*جامعة تشرين*

*كلية الهندسة المعلوماتية*

*قسم البرمجيات و نظم المعلومات*

*السنة الرابعة*

*Transitive closure*

*مشروع فصلي*

*إعداد*

*راميا يونس زينب جمعة*

*إشراف*

*الدكتور رائد جابري*

***مقدمة:***

* ***العديد من الأشياء في العالم لم تكن لتظهر للوجود لو لم يكن هناك مشكلة في الأساس بحاجة لحل.***
* ***هذه الحقيقة تشمل كل شيء يطرأ في حياتنا اليومية إلا أنها تبرز في علوم الحاسوب بكثرة وفي السنوات القليلة الماضية قدمت نظرية البان نفسها على أنها أداة رياضية هامة في مجموعة واسعة من المواضيع بدءاً من البحث العلمي و الكيمياء إلى علم الوراثة و اللغويات والهندسة الكهربائية والجغرافية ....إلخ.***
* ***ومن تلك المشاكل الهامة والتي تحتاج لحل مناسب بزمن فعال مشكلة قابلية الوصول أي الإجابة عن السؤال التالي: هل بالإمكان الوصول من وجهة ما إلى هدف محدد؟***

***والجواب سيكون نعم أم لا لكن بزمن فعال جداً .***

* ***ومن هنا سنرى أننا بحاجة إلى التفكير ببنية معطيات تجعل من الممكن الإجابة عن أسئلة قابلية الوصول بفعالية.***
* ***بنية المعطيات هذه تدعى بال Transitive closure سيصبح وبعد استخراجها سيصبح بالإمكان تحديد قابلية الوصول بزمن constant time أي O(1) .***
* ***سنتناول عدة طرق لحساب ال Transitive closure لكل منها حسنات و سلبيات مع مقارنات بين تلك الطرق من حيث زمن التنفيذ.***

***الفهرس :***

1. *مقدمة........................................................................................ 2*
2. *الفهرس....................................................................................... 3*
3. *المدخل....................................................................................... 4*
4. *الفصل الأول 5*
   1. *التعريف بالمسألة....................................................................... 6*
   2. *الطرق المستخدمة...................................................................... 7*
   3. *خوارزمية وارشال....................................................................... 7*
   4. *خوارزمية تكرار البحث بالعمق......................................................... 12*
   5. *الطريقة المقترحة بالاعتماد على ال Scc ................................................ 16*
      1. *مفهوم المكونات المرتبطة بقوة.................................................. 16*
      2. *خوارزمية تارجان............................................................. 18*
      3. *خوارزمية تارجان المحسنة...................................................... 22*
      4. *طريقة ال chunk graph.................................................... 25*
5. *الفصل الثاني 35*
   1. *جداول المقارنات......................................................................36*
6. *المراجع......................................................................................72*

***مدخل:***

*لدينا الفصل الأول كما يلي:*

* ***في البداية تم تعريف مسألة ال*** *Transitive closure* ***بالاعتماد على المراجع*** *[1,2]* ***.***
* ***من ثم تطرقنا للطرق المستخدمة لحسابه و أولها*** *Warshall Algorithm* ***وقد تم تعريفها بالاعماد على المراجع****,4] [1* ***ثم استعرضنا طريقة قائمة على تكرار خوارزمية البحث بالعمق من أجل كل عقدة و تم الاعتماد على المرجع*** *[2]* ***من أجل توصيفها , يأتي بعد ذلك طريقة مقترحة قائمة على استخراج ما نسميه ب*** *Chunk graph* ***والذي يعتمد على رصد المكونات المرتبطة بقوة في بيان موجه وقد تم استخدام خوارزميتين من أجل إيجاد تلك المكونات وهما :***

***خوارزمية*** *تارجان* ***وتم توصيفها بالاعتماد على المراجع*** *[1,2,3,5,6]* ***وخوارزمية تارجان المحسنة والتي تم توصيفها أيضاً بالاعتماد على المراجع*** *[1,3,6] .*

*وفي الفصل الثاني سنتستعرض جداول مقارنات من حيث زمن التنفيذ للخوارزميات وستبرز مزايا الخوارزميات من أجل حالات معينة.*

***أهداف المشروع :***

* ***مقارنة بين الخوارزميات المستخدمة من حيث زمن التنفيذ.***
* ***إيجاد طرق فعالة للإجابة عن قابلية الوصول بزمن*** *Constant time****.***
* ***طرح حل مقترح (فكرة مقترحة) لإيجاد ال*** *Transitive closure* ***بالاعتماد على مفهوم ال Scc.***

***الأدوات المستخدمة:***

* ***لغة الجافا لما تدعمه من حيث البرمجية غرضية التوجه وسهولة التعامل معها***
* ***كما تم أيضاً استخدام*** *مكتبة JGraphT* ***من أجل رسم بيان الخرج.***

***الفصل الأول:***

*نعلم بأن البيان هو بنية هامة جداً لوصف العديد من المسائل أو المشاكل التي يتوجب حلها بأفضل الطرق .*

*مثلا :ليكن لدينا مجموعة مدن ومجموعة الطرق التي تصل بينها,يُعبر عن كل مدينة بعقدة في البيان وكل طريق يصل بين مدينتين هو وصلة في ذلك البيان .*

