مجلة جامعة تشرين للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (30) العدد (1) 2008
Tishreen University Journal for Research and Scientific Studies - Engineering Sciences Series Vol. (30) No. (1) 2008

تحليل طريقة تقليل عدد العمليات الحسابية والمنطقية عن طريق إدخال علاقات تكرارية إضافية إلى الخوارزميات.

الدكتور إبراهيم الشامي*

(تاريخ الإيداع 1 / 10 / 2007. قُبِل للنشر في 2008/3/4)

□ الملخّص □

تبحث هذه الورقة في تحليل طريقة تحسين التعقيد الحسابي عن طريق إدخال علاقات تكرارية إضافية إليها، وذلك بهدف تقليل عدد العمليات الحسابية والمنطقية وبالتالي زيادة سرعة تنفيذها ويستنتج بأن هذه الطريقة ليست فعالة دائماً باعتبار أن العلاقات التكرارية المضافة ستؤدي إلى زيادة عدد المتحولات التي ستضمها الخوارزميات وهذا له تأثير عكسي في سرعة التنفيذ. يحدد الباحث في هذه الورقة الحالات المناسبة لاستخدام الطريقة المطروحة.

تم التحقق من صحة الاستنتاجات بتطبيق الخوارزميات من خلال الحزمة MATLAB\R2006a، وبالنسبة للخوارزميات ذات المتحولات الصحيحة بالبرمجة بلغة C++ ومن خلال تصميم برنامج لقياس زمن التنفيذ.

الكلمات المفتاحية: العلاقات التكرارية – التعقيد الحسابي للخوارزميات– تصميم الخوارزميات:

* أستاذ مساعد – قسم هندسة التحكم الآلي والحواسيب– كلية الهندسة الميكانيكية والكهربائية –جامعة البعث – حمص– سورية.

مجلة جامعة تشرين للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (30) العدد (1) 2008
Tishreen University Journal for Research and Scientific Studies - Engineering Sciences Series Vol. (30) No. (1) 2008

Analyzing the Method of Improving Computational Complexity by Inserting Recurrence Formulas to Algorithms

Dr. Ibrahim Chami*

(Received 1 / 10 / 2007. Accepted 4/3/2008)

□ ABSTRACT □

This paper handles the method of improving the computational complexity of algorithms by inserting additional recurrence formulas in order to decrease the arithmetical and logical operations. Consequently, execution speed increases. So, this method is not always active because the additional recurrence formulas increases the variables in the algorithms and this, in turn, has an opposite effect on execution speed. In this paper, the writer determines suitable states for using his method.

The results were verified by implementing the algorithms in MATLAB\R2006a language, and in C++ language (for algorithms which contained integer variables) and via designing a program to measure the execution time.

Keywords: Recurrence formulas, Computational complexity of algorithms, Algorithms designing

* **Associate Professor, Department of Computer Engineering and Automatic Control, Faculty of Mechanical and Electrical Engineering, Al Baath University, Homs, Syria.**

المقدمة:

على الرغم من السرعة الهائلة التي تتمتع بها المعالجات الحاسوبية في الوقت الحاضر إلا أن البحث عن طرق جديدة تؤدي إلى تقليل التعقيد الحسابيComputational complexity وبالتالي زيادة سرعة تنفيذ الخوارزميات Algorithms ، وخصوصاً عند تنفيذها بالدارات Hardware مازال مستمراً.

عموماً تصمم الخوارزميات لتطبيقها بالدارات أو بالبرمجة Software أو ببرامج صغرية (ميكروية) Fermware (Carlet, Fossorier and Jmesi,2006) وقبل إخراج الخوارزمية بالشكل النهائي تجرى عليها الكثير من التعديلات و التحسينات التي تهدف إلى الوصول إلى درجة مقبولة من الفعالية وفي مرحلة تحليل فعالية الخوارزمية يمكن أن نميز ثلاثة أنواع رئيسة من التحليلات ، بحسب مجال الاستخدام (Banachowski, Kreczmar and Rytter,1989):

- تحليلات زمن التنفيذ أو كلفة الخوارزمية.
- تحليلات دقة التنفيذ ويشمل ذلك دراسة تحليلية للأخطاء في الخوارزميات والتي يمكن أن يصل عدد أنواعها بحسب المواصفات القياسية PN - 71/N - 02050 إلى 30 نوعاً (Polanscy, 1991).
- حسابات طول الخوارزميات أو سعة تخزينها في الذاكرات (سعة فراغ الذاكرة المستعمل في أثناء التنفيذ).

هناك الكثير من العوامل التي تؤثر في زمن التنفيذ نذكر منها:

1. عدد العمليات الحسابية والمنطقية التي تضمها الخوارزمية
2. عدد المتحولات المعرفة ضمن الخوارزمية
3. طول الخوارزمية(السعة التخزينية)
4. أسلوب تخزين المتحولات في الذاكرة

إن حساب عدد العمليات الحسابية والمنطقية في الخوارزميات يتم بطرق رياضية (وغالباً ما تكون طرق حل عددية Numerical methods) تغطي الاحتمالات كافة (Mokrzyski,1990) بمعنى أنه لا يمكن تحليل سرعة التنفيذ تماماً عن طريق تطبيق الخوارزميات ببرامج حاسوبية؛ لأن قياسات زمن تنفيذ البرامج تتم من أجل معطيات محددة (Jankowska and Jankowski, 1989)، وهذا يسمح للمبرمج بإختيار المعطيات التي تجعل البرنامج ينفذ على نحو أسرع ، وبالتالي يمكنه تحاشي المعطيات التي تجعل برنامجه أبطأ، إلا أن هذا التطبيق يبقى في النهاية ضرورياً للتحقق Verification من صحة التحليلات المنجزة. لهذا في هذا البحث سيتم تحليل سرعة التنفيذ من خلال إحصاء عدد العمليات الحسابية والمنطقية من أجل الاحتمالات كافة، وبعد ذلك يجري التطبيق على الحاسوب.

