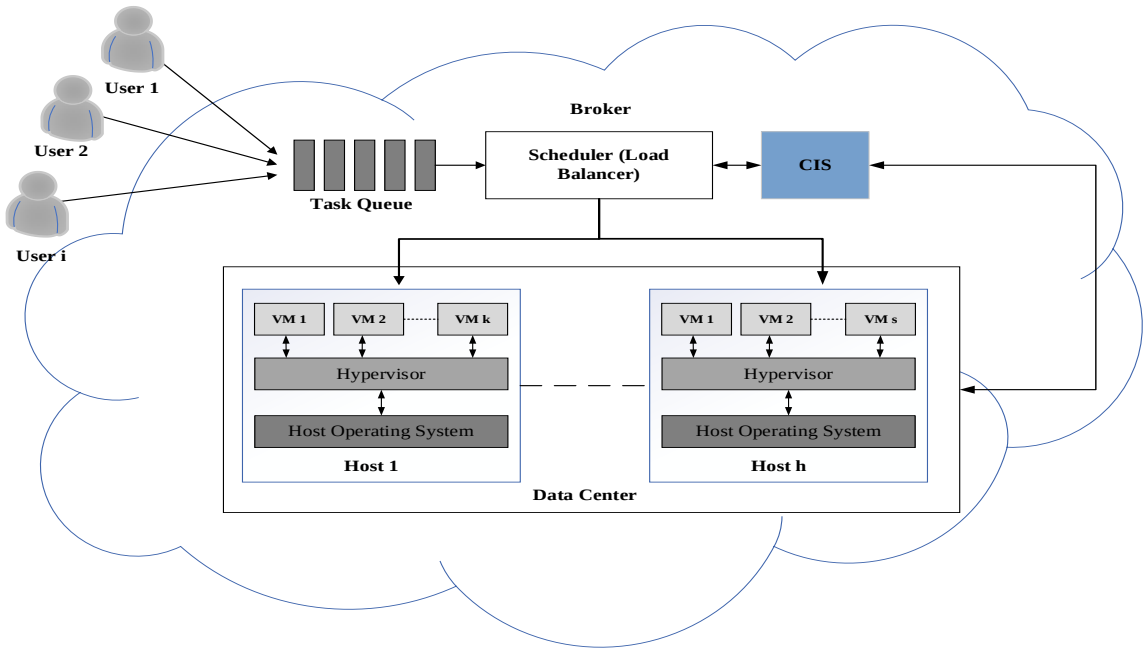
لتحقيق هدف البحث، تمَّ اتباع المنهجية التّالية:

- ✓ دراسة نظرية لبيئة مركز البيانات السحابي والمعايير المستخدمة لتقييم كفاءة خوارزميات موازنة الحمل.
- ✓ دراسة لمجموعة من الأعمال السابقة المقدمة في هذا المجال مع تحليل نقاط قوتها وضعفها بهدف تصميم خوارزمية قوية قادرة على تحقيق أهداف البحث.
- ✓ اعتماد المُحاكاة الحاسوبية للتحقّق من أداء الخوارزمية المقترحة، ثمَّ القيام بمناقشة النتائج، وصياغة الاستنتاجات.

4–الدراسة النظرية:

4–1 نموذج النظام ومعايير الأداء:

يظهر الشكل(1) المكونات الأساسية لنظام الحوسبة السحابية الذي يتكون من مجموعة من العناصر التي تتفاعل مع بعضها البعض. يقوم المستخدمون بإرسال مهامهم إلى رتل انتظار ذو طول محدد مسبقاً (حيث تتم الجدولة خلال فواصل زمنية دورية أو حالما يتم ملء الرتل بالمهام) لتتم جدولتهم على الآلات الافتراضية بالاعتماد على خوارزمية موازنة الحمل المستخدمة من قبل وسيط مركز البيانات (Datacenter Broker)،وهو المسؤول عن تحديد وجمع المعلومات عن الموارد المتوافرة في مركز البيانات عبر إرسال استعلام إلى خدمة المعلومات السحابية(cloud information service(CIS)). ويوجد على كل مخدم مراقب الآلات الافتراضية (hypervisor)، الذي يمثل واجهة بين نظام تشغيل المخدم والآلات الافتراضية التي يستضيفها[11].



الشكل(1): نموذج النظام[11]

يتضمن النموذج الرياضي لمشكلة البحث مجموعة من المهام المستقلةغير المتجانسة (ذات أطوال مختلفة) وغير الشفعية (non-preemptive) والتي لا يمكن مقاطعة تنفيذها: $\{T_1,T_2,T_3,\ldots T_n\}$ والمطلوب جدولتها على مجموعة من الآلات الافتراضية غير المتجانسة (ذات قدرات معالجة مختلفة): $\{VM_1,VM_2,\ldots VM_m\}$. نعرف زمن الانتهاء للآلة الافتراضية CT_j على أنه زمن انتهاء تنفيذ آخر مهمة تمت جدولتها على هذه الآلة. وبناء على ما سبق نقوم بتعريف مجموعة من المعايير المستخدمة في تقييم كفاءة خوارزمية موازنة الحمل.

1. زمن التنفيذ الكلي للمهام (makespan) ويعرّف على أنه أكبر قيمة زمن انتهاء بين جميع الآلات الافتراضية، ويتم حسابه كما هو موضح بالمعادلة(1)[4]:

$$\text{Makespan} = \max\{CT_j \mid j = 1,2,\ldots.m\} \qquad (1)$$

2. معدل استخدام الموارد الوسطي (Average Resource Utilization Ratio) وهو يمثل معدل انشغال الآلات الافتراضية نسبة لزمن التنفيذ الكلي للمهام ويعطى بالمعادلة(2) [12]:

$$ARUR = \sum_{j=1}^{m} \frac{CT_j}{\text{makespan} * m} \qquad (2)$$

حيث يمثل m عدد الآلات الافتراضية.

3. درجة عدم التوازن (Degree of imbalance) وهي مقياس لكفاءة توزيع الحمل بين الآلات الافتراضية المتوفرة عن طريق حساب الفرق بين أعلى وأقل زمن انتهاء بين الآلات الافتراضية مقسوماً على زمن الانتهاء الوسطي،وكلما صغرت هذه القيمة كلما زادت فعالية خوارزمية موازنة الحمل وتعطى بالمعادلة(3) [13]:

$$DI = \frac{CT_{max} - CT_{min}}{CT_{avg}} \qquad (3)$$

4-2 خوارزميات موازنة الحمل:

نظراً للتأثير الكبير لخوارزميات موازنة الحمل على أداء نظام الحوسبة السحابية قام الباحثون بتقديم العديد من الخوارزميات المبتكرة وسوف نقوم باستعراض بعض هذه الأبحاث والتي شكلت الأساس الذي انطلقنا منه في تطوير خوارزميتنا المقترحة.