*إذا كنا مهتمين بمعرفة إذا كان هناك طريق من المدينة A الى مدينة B نستطيع تطبيق خوارزمية البحث بالعمق* DFS *أو البحث بالعرض BFS بدءاً من A ومن ثم نلاحظ فيما إذا كانت B قد تمت زيارتها وذلك بزمن O(n+m) حيث*

*n : عدد العقد (المدن) و m : عدد الوصلات (الطرق)*

*لكن كما نلاحظ في العديد من التطبيقات الهامة جداً يكون من الضروري الإجابة عن مسائل قابلية الوصول "Reachability" بشكل فعّال أكثر ومن هنا جاء مفهوم ال Transitive closure وهو الإجابة عن Reachability Quires بزمن : O(1) Constant time .*

*سنبدأ بالتعريف التالي لل Transitive closure :*

*ليكن G=(V,E) بيان موجه حيث أن V :مجموعة منتهية من العقد و E :مجموعة منتهية من الوصلات .*

*إن ال Transitive closure من أجل البيان الموجه G هو بيان موجه أيضا ً حيث يحوي وصلة ولتكن (U,V) إذا وفقط إذا يوجد طريق "a non-null path" من U إلى V في البيان الأصلي G .*

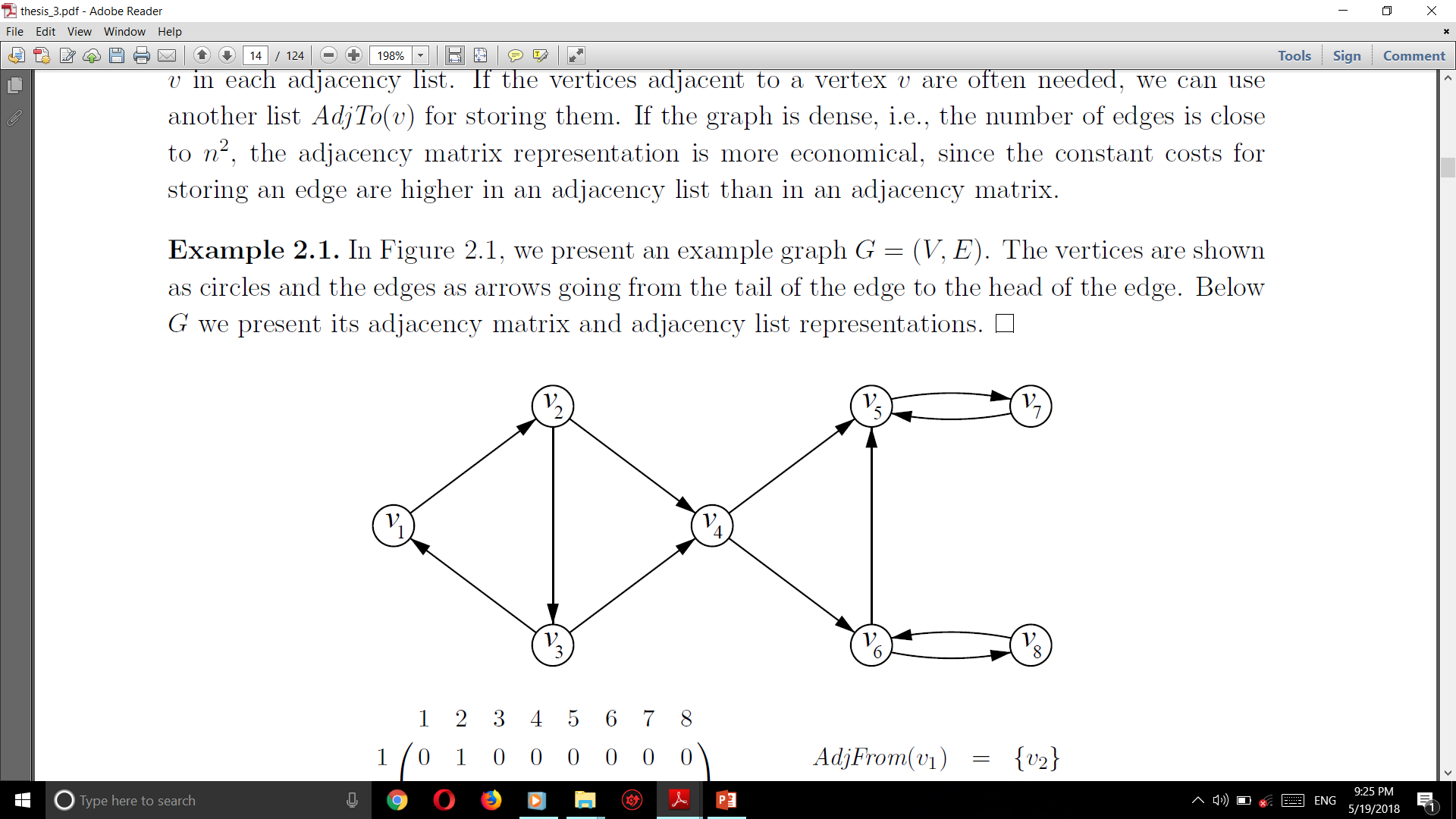
*تعريف : الطريق من العقدة بحيث أن كل وصلة تنتمي لE مجموعة الوصلات وطول الطريق هو عدد الوصلات المكونة له .*

