المجلة العراقية للعلوم الاحصائية (17) 2010
عدد خاص بوقائع المؤتمر العلمي الثاني للرياضيات-الاحصاء والمعلوماتية
ص ص [261- 274]

حد أدنى هجيني لمسائل جدولة المهام المتعددة المعالجات في بيئة الورشة الانسيابية الهجينة المتعددة المراحل

احمد محمود السبعاوي* مناف حازم احمد مطرود**

الملخص

تم في هذا البحث تحسين الحد الأدنى المقدم من لدن Oğuz (حد ادنى هجيني), لقيم وقت التنفيذ المثلى لمسائل جدولة المهام المتعددة المعالجات في الورشة الأنسيابية الهجينة المتعددة المراحل والتي يمكن استخدامها في تقييم اداء خوارزميات الحل لهكذا مسائل ,وكما يمكن استخدام الحدود المحسنة حلولا مثلى حين تكون المسائل صغيرة الحجم.

وقد أظهرت النتائج بعد التطبيق على المسائل المولدة عشوائيا مدى كفاءة هذه الحدود في تقدير الحل الامثل لقيمة وقت التنفيذ المثلى اذ اعطت قيمة قريبة من الحل الامثل .

A hybrid lower bound for multi-processor task scheduling problems in multi-stage hybrid flow shop environment

ABSTRACT

In this paper, a lower bound for the optimal makespan value have been improved (Hybrid lower bound) which can be used to evaluate the performance algorithm for multi-processor task scheduling problems in multi-stage FSMP(flow shop with multi-processor).

So, the values of hybrid lower bounds can be used as an optimal solution when the problems are small size.

After applying the hybrid bounds, the result has shown that the bounds are efficient in estimating the optimal makespen value.

---

* استاذ مساعد/ قسم الاحصاء والمعلوماتية/ كلية علوم الحاسبات والرياضيات

** مدرس مساعد/ قسم الاحصاء والمعلوماتية/ كلية علوم الحاسبات والرياضيات

تاريخ التسلم :2009/7/1 ـــــــــــــــــــ تاريخ القبول : 6 /12 /2009

In addition, the high competence of this program in calculating the lower bound and execution speed are proofed

1- مقدمة

ان مسائل الورشة الانسيابية الهجينة تجمع بين خصائص مسائل الورشة الانسيابية والماكنات المتوازية ، ويطلق على هذه المسائل ايضا بمسائل الورشة الانسيابية المتعددة المعالجات(flow shop with multi-processor or hybrid flow shop)(FSMP).

في مسائل الورشة الانسيابية(flow shop) الأعمال I_i تمر بالمراحل بالترتيب نفسه للماكنات, إذ يوجد هناك ماكنة واحدة عند كل مرحلة ، أما مسائل الورشة الانسيابية الهجينة(hybrid flow shop) فيوجد في كل مرحلة ماكنة واحدة او اكثر من الماكنات المتوازية والمتماثلة. وهذا يعطي مرونة إضافية في تخطيط الانتاج وتقليل أوقات الانتظار بين الأعمال [(1)] .

وبذلك تغلبت مسائل الورشة الانسيابية الهجينة على احد محددات مسائل الورشة الانسيابية التقليدية بإضافة الماكنات المتوازية والمتماثلة .

أما مسائل المهام المتعددة المعالجات فتغلبت أيضا على محددات الماكنات المتوازية التقليدية ، بمعالجة المهام بأكثر من ماكنة واحدة وبصورة آنية .

وبدمج مسالة الورشة الانسيابية الهجينة ومسألة المهام المتعددة المعالجات تتكون مسائل الورشة الانسيابية الهجينة بمهام متعددة المعالجات ، أذ تم برهان ان وقت التنفيذ لهذه المسالة من دون قيود الأسبقية وبافتراض أن جميع أوقات المعالجة محددة قابلة للحل على شكل متعدد حدود (Polynomial) في حين وجود قيود الأسبقية فان المسالة تكون (NP-hard) حتى عند أبسط المسائل[(7)].

