أساليب جديدة لتطوير طرق متوازية للبحث المباشر في الامثلية العددية

أ.د. بشير محمد صالح خلف
قسم علوم الحاسبات / كلية التربية
جامعة الموصل

فراس محمود سعيد
قسم الرياضيات / كلية التربية
جامعة الموصل

الاستلام 2011 / 05 / 11

القبول 2011 / 10 / 05

Abstract

The Purpose of this research is the development Direct Search Methods for minimizing the objective function; $f: R^n \to R$
The Developed Methods reduces the approach time to the minimum through two techniques:

1) by reducing the number of function evaluation.
2) by developing algorithms suitable for running on MIMD(Multiple Instruction Stream Multiple Data Stream).

We developed two methods, the first method depends on the "pattern search" and the other depends on the "direct search" technique. The comparison of results shows the new algorithms are faster than corresponding algorithms.

الخلاصة

هدفنا في هذا العمل هو تطوير خوارزميات متوازية لطرق البحث المباشر لإيجاد أقل أو اكبر قيمة للدالة $f: R^n \to R$ والتي تحتاج حساب قيمتها الى وقت طويل.

أن هذا التطوير يعتمد بصورة أساسية إلى الوصول إلى الحل الأمثل بوقت قصير.

أ- عن طريق تقليل عدد مرات إيجاد الدالة f.

ب- تطوير طرائق متوازية مناسبة لحاسبات من نوع MIMD.

تم تطوير نوعين من الخوارزميات، الأولى تعتمد على أسلوب الحركة النموذجية Pattern (search) والثانية على أسلوب هووك وجيفس باستخدام طرائق البحث المباشر (Direct search method) حيث تم تطوير هاتين الخوارزميتين ومن النتائج التي حصلنا عليها ثبت إن الخوارزميتين الجديدتين أسرع في الوصول إلى النتائج من التي سبقتها.

1- المقدمة

في بحثنا هذا سوف نهتم في الخوارزميات المتوازية لحل مسائل الامثلية للقيم الصغرى غير المقيدة والتي بالصيغة: $\min_{x\in R^n} f(x): R^n \to R \quad \ldots\ldots \quad (1)$

عندما f هي دالة مستمرة وقابلة للاشتقاق ومن الدرجة ذات بعد n في R حقل الإعداد الحقيقية.

إن مسائل الامثلية غير المقيدة تتضمن تحديد عدة متغيرات والتي هي إما أن تكون قيم عظمى (Maximize) أو قيم صغرى (Minimize) للدوال لهذه المتغيرات. ونتيجة التطبيقات العملية لمسائل الامثلية غير المقيدة فان الكثير من الجهد قد تم بذله في تطوير الخوارزميات المتسلسلة لحلول هذه المسائل . ونظرا للتقدم الحديث في تقنيات الخوارزميات المتوازية أصبح بالإمكان حل مسائل الامثلية أكثر كفاءة [11].

إن طرق البحث المباشر لاتتطلب حساب مشتقات الدالة لذلك عادةً ما تكون بطيئة في الاقتراب من الطرق التي تعتمد على المشتقات. ولكنها أكثر فعالية في الحالات التي تكون فيها قيم الدالة موضع تشويش أو تحليل المشتقة غير متوفر أوتخمين الفروقات المنتهية للمشتقة غير موثوق به [1,6,8,11].

بالإضافة إلى ذلك فأن مخططات طرائق البحث المباشر تعطينا توازي جيد بالإضافة إلى إمكانية بنائها باستخدام الطرائق المتسلسلة [5,8,9,12] . لذلك فان هدفنا في هذا البحث هو تطوير طرائق بحث مباشرة تتلائم للعمل على حاسبات نوع MIMD.

إن هذه الحاسبات تحتوي على عدة معالجات وكل معالج له ذاكرة خاصة به ووحدة معالجة ، هذه المعالجات تتصل فيما بينها عن طريق شبكة اتصال ملائمة. إن هذه الحاسبات ملائمة وبصورة جيدة للخوارزميات المتوازية التي بالإمكان التعامل مع مسائلها بصورة منفصلة عن المعالجات الأخرى [8].

2- طرائق البحث المباشر Direct Search Methods

ان طرائق البحث المباشر كما وصفها هوك و جيفس [6] تتطلب فضاء من النقاط لتكن $p \in R^n$ حيث إن p=(x1,x2,…,xn) والتي تمثل النقاط المرشحة لحل مسألة الامثلية. لذلك إذا أمكن القول إن p1 هي "أفضل" نقطة مرشحة من p2 فسوف نكتب $(p_1 \subset p_2)$ لأي نقطتين في الفضاء، هنالك على نحو محتمل نقطة منفردة لتكن p^* هي الحل، وتحقق الخاصية $(p^* \subset p)$ لكل $\mathbf{p} \neq \mathbf{p}^*$. وأن الخوارزمية التالية توضح ذلك [8].

```
1. select a point B_o arbitrarily as the first "base point"
2. i=1
3. repeat
4. select a new point p_1
5. if p_i ⊂ B_{i-1} then
6. B_i = p_i
7.   else
8.   B_i = B_{i-1}
9.  endif
10. i=i+1
11. until No better point are found
Ensure  P* = B_i
```

في الخوارزمية السابقة فان أول خطوة هي عملية اختيار نقطة جديدة التي هي "قاعدة" يتم الانطلاق منها.

أن طرائق البحث المباشر يمكن أن تصنف إلى عدة أصناف:

- Random Search.
- Pattern Search methods.
- Simplex based methods.
- Method with adaptive sets of search directions.

إن كل صنف من الأصناف أعلاه يستند إلى قاعدة وفكرة هي إيجاد نقطة جديدة في فضاء البحث والانطلاق منها [8].

أن ما سيتم التركيز عليه من الأصناف أعلاه هو"pattern search" . حيث إن طريقة هوك وجيفس تندرج ضمن هذا التصنيف. ومن خلالها سنقوم بدراسة الطرق المتوازية وتطويرها.

3- طرائق البحث المباشر المتوازية Parallel Direct search method

أن طرائق البحث المتوازية مصممة لحل مسائل القيم الصغرى غير المقيدة :

$$\min_{x \in R^n} f(x) \quad \text{where} \quad f: R^n \to R$$

أن ما يميز طرق البحث المباشر من الأخرى في الامثلية أنها تتطلب دالة مستمرة فقط. وكذلك كما ذكرنا إنها لا تحتاج لحساب مشتقات الدالة وكون مخططات طرق البحث المباشر تعطينا توازي جيد على الرغم من أمكانية استخدام الطرق المتسلسلة [14]. لقد عرض المصدر [8]. خوارزميتان متوازيتان لحل طرائق البحث المباشر سنتطرق إلى ذكرهما وسنقترح خوارزميتان متوازيتان مطورة تعتمد الأولى على أسلوب "Pattern Search" والثانية على ما ذكره المصدر. الخوارزميات المتوازية المقدمة من قبل كل من Khalaf & Neama المصدر [8].

3-1- الخوارزمية المتوازية الأولى: The first Parallel algorithm

من الواضح من خلال خطوات خوارزميات طرق البحث المباشر أنها تمتاز باستقلالية العمليات لذلك فان كل عملية حساب لدالة سوف يمكن أن تعطى إلى واحد من معالجات الحاسوب MIMD. نلاحظ في هذه الخوارزمية إن عدد المعالجات المستخدمة هي 2n+1 حيث إن n هو عدد المتغيرات في المسالة ونستطيع حساب عامل السرعة (S_p) بواسطة [8].

$$S_p = \frac{\text{معدل وقت التنفيذ في معالج واحد}}{\text{معدل وقت التنفيذ ل } p \text{ من المعالجات}} \qquad \ldots\ldots(2)$$

p عدد المعالجات المستخدمة في العملية.