*a non-null path :يرمز له ب وهو الطريق الذي طوله موجب .*

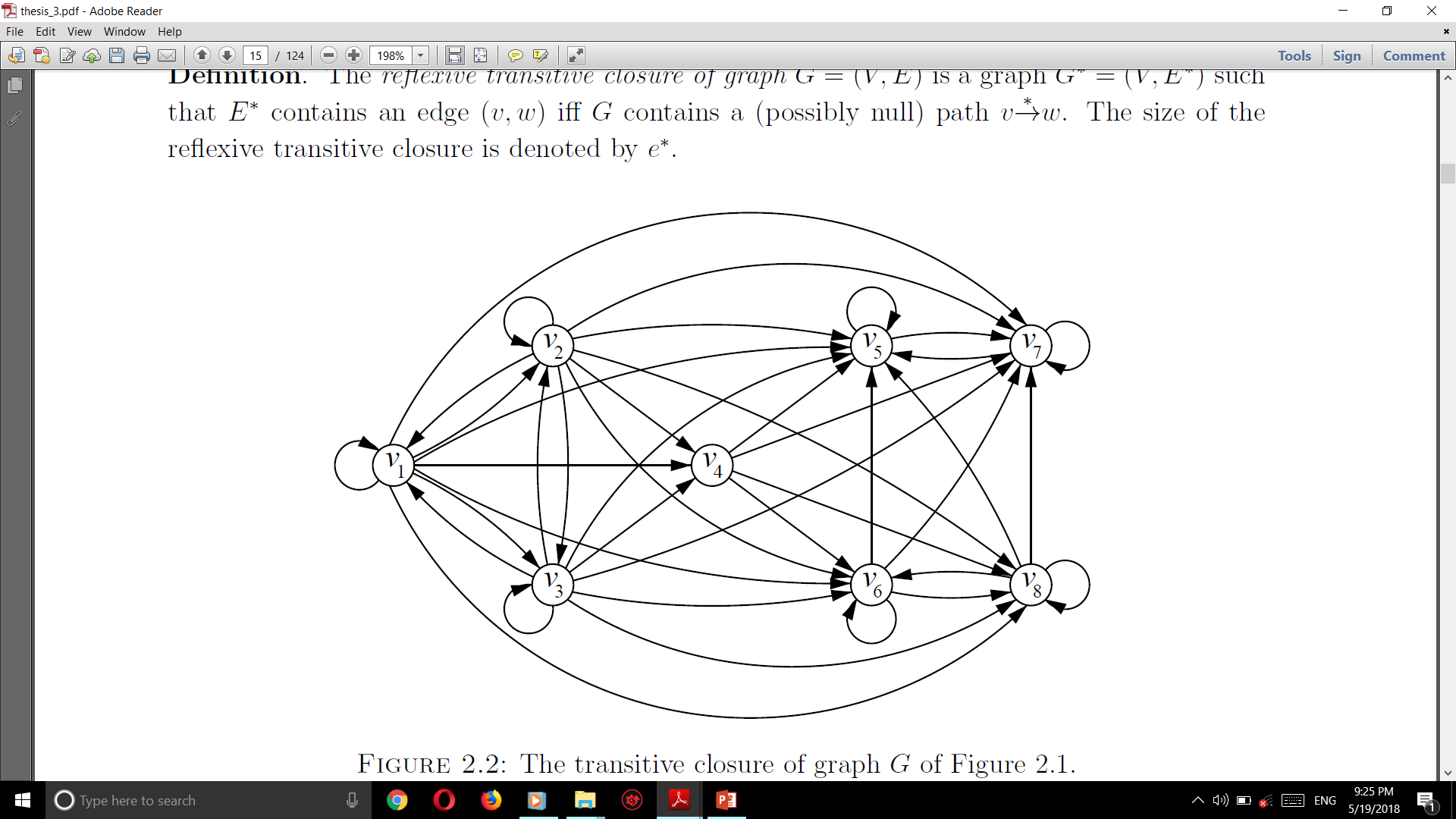
*سيتم تمثيل بيان الخرج بمصفوفة ثنائية البعد بحيث يكون تقاطع السطر i مع العمود j في تلك المصفوفة يساوي الواحد إذا وفقط إذا كان يوجد a non-null path من العقدة من i إلى j وصفر عدا ذلك وندعو تلك المصفوفة ب*

*Transitive closure of a graph .*

*Example :*

**

*شكل 1 يوضح بيان موجه*

**

*شكل 2 :يبين ال Transitive closureللبيان السابق*

*في هذا الفصل سنستعرض أهم الخوارزميات المستخدمة لحساب*

*ال Transitive closure*  *من أجل بيان موجه مع حل مقترح علماً أنه تم استخدام طريقتي تمثيل للبيان في المشروع وهما*

*Adjacency matrix*

*و Adjacency* *array*

***والطرق المستخدمة هي :***

*1) Warshall’s Algorithm .*

*2) Transitive closure using DFS traversal .*

*3) Transitive closure depending on detecting strongly connected components* ..