وتعرف مسالة الورشة الانسيابية الهجينة بمهام متعددة المعالجات كالآتي :

على إفتراض المجموعة I تحتوي على (n) من الأعمال . $I=\{1,\ldots,n\}$ ، كل عمل فيها يعالج بـ (m) من المراحل .وكل عمل في هذه المجموعة يحتوي على (m) من المهام ، كل مهمة تعالج عند كل مرحلة ، ولا يمكن أن تبدأ معالجة أية مهمة الا بعد الانتهاء من معالجة المهمة التي سبقتها ، علما ان كل مهمة تتطلب معالجتها عدد من الماكنات وبصورة آنية . في حالة كون الأعمال جاهزة عند الوقت $t = o$.

الماكنات عند كل مرحلة متماثلة ولايمكن ان تعالج هذه الماكنات المهام المرتبطة بأيـة مرحلة اخرى ، وبافتراض ان (m_j) هي عدد الماكنات في المرحلة j ، وان Size(i,j) هـي عدد الماكنات التي يتطلبها العمل للمعالجة عند المرحلة j.

تبدأ الماكنات التي عددها مساوٍ لـ Size(i,j) بمعالجة العمل I_i لفترة طولهـا P(i,j) ، إن إمكانية القطع غير مسموح بها في أثناء المعالجة. ويمكن القول بـان Size(i,j) و P(i,j) هي متطلبات المعالجة ووقت المعالجة للمهمة O_{ij} في العمل I_i عند المرحلة j على التوالي .

لايمكن ان تعالج كل ماكنة أكثر من عمل واحد في الوقـت نفـسه . كـون جميـع الماكنات عند كل مرحلة متماثلة وامكانية القطع غير مسموح بها.

وباسـتخدام الحقـول الثلاثـة $\alpha/\beta/\gamma$ يمكـن ان نرمـز لهـذه المـسالة بـ $Fm(Pm_1,\ldots Pm_m)/Size(i,j)/C_{max}$ [(2)].

2- الحد الأدنى (LB) Lower bound

في معظم مسائل الـ(NP-hard) من الصعب تقييم اداء الخوارزميات التي تحل هـذا النوع مـن المـسائل ، التـي تعتمـد علـى إيجـاد الحلـول المثلـى . وأن مـسالة الـ $Fm(Pm_1,\ldots,Pm_m)/Size(i,j)/C_{max}$ تقع تحت هذا الصنف من المسائل[(4)],[(5)] .

يعد الحد الأدنى (LB) اداة فعالة في تقدير وقت التنفيذ الامثل لهذه المسائل حين يكـون الحل لهذه المسائل غير معلوم[(10)]. كما تستخدم الحدود الدنيا في ايجاد الحلول المثلى للمـسائل الصغيرة والمتوسطة الحجم[(6)].

يعرف الحد الأدنى العام (LB) لقيم وقت التنفيـذ المثلـى (Optimal makespan) ، على انه أعظم حد أدنى من بين الحد الأدنى الذي يعتمد علـى المرحلـة (LBs) (Lower bound based stage) والحـد الأدنـى الـذي يعتمـد علـى العمـل (LB_O) (Lower bound based job)[(6)], [(8)], [(9)] .

أي أن : $LB = \max\{LB_O , LBs\}$

ويعرف الحد الأدنى الذي يعتمد على المرحلة (LBs) ، على انه أعظم الحـدود الـدنيا للمراحل جميعا ((LB(j)) بحيث $j = 1, \ldots, m$

$$LBs = \max_{J}\{LB(j)\}$$

$$LB(j)=\{\min_{I}\sum_{j'=1}^{j-1}p(i,j')\}+\max\{x_1(j),x_2(j)\}+\{\min_{I}\sum_{j'=j+1}^{m}p(i,j')\}$$

أذ ان:

$$x_1(j)=\frac{1}{m_j}\sum_{I}P(i,j)\,size(i,j)$$

حيث ان $x_1(j)$ هي طريقة حساب وقت المعالجة عند المرحلة (j) على اعتبار ان المسألة هي $P//C_{max}$ (11)

$$x_2(j)=\sum_{I\in A_j}P(i,j)+\left\lceil\frac{1}{2}\sum_{I\in B_j}P(i,j)\right\rceil$$