في [14] قام الباحثون بتقديم خوارزمية Enhanced Load Balanced Min-Min(ELBMM) من النوع الإرشادي الساكن والتي تعتبر نسخة محسنة عن الخورازمية المقدمة في [15]، حيث تتكون هذه الخورازمية من مرحلتين: في المرحلة الأولى يتم تحصيص المهام وفقاً لخورازمية Min-Min التقليدية (تحصيص أصغر مهمة للآلة الافتراضية ذات زمن الانتهاء الأدنى في حال تحصيصها لها). في المرحلة الثانية يتم البحث عن الآلة الافتراضية التي لديها أطول زمن انتهاء (المسؤولة عن قيمة makespan)، ومن ثم اختيار أطول مهمة (التي تستغرق أطول وقت للتنفيذ)، ثم البحث عن الآلة الافتراضية التي في حال تحصيص هذه المهمة لها يكون لديها أكبر زمن انتهاء بحيث يكون هذا الزمن أقل من زمن الانتهاء للآلة الافتراضية التي نقلت منها. تستمر هذه العملية حتى لا نستطيع إيجاد أي آلة افتراضية يمكن نقل المهمة إليها بحيث تحقق الشرط.

في [16] قدم الباحثون خوارزمية موازنة الحمل المدركة للموارد (Resource-Aware Load Balancing Algorithm (RALBA) من النوع الإرشادي الديناميكي الدفعي وتعتمد على فكرة أن لكل آلة افتراضية حصة (VMShare) من الحمل الكلي (مجموع أطوال جميع المهام جدولتها) تتناسب مع قدرة المعالجة الخاصة بها، حيث تمر هذه الخورازمية بمرحلتين أيضاً: في المرحلة الأولى وتسمى مرحلة الملء (Fill) يتم اختيار الآلة الافتراضية صاحبة أكبر حصة من الحمل ويتم انتقاء أطول مهمة يمكن أن يتم إسنادها لهذه الآلة بحيث لا يتجاوز مجموع المهام المسندة إليها الحصة المقررة لها، ومن ثم يتم إنقاص حصة الآلة بمقدار طول المهمة. تستمر هذه المرحلة حتى تكون الآلة الافتراضية صاحبة أكبر حصة أقل من طول أصغر مهمة في لائحة المهام المتبقية. في المرحلة الثانية وتسمى مرحلة الانسكاب (Spill) ويتم فيها اختيار أطول مهمة من المهام الباقية وإسنادها للآلة الافتراضية ذات زمن الانتهاء الأدنى في حال التحصيص. تستمر هذه المرحلة حتى تتم جدولة جميع المهام الموجودة في رتل الانتظار.

تعتبر خوازرميات ذكاء السرب (Swarm Intelligence) من الخوارزميات ذاتية الإرشاد التي تعمل على مشاركة المعلومات بين عناصر السرب خلال البحث في فضاء الحلول عن الحل الأمثل (أو القريب من الحل الأمثل) خلال وقت قصير [17]. وهناك العديد من الخورازميات مثل أمثلة سرب العناصر PSO وأمثلة مملكة النمل ACO. من بين هذه الخورازميات أظهرت خوازرمية PSO أداء أفضل وتعقيداً أقل من بقية الخورازميات، حيث أنها تتميز بالسرعة وبقلة البارامترات التي هي بحاجة للضبط، بالإضافة إلى قدرتها على استخدام قيم صحيحة لتمثيل الحلولعوضاً عن التمثيل الثنائي المستخدم في الخورازمية الجينية[18]. بناء على ما سبق سوف نقوم بالتركيزعلى خوارزميات PSO في البحث.

في خوارزمية PSO القياسية يتم تهيئة فضاء الحلول بشكل عشوائي، إلا أن هذه العشوائية تخفض من فرصة الوصول إلى الحل الأمثل على اعتبار أن الخورازمية تعتمد بشكل كبير على الحلول الأولية التي يتم الانطلاق منها في عملية البحث [19]. بناء عليه قام الباحثون بمحاولة حل هذه المشكلة عن طريق تحسين جودة الحلول الأولية. بعض هذه الحلول اعتمد على تهيئة الحلول الأولية باستخدام خورازميات إرشادية تقليدية كما في [11]، حيث وجد الباحثون بأن استخدام خورازمية زمن الانتهاء الأدنى ((minimum completion time (MCT) يعطي أفضل النتائج. في

حين اعتمد باحثون آخرون على استخدام خوارزميات ذاتية الإرشاد أخرى، حيث تتم تهيئة فضاء الحلول الأولي بناء على الحلول التي يتم الوصول إليها من قبل هذه الخورازميات مثل الخورازميات الهجينة GA–PSO في [20] و Firefly–PSO في [21]. إلا أن المشكلة التي تواجه هذا النوع من الحلول هو إمكانية وقوع الخورازمية في فخ الأمثلة المحلية والتقارب المبكر عدا عن التعقيد الحسابي الكبير في حالة الخوارزميات الهجينة.

من جهة أخرى حاول باحثون آخرون تحسين جودة الحلول العشوائية الأولية عن طريق تطبيق خوارزميات ذاتية الإرشاد بهدف تحقيق أكبر قدر ممكن من موازنة الحمل على هذه الحلول مثل خوارزمية البحث المحظور Tabu–Search التي تسعى إلى تخفيض قيمةmakespan في [22]، وخوارزمية مستعمرة النحل الاصطناعية((Artificial bee colony(ABC) لتخفيض انحراف أزمنة انتهاء الآلات الافتراضية عن زمن الانتهاء الوسطي في [4]. ومما لا شك فيه بأن استخدام هذه الخورازميات سيضيف الكثير من التعقيد على سير عمل الخوارزمية.

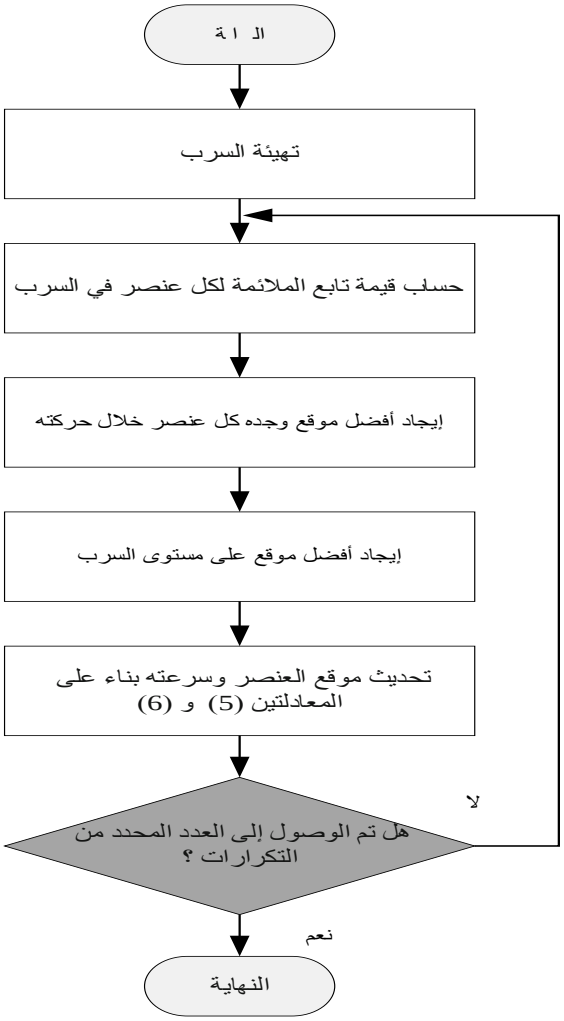
بالإضافة إلى ما سبق تختلف الخوارزميات فيما بينها من حيث تابع الملاءمة الذي تقيم على أساسه جودة الحلول التي يتم الوصول إليها. بعض الخوارزميات تعتمد تابع ملائمة يسعى إلى تخفيض زمن التنفيذ الكلي كمافي [23]و [24]، في حين يعتمد بعضها الآخر على النسبة ما بين زمن التنفيذ الكلي ومعدل استخدام الموارد الوسطي كما في [4] و [25]. ولا شك بأن كفاءة اختيار تابع الملاءمة يلعب دوراً كبيراً في جودة الحلول التي يتم الوصول إليها.