*أولا :*

*1) Warshall’s Algorithm :*

*وهي من أشهر الخوارزميات المعروفة لحساب Transitive closure سنبدأ بتعريف لها ثم نقوم على شرح فكرتها*

*Let G=(V,E) a directed graph .*

*If G contains paths v🡪\*w and w🡪\*v*

*Whose intermediate vertices come from a specify sets S, then the graph also contains a path u🡪\*v such that its intermediate vertices come from the set Su{w} .*

*خوارزمية وارشال تكرر من 0 ل V وفي اللفة ذات الرقم K تدرس الطرق التي عقدها تأتي من المجموعة*

{ , ,,…., }

*وفكرتها على الشكل التالي :*

*سنحسب كل الطرق التي تحوي العقدة 1 ثم كل الطرق التي تحوي العقدة 1 أو 2 أو2 وهكذا حتى نحسب جميع الطرق من عقد مختارة من مجموعة محددة*

*S={1 , 2 ,…….. ……….. , r}*

*علماً أنه يتم تمثيل بيان الدخل Adjacency matrix ولتكن A وسنقوم بحساب متتالية من المصفوفات .*

, *بحيث أن*

*Tc : is the path matrix*

*و تعبر عن الطرق للعقد المختارة من مجموعة العقد*

*S={1 , 2 ,…….. ……….. , r}*

*نبدأ عمل الخوارزمية بالشكل التالي :*

*Let = A*

*Let a matrix such that :*

*1 : if and only of there is a path*

*Connecting nodes i and k through*

*one or more nodes {1 , 2 , ……, r}*

*0 : other wise*

* *Let a matrix where*

*1 : if and only of there is a path*

*Connecting nodes i and k* through one or more of nodes {1, 2, …, r, r+1}

*=*

*0 : otherwise*

*نحصل على على الشكل التالي :*

*if then*

*else AS a result we have :*

*1: if and only of there is a path*

*Connecting nodes i and k through*

*one or more nodes {1 , 2 , ……, r}*

*0 : other wise*

*نعبر عن هذه الخوارزمية بالشكل :*

*Input : G=(V,E) where G is a directed graph represented by adjacency matrix*

*Output : Tc[ ][ ] : is the transitive closure of G.*

*1-initalize Tc so that Tc[ ][ ]=A[ ][ ]*

*for int k=0,k<v do*

*for int i=0,i<v, do*

*for int j=0, j<v do*

*Tc[i,j]=Tc[i , j] or (Tc[i,k] and Tc[k,j] )*

*Example:*

شكل 3 يوضح بيان الدخل

*A==*

*n=1 :*

*=*

*=*

*=*

شكل 4 يوضح بيان الخرج

ملاحظة هامة : يمكن جعل عناصر القطر الرئيسي مساوية للواحد أو يمكن إهمالها كونها

*فمن الواضح أن كل عقدة قابلة للوصول من نفسها.Trivial components*

*تعقيد الخوارزمية : نلاحظ أن لدينا 3 حلقات for ضمن هذه الخوارزمية وكل حلقة O(V) بالتالي تعقيد هذه الخوارزمية O(= O(V\*V\*V)*

* *ومن هنا نستطيع القول بأن هذه الخوارزمية ليست فعالة من أجل بيان قليل الوصلات أي* *Sparse graph .*

***ثانياً :***

*Transitive closure using DFS traversal* :

* *في البداية سنتطرق لمبدأ عمل خوارزمية البحث بالعمق حيث تستخدم هذه الخوارزمية مبدأ العودية وتقوم على زيارة كل عقدة وكل وصلة في البيان يمكن الوصول إليها ابتداء من عقدة معينة مرة واحدة فقط وفق ترتيب محدد .*
* *عند اختيار عقدة البداية ولتكن U يتم وضع علامة على هذه العقدة بأنه قد تمت زيارتها ثم ننظر لجميع الوصلات (U,V) .*
* *إذا كانت V مزارة من قبل عندها نعود للعقدة U ونبحث عن عقدة أخرى V غير مزارة .*
* *إذا كانت V غير مزارة عندها يتم تعليمها بأنها مزارة ويتم جعلها العقدة الحالية ثم نعود ونكرر ما ذكرناه من عمليات حتى نصل لحالة أن كل الأضلع التي تبدأ من العقدة الحالية U تعود إلى عقد تمت زيارتها سابقا .*
* *إذا بقي في البيان عقد غير مكتشفة يتم اختيار إحداها كنقطة بداية ويتم تكرار ما سبق من خطوات .*
* *من أجل إيجاد ال Transitive* *closure سنقوم بتكرار خوارزمية البحث بالعمق من أجل كل عقدة في البيان وسيتم اكتشاف فيما إذا كانت عقدة ما ولتكن V قابلة للوصول reachable من عقدة أخرى ولتكن U وذلك من*

*أجل جميع أزواج العقد (U,V) في البيان المُعطى .*

***ملاحظة :*** *نقصد بقابلية الوصول أي أنه يوجد طريق من العقدة U إلى العقدة V وسيتم تمثيل الخرج بمصفوفة ثنائية البعد تعبًر عن بيان الخرج .*

*خطوات الطريقة الأولى :*

* *نُنشئ مصفوفة 2D* *matrix بحجم V حيث V هو عدد عقد البيان المعطى ولتكن هذه المصفوفة Tc[v][v] والتي ستمثل في النهاية*