من الواضح انه بالإمكان حساب التالي:

$$f(x_1+h,x_2,\ldots x_n), f(x_1-h,x_2,\ldots,x_n), f(x_1,x_2+h,\ldots,x_n), f(x_1,x_2-h,\ldots,x_n),\ldots$$
$$\ldots\ldots, f(x_1,x_2,\ldots,x_n+h), f(x_1,x_2,\ldots,x_n-h)\ldots\ldots(3)$$

في معالجات مستقلة لذلك يمكن ان نحسب هذه العمليات بشكل متزامن إي في وقت واحد على جميع المعالجات ومن ثم يتم المقارنة بين القيم :-

$$f(x_1+h,x_2,\ldots,x_n) \text{ and } f(x_1,x_2,\ldots,x_n),$$
$$f(x_1-h,x_2,\ldots,x_n) \text{ and } f(x_1,x_2,\ldots,x_n),$$

وكذلك:-

$$f(x_1,x_2+h,\ldots,x_n) \text{ and } f(x_1,x_2,\ldots,x_n),$$
$$f(x_1,x_2-h,\ldots,x_n) \text{ and } f(x_1,x_2,\ldots,x_n),$$

وكذلك:-

$$f(x_1,x_2,\ldots,x_n+h) \text{ and } f(x_1,x_2,\ldots,x_n),$$
$$f(x_1,x_2,\ldots,x_n-h) \text{ and } f(x_1,x_2,\ldots,x_n),$$

سوف نحسب هذه العمليات كل منها مستقل عن الأخر وبالإمكان تنفيذها بخوارزمية متوازية .

وللسهولة سوف نفترض التالي:-

$$\left.\begin{aligned} p &= (x_1,x_2,\ldots,x_n) \\ \Delta p_i &= (x_1,x_2,\ldots,x_i+h,\ldots); \\ \nabla p_i &= (x_1,x_2,\ldots,x_i-h,\ldots); \end{aligned}\right\} \qquad \ldots\ldots\ldots(4)$$

وأن الخوارزمية التي سوف نستخدمها هي التالية:-

1- Choose an initial point (P_o).
2- Choose an initial step size (h).
3- Do in Parallel:-

cpu_1 : find $f(\Delta p_1)$
If $f(\Delta p_1) < f(p_1)$
$x_1 = x_1$ +h ; send x_1 to cpu_{2n+1}
Else
Get new (p_1) from cpu_{2n+1}

cpu_2 : find $f(\nabla p_1)$
if $f(\nabla p_1) < f(p_1)$
$x_1 = x_1$ -h ; send x_1 to cpu_{2n+1}
Else
Get new (p_2) from cpu_{2n+1}

cpu_3 : find $f(\Delta p_2)$
if $f(\Delta p_2) < f(p_2)$
$x_2 = x_2$ +h ; send x_2 to cpu_{2n+1}
Else
Get new (p_2) from cpu_{2n+1}

cpu_4 : find $f(\nabla p_2)$
if $f(\nabla p_2) < f(p_2)$
$x_2 = x_2$ -h ; send x_2 to cpu_{2n+1}
Else
Get new (p_2) from cpu_{2n+1}

.

.

cpu_{2n-1} : find $f(\Delta p_n)$
if $f(\Delta p_n) < f(p_n)$
$x_n = x_{n2}$+h ; send x_n to cpu_{2n+1}
Else
Get new (p_n) from cpu_{2n+1}

cpu_{2n} : find $f(\nabla p_n)$
if $f(\nabla p_n) < f(p_n)$
$x_n = x_{n2}$ -h ; send x_n to cpu_{2n+1}
Else
Get new (p_n) from cpu_{2n+1}

4- On cpu_{2n+1} :
Check if one of these steps yields to a smaller f(p).
$\Longrightarrow$ New iterate (P^*).
5- Otherwise: tray again with a step size (h=h/2).
6- As (P^*) approaches the solution, the algorithm reduces the length steps.
7- Stopping criteria: Step length falls below a certain tolerance.

الخوارزمية أعلاه تشرح طريقة توزيع المهام على عدة معالجات. أن عدد المعالجات في هذه الطريقة هو 2n+1 حيث إن n هو عدد المتغيرات [8].

ان الخوارزمية الثانية المقدمة من نفس المصدر [8] والتي يتم فيها اختزال عدد المعالجات المستخدمة والتي سيتم شرحها.

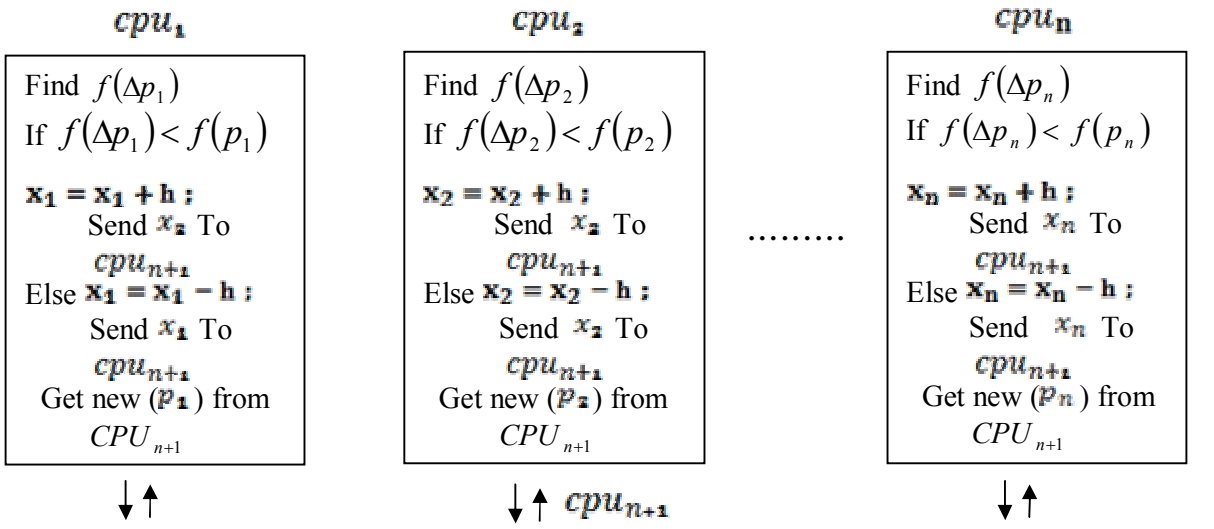
3-2- الخوارزمية المتوازية الثانية: The Second Parallel Algorithm.

هنا سيتم تقليل عدد المعالجات المستخدمة في الخوارزمية الأولى لتقليل الكلفة، في هذه الخوارزمية كل عملية حساب دالة يمكن إن تخصص إلى احد معالجات الحاسوب المتوازي *MIMD*. هذه الطريقة تحتوي على $n+1$ من المعالجات ويمكن أن تلخص الخوارزمية بالخطوات التالية:

- الخطوة الأولى: عند الـ cpu_1 سيقوم بحساب $f(\Delta p_1)$ ومن ثم يقارن هذه القيمة مع $f(p_1)$.