4–3 خوازرمية أمثلة سرب العناصر القياسية:

خوارزمية أمثلة سرب العناصر هي خوارزمية ذاتية الإرشاد تم تقديمها من قبل Eberhart and Kennedy في العام 1995[26]. وهي تحاكي السلوك الاجتماعي لسرب من الطيور أثناء قيامهم بالبحث عن الطعام. يدعى كل طير داخل الخورازمية بالعنصر (particle)، حيث يمتلك كل عنصر شعاعاً[1] يعطي موقعه الحالي (position vector)، والذي يمثل أحد الحلول الممكنة للمشكلة، وشعاع سرعة (velocity vector) يعبر عن حركته ضمن فضاء الحلول. بداية يتم توليد المواقع بشكل عشوائي ومن ثم يتغير الموقع في كل تكرار من تكرارات الخورازمية، حيث يعتمد الموقع الجديد للعنصر على قيمة كل من: سرعته في التكرار السابق، أفضل موقع وجده العنصر(pbest)، موقع أفضل عنصر موجود في السرب (gbest).

يتم تقييم المواقع أو جودة الحلول المقدمة باستخدام ما يسمى بتابع الملاءمة (fitness function) في بداية كل تكرار من تكرارات الخوارزمية. يظهر الشكل(2) المخطط التدفقي للخوارزمية.

[1]طول الشعاع يعتمد على بُعد فضاء الحلول، في خورازميات الجدولة يكون طول الشعاع مساوياً لعدد المهام المطلوب جدولتها



الشكل(2): المخطط التدفقي لخوارزمية PSO[27]

يتم حساب كل من شعاع الموقع والسرعة باستخدام المعادلتين (5) و (6) [4] :

$$\overrightarrow{v_i^{t+1}} = w * \overrightarrow{v_i^t} + c_1 * rand_1 * \left(\overrightarrow{pbest_i} - \overrightarrow{x_i^t}\right) + c_2 * rand_2 * \left(\overrightarrow{gbest} - \overrightarrow{x_i^t}\right) \quad (5)$$

$$\overrightarrow{x_i^{t+1}} = \overrightarrow{x_i^t} + \overrightarrow{v_i^{t+1}} \quad (6)$$

حيث : $\overrightarrow{v_i^t}$ يمثل سرعة العنصر i في التكرار t، $\overrightarrow{v_i^{t+1}}$ يمثل سرعة العنصر في التكرار t+1 ، w يمثل وزن القصور الذاتي، c_1, c_2 تمثل معاملات التسارع، $rand_1$و $rand_2$ عدد عشوائي ضمن المجال [0,1]، $\overrightarrow{x_i^t}$ الموقع الحالي للعنصر، $\overrightarrow{pbest_i}$ أفضل موقع وجده العنصر، $\overrightarrow{gbest}$ موقع أفضل عنصر في السرب، $\overrightarrow{x_i^{t+1}}$ موقع العنصر في التكرار t+1.

5- الخوارزمية المقترحة:

بناء على دراستنا المرجعية وتحليلنا لأداء الخوارزميات المقدمة في مجالموازنة الحمل في بيئة الحوسبة السحابية قمنا بتصميم خورازمية جديدة وهي خوارزمية الموازنة الديناميكية المدركة لعدم التجانس باستخدام أمثلة سرب العناصر (Heterogeneity aware Dynamic Load Balancing using Particle Swarm Optimization (HADLB-PSO))، والتي تتألف بشكل أساسي من مرحلتين:يتم تنفيذ المرحلة الأولى قبل الإسناد الفعلي للمهمات إلى الآلات الافتراضية باستخدام خوارزمية أمثلة سرب العناصر، والثانية تتم خلال مرحلة التنفيذ عبر مراقبة الأحداث التي يمر بها النظام.

5-1 المرحلة الأولى:

في هذه المرحلة نقوم ببناء فضاء الحلول الأولية باستخدام طريقة مبتكرة تحاول تلافي التعقيد الحسابي قدر الإمكان في سبيل إعطاء العناصر مواقع جيدة للانطلاق منها في عملية البحث عوضاً عن الحلول العشوائية أو التقليدية أو الهجينة، ومن ثم صياغة تابع الملاءمة الخاص بخوارزميتنا وتحديد البارامترات المستخدمة من قبل الخوارزمية.

- ✓ **بناء فضاء الحلول:**لضمان أن يتم توزيع الحمل بشكل عادل على الآلات الافتراضية غير المتجانسة بشكل يتناسب مع قدراتها وسرعاتها نقوم بتحديد عتبة حمل لكل آلة افتراضية بشكل مشابه لما هو متبع في خوارزمية RALBA حيث نقوم بتعريف عتبة الحمل للآلة الافتراضية j وفق العلاقة(7)[5]:

$$VM^{j}_{Threshold} = \frac{V^{j}_{Capacity}}{C} * TotalLoad \qquad (7)$$

حيث تمثل Cمجموعقدرات جميع الآلات الافتراضية الموجودة في النظام، TotalLoadيمثل مجموع أطوال جميع المهام المطلوب جدولتها. كما نُعرّف حمل الآلة الافتراضيةVM^{j}_{Load} على أنه مجموعة أطوال المهام التي تم إسنادها إلى هذه الآلة.