*The Transitive closure of a given graph.*

* *نقوم بتهيئة جميع عناصر هذه المصفوفة بالقيمة 0 .*
* *نستدعي خوارزمية البحث بالعمق من أجل كل عقدة في البيان وذلك لتعليم "تحديد" ال reachable vertices في المصفوفة .Tc*

***ملاحظة :****أثناء الاستدعاء العودي لن نستدعي DFS من أجل عقدة U مجاورة ل* V *إذا كانت*

*Tc[u][v]=1 أي تمت زيارتها سابقا .*

*وفيما يلي pseudocode الخوارزمية :*

*Input : G=(V,E) | G is a Dgraph.*

*Output : Tc[][] | TC is the transitive closure of a given graph G.*

*1.1 init ()*

*1.2 - initialize all entries of Tc[][] as 0.*

*2.1 - For each vertex vV do*

*2.2- DFSVisit (V,V);*

*3.1- DFSVisit (int v ,int u )*

*3.1- Tc [u][v]=1*

*3.2- For each edge (v,w) E do*

*3.3- if Tc[v][w]=0 then*

*3.4- DFSVisit (v,u)*

*Example:*

شكل 5 يوضح بيان الدخل

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *V* | *DFSVisit (v,u)* | *Tc[U][V]=1* | *w* | *Tc[V][w]=1* |
| *0* | *DFSVisit(0,0)* | *Tc[0][0]=1* | *1* | *True* |
| *0* | *DFSVisit(0,1)* | *Tc[0][1]=1* | *2* | *True* |
| *0* | *DFSVisit(0,2)* | *Tc[0][2]=1* | *3* | *True* |
| *0* | *DFSVisit(0,3)* | *Tc[0][3]=1* | *-* | *-* |
| *1* | *DFSVisit(1,1)* | *Tc[1][1]=1* | *2* | *True* |
| *1* | *DFSVisit(1,2)* | *Tc[1][2]=1* | *3* | *True* |
| *1* | *DFSVisit(1,3)* | *Tc[1][3]=1* | *-* | *-* |
| *2* | *DFSVisit(2,2)* | *Tc[2][2]=1* | *3* | *True* |
| *2* | *DFSVisit(2,3)* | *Tc[2][3]=1* | *-* | *-* |
| 3 | DFSVisit(3,3) | Tc[3][3]=1 | - | - |

*نلاحظ أن زمن تنفيذ هذه الخوارزمية هو O(V(V+E)) = O()*

شكل 6 يوضح بيان الخرج

*ثالثا :*

*3) Transitive closure depending on deleting strong connected components.*

* *في البداية سنتطرق لمفهوم المكونات المتصلة بقوة ونقول عن عقدتين wوv أنهما متصلتان بقوة إذا كان هناك طريق من v إلى w ومن w إلى v ونستطيع القول أن تلك العقدتين متكافئتان ويمكن أن نكتب v وندعو تلك العلاقة بعلاقة تكافؤ والتي تحقق الخواص الهامة التالية :*

1. *علاقة انعكاسية : أي w ذلك لأن هناك طريق طوله صفر من w إلى w وبالعكس .*
2. *علاقة تناظرية : إذا كان v أيضاً وهذا الامر محقق بالتعريف .*
3. *علاقة متعدية : إذا كان هناك طريق من v لw وطريق من w إلى العقدة x بالتالي يوجد طريق من v إلى x .*

*إن علاقة التكافؤ تلك تحدد تقسيمات أو ما نسميه strong component لعقد البيان وبحيث كل strong component يحوي عقدا وكل زوج من تلك العقد في المكّون الواحد متكافئ .*

*وهذا ما ندعوه بالمكونات المتصلة بقوة لبيان الموجه .*

*الان سوف نستعرض طريقتي حساب للمكونات المتصلة بقوة وهما :*

1. *خوارزمية تارجان .*
2. *خوارزمية تارجان المعدّلة .*

***أولا خوارزمية تارجان :***

* *سميت بهذا الاسم نسبة إلى مبتكرها البروفسور روبرت تارجان وهي تعتمد على خوارزمية البحث بالعمق بالإضافة لمكدّس .*
* *الخوارزمية تكتشف كل العقد في البيان باستخدام خوارزمية البحث بالعمق إذ تبدأ من عقدة مدخلة وتتجاهل جميع العقد التي تمت زيارتها سابقا .*
* *يتم وضع العقد في المكدس بالترتيب الذي تم اكتشافها فيه .*

*قبل البدء بخوارزمية تارجان يجب التطرق لمفهوم Low link value و Dfs-number لعقدة ما وهما على الشكل التالي :*

* *Dfs-number :*

*عند المرور على العقد في خوارزمية البحث بالعمق يتم إعطاء كل عقدة id مميز لها يعرف باسم Dfs*-*number وهذا الid يختلف باختلاف الترتيب الذي تتم زيارة العقد من خلاله .*

* *Low link value :*

*إن Low link value لعقدة ما هو الid الأصغر التابع لعقدة ما في نفس المكون المتصل بقوة.*