وان $x_2(j)$ هي طريقة حساب وقت المعالجة عند المرحلة (j) حـسب فرضـية الباحـث Oğuz (6)

$$A_j=(i/size(i,j)>\frac{m_j}{2})$$

$$B_j=(i/size(i,j)=\frac{m_j}{2})$$

اما الحد الأدنى الذي يعتمد على العمل (LB_O) فيمكن الحصول عليه بالطريقة نفـسها المستخدمة في مسائل الورشة الانسيابية ، وذلك باخذ اكبر مجموع أوقات المعالجة لمجموعـة الأعمال من المرحلة الاولى الى المرحلة الاخيرة.

$$LBo=\max_{I}\left\{\sum_{j=1}^{m}P(i,j)\right\}$$

3- تحسين الحد الادنى (الحد الأدنى الهجيني) (NLB)

The improvement of the lower bound (Hybrid lower bound)

في هذِهِ الفقرة سوف يتم تحسين الحد الادنى اعلاه المقدم من لـدن Oğuz (6) الـذي افترض بأن المهام التي لها متطلبات معالجة اقل من نصف عدد الماكنات تعالج آنيا مع المهام التي لها متطلبات معالجة اكبر من نصف عدد الماكنات (أي بالماكنات غير المستخدمة من قبل هذه المهام) لا تؤثر في قيمة الحد الأدنى.

ولكن في الحقيقة ان هذه المهام التي لها متطلبات معالجة اقل من نصف عدد الماكنـات لايمكن معالجتها جميعا بالماكنات غير المستخدمة ، مما يجعل الافتراض اعلاه غير دقيق في بعض الاحيان.

بعبارة اخرى من المحتمل أن عدد المكائن غير المستخدمة من قبل المهـام التـي لهـا متطلبات معالجة اكبر من نصف عدد الماكنات لا تكفي لمعالجة جميع المهام التي لها متطلبات معالجة اقل من نصف عدد الماكنات ، مما يؤدي الى معالجة بعض هذه المهام آنيا والـبعض الأخر لا يمكن معالجته ، ولهذا فان وقت المعالجة لهذه المهام سيضاف الى وقـت المعالجـة الذي حسب بالطريقة $x_2(j)$ ، أي سوف يؤثر في قيمة الحد الأدنى (تزداد قيمة الحد الأدنى).

وبالاعتماد علـى هـذا الافتـراض تـم تحـسين الحـد الأدنـى (NLB) لمـسائل الـ $Fm(Pm_1,\ldots,Pm_m)/Size(i,j)/C_{max}$ الذي يعالج هذه المشكلة إذ تـم حـساب وقـت المعالجة للمرحلة (j) بصورة دقيقة مع الاخذ بنظر الاعتبار عـدم كفايـة الماكنـات غيـر المستخدمة لمعالجة المهام التي لها متطلبات معالجة اقل من نصف عدد الماكنات.

الحد الهجيني NLB(j) المقابل للمرحلة (j) يدمج الطـريقتين $x_1(j)$ ، $x_2(j)$. إذ يـتم حساب وقت المعالجة عند المرحلة j بالطريقة $x_2(j)$ وبعدها تضاف نسبة معينة ولـتكن (λ). بحيث $0 \le \lambda \le 1$ من وقت معالجة المهام التي لها متطلبات معالجة اقل مـن نـصف عـدد الماكنات ويحسب هذا الوقت بالطريقة $x_1(j)$.

ولعدم معرفتنا أية من المهام التي لايمكن معالجتها بالماكنات غير المستخدمة فانه سوف يتم حساب وقت المعالجة لجميع المهام التي لها متطلبات معالجة اقل من نصف عدد الماكنات وبضرب قيمة وقت التنفيذ بـ (λ) التي هي عبارة عن عدد متطلبات المعالجة (التكـرارات) التي لايمكن معالجتها بالماكنات غير المستخدمة ويرمز لها بـ (P) مقسومة علـى مجمـوع متطلبات المعالجة للمهام التي لها متطلبات معالجة اقل من نصف عدد الماكنـات بـ ـ (Z_C) وبهذا يكون وقت المعالجة للمرحلة (j) على النحو الآتي :

$$x_3(j) = x_2(j) + \lambda f(j)$$

إذ أن :

$$f(j)=\frac{1}{m_j}\sum_{I\in c_j}P(i,j)\,size(i,j)$$

$$C_j=\left(i/size(i,j)<\frac{m_j}{2}\right)$$

$$\lambda_1=P/Z_C$$

أما أوقات التنفيذ للمراحل السابقة للمرحلة (j) واللاحقة للمرحلة (j) فتحسب بالطريقة نفسها المستخدمة في الفقرة 2.