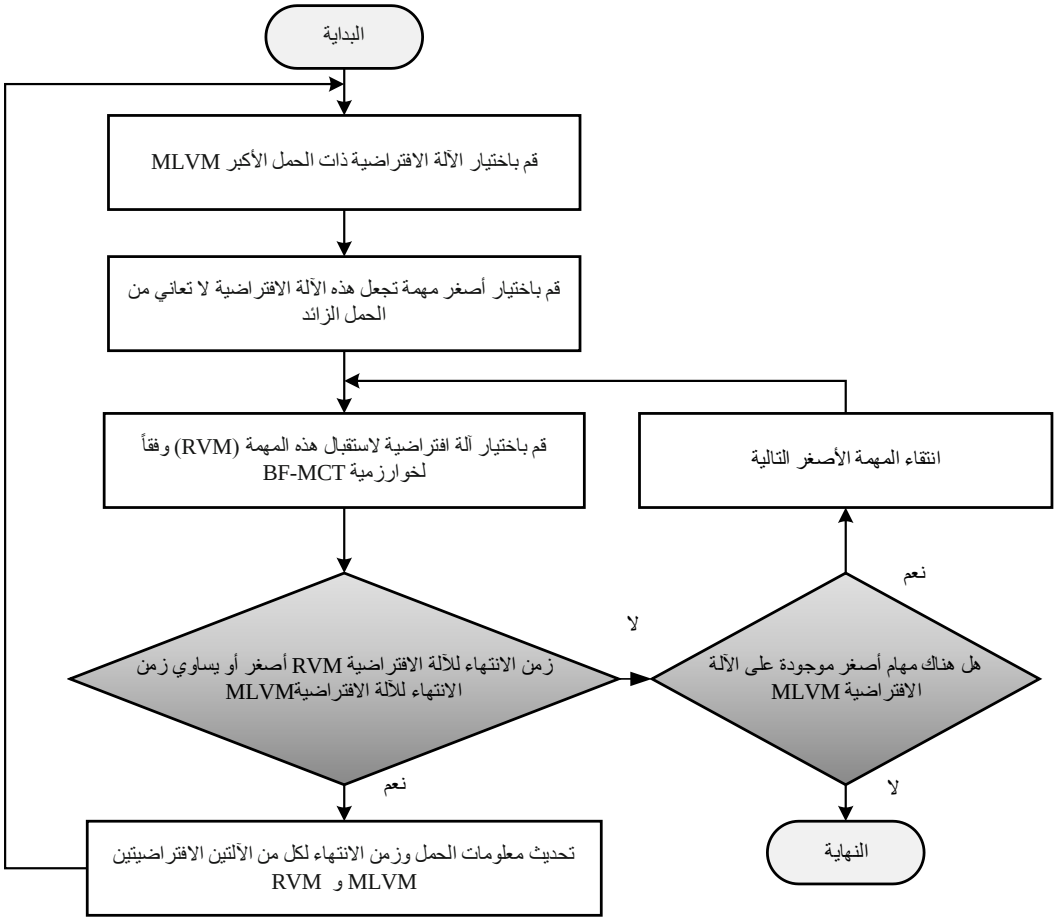
في البداية يتم تهيئة المواقع الأولية للعناصر وفق الخطوات التالية التي يتم تكرارها طالما أن هناك مهمة لم يتم تخصيصها:

1. اختيار إحدى المهام بشكل عشوائي من قائمة المهام المطلوب جدولتها.
2. استخدام خوارزمية الملائم الأفضل – زمن الانتهاء الأدنى (Best Fit – Minimum Completion Time)في اختيار الآلة الافتراضية التي سوف يتم إسناد المهمة إليها والتي تعتمد على إيجاد الآلة الافتراضية التي يؤدي تحصيص المهمة إليها إلى جعل الفرق بين الحمل المطبق والعتبة أقل ما يمكن (حيث يجب أن تكون قيمة الحمل أصغر من العتبة)، وفي حال عدم وجود هكذا آلة يتم تحصيص المهمة للآلة الافتراضية ذات زمن الانتهاء الأدنى في حال التحصيص.
3. تحديث معلومات الحمل وزمن الانتهاء للآلة الافتراضية التي تم اختيارها.

نتيجة للعشوائية التي يتم بها اختيار المهمات لجدولتها وللحل الثاني في خوارزمية BF-MCT يمكن أن يتجاوز الحمل المطبق على إحدى الآلات عتبة الحمل الخاصة بها، لذلك سوف نقوم بتطبيق خورازمية لموازنة الحمل على موقع كل عنصر بهدف تحسين جودة الحل وضمان عدم وجود آلات افتراضية تعاني من حمل زائد

في حين أن هناك آلات أخرى ذات حمل منخفض. حيث يعتمد عدد مرات تكرار الخوارزمية على جودة الحل الأولي وهو بشكل عام يكون أقل بكثير فيما لو طبقنا الخورازمية على حلول عشوائية صرفة.

وكما هو موضح في الشكل(3) يتم اختيار الآلة الافتراضية ذات الحمل الزائد التي تكون قيمة الحمل المطبق عليها أكبر من العتبة الخاصة بها، بحيث يكون الفرق بين القيمتين هو الأكبر بين جميع الآلات الافتراضية. ثم يتم انتقاء أصغر مهمة، والذي يؤدي نقلها من هذه الآلة إلى جعل الحمل المطبق أقل أو يساوي من العتبة. يتم البحث عن الآلة الافتراضية التي يتوجب نقل المهمة إليها باستخدام خورازمية BF-MCT، ويتم نقل المهمة إليها تحت شرط أن يكون زمن الانتهاء في حال التحصيص أقل من زمن انتهاء الآلة الافتراضية التي تم النقل منها، وفي حال تحقق الشرط يتم نقل المهمةو تحديث معلومات الحمل وزمن الانتهاء لكل من الآلتين، ومن ثم البحث مجدداً عن الآلة الافتراضية التي تعاني من الحمل الزائد. في حال عدم تحقق الشرط يتم اختيار المهمة الأصغر التالية، ومحاولة نقلها وفي حالة عدم وجود إمكانية لنقل أي مهمة من الآلة الافتراضية ذات الحمل الزائد الأكبر يتم الخروج من الخورازمية، ويعود السبب في ذلك إلى أننا نتعامل مع حلول أولية ستتم محاولة تحسينها لاحقاً خلال سير عمل خوارزمية PSO وبهدف تخفيف التعقيد الحسابي قدر الإمكان.



الشكل(3): خوارزمية موازنة الحمل المطبقة على مواقع العناصر

✓ **تابع الملاءمةFitness Function:** لضمان أن يتم تنفيذ المهام بأسرع وقت وأن تبقى الآلات الافتراضية مشغولة بنفس مقدار الوقت قدر الإمكان نعرف تابع الملاءمة الخاص بخوارزميتنا وفق العلاقة (8):

$$\text{Fitness Function} = \text{makespan} * \sigma \qquad (8)$$

حيث يمثل σالانحراف المعياري لأزمنة انتهاء الآلات الافتراضية عن قيمة المتوسط الحسابي لهذه الأزمنة، وكلّما صغرت قيمة هذا التابع كلّما كان موقع العنصر أفضل .

✓ **بارامترات الخورازمية :** تعتبر القيمة التي يأخذها وزن القصور الذاتي wخلال مراحل تكرار الخورازمية واحدة من أهم المشاكل التي حاول الباحثون إيجاد حل لها، بهدف تجنب الوقوع في مشكلة الأمثلة المحلية ونتيجة لدراستنا وتجريبنا للعديد من الطرق وجدنا بأن المنهج المستخدم في [25] يقود إلى نتائج جيدة، حيث يعتمد على تخفيض قيمة w بشكل خطي لما نسبته 70% في المئة من التكرارات، وفي المراحل الأخيرة من الخورازمية يتم إعطاء w قيمة عشوائية ضمن المجال$(0.4,0.7)$، وهذا ما يعطي قيمة أكبر ويسمح للخوارزمية بالقفز خارج الحل الأمثلي المحلي والبحث عن الحل الأمثلي العام. تعطى معادلة حساب w وفق العلاقة (9)[25]:

$$w = \begin{cases} w_{max} - \frac{t(w_{max} - w_{min})}{T} & t < 0.7 * T \\ 0.4 + 0.3 * rand() & t > 0.7 * T \end{cases} \quad (9)$$

حيث يمثل t التكرار الحالي، T العدد الكلي للتكرارات. بقية البارامترات الخاصة بالخوارزمية موضحة بالجدول(1)[25].