هناك كثير من الطرق التي تتبع لرفع سرعة التنفيذ، وكل منها يلائم حالات محددة. نناقش في هذا البحث طريقة تقليل عدد العمليات من خلال إدخال علاقات تكرارية إضافية إلى جسم الخوارزمية، فيمكن للعلاقات التكرارية في حالات عديدة أن تزيد زمن تنفيذ الخوارزمية و لكن مع هذا يمكن أيضاً أن ترفع من فعالية تلك الخوارزمية عند تنفيذها على الحاسوب من خلال تطبيقها ببرنامج معين ، نظراً للخطأ المرتكب عند تمثيل المعطيات في عالم (داخل) الحاسوب، فمثلاً بالنسبة لخوارزمية إيجاد المجموع :

$$z = \sum_{i=1}^{n} a_i$$

التي تكون رياضياً كالآتي:

```
z :=0;
```

```
for i:=1 step 1 until n do
v:= <  i قيمة العنصر الذي ترتيبه >;
z:= z+v;
end;
```

ولتلافي الخطأ المرتكب عند تمثيل المعطيات لدينا خوارزمية Gill-Möller من (Möller,1965) التي فيها تم إدخال علاقات تكرارية إضافية إلى جسم الخوارزمية السابقة لتصبح بالشكل الآتي:

```
z :=0;
for i:=1 step 1 until n do
v:= <  i قيمة العنصر الذي ترتيبه >;
s := u+v;
p := u- s+v+p;
z:= s;
end;
z := s+p;
```

حيث نلاحظ أن الخوارزمية المعدلة أبطأ بكثير (و ذلك بسبب وجود عمليات حسابية و متحولات أكثر) من الأساسية، ولكنها تعطي نتائج أدق، وهذا الأسلوب يمكن استخدامه عند الحاجة إلى إظهار نتائج دقيقة جداً للبرامج بحيث نتمكن من التغلب على محدودية التمثيل في الحاسوب و ليس مبرراً على الإطلاق أن نستخدمها في خوارزميات تتطلب السرعة في التنفيذ مثل خوارزميات البحث و التصنيف.

نناقش في هذا البحث الحالات المفيدة و فعالية الطريقة المقترحة ونخرج من ذلك ببعض الاستنتاجات المفيدة لمصممي الخوارزميات عموماً.

أهمية البحث و أهدافه:

عند تصميم الخوارزميات نأخذ بالحسبان نوع المتحولات و عددها وكذلك نوع العمليات وعددها ونسعى أن تكون المتحولات من نوع صحيح (حيث أن تمثيل النوع الصحيح ضمن بنية الحاسوب مختلفة عن تمثيل النوع الحقيقي و بشكل يجعل العمليات الحسابية تنفذ بشكل أسرع) إن أمكن ذلك و أن يكون عددها أقل ما يمكن وكذلك نوع العمليات بحيث يكون أبسط ما يمكن وعددها بحيث يكون أيضاَ أقل ما يمكن.

يتم التركيز في هذا البحث على تقليل عدد العمليات من خلال إدخال علاقات تكرارية إضافية بحيث لا يؤدي ذلك إلى زيادة عدد المتحولات بالقدر ذاته ، أي بحيث يكون معدل تقليل العمليات أعلى من معدل زيادة المتحولات، وهذا له أهمية خاصة من أجل الخوارزميات التي تتعامل مع المتحولات من نوع صحيح والتي يتم تطبيقها بدارات إلكترونية.

تمكن هذه الدراسة من تحديد الحالات الأنسب لاستخدام العلاقات التكرارية الإضافية من أجل تقليل كلفة الخوارزميات.

تم تنفيذ البرامج على حاسوب شخصي مزود بمعالج بنتيوم IV ذي تردد 2400Hz وذاكرة مخبئية Cache memory سعتها 512Byte و ذاكرات RIMM سعتها 384MB

طريقة البحث:

تستند طريقة البحث على إستبدال علاقة تكرارية معقدة بعدة علاقات تكرارية أبسط وهذه العلاقة التكرارية يمكن أن تكون متزايدة أو متناقصة أي:

$$d_{i+1} = d_i \pm D_i \quad \text{...............................(1)}$$

والحد D_i هو عبارة عن كثير حدود ذي متحولات (واحد أو أكثر) ، وفي حالة خاصة عندما يكون الحد D_i ثابتاً (بلا متحولات) تكون العلاقة التكرارية (1) بسيطة و لا يمكن إستبدالها بعدة علاقات تكرارية أبسط.

نناقش عدة حالات يمكن أن تصادف بكثرة:

1− حالة D_i بمتحول واحد:

لتكن العلاقة التكرارية المعقدة ذات الشكل العام الآتي:

$$d_{i+1} = d_i \pm \underbrace{AX_i \pm B}_{D_i} \quad \text{...............................(2)}$$

لنستبدل كثير الحدود D_i بالمتحول U_i وهذا يعني إضافة متحول لحلقة التكرار ، وبفرض أن الخطوة لتغير X هيδ أي:

$$X_{i+1} = X_i \pm \delta$$

$$U_{i+1} = AX_{i+1} \pm B = A(X_i \pm \delta) \pm B = U_i \pm A\delta$$

حيث الحد $A\delta$ عبارة عن مقدار ثابت ولا يحتوي على أية عملية حسابية (المبرمج يعوضه في الخوارزمية مباشرة) وتصبح العلاقة التكرارية(2) عبارة عن علاقتين تكراريتين بسيطتين:

$$\left.\begin{aligned} d_{i+1} &= d_i \pm U_i \\ U_{i+1} &= U_i \pm A\delta \end{aligned}\right\} \quad \text{...................................(3)}$$

ونلاحظ أن العلاقة (2) تحتوي على عمليتي جمع (طرح) وعملية ضرب واحدة بينما العلاقتين (3) تحتويان على عمليتي جمع (طرح) فقط ، أي اختصرنا عملية ضرب مقابل زيادة متحول واحد . ولنختبر مقدار التسريع من خلال الحزمة MATLAB . لنفرض أولاً قيمتا الثابتين: A=5 و B=7 ، ولنفترض أن طول الخطوة δ=0.7 ، وهذا يعني أن القيمة الابتدائية للمتحول u هي 12 و مقدار تزايد u هو 3.5 =5*0.7 =Aδ ، و لابد من افتراض قيمة ابتدائية للحد d ولتكن صفراً ، فمن أجل قيم منتظمة للمتحول x تقع بين الواحد و المئة نكتب:

```
>> d=0;tic;for x=1:0.7:100
d=d+5*x+7;
end; t=toc
t =
  3.3775e-004
```