*لكن يجب ملاحظة الامر التالي :*

1. *إن قيم* Low link *تعتمد كثيرا على عقدة البداية* Dfs *وعلى أي وصلات تمر أولا وهذا قد يؤدي لنتائج خاطئة في قيم* Low link *وهنا تقدم خوارزمية تارجان حلا لتلك المشكلة وذلك من خلال المكدس الذي يمنع المكونات المرتبطة بقوة بأن تؤثر على قيم* Low link *لعقد تلك المكونات حيث سيحوي المكدس العقد المتاحة التي تستطيع تحديث قيم* Low link *منها .*
2. *تضاف العقد إلى المكدس عندما تكتشف للمرة الأولى ويتم حذفها في كل مرة نجد فيها مكون مرتبط بقوة .*

* *شرط تحديث قيمة Low**link لعقدة* ***:***

*إذا كانت* vوu *عقد في البيان وقد قمنا حاليا باكتشاف v عندها لتحديث قيمة Low* *link* *للعقدة u إلى قيمة Low link للعقدة v يجب أن يتحقق شرطان :*

1. *هناك طريق من* u *إلى* v *.*
2. *العقدة* v  *يجب أن تكون في المكدس .*

* *خطوات الخوارزمية على الشكل التالي :*

1. *نجعل ال Dfs*-*num لكل عقدة مساويا ل-1 وذلك للدلالة على أن تلك العقدة غير مزارة بعد .*
2. *نختار عقدة بداية لتكون نقطة انطلاق ال*Dfs *.*
3. *عند زيارة عقدة نعطيها قيمة Dfs*-*num موجبة وهذه القيمة ستدل على أن تلك العقدة قد تمت زيارتها .*

*وأيضاً نعطيها قيمة Low* *link* *تكون في البداية مساوية لقيمة* Dfs-num *ثم نضيف تلك العقدة إلى المكدس .*

1. *ال Dfs-callback :*

*إذا كانت العقدة السابقة ولتكن u في المكدس عندها ستأخذ هذه العقدة الحالية ولتكن v قيمة Low link على الشكل التالي :*

*Low link [u]=min (Low link [v] , Low link[u] )*

1. *بعد زيارة كل العقد إذا كانت العقدة الحالية قد بدأت مكون مرتبط بقوة عندها سنسحب العقد من المكدس حتى نحصل للعقدة الحالية في المكدس .*
2. *إذا بقي لدينا عقد غير مزارة نعود ونختار عقدة بداية ونكرر ما سبق من عمليات .*

* *لدينا المبرهنة الهامة التالية :*

*تكون عقدة ما ولتكن u جذر للمكون المرتبط بقوة إذا وفقط إذا كان :*

*Dfs-num [u] =low link [u]*

* *وهنا لدينا pseudocode التالي للخوارزمية :*
* *init ( )*

*counter =0 ,*

*for each vertex u do*

*Dfs-num[v]=-1*

* *for each vertex x do*

*if Dfs-num[x]=-1 Then*

*visit(x)*

* *visit (u)*

*Dfs-num[u]=counter*

*low link [u]=counter*

*counter=counter+1*

*Push u into the stack*

*for each edge (u,w) do*

*if Dfs-num[w]=-1 Then*

*visit(w)*

*Low link [u]=min (Low link [u] , Low link[w] )*

*Else if Dfs-num(w)<Dfs-num(v) Then*

*if w*

*Low link [u]=min (Low link [u] , Low link[w] )*

*if Low link [u]= Dfs-num(u) Then*

*pop vertices off the stack*

*Dfs-num(stack, top)≥ Dfs-num[u]*

* *إن زمن تنفيذ خوارزمية تارجان هو O(V+E) حيث أنها قائمة على Dfstrarersal .*

شكل 7يوضح بيان موجه

لتكن العقدة 1 هي عقدة البداية :

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *num(S.top)*  *>=num(u)* | *Low(u)*  *=num(u)* | *w€S* | *num(w)*  *<num(u)?* | *w is new?* | *w* | *S* | *count* | *num=low* | *operation* | *u* |
| ---- | ---- | ---- | ---- | T | 2 | {1} | 1 | 0 | Visit(1) | 1 |
| ---- | ---- | ---- | ---- | T | 3 | {2,1} | 2 | 1 | Visit(2) | 2 |
| ---- | ---- | T | T | F | 1 | {3,2 1} | 3 | 2 | Visit(3) | 3 |
| ---- | F | ---- | ---- | ---- | -- | {3,2 1} | 3 | ---- | Low(3)=0 | 3 |
| ---- | ---- | T | T | F | 3 | {3,2 1} | 3 | ---- | ---- | 2 |
| ---- | F | ---- | ---- | ---- | -- | {3,2 1} | 3 | ---- | Low(2)=0 | 2 |
| T | T | ---- | ---- | ---- | -- | {3,2 1} | 3 | ---- | ---- | 1 |
| T | ---- | ---- | ---- | ---- | -- | {2,1} | 3 | ---- | Pop(3) | 1 |
| T | ---- | ---- | ---- | ---- | -- | {1} | 3 | ---- | Pop(2) | 1 |
| ---- | ---- | ---- | ---- | ---- | -- | {} | 3 | ---- | Pop(1) | 1 |
| ---- | ---- | F | T | F | 3 | {0} | 4 | 3 | Visit(0) | 0 |
| ---- | ---- | F | T | F | 1 | {0} | 4 | ---- | ---- | 0 |
| ---- | ---- | ---- | T | T | 4 | {0} | 4 | ---- | ---- | 0 |
| ---- | ---- | F | T | F | 1 | {4,0} | 5 | 4 | Visit(4) | 4 |
| ---- | ---- | T | T | F | 0 | {4,0} | 5 | ---- |  | 4 |
| ---- | F | ---- | ---- | ---- | -- | {4,0} | 5 | ---- | Low(4)=3 | 4 |
| T | T | ---- | ---- | ---- | -- | {4,0} | 5 | ---- |  | 0 |
| T | ---- | ---- | ---- | ---- | -- | {0} | 5 | ---- | Pop(4) | 0 |
| F | ---- | ---- | ---- | ---- | -- | {} | 5 | ---- | Pop(0) | 0 |