الجدول(1): البارامترات الخاصة بخورازمية PSO

Number of particles	100
Initial inertia weight w	1.4
Minimum Value of w	0.4
Learning factors c1,c2	2
Number of iterations	100

5-2 المرحلة الثانية:

لا تضمن خوارزمية PSO الوصول بشكل مؤكد إلى الحل الأمثل إلا أنها تضمن حلاً قريباً منه. لزيادة فعالية الخورازمية المقترحة يقوم موازن الحمل بمراقبة الآلات الافتراضية خلال مرحلة التنفيذ بشكل مستمر وفي حالة إنهاء إحدى الآلات الافتراضية (VM_i) واحدة من المهام المسندة إليها يقوم موازن الحمل بالبحث عن الآلة الافتراضية ذات زمن الانتهاء الأعلى(VM_j)، ومن ثم يقوم بفحص قائمة المهام التي هي في طور الانتظار على هذه الآلة(من أطول مهمة إلى أصغر مهمة)، ويقوم باختبار فيما إذا كانت VM_i قادرة على إنهاء إحدى هذه المهام بحيث يكون زمن الانتهاء الخاص بها أقل من VM_j، عندئذ يقوم بإلغاء تنفيذ المهمة المختارة على VM_j وإرسالها إلى VM_i. يضمن هذا الحل تحقيق أي تخفيض ممكن لزمن التنفيذ الكلي(makespan). كما أن هذا الحل يمكن توسيعه ليعالج الحالة التي يمكن أن تصل فيها إحدى الآلات الافتراضية خارج أوقات الجدولة المحددة مسبقاً، وأيضاً الحالة التي يمكن أن تتجاوز فيها إحدى الآلات الافتراضية الزمن المتوقع لتنفيذها إحدى المهام نتيجة لوجود حلقة ما ضمن المهمة.

6. بيئة المحاكاة:

نستخدم في هذا البحث أداة المحاكاة cloudsim[28]،وهي واحدة من أشهر الأدوات المستخدمة في محاكاة البيئات السحابية،يظهر الجدول(2) البارامترات الخاصة ببيئة المحاكاة، حيث تدعى المهام في هذه الأداة بالـ cloudlets التي يعبر عن طولها بواحدة مليون تعليمة ((million instructions (mi)، في حين تقاس سرعة المعالجة في المضيفات والآلات الافتراضية بواحدة مليون تعليمة في الثانية (million instructions per second (mips))، حيث تم استخدام عشرة أنواع من الآلات الافتراضية تتراوح سرعاتها مابين 500−5000 mips.

سوف نقوم بمقارنة أداء خوارزميتنا المقترحة (HADLB−PSO) مع كل من الخوارزميات:ILBMM، RALBA، (MCT−PSOتم استخدام نفس تابع الملاءمة والبارامترات المستخدمة في خوارزميتنا)، IPSO[25].وسوف تتم المقارنة من حيث زمن التنفيذ الكلي للمهام makespan، معدل استخدام الموارد الوسطي ARUR، درجة عدم التوازن.

الجدول(2) البارامترات الخاصة ببيئة المحاكاة

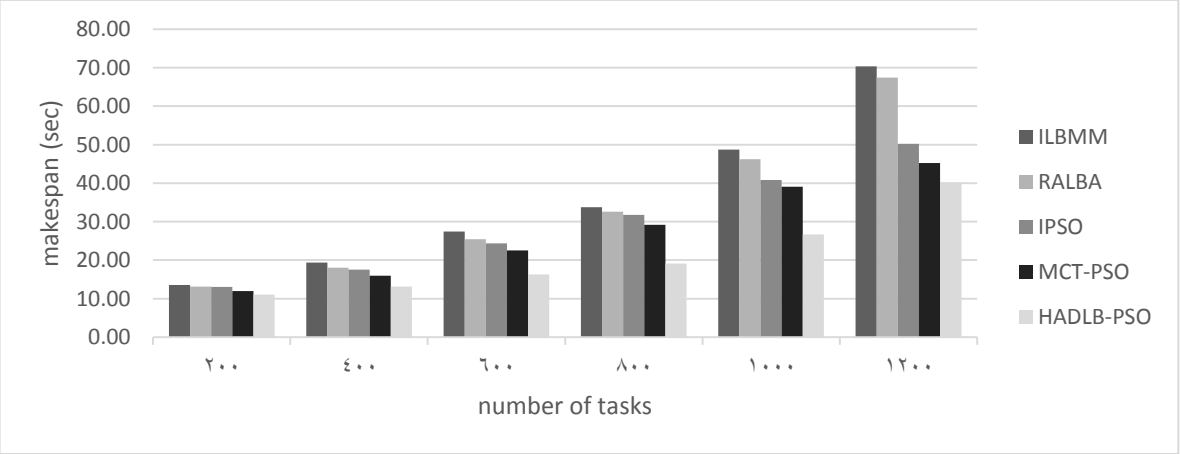
Type	Parameter	Description
Data center	Number of Datacenter	1
	Number of Host	20
	Number of Processingelements	4/8
	Processing Speed	10000 MIPS
	Operating system	Linux
	Hypervisor	Xen
	Host bandwidth	10000
	Host memory	4096
	System architecture	X86
Virtual machine (VM)	Total number of VM	20-100
	Number of processing element (PE) per VM	1
	VM memory	512 MB
	VM bandwidth	1000
	Processing Speed	500-5000 MIPS
Tasks (Cloudlets)	Input file size	300
	Output file size	300
	Utilization model	Full
	Number of cloudlets	200-1200
	Length of cloudlets	100-100000 MI

7-النتائج والمناقشة:

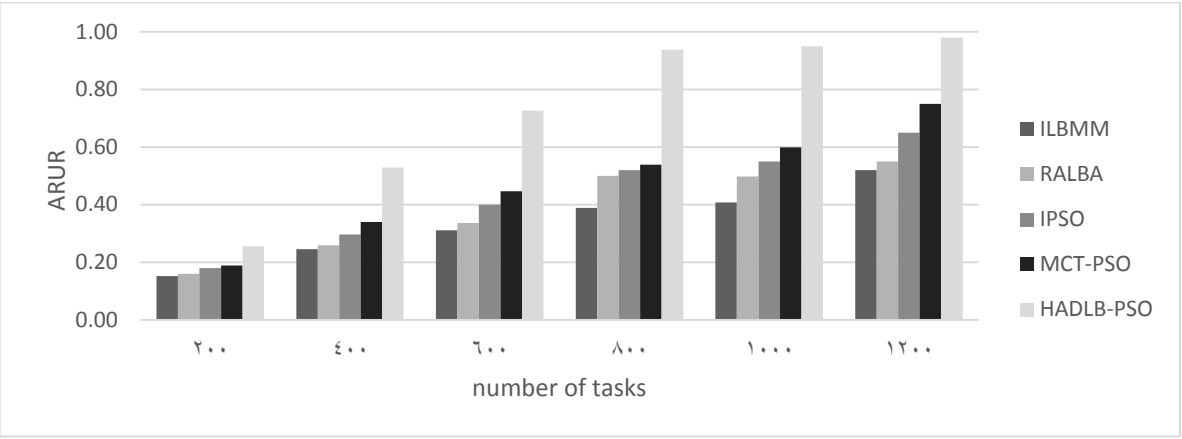
7-1 السيناريو الأول:في هذا السيناريو قمنا باستخدام 100 آلة افتراضية مع القيام بتغيير عدد المهام المطلوب جدولتها دفعة واحدة ضمن المجال 1200-200، حيث تتيح هذه التجربة مقارنة أداء الخوارزميات مع زيادة الحمل المطبق كما تختبر قدرة خوارزميات PSO على التعامل مع فضاء حلول أكبر.