وبهذا يكون الحد الأدنى الهجيني المقابل للمرحلة j ‏NLB(j) على النحو الآتي :

$$NLB(j)=\min_I\{\sum_{j'=1}^{j-1}p(i,j')\}+\max\{x_1(j),x_3(j)\}+\min_I\{\sum_{j'=j+1}^{m}P(i,j')\}$$

وبالطريقة السابقة نفسها يكون الحد الأدنى الذي يعتمد على المرحلة NLBs مساوياً لـ

NLBs = max {NLB(j)}

ويحسب الحد الأدنى الذي يعتمد على العمل ايضا بالطريقة السابقة نفسها، اذن الحد الأدنى الهجيني العام المقترح NLB.

NLB = max {LBo ,NLBs}

4- الخوارزمية المصممة للحد الأدنى الهجيني
Design algorithm for Hybrid lower bound

الخطوة (1) : ادخل (n) الذي يمثل عدد الأعمال ، و (m) الذي يمثل عدد المراحل ، و (m_j) الذي يمثل عدد الماكنات عند المرحلة j.

الخطوة (2) : اجعل j = 1.

الخطوة (3) : ايجاد وقت المعالجة (u) للمراحل التي تسبق المرحلة j.

$$u=\min_I\left\{\sum_{j'=1}^{j-1}P(i,j')\right\}$$

الخطوة (4) : إيجاد وقت المعالجة (w) للمراحل التي تلي المرحلة j.

$$w=\min_I\left\{\sum_{j'=j+1}^{m}P(i,j')\right\}$$

الخطوة (5) : إيجاد وقت المعالجة عند المرحلة j بالطريقة $x_1(j)$.

$$x_1(j) = \frac{1}{m_j}\sum_{i=1}^{n} P(i,j)\, size(i,j)$$

الخطوة (6) : تصنيف الأعمال عند المرحلة j حسب المجموعات الثلاث.

$$A_j = \left(i / size(i,j) > \frac{m_j}{2} \right)$$

$$B_j = \left(i / size(i,j) = \frac{m_j}{2} \right)$$

$$C_j = \left(i / size(i,j) < \frac{m_j}{2} \right)$$

الخطوة (7) : ايجاد وقت المعالجة عند المرحلة j بالطريقة $x_2(j)$.

$$x_2(j) = \sum_{I \in Aj} P(i,j) + \left[\frac{1}{2} \sum_{I \in Bj} P(i,j) \right]$$

الخطوة (8) : ايجاد عدد الماكنات غير المستخدمة من قبل الأعمال التي تنتمي الى المجموعة A_j، ولإيجاد عدد هذه الماكنات يمكن اتباع الخطوات الآتية :

1. إيجاد عدد الأعمال (Na) التي تنتمي للمجموعة A_j وضربها في عدد الماكنات بالمرحلة (m_j).

$r = Na * m_j$

2. ايجاد مجموع متطلبات المعالجة Z_A للأعمال التي تنتمي الى المجموعة A_j

$$Z_A = \sum_{I \in A_j} Size(i,j)$$

3. عدد الماكنات غير المستخدمة (q) هو حاصل الفرق ما بين r و Z_A

$q = r - Z_A$

الخطوة (9) : ايجاد مجموع متطلبات المعالجة Z_c للاعمال التي تنتمي الى المجموعة C_j

$$Z_C = \sum_{I \in C_j} size(i,j)$$

الخطوة (10) : اذا كانت $Z_C \le q$ فان $\lambda = 0$ ، اما اذا كانت $Z_C > q$ فهذا يعني ان الماكنات غير المستخدمة من قبل الأعمال التي تنتمي للمجموعة A_j لاتكفي لمعالجة جميع الأعمال التي تنتمي الى المجموعة C_j. وتحسب قيمة (λ) على النحو الآتي :

1. ايجاد عدد متطلبات المعالجة التي لايمكن معالجتها بالماكنات غير المستخدمة (P) والتـي هي عبارة عن حاصل الفرق بين عدد الماكنات الواجب توفرها لمعالجة هذه الأعمال (المهام) والتي تكون مساوية الى Z_C وعدد الماكنات غير المستخدمة .