  إذا كان $f(\Delta p_1) < f(p_1)$ اذن نعتبر x_1=x1+h; وألا x_1=x1-h; نفس الإجراء يطبق على $cpu_n, \ldots, cpu_4, cpu_3, cpu_2$

وفي الخطوة التالية. فأن الـ cpu_{n+1} سوف يستلم القيم الجديدة من $cpu_n, \ldots, cpu_1$ اذا لم يكن هناك تخفيض في القيم الجديدة للدالة نقوم بتقليل الخطوة إلى النصف (h=h/2) ومن ثم نرسل قيمة h الجديدة إلى $cpu_n, \ldots, cpu_1$. أما إذا حصلنا على تخفيض في قيم الدالة الجديدة فنعتبر القيم الجديدة قواعد ونرسلها إلى $cpu_n, \ldots, cpu_1$ وتستمر العملية إلى حد الوصول إلى قيمة صغيرة للخطوة h مناسبة. والمخطط التالي يوصف الخوارزمية الثانية:



1. Receive the new values from $Cpu_1, \ldots, cpu_n$.
2. If no reduction on the function values, Then h=h/2,send h to $Cpu_1, \ldots, Cpu_n$.
3. Else: send new values to $Cpu_1, \ldots, Cpu_n$.
4. Until no better point are found.

شكل (1): توزيع المهام على المعالجات في الخوارزمية المتوازية [8]

الخوارزمية الأولى

1. نحتاج إلى 2n+1 من المعالجات مهما كان عدد المتغيرات.
2. البرنامج يأخذ وقت اقل من الخوارزمية الثانية عندما عدد المتغيرات يكون في ازدياد.
3. تعتبر الخوارزمية الأولى مكلفة جدا عندما تحل مسائل ذات أبعاد كبيرة.

الخوارزمية الثانية

1. تحتاج إلى n+1 من المعالجات.
2. البرنامج يأخذ وقت أطول من الأولى عندما يكون عدد المتغيرات في ازدياد.
3. تعتبر الخوارزمية الثانية غير مكلفة للمسائل المعقدة وذلك لان كل المسائل تحتاج إلى n+1 من المعالجات.

4- أساليب تطوير الطرق المتوازية

سنقوم بطرح طريقتين متوازيتين مطورتين الأولى تعتمد على أسلوب "pattern move" والثانية تعتمد على أسلوب البحث المباشر"Direct search method". سنقوم باستعراض طريقة هووك وجيفس للحركة النموذجية"Hook and Jeeves pattern Search" كما في المصدر [6] حيث تتلخص الطريقة كالتالي:

- **الحركة الاستكشافية "Exploratory move":**

في هذه العملية فان كل الإحداثيات سوف تفحص ضمن دوره واحدة كما يلي.نتحرك بخطوة واحدة باتجاه الاحداثي (i) (وذلك بإضافة زيادة Δ الى المتغير x_i). إذا كان التحرك ناجح، إذن القيمة الجديدة للمتغير سوف تؤخذ.أما إذا لم تنجح عملية التحرك فسوف نأخذ الاتجاه السالب (وذلك بطرح ∇ من المتغير x_i). إذا كان هذا التحرك ناجح إذن هذه القيمة للمتغير هي التي سوف تؤخذ، ماعدا ذلك فأن قيمة المتغير x_i تبقى كما هي.

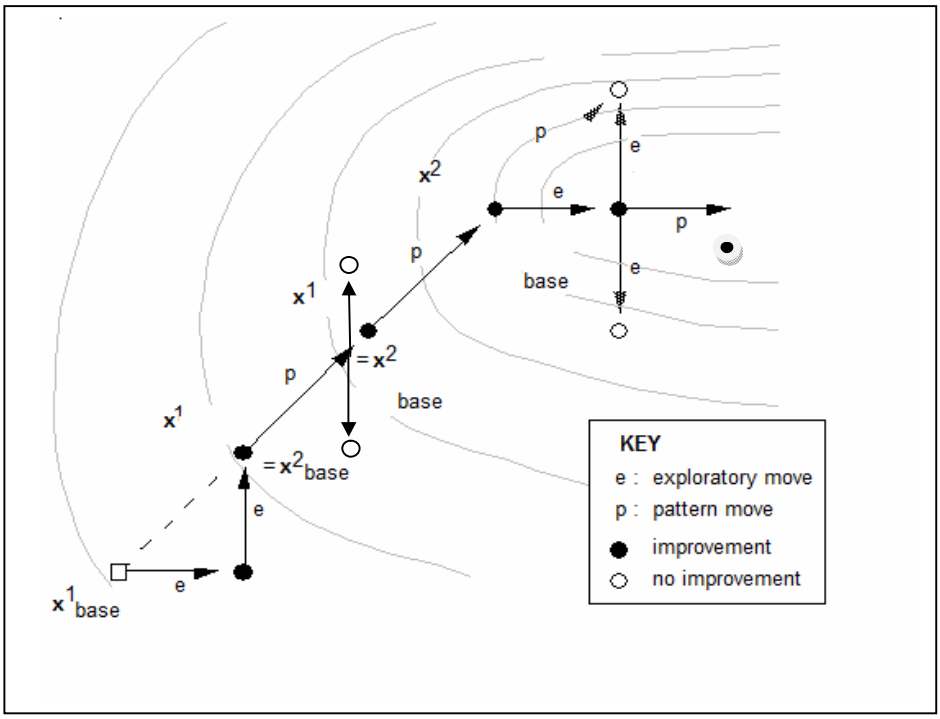
عندما يتم اختبار جميع المتغيرات التي عددها n فأن عملية الاستكشاف سوف تنتهي. إن النقطة التي نصلها نتيجة هذا الإجراء هي نقطة قد تكون أو لا تكون متميزة عن النقطة التي تحركنا منها، النقطة الجديدة تسمى قاعدة [9].

- **التحرك النموذجي "pattern move":**

نقطة القاعدة التي حصلنا عليها باستخدام الحركة الاستكشافية سوف تحدد اتجاه عملية التحرك النموذجي "pattern move" أو اتجاه البحث "Search Direction". إن التحرك النموذجي يأخذ شكل خطوات مفردة من النقطة الحالية "نقطة القاعدة" إلى اتجاه

معين يحدده قانون التحرك النموذجي. هذا الاتجاه عبارة عن نقطة جديدة سوف تصبح نقطة بداية جديدة إلى حركة استكشافية جديدة [1].

إذا فشل كل من التحرك النموذجي والحركة الاستكشافية فأن الخوارزمية تعود إلى نقطة القاعدة السابقة.إذا فشلت الحركة الاستكشافية حول هذه القاعدة أيضا فسوف يتم إلغاء الحركة النموذجية وان الزيادة Δ سوف يتم تقليلها. وعندها يتم إعادة جميع خطوات الخوارزمية حول هذه القاعدة.