أي أن زمن التنفيذ هو 3.3775e-004 ، بينما مع إدخال علاقة تكرارية إضافية نجد:

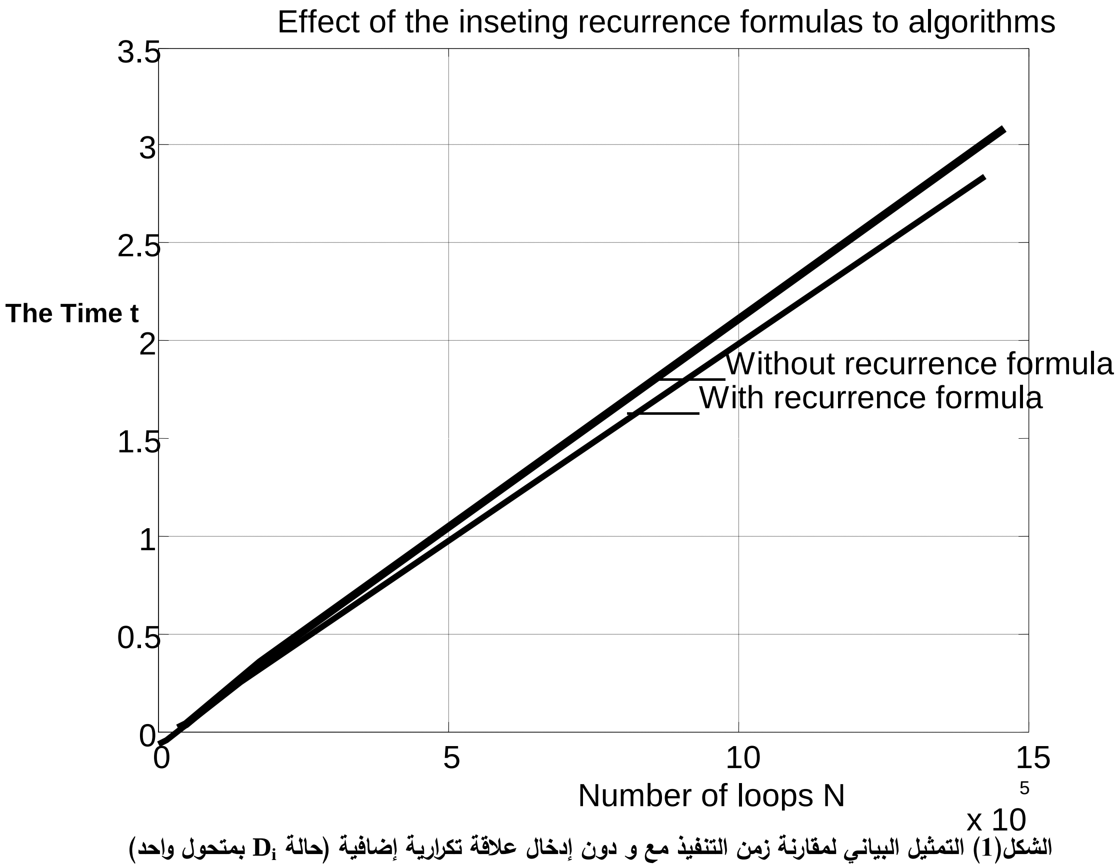
```
>> d=0;tic;u=12;for x=1:0.7:100
d=d+u;
u=u+3.5;
end;t=toc
t =
  3.3747e-004
```

أي أن زمن التنفيذ أصبح 3.3747e-004 أي نقص بمقدار 0. 75e-006، و إذا ما تابعنا التنفيذ من أجل مجالات أوسع للمتحول x لحصلنا على أزمنة تنفيذ متزايدة وفق الجدول الآتي:

الجدول (1) مقارنة زمن التنفيذ مع و بدون إدخال علاقة تكرارية إضافية (حالة D_i بمتحول واحد)

مجال تغير x	1÷100	1÷1000	1÷10000	1÷100000	1÷1000000
زمن التنفيذ دون إدخال علاقة تكرارية إضافية	3.3775e-004	0.0031	0.0306	0.3401	3.0758
زمن التنفيذ مع إدخال علاقة تكرارية إضافية	3.3747e-004	0.0030	0.0289	0.3216	2.9158
نسبة تقليل الزمن	0.22%	3.23%	5.56%	5.44%	5.21%

نلاحظ أن نسبة تقليل زمن التنفيذ في هذه الحالة هي أعلى من 5% بقليل، و يتضح ذلك عند زيادة مجال تغير المتحول x ، والشكل(1) يوضح بيانياً الجدول(1).



الشكل(1) التمثيل البياني لمقارنة زمن التنفيذ مع و دون إدخال علاقة تكرارية إضافية (حالة D_i بمتحول واحد)

وعند فرض أن طول خطوة التزايد (التناقص) X تساوي الواحد الصحيح يكون :

$$X_{i+1} = X_i \pm 1$$

$$U_{i+1} = AX_{i+1} \pm B = A(X_i \pm 1) \pm B = U_i \pm A$$

$$d_{i+1} = d_i \pm U_i$$

وتصبح العلاقة التكرارية(2) عبارة عن علاقتين تكراريتين بسيطتين ولكن بشكل أبسط من العلاقتين(2):

$$\left.\begin{aligned} d_{i+1} &= d_i \pm U_i \\ U_{i+1} &= U_i \pm A \end{aligned}\right\} \qquad (4)$$

2- حالة D_i بمتحولين:

لتكن العلاقة التكرارية المعقدة ذات الشكل العام الآتي:

$$d_{i+1} = d_i \pm \underbrace{AX_i \pm BY_i \pm C}_{D_i} \qquad (5)$$

لنستبدل كثير الحدود D_i بالمتحول U_i وهذا يعني إضافة متحول لحلقة التكرار ، وبفرض أن الخطوة لتغير X هي δ_1 و لتغير Y هي δ_2 أي:

$$X_{i+1} = X_i \pm \delta_1$$

$$Y_{i+1} = Y_i \pm \delta_2$$

$$U_{i+1} = AX_{i+1} \pm BY_{i+1} \pm C = A(X_i \pm \delta_1) \pm B(Y_i + \delta_2) = U_i \pm A\delta_1 \pm B\delta_2$$

حيث الحدان $A\delta_1 \pm B\delta_2$ عبارة عن مقدار ثابت ولا يحتوي على أية عملية حسابية (المبرمج يعوض مقدارهما في الخوارزمية مباشرة) وتصبح العلاقة التكرارية(5) عبارة عن علاقتين تكراريتين بسيطتين:

$$\left.\begin{aligned} d_{i+1} &= d_i \pm U_i \\ U_{i+1} &= U_i \pm A\delta_1 \pm B\delta_2 \end{aligned}\right\} \qquad (6)$$

ونلاحظ أن العلاقة (5) تحتوي على ثلاث عمليات جمع (طرح) وعمليتي ضرب بينما العلاقتين (6) تحتويان على عمليتي جمع (طرح) فقط ، أي اختصرنا عملية جمع (طرح) واحدة و عمليتي ضرب مقابل زيادة متحول واحد . ولنختبر مقدار التسريع من خلال الحزمة MATLAB. لنفرض أولاً قيم الثوابت: A=5 و B=7 و C=4 ، ولنفترض أن طول الخطوة $\delta 1=0.7$ وطول الخطوة $\delta_2=1$ ، وهذا يعني أن القيمة الابتدائية للمتحول u ستختلف بحسب قيمتي المتحولين x و y الابتدائيتين فمن أجل قيمة ابتدائية للمتحول x قدرها 1 ولـ y قدرها 99 نجد:

$$U_1 = 5*1 - 7*99 + 4 = -684$$

و مقدار تزايد u هو $A\delta_1 - B\delta_2 = 5*0.7 - 7*1 + 4 = 0.5$ ، و لابد من افتراض قيمة ابتدائية للمتحول d ولتكن صفراً ، فمن أجل قيم منتظمة للمتحول x تقع بين الواحد و المئة ، ومن أجل قيم منتظمة للمتحول y تقع بين 142 و الصفر نكتب:

```
 >> d=0; y=142 ;tic;for x=1:0.7:100
d=d+5*x-7*y+4;
y=y-1;
end;t=toc
t =
  6.7802e-004
```

أي أن زمن التنفيذ هو 6.7802e-004 ، بينما مع إدخال علاقة تكرارية إضافية نجد:

```
>> d=0; y=142 ;tic; u=-684;for x=1:0.7:100
d=d+u;
u=u+0.5;
y=y-1;
```

```
end;t=toc
    t =
        4.6011e-004
```

أي أن زمن التنفيذ أصبح 4.6011e-004 أي نقص بمقدار 2.1791e-004 ، و إذا ما تابعنا التنفيذ من أجل مجالات أوسع للمتحول x و للمتحول y لحصلنا على أزمنة تنفيذ متزايدة وفق الجدول(2).

الجدول (2) مقارنة زمن التنفيذ مع و دون إدخال علاقة تكرارية إضافية (حالة D_i بمتحولين)

مجال تغير x	1÷100	1÷1000	1÷10000	1÷100000	1÷1000000
مجال تغير y	142÷0	1428÷0	14285÷0	142856÷0	1428571÷0
زمن التنفيذ دون إدخال علاقة تكرارية إضافية	6.7802e-004	0.0061	0.0595	0.6195	5.9698
زمن التنفيذ مع إدخال علاقة تكرارية إضافية	4.6011e-004	0.0043	0.0425	0.45	4.2210
نسبة تقليل الزمن	32.14%	35. 29%	28.6%	27.36%	29.3%

نلاحظ أن نسبة تقليل زمن التنفيذ في هذه الحالة هي بشكل متوسط 30% ، والشكل(2) يوضح بيانياً مضمون الجدول(2).

النتائج والمناقشة:

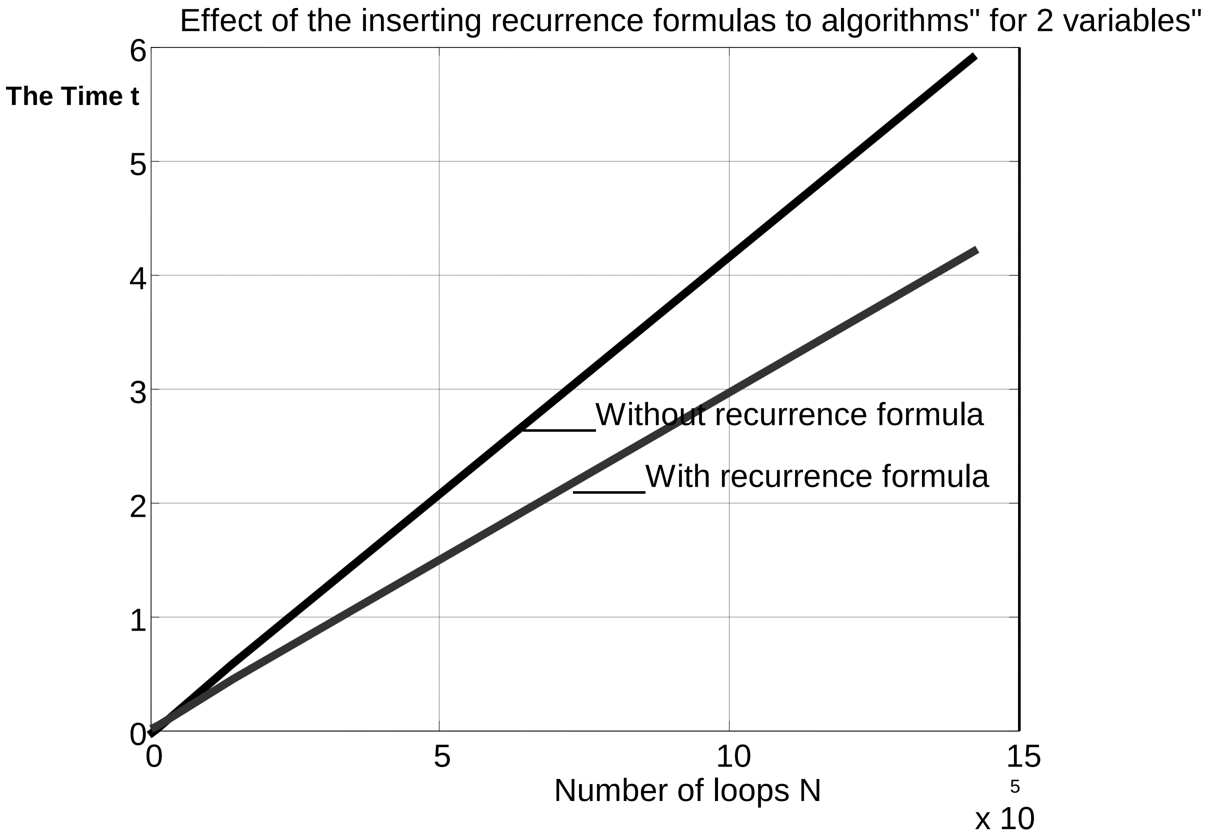
نناقش في هذه الفقرة مسألتين وهما:

1. أهمية استخدام الطريقة المطروحة في تقليل كلفة الخوارزميات ذات المتحولات من نوع صحيح، والتي غالباً ما تطبق بالدارات الرقمية Digital circuits
2. تحليل فعالية إدخال علاقات تكرارية إضافية إلى خوارزميات ذات تفرعات.

إن التأكد من صحة الطريقة المطروحة من خلال الحزمة MATLAB\R2006a غير ممكن في حالة التعامل مع المتحولات الصحيحة (http://www.mathworks.com/pl_homepage)، لهذا سنعتمد على لغة البرمجة المعروفة C++ زمن تنفيذ البرنامج قد تم قياسه بوساطة التابع:

gettime (الزمن الآني&)

وتم الإعلان (تعريف) عن البنية (الزمن الآني&) على أنها من النوع struct time



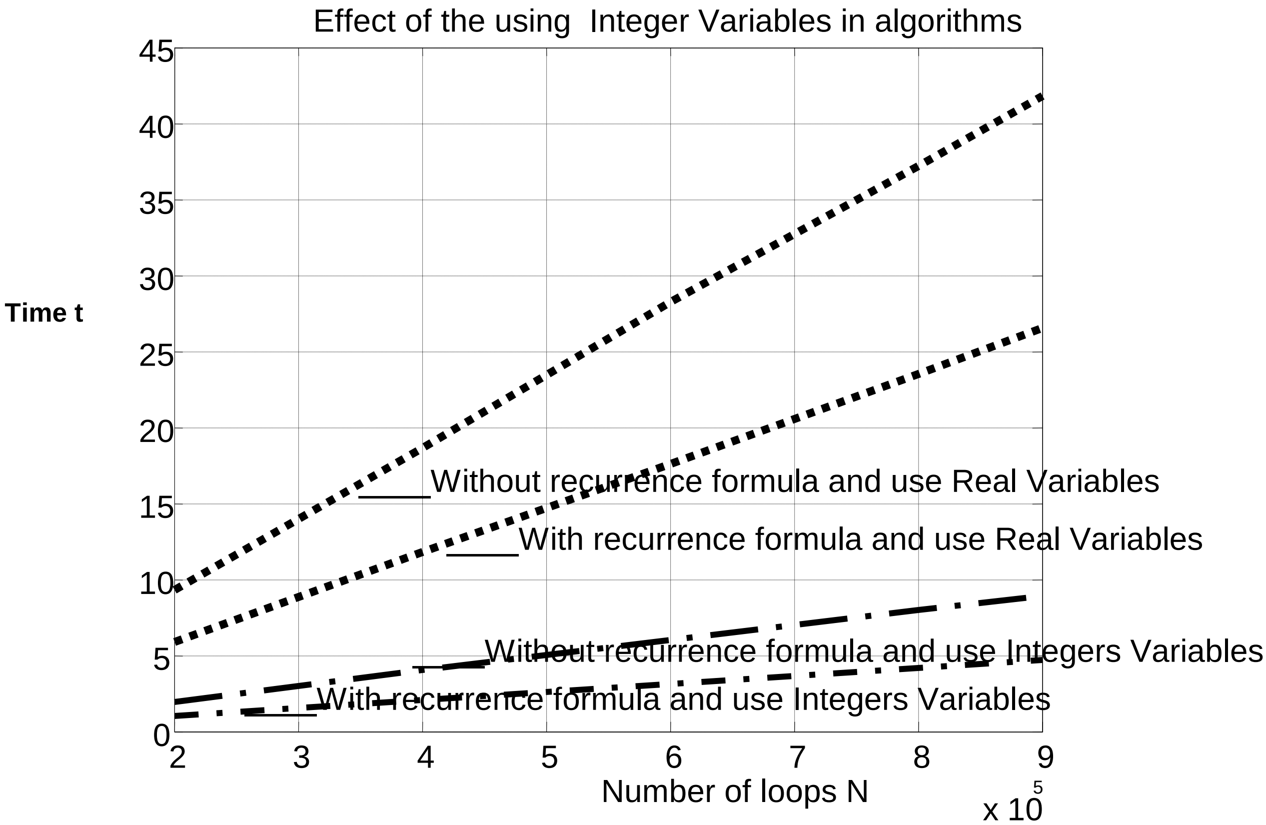
الشكل(2) التمثيل البياني لمقارنة زمن التنفيذ مع و دون إدخال علاقة تكرارية إضافية (حالة D_i بمتحولين)

وعلى هذا الأساس يتم قياس الزمن باستخدام ساعة معالج الحاسوب وذلك من خلال مناداة التابع (الزمن الآني& gettime) مرتين في البرنامج الرئيسي قبل الجزء المراد قياس زمن تنفيذه وبعده. ونظراً لصغر الزمن وتوخياً للدقة يتم تكرار تنفيذ ذلك الجزء لعدة مرات كما يبين الجدول(3).