$P = Z_C - q$

2. ايجاد قيمة (λ) بوساطة العلاقة الآتية :

$\lambda = P/Z_C$

الخطوة (11) : ايجاد وقت المعالجة f(j) عند المرحلة j للاعمال التي تنتمي الى المجموعـة C_j فقط وبالطريقة $x_1(j)$:

$$f(j) = \frac{1}{m_j} \sum_{I \in C_j} P(i,j)\, Size(i,j)$$

الخطوة (12) : ايجاد وقت المعالجة عند المرحلة j بالطريقة $x_3(j)$:

$X_3(j) = x_2(j) + \lambda .\ f(j)$

الخطوة (13) : ايجاد الحد الأدنى الهجيني NLB(j) المقابل للمرحلة j :

$NLB(j) = u + \max\ \{x_1(j)\ ,\ x_3(j)\} + w$

الخطوة (14) : اجعل $j = j + 1$

الخطوة (15) : اذا كانت $j \le m$ اذهب للخطوة (3).

الخطوة (16) : ايجاد الحد الأدنى الهجيني الذي يعتمد على المرحلة NLBs

$NLBs = \max\ \{NLB(j)\}$

الخطوة (17) : ايجاد الحد الأدنى الذي يعتمد على العمل

$$LBo = \max_I \left\{ \sum_{j=1}^{m} P(i,j) \right\}$$

الخطوة (18) : الحد الأدنى الهجيني العام

$NLB = \max\ \{LBo\ ,\ NLBs\}$

5- **توليد البيانات** [11], [8], [6],[3]

لـ ـــصعوبة الحـ ـــصول علـ ـــى بيانـ ـــات تلائـ ـــم مـ ـــسالة الـ ـــ ـــ ـ $Fm(Pm_1,...Pm_m)/Size(i,j)/C_{max}$ تم توليد البيانات للمسائل اعلاه عشوائيا بالاعتماد على بعض الدراسات السابقة.

ولغرض اعطاء شمولية في المسائل المولدة واختبار الحدود الدنيا المحسنة في جميع حالات المسائل المولدة تم افتراض عدد الأعمال مساوياً لـ ـ ($n = 100$) وعـ ـدد المراحـ ـل مساوياً لـ ـ ($m = 5,10,25$) وكل تركيبة من (n) و(m) تمثل مسالة سوف يتم تطبيق الحدود الدنيا المحسنة عليها بحيث تم التوليد على أساس الفقرات الثلاث الآتية :

1. أوقات المعالجة للأعمال جميعها في جميـ ـع المـ ـسائل تـ ـم توليـ ـدها عـ ـشوائيا مـ ـن التوزيع المنتظم (Uniform distribution) ضمن الفترة [1,40] بحيـ ـث ان جميـ ـع اوقات المعالجة اعداد صحيحة وموجبة ($j=1,....,m$) ($i=1,...,n$)

$$P(i,j) \sim U[1,40]$$

بحيث ان $p(i,j)$ عدد صحيح وموجب .

2. عدد الماكنات عند كل مرحلة j تم إيجاده بالصيغة الآتية :

$$m_j = 2^V$$

إذ أن V مجموعة من القيم مولدة عشوائيا من التوزيع المنتظم ضمن الفتـ ـرة مـ ـن [1,4] ويكون عدد المشاهدات المولدة مساوياً لعدد المراحل (m) في المسألة .

$$V \sim U[1,4]$$
$$V = \{ V_1 , V_2 ,.....V_m\}$$

بحيث ان جميع قيم V اعداد صحيحة وموجبة

3. كل عمل I_i تم توليد متطلبات المعالجة له Size(i,j) . عند كل مرحلة j عشوائيا من التوزيع المنتظم ايضا ضمن الفترة [1,mj] بحيث ان جميع متطلبات المعالجة اعـ ـداد صحيحة وموجبة .

$$Size(i, j) \sim U[1,m_j]$$

6- عرض النتائج والمناقشة

فيما يأتي النتائج التي تم التوصل إليها .