تظهر الأشكال من (4) حتى (6) نتائج المقارنة بين الخوارزميات، حيث تظهر النتائج بشكل عام تفوق خوارزميات PSO على الخورازميات التقليدية، كما تظهر الدور الكبير الذي يقدمه البناء الفعال لفضاء الحلول الأولي في الوصول إلى حلول أكثر قرباً من الحل المثالي حيث تتفوق كل من خورازمية HADLB-PSO و MCT-PSO على خوارزمية IPSO التقليدية.

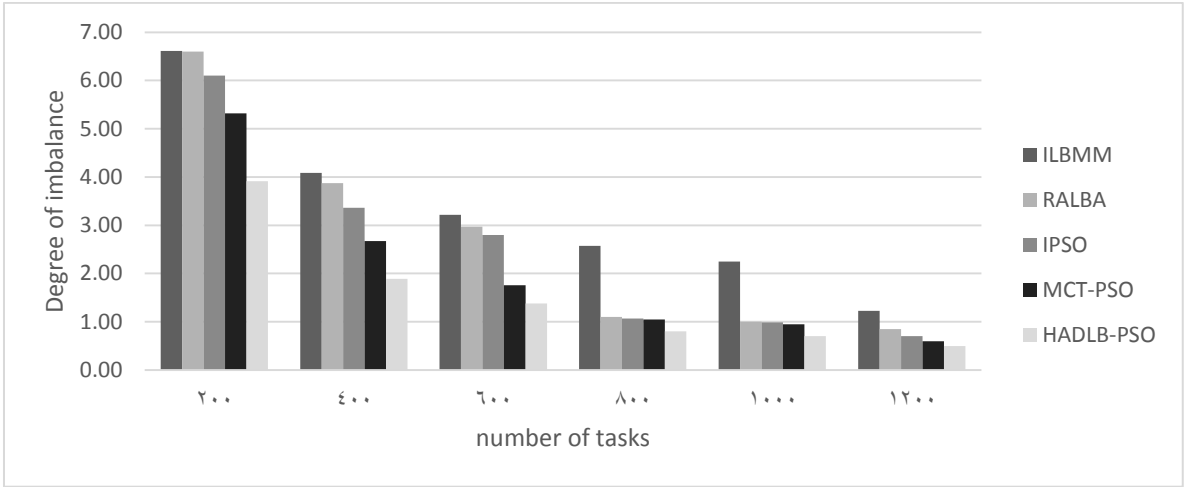
تبين النتائج بأن زمن التنفيذ الكلي للمهام makespan ومعدل استخدام الموارد الوسطيARUR يزداد مع زيادة عدد المهام في حين تنخفض درجة عدم التوازن DI، ويعود السبب في ذلك إلى مشاركة المزيد من الآلات في تنفيذ المهام، حيث يتم التركيز بداية على الآلات الافتراضية الأكثر سرعة وكلما زاد الحمل كلما أصبحت هناك حاجة إلى اختيار آلات أقل سرعة ولكنها ذات حمل خفيف، ويمكنها إنجاز المهام بشكل أسرع من الآلات السريعة ذات الحمل المرتفع.



الشكل(4): نتائج مقارنة الخوارزميات من حيث زمن التنفيذ الكلي makespan



الشكل(5): نتائج مقارنة الخوارزميات من حيث معدل استخدام الموارد الوسطيARUR



الشكل(6) نتائج مقارنة الخوارزميات من حيث درجة عدم التوازنDI

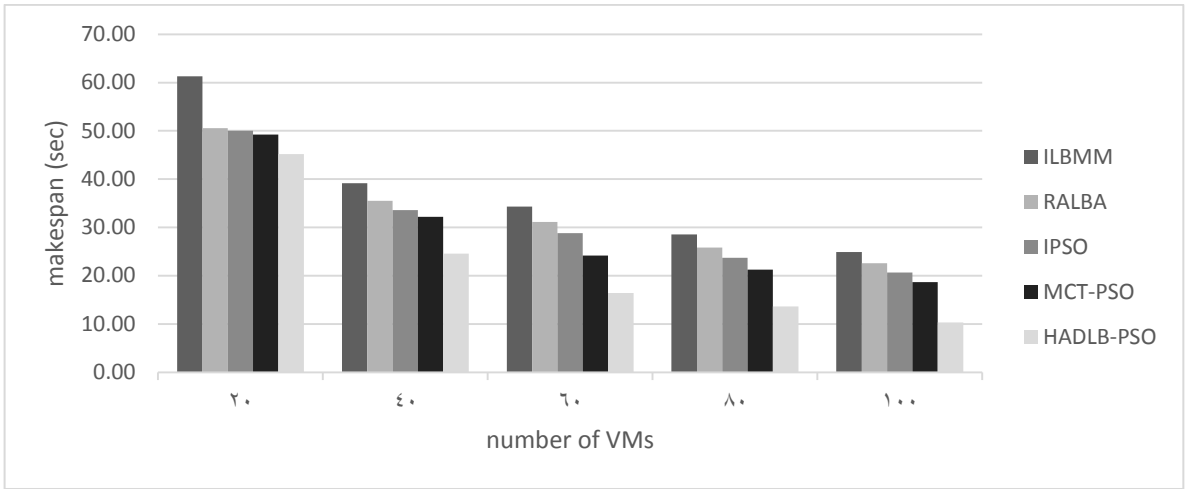
تظهر النتائج بوضوح تفوق الخوارزمية المقترحة على بقية الخوارزميات، ويعود السبب في ذلك إلى جودة الحلول التي تم الانطلاق منها في عملية البحث، والتي تأخذ بعين الاعتبار عدم التجانس الموجود في الموارد والمهام، وتحاول تقسيم الحمل بشكل يتناسب مع قدرات الآلات الافتراضية، وهذا ما يظهر بوضوح في معيار ARUR وDI، حيث تضمن الخوارزمية بأن تنشغل الآلات الافتراضية لأوقات متقاربة من بعضها سواء في مرحلة بناء فضاء الحلول أو تابع الملاءمة الذي يتم تقييم جودة الحلول على أساسه. كما أن سياسة الخورازمية خلال مرحلة التنفيذ تضمن بأن يتم تخفيض زمن التنفيذ الكلي للمهام عن طريق تهجير المهام بين الآلات الافتراضية. نلاحظ بأن توسع فضاء البحث لم يؤثر على أداء الخوارزمية كما هو الحال عند خورازميتي MCT–PSO و IPSO. يظهر الجدول(3) نسبة التحسين الوسطية التي تقدمها الخوارزمية المقترحة بالمقارنة مع بقية الخوارزميات.