شكل (2): مخطط طريقة هوك وجيفس للحركة النموذجية

4-1- الخوارزمية المطورة المقترحة الأولى: The first parallel algorithm

إن هذه الخوارزمية تعتمد على قانون الحركة النموذجية عند هووك وجيفس حيث في طريقة هووك وجيفس يتم استخدام هذا القانون بعد إجراء سلسلة من عمليات الاستكشاف ثم التحرك النموذجي كما تم شرحه في الطريقة. أما في الخوارزمية الجديدة فقد قمنا بدارسة الحركة النموذجية بعد إجراء سلسلة من العمليات تتلخص بالتالي:

بعد اختيار نقطة قاعدة (P_0) يتم حساب التالي:

$$P_2 = (x_1, x_2 + h, \ldots, x_n)$$
$$P_2' = (x_1, x_2 - h, \ldots, x_n)$$

$$\left.\begin{aligned} p_n &= (x_1, x_2, \ldots, x_n + h) \\ \mathbf{p_n'} &= (x_1, x_2, \ldots, x_n - h) \end{aligned}\right\} \quad \ldots\ldots\ldots(5)$$

بعد حساب هذه القيم قمنا بدراسة الحركة حسب قانون الحركة النموذجية بين تلك القيم وهل الحركة من القيمة الاوطأ للدالة إلى الأعلى أفضل أم العكس.تبين وبصورة عملية إن الحركة أو القفز باستخدام قانون الحركة النموذجية من القيمة الأعلى أفضل من القفز من القيمة الاوطأ إي أن الحركة باتجاه النقصان باستخدام القانون يقربنا من نقطة الـ min حيث يتم اعتبار النقطة التي يتم الوصول إليها نقطة قاعدة جديدة. عندها يتم إعادة نفس الخطوات السابقة على هذه النقطة الجديدة وباستمرار عملية القفز بين القيم سنصل إلى قيمة الـ min الحقيقة .بحيث بعد هذه النقطة لا نحصل على تقليل في قيمة الدالة عند إجراء عملية القفز. عندها نعمد إلى تقليل قيمة الخطوة h إلى قيمة صغيرة يتم التوقف عندها.

إن قانون الحركة النموذجية هو كالتالي:

$$P_i = b_i + 2(b_{i+1} - b_i) \qquad \text{.....} \ (6)$$

حيث لأي نقطتين b_2, b_1 وان $b_1 = (x_1, x_2, ..., x_n)$ و $b_2 = (x_1 + h, x_2, ..., x_n)$

فأن: $P_i \equiv$ إحداثيات النقطة الجديدة التي نريد الحصول عليها

$b_i \equiv$ إحداثيات النقطة الأولى (المنطلق)

$b_{i+1} \equiv$ إحداثيات النقطة الثانية (الهدف)

وهذا القانون هو قانون الحركة النموذجية المستخدم في مصدر [6] لكل من هووك وجيفس ويمكن تلخيص الخوارزمية المتوازية الجديدة في الخطوات التالية:

للسهولة سوف نفترض التالي:

$$\left.\begin{aligned} p &= (x_1, x_2, \ldots, x_n) \\ \Delta p_i &= (x_1, x_2, \ldots, x_i + h, \ldots); \\ \nabla p_i &= (x_1, x_2, \ldots, x_i - h, \ldots); \end{aligned}\right\} \quad \text{.....} \ (7)$$

1- Choose an initial point (P_o).
2- Choose an initial step size (h).
3- Do in Parallel:-

cpu_1 : find $f(\Delta p_1)$
If $f(\Delta p_1) < f(p_1)$
$x_1 = x_1$ +h ; send x_1 to cpu_{2n+1}
Else
Get new (p_1) from cpu_{2n+1}

cpu_2 : find $f(\nabla p_1)$
if $f(\nabla p_1) < f(p_1)$
$x_1 = x_1$ -h ; send x_1 to cpu_{2n+1}
Else
Get new (p_2) from cpu_{2n+1}

cpu_3 : find $f(\Delta p_2)$
if $f(\Delta p_2) < f(p_2)$
$x_2 = x_2$ +h ; send x_2 to cpu_{2n+1}
Else
Get new (p_2) from cpu_{2n+1}

cpu_4 : find $f(\nabla p_2)$
if $f(\nabla p_2) < f(p_2)$
$x_2 = x_2$ -h ; send x_2 to cpu_{2n+1}
Else
Get new (p_2) from cpu_{2n+1}

.
.

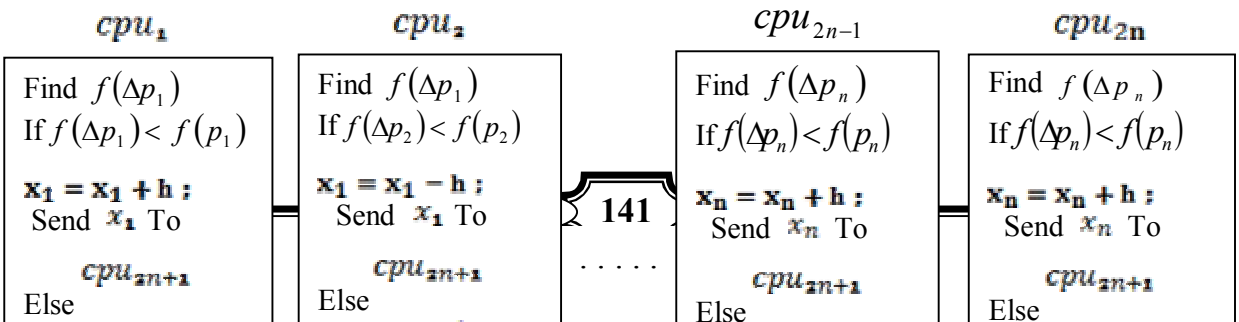
cpu_{2n-1} : find $f(\Delta p_n)$
if $f(\Delta p_n) < f(p_n)$
$x_n = x_{n2}$+h ; send x_n to cpu_{2n+1}
Else
Get new (p_n) from cpu_{2n+1}

cpu_{2n} : find $f(\nabla p_n)$
if $f(\nabla p_n) < f(p_n)$
$x_n = x_{n2}$-h ; send x_n to cpu_{2n+1}
Else
Get new (p_n) from cpu_{2n+1}

4- On cpu_{2n+1} :
Check if one of these steps yields to a smaller f(p).
Then make pattern move by:
$P_i = b_i + 2(b_{i+1} - b_i)$
Send new value P^* to $cpu_1, cpu_2, \ldots, cpu_{2n-1}, cpu_{2n}$
5- Otherwise: tray again with a step size (h=h/2).
6- As (P^*) approaches the solution, the algorithm reduces the length steps.
7- Stopping criteria: Step length falls below a certain tolerance.

من الملاحظ إن الخوارزمية الجديدة تستخدم معالجات عددها 2n لحل مسألة المحاور (الاتجاهات) حيث أن n هو عدد المتغيرات والمعالج 2n+1 هو المعالج الأخير الذي يقوم بأجراء الـ "pattern move" وكذلك يكون مسؤول عن تغيير طول الخطوة وتوقف الخوارزمية.