الجدول (6.1)

يبين نتائج تطبيق الحد الأدنى الهجيني الذي يعتمد على المرحلة لمسألة عدد مراحلها مساو لـ 5 مقارنة بالحد الأدنى الذي يعتمد على المرحلة المقدم من لدن Oğuz (2005)

J	m_j	u_j	$X_1(j)$	$X_2(j)$	W_j	q	Zc	λ	λ f(j)	$X_3(j)$	LB(j)	NLB(j)
1	4	0	1283	1399	37	40	17	0	0	1399	1436	1436
2	8	1	1143	1129	15	81	92	0.12	28	1157	1159	1173
3	4	7	1207	1266	7	26	17	0	0	1266	1280	1280
4	16	11	1075	1033	2	158	213	0.26	70	1103	1088	1116
5	8	14	1292	1404	0	100	71	0	0	1404	1418	1418

LBS	LBO	NLB
1436	157	1436

الجدول (6.2)

يبين نتائج تطبيق الحد الأدنى الهجيني الذي يعتمد على المرحلة لمسألة عدد مراحلها مساو لـ 10 مقارنة بالحد الأدنى الذي يعتمد على المرحلة المقدم من لدن Oğuz (2005)

J	mj	Uj	X1(j)	X2(j)	Wj	q	Zc	λ	λf(j)	X3(j)	LB(j)	NLB(j)
1	8	0	1017	977	100	74	89	0.17	36	1013	1117	1117
2	16	1	969	961	87	154	179	0.14	26	987	1057	1075
3	16	9	1047	1013	74	177	212	0.17	45	1058	1130	1141
4	16	17	1108	1036	64	154	244	0.37	109	1145	1189	1226
5	4	28	1316	1378	47	32	21	0	0	1378	1453	1453
6	8	54	1166	1185	27	74	87	0.15	28	1213	1266	1294
7	8	66	1305	1361	14	87	68	0	0	1361	1441	1441
8	4	75	1250	1329	8	36	16	0	0	1329	1412	1412
9	8	93	1003	1078	2	97	71	0	0	1078	1173	1173
10	4	113	1347	1463	0	33	12	0	0	1463	1576	1576

LBS	LBO	N1LB
1576	289	1576

الجدول (6.3)

يبين نتائج تطبيق الحد الأدنى الهجيني الذي يعتمد على المرحلة لمسألة عدد مراحلها مساو لـ 25 مقارنة بالحد الأدنى الذي يعتمد على المرحلة المقدم من لدن Oğuz (2005)

J	m_j	U_j	$X_1(j)$	$X_2(j)$	W_j	q	Zc	λ	λ f(j)	$X_3(j)$	LB(j)	NLB(j)
1	8	0	1181	1236	358	90	61	0	0	1236	1594	1594
2	16	1	1155	1103	350	174	218	0.20	59	1162	1506	1513
3	2	7	1586	1586	328	0	0	0	0	1586	1921	1921
4	2	27	1579	1579	303	0	0	0	0	1579	1909	1909
5	4	44	1209	1316	282	40	12	0	0	1316	1642	1642
6	2	51	1617	1617	277	0	0	0	0	1617	1945	1945
7	16	60	1075	1075	268	168	192	0.13	29	1104	1403	1432
8	4	84	1231	1274	251	30	20	0	0	1274	1609	1609
9	16	103	1054	1101	243	185	180	0	0	1101	1447	1447
10	16	107	1110	996	234	147	216	0.32	94	1090	1451	1451
11	8	130	1331	1367	215	95	77	0	0	1367	1712	1712
12	4	151	1259	1286	183	23	16	0	0	1286	1620	1620
13	4	160	1324	1399	179	28	20	0	0	1399	1738	1738
14	2	170	1448	1448	162	0	0	0	0	1448	1780	1780
15	2	194	1647	1647	125	0	0	0	0	1647	1966	1966
16	8	220	1146	1145	111	81	78	0	0	1145	1477	1477
17	16	226	1133	1087	103	147	221	0.33	93	1180	1462	1509
18	16	261	1213	1364	77	238	120	0	0	1364	1702	1702
19	4	274	1344	1323	62	19	25	0.24	34	1357	1680	1696
20	16	303	1167	1175	53	163	175	0	15	1190	1531	1546
21	4	311	1419	1477	33	30	17	0.7	0	1477	1821	1821
22	4	327	1335	1386	10	28	21	0	0	1386	1723	1723
23	2	336	1469	1469	9	0	0	0	0	1469	1814	1814
24	8	348	1155	1135	1	87	82	0	0	1135	1504	1504
25	4	350	1328	1440	0	40	12	0	0	1440	1790	1790