الجدول (3) مقارنة زمن التنفيذ مع و دون إدخال علاقة تكرارية إضافية إلى الخوارزميات ذات المتحولات الصحيحة و الحقيقية

عدد مرات التكرار		200000	400000	600000	800000	900000
زمن التنفيذ دون إدخال علاقة تكرارية إضافية	لمتحولات من نوع حقيقي	9.29	18.57	28.18	37.19	41.82
	لمتحولات من نوع صحيح	1.87	3.95	5.93	7.91	8.93
زمن التنفيذ مع إدخال علاقة تكرارية إضافية	لمتحولات من نوع حقيقي	5.82	11.75	17.63	23.5	26.47
	نسبة تقليل الزمن	37.35%	36.73%	37.44%	36%	36.7%
	لمتحولات من نوع صحيح	1.02	2.05	3.08	4.15	4.67
	نسبة تقليل الزمن	45.45%	48.1%	48.06%	47.53%	47.7%

نلاحظ أن نسبة تقليل الزمن بالنسبة للخوارزميات ذات المتحولات الصحيحة تقترب من 48% ، بينما تقليل الزمن بالنسبة للخوارزميات ذات المتحولات الحقيقية تقترب من 37% (الشكل(3) يوضح ذلك)، ما تعليل ذلك؟ لاشك في أن نسبة تقليل عدد العمليات هو ذاته في كلا الحالتين ، و بالمقابل عدد المتحولات التي إزدات هو ذاته في كلا الحالتين، و لكن نظراً لإختلاف تمثيل المتحولات الصحيحة عن الحقيقية داخل الحاسوب فإن تأثير المتحولات الزائدة يكون تأثيره أكبر في الخوارزميات ذات المتحولات الحقيقية مقارنة مع الخوارزميات ذات المتحولات الصحيحة.



الشكل(3) التمثيل البياني لمقارنة زمن التنفيذ مع و دون إدخال علاقة تكرارية إضافية (حالة D_i بمتحولين من نوع صحيح و حقيقي)

إن ما أورد سابقاً لا يعتبر صحيحاً في حالة وجود تفرعات في الخوارزمية ، فمثلاً في حالة وجود فرعين كما في الشكل(4)، حيث لدينا جزءاً من خوارزمية بريسنهام من(Bresenham,1985) ، و البرمجة الكاملة لهذه الخوارزمية متوافرة في كثير من مراجع التمثيل البياني مع الحاسوب مثل (DAWARA,2005) حيث استخدم لغة C المزودة بمكتبة قياسية ضخمة تخص التمثيل البياني مع الحاسوب ، ويمكن استخدام لغات أخرى عالية المستوى مثل لغة BASIC (Ducamp and Reverchon, 1985) .

نلاحظ في خوارزمية الشكل(4) أن قيمة المتحول d تزداد أو تتناقص في كلا الفرعين، و عندما $d > 0$:

$$d = d + 2X - 2Y + 1 \qquad (7)$$

Y = Y -1
X= X +1

وعندما $d \le 0$:

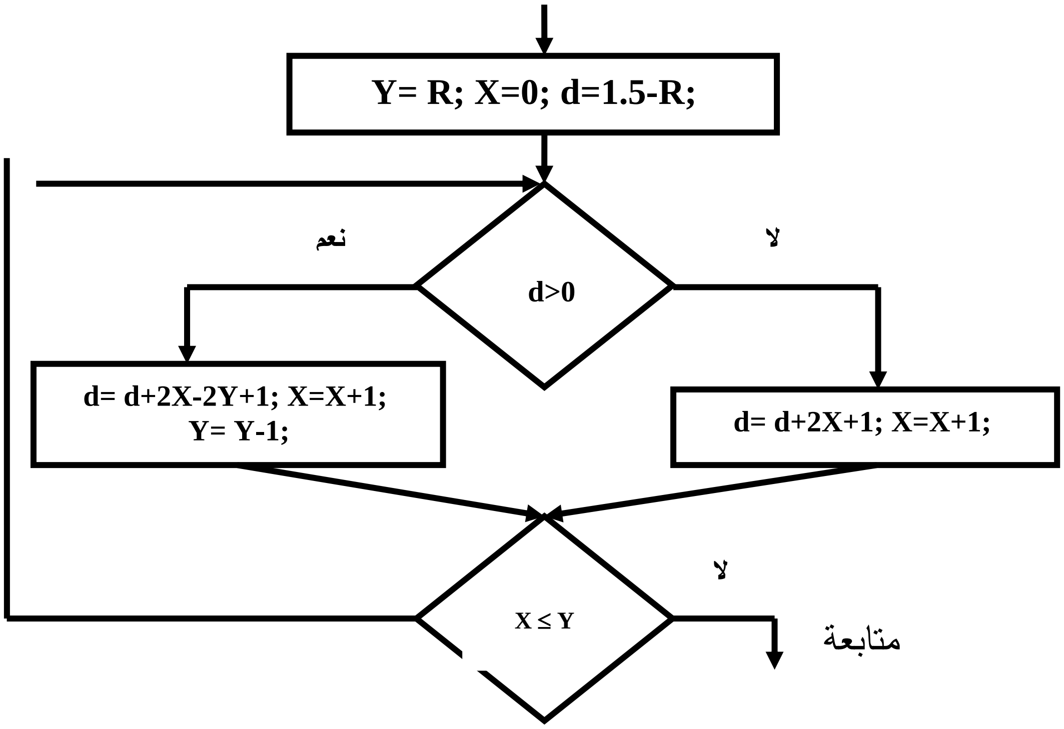
$$d = d + 2X + 1 \qquad (8)$$

X = X +1

ويتم إنجاز عملية الاختبار $\{X \le Y\}$ في نهاية كل دورة من أجل الوصول إلى نهاية جزء الخوارزمية.