ءالمخطط التالي يوضح سير العمليات:



شكل (3): توزيع المهام على المعالجات في الخوارزمية المتوازية المقترحة الأولى

مثال (4−1):

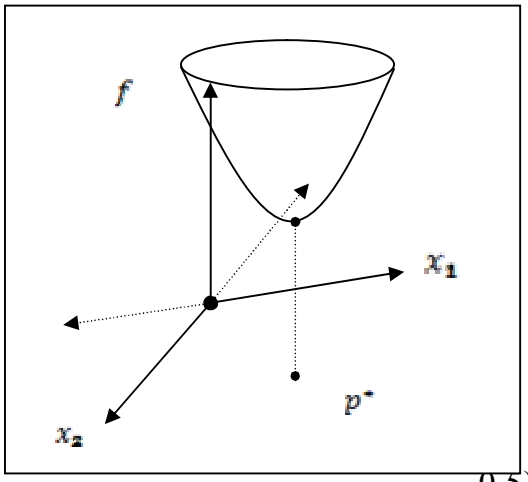
تم تصميم برنامج للخوارزمية المطورة المقترحة الأولى بلغة الماتلاب وتم تنفيذه على الحاسوب واستخدمنا المثال التالي:

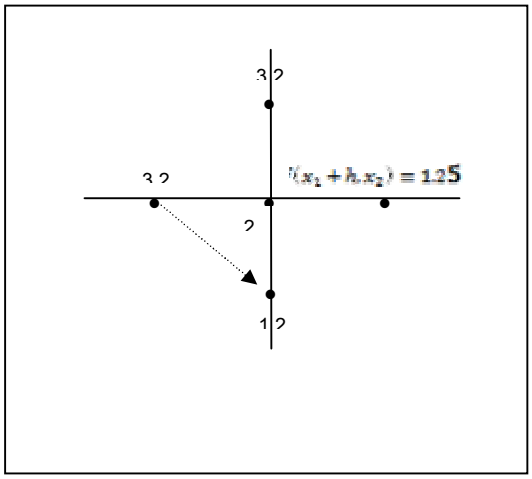
Find min of $f(x_1,x_2)=(x_1-1)^2+(x_2+1)^2$, $p_0=(0,0)$, **h=0.5**

Sol. $f(p_0)=2$

$f(x_1+h,x_2)=1.25$ at (0.5,0), $f(x_1-h,x_2)=3.25$ at (-0.5,0)

$f(x_1,x_2+h)=3.25$ at (0,0.5), $f(x_1,x_2-h)=1.25$ at (0,-0.5)





شكل (4): أسلوب القفز وافتراض الدالة

شكل (5): حساب قيمة الدالة عند الجهات الأربعة

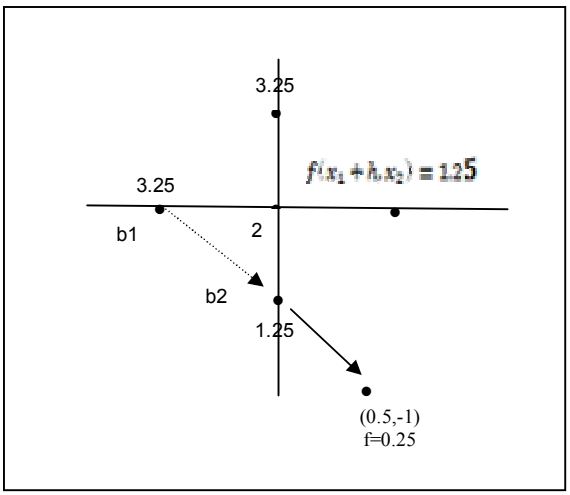
$b_1=$ و (-0.5,

النموذجي $P_i=b_i+2(b_{i+1}-b_i)$ سوف نستنتج $\mathbf{p_1}$ من الصيغة (6) كالتالي.

$$P_1 = b_1 + 2(b_2 - b_1)$$

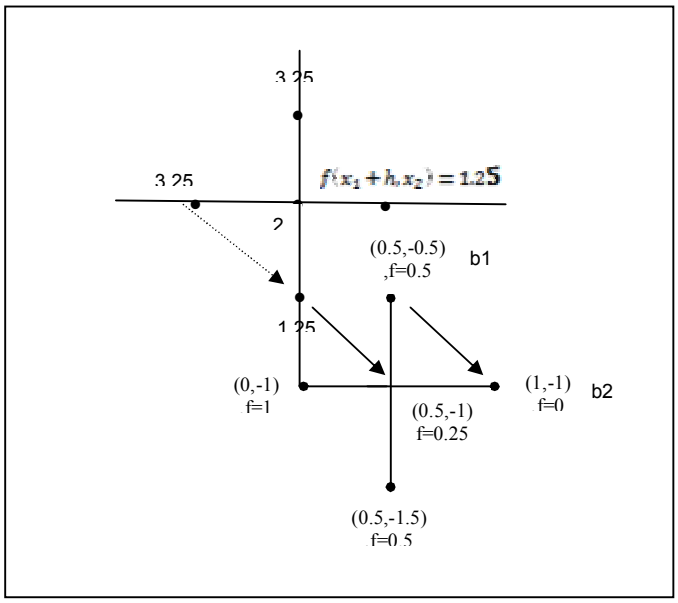
والناتج هو $P_1 = (0.5, -1)$ وهي القيمة الجديدة التي سنحسب قيمة الدالة عندها كالتالي f(0.5,-1)=0.25

وهذا يؤدي إلى تقليل في قيمة الدالة إي إن التحرك من القيمة الأعلى إلى الاوطأ تؤدي إلى نتيجة جيدة وحسب الخوارزمية سنعتبر هذه النقطة قاعدة جديدة ويتم حساب المحاور بزيادة h ونختبر النقاط كما في الشكل (6).



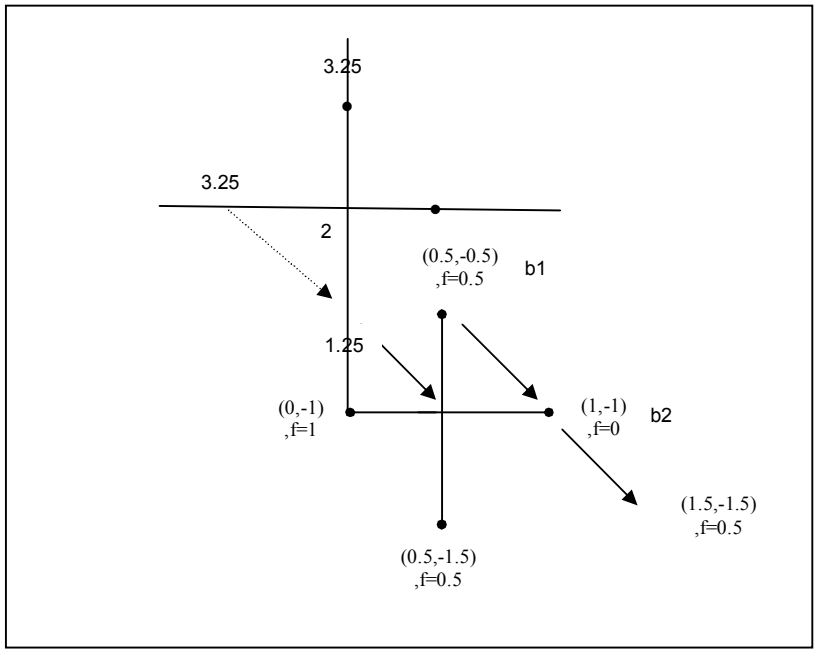
شكل (6): استنتاج نقطة جديدة بعد تطبيق قانون القفز

سنعتبر النقطة (0.5,-1) قاعدة للانطلاق بحيث يتم زيادة المحاور بمقدار (h) وبنفس الأسلوب نتفحص أماكن الزيادة والنقصان لتحديد وجهة التحرك النموذجي كما في الشكل (7):



شكل (7): إعادة تطبيق أسلوب الخوارزمية على النقطة الجديدة

سنعتبر النقطة (0.5,-1) قاعدة للانطلاق بحيث يتم زيادة المحاور بمقدار (h) وبنفس الأسلوب نتفحص أماكن الزيادة والنقصان لتحديد وجهة التحرك النموذجي كما في الشكل(8):



شكل (8): تطبيق القفز في النقاط الجديدة وتوضيح أسلوب التوقف

وألان سنقوم بالتحرك باعتبار $b_1=(0.5,-0.5)$ و $b_2=(1,-1)$, وبعد تطبيق قانون التحرك النموذجي $P_i = b_i + 2(b_{i+1} - b_i)$ سوف نستنتج $\mathbf{P_1}$ كالتالي :

$$P_1 = b_1 + 2(b_2 - b_1)$$

والناتج هو $P_1 = (1.5, -1.5)$ وهي القيمة الجديدة التي سنحسب قيمة الدالة عندها كالتالي

$f(1.5,-1.5)=0.5$

وهذا يؤدي إلى عدم وجود تقليل في قيمة الدالة لذلك نعمد إلى تقليل الخطوة *(h)* حسب الخوارزمية ونستمر بنفس الأسلوب إلى أن يتم التوقف عند قيمة الـ *min* الحقيقية كما في الشكل(8).