LBS	LBO	N1LB
1966	668	1966

الحالة (1):

يتضح في الجدول (6.2) عند المرحلة الاولى (j=1) ان الطريقة $X_2(1)$ كانت غير دقيقة في حساب وقت المعالجة عند المرحلة (1) بسبب فرضيتها القائلة بأن الأعمال التي تنتمي الى المجموعة C_j تعالج آنياً. هذا واضح من خلال قيمتي (q) و (Z_c) إذ أن عدد

الماكنات الواجب توافرها لمعالجة الأعمال التي تنتمي الى المجموعة C_j هي (89) في حـين ان الماكنات غير المستخدمة (q) هي (74).

وبذلك تصبح الطريقة $X_3(1)$ هي الادق في حساب وقت المعالجة عند المرحلة نفسها ، إذ أعطت هذه الطريقة زيادة بوقت المعالجة . هذه الزيادة بوقت المعالجة هي وقت المعالجـة للاعمال التي لايمكن معالجتها بالماكنات غير المستخدمة. إذ ان مقدار هذه الزيادة هو $f_1=36$ وحدة زمنية .

ولكن المرحلة (j=1) هي اقرب إلى مسالة الماكنات المتوازية $P//C_{max}$ وهذا واضـح من خلال ان قيمة $X_1(1)$ هي اكبر من$X_3(j)$ ($X_1(1)> X_3(1)$) ومن ثم فان الحـد الادنـى الهجيني المقابل للمرحلة (1) ((NLB(1)) يكون مساوياً للحد الأدنى ((LB(1)) والمقـدم مـن لدن Oğuz (2005) [6].

الحالة أعلاه تعزز من قوة الحد الأدنى الهجيني ((NLB(j)) وذلك لعدم تجاهله للطريقة $X_1(1)$. إذ يمكن ملاحظة هذه الحالة نفسها في الجدول (3) عند المرحلة (j=10).

الحالة (2):

يتضح في الجـداول (6.1) و (6.2) و (6.3) عنـد المراحـل j=2,4 و j=2,3,4 و j=2,17,19 على التوالي ، ان الطريقة التي استخدمت في حساب الحد الادنى هـي $X_1(j)$ ، وهذا واضح لكون قيمة $X_1(j)$ اكبر من $X_2(j)$ أي تعد المراحل أعلاه هي مسائل مـن نـوع ($P//C_{max}$) .

ولكن من خلال قيمتي (q) و (Z_C) ، نلاحظ أن المراحل أعلاه هـي ليـست بمـسائل ($P//C_{max}$) ولكن الفرضية القائمة على اساسها الطريقة $X_2(j)$ ، قد تسبب في اسـتبعاد هـذه الطريقة ، في حين ان الطريقة $X_3(j)$ تغلبت على هذه الحالة وأعطـت وقـت معالجـة أدق لهذه المراحل أعلاه. مما تم استخدامها في الحد الادنى الهجيني I لكـون قيمـة $X_3(j)$ اكبـر من $X_1(j)$.

الحالة (3):

أما الحالة الظاهرة في الجدولين (6.2) و (6.3) عنـد المراحـل j=6 و j=18,20 على التوالي : فهي أن الطريقة المحتسبة في ايجاد وقت المعالجة عند المرحلة j هـي $X_2(j)$

ولكن الفرضية القائمة على أساسها هذه الطريقة تجاهلت اوقات المعالجة للاعمال التي لايمكن معالجتها بالماكنات غير المستخدمة . هذا واضح ايضا من خلال قيمتي (q) و (z_c) . في حين $X_3(j)$ تغلبت على هذه الحالة وأضافت أوقات المعالجة للإعمال التـي لا يمكـن معالجتهـا بالماكنات غير المستخدمة إلى قيمة $X_2(j)$ وكما هو ملاحظ في المراحل أعلاه .

الحالة (4) :

يمكن الملاحظة من خلال الجداول الثلاثة المذكورة انفا أن قيمة كل مـن الطرائـق الثلاث $X_1(j)$ و$X_2(j)$ و$X_3(j)$ تتساوى حين يكون عدد الماكنات في المرحلـة يـساوي 2 $(m_j=2)$ وغالباً ما تكون الحدود الدنيا المقابلة للمراحل التي فيها عدد الماكنات يساوي 2 هي الحدود الدنيا التي تعتمد على المرحلة .