وعند بتعويض الآتي في العلاقة (8):

U = 2X +1



الشكل(4) جزء من خوارزمية يحتوي على تفرع (إلى فرعين)

تصبح تلك العلاقة بالشكل المبسط الآتي:

$$d = d + U \qquad (9)$$

وبشكل مشابه عند افتراض في العلاقة (7) أن:

V = 2X - 2Y +1

لتصبح العلاقة (1) بالشكل المبسط الآتي:

$$d = d + V \qquad (10)$$

ولوجود تأثير متبادل للفرعين في بعضهما البعض نجد أنه عندما $d > 0$ يكون:

U = U + 2
V = V + 4
d = d + V
Y = Y -1
X = X + 1

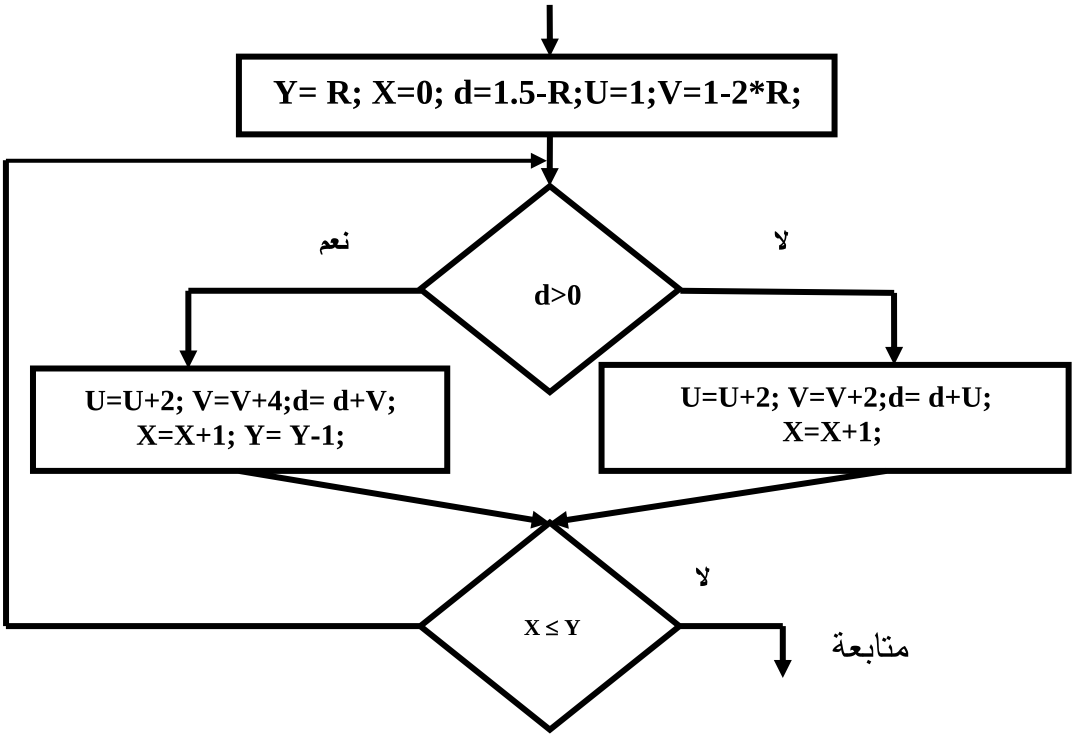
وعندما $d \le 0$ يكون:

U = U + 2
V = V +2
d = d + U
X = X + 1

ويصبح جزء الخوارزمية من الشكل(4) معدلاً إلى الشكل(5).

لنحلل فائدة هذا التعديل ، حيث نظرياً أصبح لدينا أربع عمليات جمع في كلا الفرعين مع عملية طرح إضافية في الفرع الأيسر (و بافتراض أن عدد مرات تنفيذ الفرعين متساو يكون لدينا وسطياً 4.5 عملية)، بدلاً من ثلاث عمليات جمع في كلا الفرعين مع عملية ضرب في الفرع الأيمن ، و عمليتي ضرب وطرح في الفرع الأيسر (وسطياً 6 عمليات) . ولنختبر مقدار التسريع من خلال الحزمة MATLAB . لنفرض أولاً قيمة الثابت: R=1000000 فيكون من أجل الشكل(4):

```
tic;X=0; R= 1000000; d=1.5-R;Y=R;
```



الشكل(5) جزء معدل من خوارزمية يحتوي على تفرع (إلى فرعين)

```
while (X<=Y)
if (d >0)
d=d+2*X-2*Y+1;
X=X+1;
Y=Y-1;
else
d=d+2*X+1;
X=X+1;
end
end; l=toc
    l =
          3.7164
```

ومن أجل الشكل(5):

```
tic; X=0; R= 1000000; d=1.5-R;Y=R;U=1;V=1-2*R ;while (X<=Y)
if (d >0)
U=U+2;
V=V+4;
d=d+V;
X=X+1;
Y=Y-1;
else
U=U+2;
V=V+2;
d=d+U;
X=X+1;
end
end; l=toc
   l =
        4.1113
```

و نلاحظ أن زمن التنفيذ قد ازداد بدلاً من أن ينقص، و عند التأكد من ذلك مع قيمة أكبر للثابت R= 10000000 نجد أن زمن التنفيذ أيضاً قد ازداد من 36.8760 ثانية إلى 40.9170 ، أي بمعدل 10.96% ، و بذلك نستنتج عدم فعالية الطريقة المطروحة في هذه الحالة و السبب هو كتابة مضمون الطرف الأيمن من خوارزمية الشكل(4) حيث تم تعديله إلى الطرف الأيمن من خوارزمية الشكل(5) ، و طرفا الشكل(5) متكافئان بينما طرفا الشكل(4) ليسا كذلك!