الجدول(3): نسبة التحسين الوسطية المحققة من قبل الخوارزمية المقترحة في السيناريو الأول

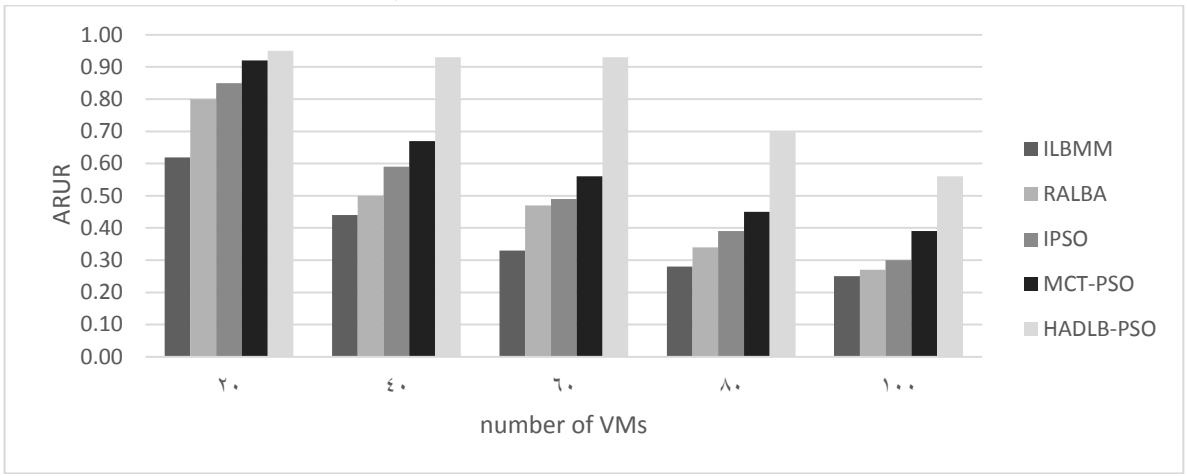
	ILBMM	RALBA	IPSO	MCT-PCO
Makespan	35%	33%	30%	23%
ARUR	115%	90%	70%	55%
DI	53%	42%	38%	24%

7–2 السيناريو الثاني:في هذه التجربة نقوم بتثبيت عدد المهام عند 500 مهمة ومن ثم نقوم بزيادة عدد الآلات الافتراضية ضمن المجال من 20 حتى 100 آلة افتراضية.

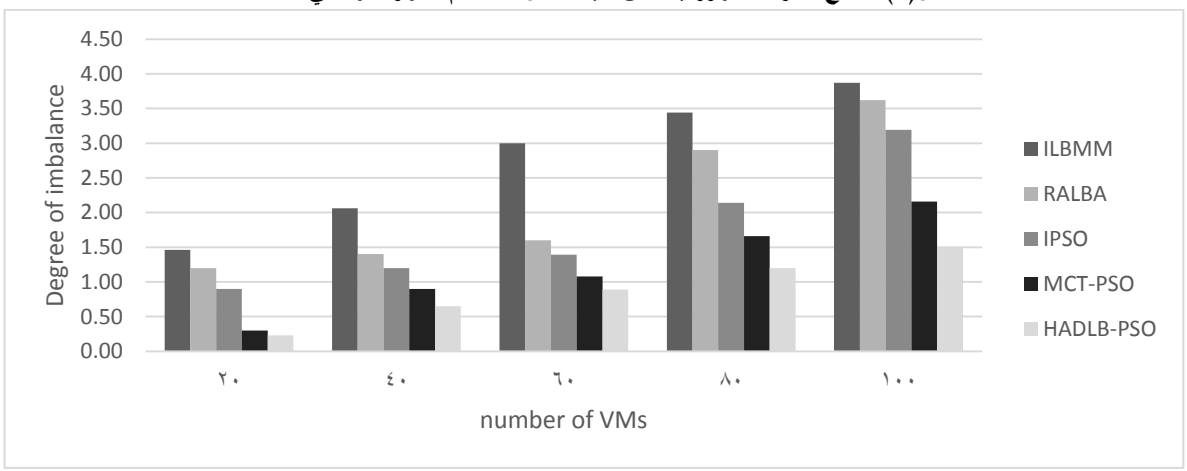
تظهر الأشكال من (7) حتى (9) نفس الاستنتاجات التي تم الوصول إليها في السيناريو الأول من حيث تفوق خوارزميات PSO على الخورازميات الإرشادية التقليدية، وتفوق الخوارزميات التي تحاول تحسين جودة الحلول الأولية على الخوارزميات التي تستخدم الحلول العشوائية.



الشكل(7): نتائج مقارنة الخوارزميات من حيث زمن التنفيذ الكلي makespan



الشكل(8): نتائج مقارنة الخوارزميات من حيث معدل استخدام الموارد الوسطي ARUR



الشكل(9): نتائج مقارنة الخوارزميات من حيث درجة عدم التوازن DI

تظهر النتائج بأن زمن التنفيذ الكلي للمهام ينخفض بشكل عام مع زيادة عدد الآلات الافتراضية وذلك يعود إلى زيادة نسبةالآلات الافتراضية السريعة التي تتم جدولة المهام عليها، إلا أن زيادة الاعتماد على هذه الآلات ينعكس سلباً على معدل استخدام الموارد الوسطي الذي ينخفض مع زيادة عدد الآلات، كما أن درجة عدم التوازن تزداد نتيجة تركيز الحمل على أسرع الآلات المتوفرة. إلا أنه ونتيجة لكفاءة الخوارزمية المقترحة تم

الحد من هذا التأثير السلبي،حيث قدمت نتائج جيدة في مختلف الحالات ويظهر الجدول(4) نسبة التحسين الذي قدمته الخورازمية مقارنة مع بقية الخورازميات.

الجدول(4): نسبة التحسين الوسطية المحققة من قبل الخوارزمية المقترحة في السيناريو الثاني

	ILBMM	RALBA	IPSO	MCT-PCO
Makespan	41%	33%	31%	29%
ARUR	112%	85%	55%	36%
DI	68%	58%	49%	27%

8-الاستنتاجات والتوصيات:

تم في هذا البحث تقديم حل فعال لمشكلة الموازنة ما بين متطلبات مزود الخدمة السحابي من حيث الاستثمار الأمثل للموارد والتوزيع العادل للحمل، ومتطلبات المستخدمين من حيث تخفيض الزمن اللازم لتنفيذ مهامهم. كما أظهرت النتائج الكفاءة التي تتمتع بها الخوارزميات التي تعتمد على أمثلة سرب العناصر وتفوقها على الخورازميات التقليدية عند استخدامها لموازنة الحمل في بيئة ذات طبيعة ديناميكية وغير متجانسة كالحوسبة السحابية. كما أثبتت النتائج بأن تحسين جودة الحلول الأولية يمكن أن يساعد بشكل كبير على الوصول إلى الحل الأمثل بطريقة أسرع. من جهة أخرى فإن هذه الخورازمية تتضمن تعقيداً حسابياً إضافياً خلال مرحلة بناء فضاء الحلول الذي يتم بشكل عشوائي في الخوارزمية التقليدية مما يؤدي إلى زيادة في وقت تنفيذ الخورازمية كلما زاد حجم فضاء الحلول، وهذا ما يفرض قيوداً على حجم رتل الانتظارأي عدد المهام التي يمكن جدولتها في كل دفعة بهدف الموازنة ما بين جودة الحل الذي تسعى الخوارزمية للوصول إليه والوقت الذي تستغرقه الخوارزمية لتحقيق ذلك.