*شكل 8 يوضح بيان الخرج*

***ثانيا خوارزمية تارجان المحسنة :***

*على الرغم من أن تارجان تقريباً مثالية من أجل إيجاد المكونات المرتبطة بقوة إلا إنها تقوم ببعض العمل غير الضروري .*

* *إذا كان بيان الدخل غير حلقي كما يوضح الشكل عندها كل مكون مرتبط بقوة سيتألف من عقدة واحدة فقط وهكذا فإن وضع تلك العقد في المكدس وحذفها منه غير مجدي .*

*شكل 9 يمثل بيان غير حلقي*

* *أيضاً بالنسبة للبيانات الحلقية التي تحوي عدد قليل جداً من العقد المنتمية لحلقة واحدة سيكون هناك الكثير من المكونات المرتبطة بقوة المؤلفة من عقدة*

*واحدة فقط بالتالي ستكون عمليات الإضافة والحذف لعقد الجذر غير نافعة أبداً كما يوضح المثال التالي:*

*شكل 10 يوضح بيان يحوي حلقة وحيدة*

* *نلاحظ أنه بالإمكان الوصول إلى عقدة الجذر دون وضعها في المكدس ومن هنا جاءت خوارزمية تارجان المحسنة والتي تضع تعديلاً على التابع visit في الخوارزمية السابقة بأن يخزن عقدة v في المكدس فقط بعد المرور على جميع الوصلات الخارجة منها وبعد أن يتأكد بأنها ليس عقدة جذر .*
* *هنا ستصيح عملية استكشاف مكون جديد مختلفة قليلاً وذلك لأن عقدة الجذر ليست في المكدس .*
* *نلاحظ بأن كل عقدة ليست عقدة جذر للمكون المرتبط بقوة قد دخلت المكدس وذلك بعد اكتشاف جذر المكون المرتبط بقوة ولهذا سنزيل العقد من المكدس ما دام*

*dfs-num للعقدة التي في قمة المكدس أكبر من dfs-num لعقدة الجذر ذاتها .*

* *للتميز بين العقد التي تنتمي لنفس المكون وبين العقد الأخرى سنعرف شعاع بولياني*

*in-component قيمته الابتدائية تساوي False من أجل كل عقدة وعندما يتم اكتشاف مكون مرتبط بقوة بشكل كامل عندها سيكون in-component من أجل تلك العقد يساوي True .*

*وما يلي هو pseudocode الخوارزمية :*

* *init ( )*

*counter =0 ,*

*for each vertex u do*

*Dfs-num[v]=-1*

* *for each vertex v do*

*if Dfs-num[v]=-1 Then*

*visit(v)*

* *visit (u)*

*Dfs-num[u]=counter*

*low link [u]=counter*

*counter++*

*for each edge (u,w) do*

*if Dfs-num[w]=-1 Then*

*visit(w)*

*if (!in-component [w]) Then*

*Low link [u]=min (Low link [u] , Low link[w] )*

*if (root [u]= Dfs-num(u) )Then*

*in-component[u]=true*

*while (Top(stack)>u) do*

*w := (stack,pop)*

*in-component[w]=true*

*else push u into the stack*

*أن زمن تعقيد هذه الخوارزمية أيضاً O(V+E)*

*والآن بعد أن تعرفنا على مفهوم* *المكونات المرتبطة بقوة ودرسنا آلية إيجادها سنبدأ بشرح الطريقة الثالثة لإيجاد ال*

*بالاعتماد على المكونات المرتبطة بقوة, و سننشئ بيان جديد يدعى بال Transitive closure*

*كما يلي:Chunk graph*

* ليكن لدينا البيان الموجه *G=(V,E)*
* نطبق إحدى خوارزميتي إيجاد المكونات المتصلة بقوة للبيان G.
* ننشئ بيان جديد وليكن *chunk* وعقده {d1,d2,…. , dm} حيث تعبر كل عقدة عن مكون مرتبط بقوة والذي سيحوي عقدة على الأقل من عقد البيان G.
* البيان الجديد هو بيان موجه أيضاً ويحوي وصلة من العقدة di الى العقدة dj إذا وفقط إذا هناك وصلة في البيان G منطلقها عقدة في المكون المرتبط بقوة الموافق للعقدة di ومستقرها عقدة في المكون الموافق للعقدة dj.
* سيتم إهمال جميع الوصلات بين عقد كل مكون مرتبط بقوة وسنضيف العقد الباقية*.*