4-2- الخوارزمية المتوازية المطورة الثانية:

من الملاحظ في الخوارزمية المتوازية الأولى المطروحة من المصدر [8] Kahlaf&Neama انه يتم البحث عن النقصان في جميع الاتجاهات. والطريقة أو الخوارزمية المقترحة الجديدة تختلف عنها بأنها تبحث فقط في الجهات التي يحصل فيها النقصان. أي بمعنى ترك الجهات التي يحصل فيها زيادة في قيم الدالة والاستمرار في الجهات التي يحصل فيها نقصان. وتتضمن هذه الخوارزمية الاعتماد على مبدأ الاستكشاف "Exploration" حيث يتم العثور على جهة النقصان بعد التحرك بخطوة مقدارها (h) ومن ثم تتم عملية الاستكشاف حول النقطة التي حصل فيها النقصان. وفي حالة وجود نقصان نستمر باتجاه هذا النقصان دون

الرجوع إلى الجهة التي تم القدوم منها. وهذا يعتبر تحسين للطريقة السابقة حيث سوف يقل عدد عمليات حساب الدالة وبذلك يقل الزمن المستغرق في أيجاد أقل قيمة.

وللسهولة سوف نفترض التالي:

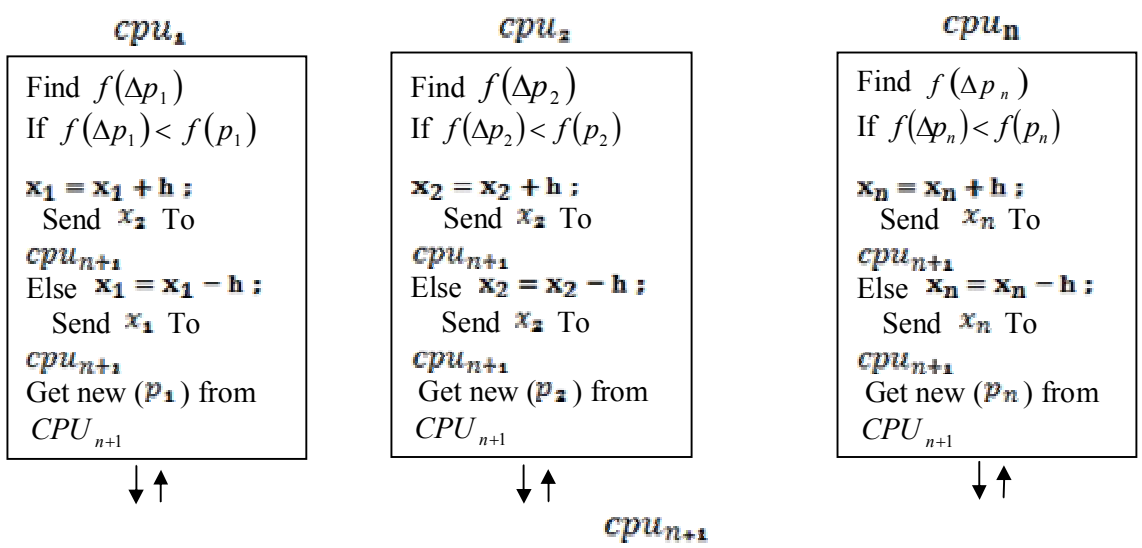
$$\left.\begin{aligned} p &= (x_1, x_2, \ldots, x_n) \\ \Delta p_i &= (x_1, x_2, \ldots, x_i + h, \ldots); \\ \nabla p_i &= (x_1, x_2, \ldots, x_i - h, \ldots); \end{aligned}\right\} \quad \ldots\ldots(8)$$

الخطوة الأولى: عند الـ cpu_1 سيقوم بحساب $f(\Delta p_1)$ ومن ثم يقارن هذه القيمة مع $f(p_1)$.

إذا كان $f(\Delta p_1) < f(p_1)$ ، اذن نعتبر $x_1 = x_1 + h$; وألا $x_1 = x_1 - h$;

- نفس الإجراء يطبق على $cpu_n, \ldots, cpu_4, cpu_3, cpu_2$
- وفي الخطوة التالية. فأن الـ cpu_{n+1} سوف يستلم القيم الجديدة من $cpu_n, \ldots, cpu_1$ اذا لم يكن هناك تخفيض في القيم الجديدة للدالة نقوم بتقليل الخطوة الى النصف (h=h/2) ومن ثم نرسل قيمة h الجديدة إلى $cpu_n, \ldots, cpu_1$.

أما إذا حصلنا على تخفيض في قيم الدالة الجديدة فأننا نقوم بإيجاد اقل قيمة من هذه القيم مع الاتجاه الذي حصل فيه هذا التخفيض فنعتبر القيم الجديدة قواعد ونرسلها إلى $cpu_n, \ldots, cpu_1$ مع ملاحظة عدم فحص الجهات التي حصل فيها زيادة وتستمر العملية إلى حد الوصول إلى قيمة صغيرة للخطوة h مناسبة. والمخطط التالي يوصف الخوارزمية الثانية:



1- Receive the new values from $Cpu_1, \ldots, cpu_n$
2- If no reduction on the function values, Then h=h/2, send h to $Cpu_1, \ldots, cpu_n$
3- Else: find min of $f(x_1, x_2, \ldots, x_n)$.
Send new values to $Cpu_1, \ldots, cpu_n$
4- Until no better point are found.

مثال (شكل (9): أسلوب توزيع المهام على المعالجات في الخوارزمية المطورة الثانية

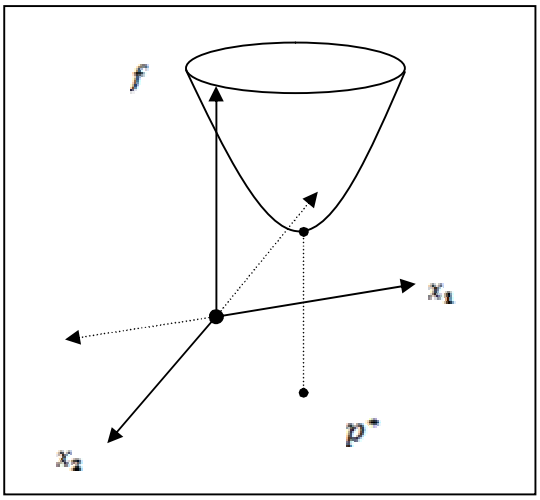
تم تصميم برنامج للخوارزمية المطورة الثانية بلغة الماتلاب وتم تنفيذه على الحاسوب واستخدمنا المثال التالي:

Find min of: $f(x_1,x_2)=(x_1-1)^2+(x_2+1)^2$, $p_0=(0,0)$, **h=0.5**

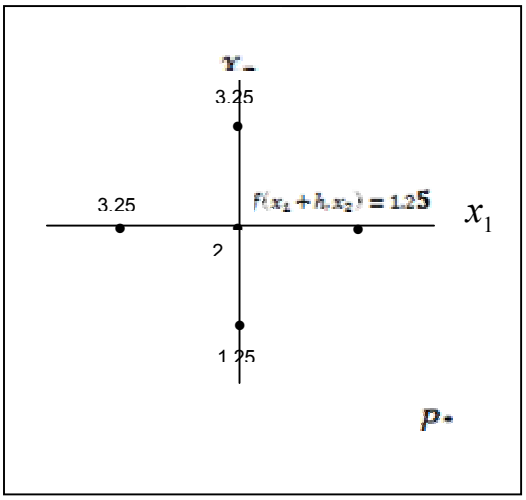
Sol. $f(p_0)=2$

$f(x_1+h,x_2)=1.25$ at (0.5,0), $f(x_1-h,x_2)=3.25$ at (-0.5,0)

$f(x_1,x_2+h)=3.25$ at (0,0.5), $f(x_1,x_2-h)=1.25$ at (0,-0.5)

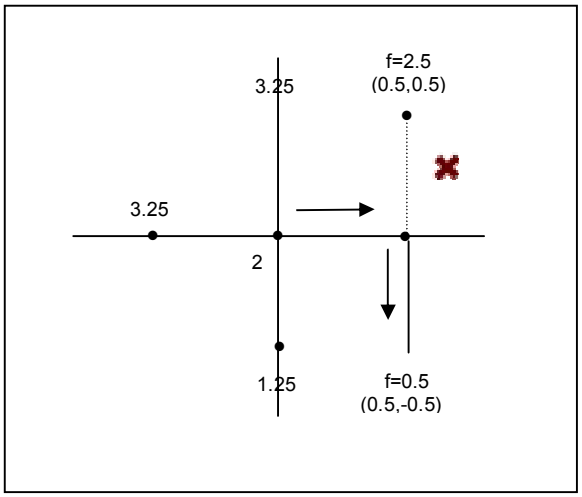


شكل (11): أسلوب البحث الباشر وافتراض الدالة



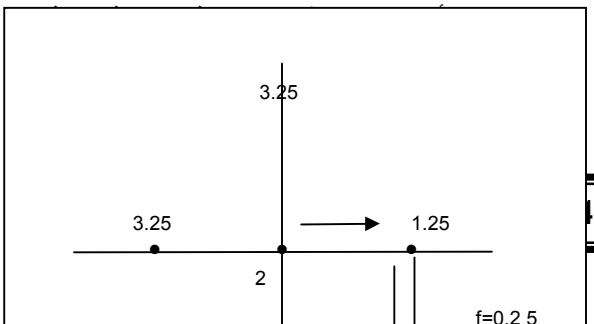
شكل (12): حساب قيمة الدالة عند الجهات الأربعة

نلاحظ أن الحركة الأولى باتجاه النقصان ستكون عند x_0+h عند قيمة الدالة *1.25* وألان سنقوم بترك المحاور الأخرى والسير باتجاه النقصان عند النقطة *(0,0.5)* كما في الشكل (13).



نلاحظ انه الحركة باتجاه الأعلى لا تعطينا نقصان لذلك سوف نترك هذا الاتجاه ونتجه أسفل أو أمام ونختبر الدالة كما في شكل (13).

وألان بأخذ الاتجاه إلى أسفل إي باتجاه النقطة(0.5,-0.5) سنحصل على نتيجة جديدة تؤدي إلى نقصان في قيمة الدالة كما في شكل (13):

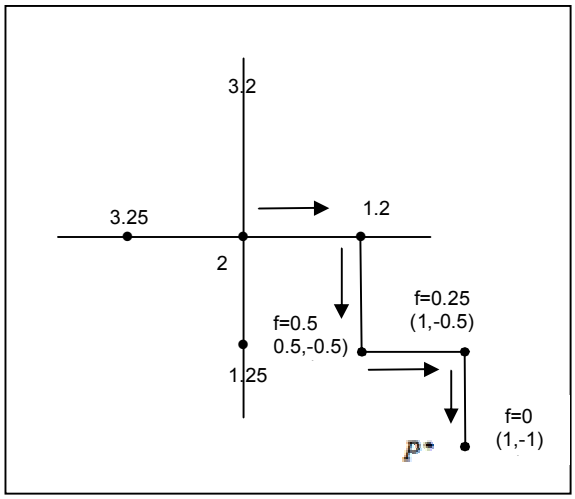


نلاحظ عند وصولنا إلى النقطة (0.5,-0.5) النقصان موجود عند هذه النقطة.

إذن تعتبر هذه النقطة قاعدة جديدة نجري عليها عملية الاستكشاف ومعرفة الاتجاه الصحيح الذي نسلكه.

نلاحظ عملية الاستكشاف بزيادة $x_1 = x_1 + h$ اي الاتجاه نحو النقطة (1,-0.5) نلاحظ نقصان في قيمة الدالة كما في شكل (14).

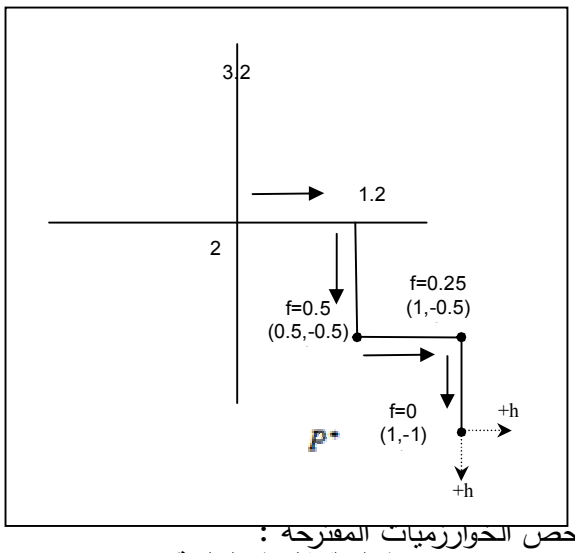
وألان عند السير باتجاه المحور الثاني أي $x_1 = x_1 - h$ سنحصل على النقطة (1,-1) وهي نقطة الـ minimum كما في الشكل (15) :



شكل (15)

شرط التوقف:

نلاحظ عند وصولنا إلى نقطة الـ min فأن إي زيادة بمقدار h يؤدي إلى زيادة قي قيمة الدالة عند ذلك نعمد إلى تقليل قيمة h إلى قيمة صغيرة جدا ϵ معطاة في المسالة عندها يتم التوقف كما في الشكل (16).



شكل (16): شرط التوقف

5- مناقشة النتائج :-

5-1: لقد تم استخدام المثال والمعايير التالية لفحص الخوارزميات المقترحة :

1- $f(x_1, x_2) = (x_1 - 1)^2 + (x_2 + 1)^2$, $p_0 = (0,0)$, **h=0.5**

Stopping criteria h<10^{-2}

كانت النتائج كالتالي:

- الخوارزمية الأولى التي تستخدم أسلوب البحث في الاتجاهات في كل المحاور كالتالي:
  تم إيجاد اقل قيمة عند p=(1.000,−1.000) وبقيمة z=0.0000038 عدد مرات حساب الدالة fe= 150.
  الزمن المستغرق للحل 0.297 sec
- الخوارزمية المحسنة والتي تترك الجهات التي يحصل فيها الزيادة كانت النتائج كالأتي:
  تم إيجاد اقل قيمة للدالة p=(1,−1)
  وبقيمة z=0
  عدد مرات حساب الدالة 73
  الزمن المستغرق 0.127 sec

نلاحظ أنه تم تقليل عدد مرات حساب الدالة وذلك لعدم الحساب في الجهات التي يحصل بها زيادة.