بالنسبة للأعمال المستقبلية فإنها تتمحور حول اختبار فعالية الخورازمية مع مسارات حمل حقيقية وليست عشوائية كما هي في هذا البحث، بالإضافة إلى تطوير الخوارزمية للتعامل مع المهام غير المستقلة التي تفرض قيوداً على الترتيب الذي يجب أن يتم فيه تنفيذ المهام.

9- المراجع:

1. Mell, P; Grance, T.(2011).*The NIST Definition of Cloud Computing.*NIST Special Publication 800-145
2. He, S;Guo, L;Ghanem, M;Guo, Y.(2012).*Improving resource utilisation in the cloud environment using multivariate probabilistic models*. In IEEE Fifth International Conference on Cloud Computing,Honolulu,USA
3. Tripathi,G;Kumar, R.(2020).*Heuristic Cloudlet Allocation Policy for QoS Improvement in Cloud*. In 2020 International Conference on Electrical and Electronics Engineering (ICE3),Gorakhpur,India
4. Ebadifard, F ;Babamir, M.(2018).*A PSO-based task scheduling algorithm improved using a load-balancing technique for the cloud computing environment.*Concurrency and Computation: Practice and Experience. vol. 30, no. 12, p. e4368.
5. Mishra, S;Khan, A.(2017).*Time efficient dynamic threshold-based load balancing technique for Cloud Computing*. In 2017 International Conference on Computer, Information and Telecommunication Systems (CITS),Dalian, China
6. Ahmad, O;Khan, Z.(2019).*Pso-Based Task Scheduling Algorithm Using Adaptive Load Balancing Approach For Cloud Computing Environment.*International Journal Of Scientific & Technology Research. vol. 8, no. 11.
7. Kumar, M;Sharma, S.(2018).*Deadline constrained based dynamic load balancing algorithm with elasticity in cloud environment.*Computers & Electrical Engineering. vol. 69, pp. 395-411.
8. Mishra, K;Sahoo, B;Parida, P.(2020).*Load balancing in cloud computing: A big picture.*Journal of King Saud University-Computer and Information Sciences. vol. 32, no. 2, pp. 149-158.
9. Deldari, A;Naghibzadeh, M;Abrishami, S.(2017).*CCA:a deadline-constrained workflow scheduling algorithm for multicore resources on the cloud.*The journal of Supercomputing. vol. 73, no. 2, pp. 756-781.
10. Masdari, M;Salehi, F;Jalali, M.(2017).*A survey of PSO-based scheduling algorithms in cloud computing.*Journal of Network and Systems Management. vol. 25, no. 1, pp. 122-158.
11. Alsaidy, S;Abbood, A;Sahib, M.(2020).*Heuristic initialization of PSO task scheduling algorithm in cloud computing.*Journal of King Saud University-Computer and Information Sciences.
12. Panwar, N;Negi, S ;Rauthan, S.(2019).*TOPSIS–PSO inspired non-preemptive tasks scheduling algorithm in cloud environment.*Cluster Computing. vol. 22, no. 4, pp. 1379-1396.
13. Kong, L;Mapetu, B;Chen, Z.(2020).*Heuristic load balancing based zero imbalance mechanism in cloud computing.*Journal of Grid Computing. vol. 18, no. 1, pp. 123-148.
14. Patel, G;Mehta, R;Bhoi, U.(2015).*Enhanced load balanced min-min algorithm for static meta task scheduling in cloud computing.*International Journal of Computer Applications. vol. 57, pp. 545-553.

15. Kokilavani, T;Amalarethinam, G.(2011).*Load balanced min-min algorithm for static meta-task scheduling in grid computing.*International Journal of Computer Applications. vol. 20, no. 2, pp. 43-49.

16. Hussain,A;Aleem, M;Khan, A.(2018).*RALBA: a computation-aware load balancing scheduler for cloud computing.*Cluster Computing vol. 21, no. 3, pp. 1667-1680.

17. Kattan, A;Fatima, S.(2012).*Pso as a meta-search for hyper-ga system to evolve optimal agendas for sequential multi-issue negotiation*. In 2012 IEEE Congress on Evolutionary Computation,Brisbane,Australia

18. Milani, S;Navin, H.(2015).*Multi-objective task scheduling in the cloud computing based on the patrice swarm optimization.*I.J. Information Technology and Computer Science,. vol. 5, pp. 61-66.

19. Zhou, Z;Li, F;Abawajy, J;Gao, C.(2020).*Improved PSO algorithm integrated with opposition-based learning and tentative perception in networked data centres.*IEEE Access. vol. 8, pp. 55872-55880.

20. Manasrah, A;Ali, H.(2018).*Workflow scheduling using hybrid GA-PSO algorithm in cloud computing.*Wireless Communications and Mobile Computing. vol. 2018, pp. 1-16.

21. Devaraj, S;Elhoseny, M;Lydia, L.(2020).*Hybridization of firefly and Improved Multi-Objective Particle Swarm Optimization algorithm for energy efficient load balancing in Cloud Computing environments.*Journal of Parallel and Distributed Computing. vol. 142, pp. 36-45.

22. Alkhashai, H;Omara, F.(2016).*BF-PSO-TS: hybrid heuristic algorithms for optimizing task schedulingon cloud computing environment.*International Journal of Advanced Computer Science and Applications. vol. 7, no. 6, pp. 207-212.

23. Saleh, H;Nashaat, H;Saber, W.(2018).*IPSO task scheduling algorithm for large scale data in cloud computing environment.*IEEE Access. vol. 7, pp. 5412-5420.

24. Khalili, A;Babamir, M.(2015).*Makespan improvement of PSO-based dynamic scheduling in cloud environment*. In 2015 23rd Iranian Conference on Electrical Engineering,Tehran,Iran

25. Luo, F;Yuan, Y.(2018).*An improved particle swarm optimization algorithm based on adaptive weight for task scheduling in cloud computing*. In Proceedings of the 2nd International Conference on Computer Science and Application Engineering,New York,USA

26. Eberhart, R;Kennedy, J.(1995).*A new optimizer using particle swarm theory*. In Proceedings of the Sixth International Symposium on Micro Machine and Human Science,Nagoya,Japan

27. Wang, D;Tan, D;Liu, L.(2018).*Particle swarm optimization algorithm: an overview.*Soft Computing. vol. 22, no. 2, pp. 387-408.

28. Calheiros, R;Ranjan, R;Beloglazov, A;De Rose, C;Buyya, R.(2011).*CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms.*Software: Practice and experience. vol. 41, no. 1, pp. 23-50.