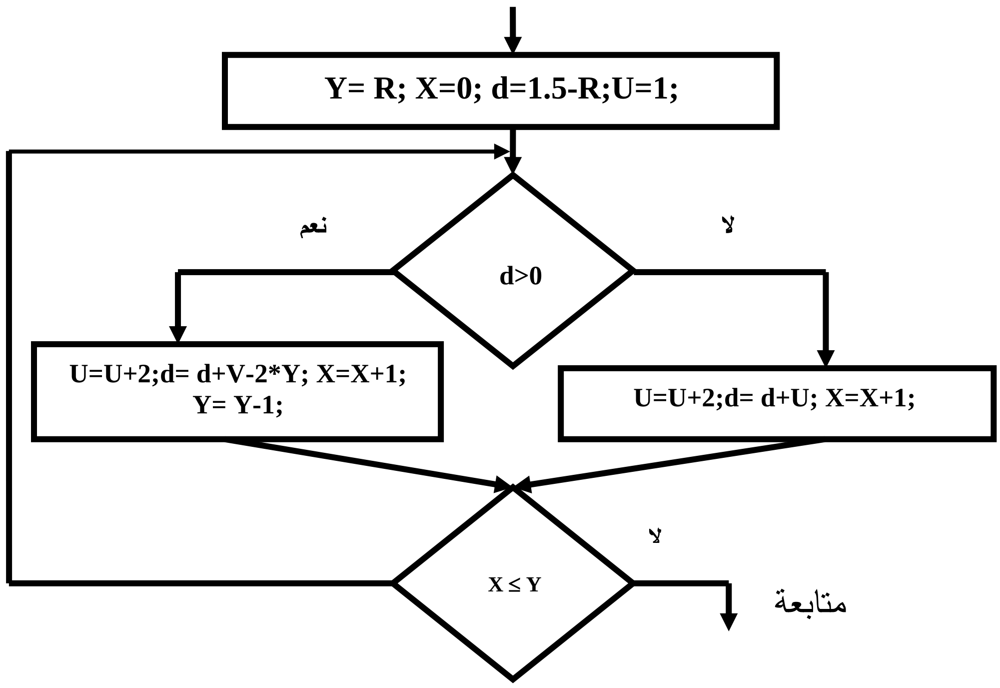
يمكن التأكد من أن المشكلة ليست في الطرف الأيسر من خلال اختباره لمفرده، للشكل(4) لدينا:

```
tic;X=0; R= 10000000; d=1.5-R;Y=R;while (X<=Y)
d=d+2*X-2*Y+1;
X=X+1;
Y=Y-1;
end; l=toc
    l =
        29.1123
```

و للشكل(5) لدينا:

```
tic;X=0; R= 10000000; d=1.5-R;Y=R;U=1;V=1-2*R;while (X<=Y)
U=U+2;
V=V+4;
d=d+V;
X=X+1;
Y=Y-1;
end; t=toc
    t =
        26.9127
```

أي أن إدخال علاقات تكرارية إضافية قد خفض زمن التنفيذ بمقدار 7.56%، وعند الاكتفاء بمتحول إضافي واحد بدلاً من اثنين يصبح الشكل(5) معدلاً إلى الشكل(6)، وفي هذه الحالة سيصبح زمن التنفيذ أقل ما يمكن فهو من أجل R=10000000 يصبح 36.2453 ثانية بدلاً من 36.8760 لحالة بلا علاقات تكرارية إضافية ، وبدلاً من40.9170 لحالة وجود علاقات تكرارية مع متحولين ، و هذا يعني إنقاص الزمن بنسبة 1.71% للحالة الأولى ، وبنسبة 11.42% للحالة الثانية.



الشكل(6) جزء معدل من خوارزمية يحتوي على تفرع (إلى فرعين) وذات متحول إضافي واحد

الاستنتاجات و التوصيات:

إن إدخال علاقات تكرارية للخوارزميات يرفع من سرعة تنفيذها في حالات محددة ، وهذه الحالات تم إلقاء الضوء عليها في سياق هذا البحث ، ومن أجل حالات أخرى يجب تحليلها من حيث التعقيد الحسابي ، ومن ثم التأكد من صحة ذلك برمجياً .

إن إدخال علاقات تكرارية للخوارزميات ذات المتحولات من نوع صحيح له تأثير أكبر مقارنة مع الخوارزميات ذات المتحولات من نوع حقيقي ، وهذا ما تم التأكد منه في هذا البحث ، و لا شك في أن هذه الخوارزميات ذات المتحولات من نوع صحيح غالباً ما يتم تطبيقها بالدارات الرقمية و خصوصاً في خوارزميات توليد الحركة في الأجهزة الروبوتية (آلات قص الألواح المعدنية، وأجهزة الحفر أو الرسم أو النقش على السطوح المعدنية مثل النحاس وآلات التفريز الرقمي).

المراجع:

1- BANACHOWSKI, L, KRECZMAR, A and RYTTER, W- *Analysis of algorithms and data structures.* ISBN Warsaw Poland, 1989,220pp
2- BRESENHAM, J. E- *Algorithms for circular arc generation*, Fandametal Algorithms for computer Graphics. Yorshie, England 1985**,** pp 197- 217
3- CARLET, C , FOSSORIER, M and JMESI, H- *Applied Algebra*. Las Vegas, NV 2006, 339p
4- DAWARA, S-*Mastering Graphics Programming in C*, FIRE wall Media 2005,286p
5- DUCAMP, M and REVERCHON, A **-** *Mathematiques et Graphisme sur Apple II.* Eyrolles, Paris, 1985, 300p.
6- JANKOWSKA, J and JANKOWSKI, M- *Survey of numerical methods and algorithms***.** Part 1,2. ISBN Warsaw, Poland 1989, 216pp
7- MOKRZYSKI, W- *Some algorithms of algebraic curve discretization on homogeneous multiconnected grids.* Comp. and Artificial Intelligence. vol.9, No:1, Bratislava, Slovakia 1990, pp 269- 273
8- MÖLLER, O- *Quasi-double-precision in floating point addition*.BIT, No: 5, 1965, pp 37-50
9- POLANSCY, R and Z- *Algorithms and programs for engineers***,** SIGMA- NOT, Warsaw Poland, 1991,312pp

http://www.mathworks.com/pl_homepage MATLAB\R2006a