- الخوارزمية المقترحة التي تعتمد أسلوب الحركة النموذجية تم التوصل إلى النتائج التالية:
  تم إيجاد اقل قيمة للدالة عند النقطة p=(1.0020,−1.0020) وبقيمة z=0.00000038 عدد مرات حساب الدالة 90 الزمن المستغرق للحل 0.06 sec

5-2: إن من الاختبارات التي يمكن من خلالها قياس سرعة تنفيذ الخوارزمية هو اختبار "Rosenborck function" وهذه الدالة هي بالشكل:

$$\mathbf{2-}\quad f(x_1,x_2)=100(x_2-x_1^2)^2+(1-x_1)^2 \quad \text{at} \quad p_0=(0,0) \quad h=0.5$$

سنحل هذه الدالة بواسطة الخوارزميات السابقة ثم نحلها بواسطة الخوارزميات المقترحة كالتالي:

1- الخوارزمية المقترحة من قبل المصدر [8] حيث يتم البحث في جميع المحاور كانت النتائج كالتالي:
   - تم إيجاد نقطة الـ min عند النقطة $p^*=(1.0000,1.0000)$
   - عدد مرات حساب الدالة 46065
   - الزمن المستغرق 6.129178 sec
   - شرط التوقف h< 10^{-6}
2- الخوارزمية المطورة التي تبحث في اتجاه النقصان فقط كانت النتائج كالتالي:
   - تم إيجاد نقطة الـ min عند النقطة $p^*=(1,1)$
   - عدد مرات حساب الدالة 30624
   - الزمن المستغرق 5.2544 sec

- شرط التوقف $h < 10^{-6}$

3- الخوارزمية المقترحة التي تعتمد على أسلوب الحركة النموذجية كانت النتائج كالتالي:

- تم إيجاد نقطة الـ min عند النقطة $p^* = (0.999, 0.999)$
- عدد مرات حساب الدالة 36271
- الزمن المستغرق 4.83785 sec
- شرط التوقف $h<10^{-6}$

5-3: لنعتبر الدالة التالية:

2- $f(x_1, x_2) = 100(x_1^2 - x_2)^2 + (1 - x_1)^2 + 1$ at $p_0 = (0,0)$ $h = 0.5$

واختبرناها على الخوارزميات المقدمة وكانت النتائج:

1- الخوارزمية المقترحة من قبل المصدر [8] حيث يتم البحث في جميع المحاور كانت النتائج كالتالي:

- تم إيجاد نقطة الـ min عند النقطة $p^* = (1.0000, 1.0000)$
- عدد مرات حساب الدالة 40762
- الزمن المستغرق 9.7641 sec
- شرط التوقف $h<10^{-6}$

2- الخوارزمية المطورة التي تبحث في اتجاه النقصان فقط كانت النتائج كالتالي:

- تم إيجاد نقطة الـ min عند النقطة $p^* = (1000, 1.000)$
- عدد مرات حساب الدالة 46065
- الزمن المستغرق 4.31420 sec
- شرط التوقف $h<10^{-6}$

3- الخوارزمية المقترحة التي تعتمد على أسلوب الحركة النموذجية كانت النتائج كالتالي:

- تم إيجاد نقطة الـ min عند النقطة $p^* = (0.999, 0.999)$
- عدد مرات حساب الدالة 36356
- الزمن المستغرق 3.665 sec
- شرط التوقف $h<10^{-6}$

6- الاستنتاجات:-

من خلال مناقشة النتائج نستنتج ان الخوارزميتين المقترحتين أفضل من الخوارزميات المعطاة في المصدر [8] من حيث عدد إيجاد قيم الدالة وزمن التنفيذ.

والجدول (1) يبن المقارنة بين الخوارزميات السابقة والمقترحة.

1. $f(x_1,x_2)=100(x_2-x_1^2)^2+(1-x_1)^2$ 'Rosenborck function'
2. $f(x_1,x_2)=100(x_1^2-x_2)^2+(1-x_1)^2+1$
3. $f(x_1,x_2)=(x_1-1)^2+(x_2+1)^2$

No.Fun	p_o	h	Kahlaf&Neama Algorithm				The first Algorithm				The Second Algorithm			
			P^*	Time(s)	$f(P^*)$	No.fe	P^*	Time(s)	$f(P^*)$	No.fe	P^*	Time(s)	$f(P^*)$	No.fe
1.	(0,0)	0.5	(1.000,1.000)	6.1291	0.5512	46065	(0.999,0.999)	4.8375	0.5272	36271	(1,1)	5.254	0.5415	30624
2.	(0.1,0.2)	0.5	(1.000,1.000)	9.7641	1.000	40762	(0.999,0.999)	3.665	1.000	36356	(1.000,1.000)	4.314	1.000	46065
3.	(0,0)	0.5	(1.000,-1.000)	0.297	0.0038	150	(1.002,-1002)	0.06	0.00038	90	(1,-1)	0.127	0	73

جدول (١) مقارنة النتائج بين الخوارزميات السابقة والخوارزميات المقترحة

References:

1) Amitay, Isaacs (2003) "Direct-Search Methods and DACE", Ph.D.Thesis, Indian Institute of technology, India.

2) Audetand C and J.E Dennis. (2003), Analysis of generalized pattern search, SIAMJ.opt., (13) PP:889-903.

3) Brocklehurst E.(1992), parallel processing, NPL News, No.372, Pp.24-27.

4) Bunday. B.D. (1984), "Basic Optimization Methods", Edward Arnold, USA.

5) Dennis. J.E, JR and Torezon. V. (1991), Direct search method on parallel machines. SIAM Journal on optimization, 1(4):448-474.

6) Flynn M. (1972), Some computer organization and their effctiveness, IEEE Trans. on computer, Vol-21, No.9 Pp:948-969.

7) Hook, Robert and T.A Jeeves. (1961), Direct search Solution of numerical and statistical problem Journal of The Association of computing Machinery, Pp:221-224.

8) Khalaf B. and M. Neama (2010), Parallel direct search method, Al-Rafidain Journal of computer and mathematics, Vol.(7), No.3, Pp:51-59.

9) Khalaf B.(1995), A Classification for computer architecture J.Educ. Sci, Vol.(24), Pp.212-222.

10) Khalaf B. (1990), Parallel numerical algorithm for solving ODEs, ph.D.thesis, School of computer studies, Leeds University.

11) Kolda G. Tamara, Robert Meichal Lewise and Vergina Torczon, (2003), Optimization by direct Search, SIAM Review, Vol.(45), No.3,Pp.385-482.

12) Kwok L. Chow. (1996), Parallel Unconstrained Optimization. Ph.D. thesis, Department of computer Scinec, University of Toronto, Toronto Ontario, canda M5s 1A4.

13) Meiko (1989), A programmer introduction to CStools, Meiko Ltd, England.

14) Murray W. (1972). "Numerical Methods for Unconstrained Optimization". Academic press.

15) Torczon V. (1989), Multi-Directional search: A direct Search Algorithm for parallel Machianes, Ph.D Thesis, Department of Mathematical Science, Rice University, Houston, USA.

16) Torczon V. (1997), On The converennce of pattern search methods. SIAM Jouurnal on Optimization, 7(1):1-25.