ويمكن حساب هذه الحدود الدنيا المقابلة للمراحل اعلاه باي من الطرائق الـثلاث . وفيما يأتي الأشكال الثلاثة التي توضح وقت المعالجة للمرحلة j المحسوبة بالطرائق الـثلاث $X_1(j)$ و$X_2(j)$ و$X_3(j)$ للمسائل التي عدد أعمالها يساوي 100 وبالمراحل $j = 5,10,25$

7- الاستنتاجات Conclusions

1. حين يكون عدد الماكنات في المرحلة j يساوي 2 $(m_j=2)$. عنـدها تتـساوى قـيم الحدود الدنيا المقابلة للمرحلة $LB(j) = NLB(j)$.
2. عند أية مرحلة من مراحل المسألة التي لها اقل عدد ماكنات غالبا مـا يكـون الحـد الأدنى المقابل لتلك المرحلة هو الحد الأدنى الذي يعتمد على المرحلة (LB_S) والحـد الأدنى العام (LB) ولجميع الحدود الدنيا المقدمة من لدن Oğuz (2005) والمحسنة
3. تم التغلب على الصعوبات الحسابية المرافقة للمسائل الكبيرة عند ايجاد الحدود الـدنيا بالبرنامج الذي تم عمله ، إذ اظهر هذا البرنامج كفاءه عالية في حساب هذه الحـدود وباوقات قياسية .
4. إذا كانت مسألة الـ $Fm(Pm_1,\ldots Pm_m)/Size(i,j)/C_{max}$ بمرحلة واحدة أي مـسالة $P/Size(i,j)/C_{max}$ ، فيمكن استخدام الخوارزمية المصممة للحد الادنى المحسن فـي حساب وقت التنفيذ (makespan) لهذه المسالة . بدلا من الخوارزميـات الأخـرى (الحدسية ، الجينية).

المصادر

1. Andre's, C., Gómez P., and García, Sabter, J.P. (2006), "Comparing dispatching rules in dynamic hybrid flow -shop", ***IEEE.***

2. Bruker, P. (2007), ***"Scheduling algorithms"***, Springer, Germany.

3. Etiler, O. Toklu, B. and Wilson, J. (2004), "A genetic algorithm for flow-shop scheduling problems", Journal of the Operational Research Society, Vol.55, P.P. 830-835.

4. Jatinder, N. and Gupta, D. (1988), "Two-stage hybrid flow-shop scheduling problem", ***Journal of the Operational Research Society***, Vol. 39, No.4, P.P. 359-364.

5. Kis, T. and Pesch, E. (2005), "A review of exact solution methods for the non-preemptive multiprocessor flow-shop problem", ***European Journal of Operational Research***, Vol. 164, P.P. 592-608.

6. Oğuz, C. and Fikret Ercan, M. (2005), "A genetic algorithm for hybrid flow-shop scheduling with multiprocessor tasks", ***Journal of Scheduling***, Vol. 8, P.P. 323-351.

7. Oğuz, C., Zinder, Y., Do, V.H. and Janiak, A. (2004), "Hybrid flow-shop scheduling problems with multiprocessor task systems", ***European Journal of Operational Research***, Vol. 152, PP. 115-131.

8. Oğuz, C., Eikret Ercan, M., Edwin Cheng, T.C., and Fung, Y.F. (2003), "Heuristic algorithms for multiprocessor task scheduling in a two-stage hybrid flow-shop", ***European Journal of Operational Research***, Vol. 149, P.P. 390-403.

9. Portmann, M.C., Vignier, A., Dardilhac, D. and Dezalay, D. (1998), "Branch and bound crossed with GA to solve hybrid flow-shop", ***European Journal of Operational Research***, Vol. 107, P.P. 389-400.

10. Santos, D.L., Hunsucker, J.L. and Deal, D.E. (1995), "Global lower bounds for flow shop with multiple processor", ***European Journal of Operational Research***, Vol. 80, P.P. 112-120.

11. Serifoğlu, F.S. and Ulusoy, G. (2004), "Multiprocessor task scheduling in multi stage hybrid flow-shop: a genetic algorithm approach", ***Journal of the Operational Research Society***, Vol. 55, P.P. 504-512.