شكل 12 يوضح تشكيل

*Chunk graph*

شكل11 يوضح بيان الدخل

* *يجب ضمان عدم وجود وصلات موازية في البيان المتشكل وسنوضحها بالشكل التالي*
* *سنعتبر كل عقدة في ال chunk graph بعقدة الجذر للمكون المرتبط بقوة في البيان G.*
* *من أجل إيجاد هذا البيان تم إضافة شعاعين إلى خوارزميتي تارجان.*

*الشعاع الأول: root\_of وسنخزن فيه جذر كل عقدة من عقد البيان G*

*الشعاع الثاني: roots وفيها يتم وضع عقد الجذر للبيان.*

* *تم تمثيل ال chunk graph في الكود ب Adjacency matrix وهي: meta\_matrix[roots.size][roots.size]*
* *ثم نمر على كل وصلات البيان والتي هي من الشكل (u,v)*

*نأخذ جذر العقدة u كما يلي:*

*x= root\_of[u]*

*وجذر العقدة v بنفس الطريقة:*

*y= root\_of[v]*

*فإذا كان x* ≠ *y وكان meta\_matrix[x][y]=0 عندها سيتم إضافة تلك الوصلة إلى ال chunk graph*

*meta\_matrix[x][y]=0 إن الشرط يضمن عدم إضافة الوصلات المتوازية*

* *و فيما يلي pseudocode لإنشاء ال chunk Graph :*
* *Input: G=(V,E) | G is a directed graph*
* *Output: chunk graph represented by meta\_matrix[][]*
* *Run Tarjan or Improved Tarjan algorithm to detect Scc.*
* *Initialize meta\_matrix*
* *For int i=0; i<V do*

*x= root\_of[i]*

*for int j=0; j<V do*

*y= root\_of[j]*

*if (x≠y && (i,j) € E && meta\_matrix[x][y] =0 ) then*

*meta\_matrix[x][y] =1*

*:Transitive closure الآن لدينا حلان من أجل إيجاد ال*

*الحل الأول:*

* *سنقوم بتطبيق الطريقة الأولى من أجل إيجاد ال Transitive closure والتي هي تنفيذ ال DFS من أجل كل عقدة من عقد ال chunk graph ونضع الخرج في matrix جديدة وهي :*

*TC\_meta\_matrix*

* *نقوم بإنشاء التابع البولياني checkConnectivityFunction على الشكل التالي:*

*هذا التابع سيفحص عقدتين مدخلتين, فإذا كان لديهما نفس الجذر سيرد التابع قيمةtrue*

*وإلا إذا كان هناك وصلة بين جذري هاتين العقدتين في المصفوفة TC\_meta\_matrix سيرد true أيضاً*

*وغير ذلك سيرد قيمة false.*

*وال pseudocode للتابع كما يلي:*

* *checkConnectivityFunction(s,d)*

*if(root\_of[s]= root\_of[d] ) then*

*return true*

*else if( TC\_meta\_matrix [root\_of[s]] [root\_of[d]=1)*

*return true*

*else*

*return false*

* *الآن سنقوم بتشكيل مصفوفة الخرج النهائية ولتكن*

*transitve\_matrix[V][V]*

*وذلك من خلال التابع*

*fillTransitiveMatrix()*

*والذي سيمر على جميع وصلات البيان الأصلي G ونستدعي التابع السابق*

*checkConnectivityFunction()*

*من أجل كل عقدتين فإذا ردّ قيمة true نضيف وصلة إلى مصفوفة الخرج*

*: للتابع كما يلي pseudocode وفيما يلي ال*

* *fillTransitiveMatrix( )*

*for int i=0;i<V do*

*for int j=0; j<V do*

*if(checkConnectivityFunction(i,j) then*

*transitve\_matrix[i][j]=1*

*وهكذا يصبح بالإمكان الاستعلام عن قابلية الوصول بزمن constant time من خلال التابع*

*checkConnactivitySecondMethod*

*كما يلي:*

* *checkConnactivitySecondMethod(s,d)*

*return transitve\_matrix[s][d] ==1*

* *أما الحل الثاني يقوم على فكرة الاستغناء عن المصفوفة transitve\_matrix من أجل توفير حجمها وعمليات الحجز بالذاكرة إذ أنها تستهلك مساحة قدرها (V×V)*

*وسيتم الاستعاضة عنها بالتابع*

*checkConnectivityFunction()*

* *هذا الامر سيوفر في الذاكرة حجز المصفوفة transitve\_matrix لكن الاستعلام عما إذا كانت عقدة u قابلة للوصول من عقدة v سيكون أطول من الحل الأول بفارق زمني مقدر بال nanosecond*

*لكن وكما نلاحظ زمن الاستعلام لا يزال Constant time.*

*Chunk graph وفيما يلي مثال يوضح إنشاء ال*

*شكل 13 يوضح بيان الدخل*

*بعد تطبيق إحدى خوارزميتي تارجان سينتج لدينا المكونات الخمسة المرتبطة بقوة التالية :*

*Scc are : {0,1,2} {3} {4} {5,6,7} {8}*

*وسيكون ال Chunk v graph مبدئياً هكذا :*

*شكل 14 يوضح بداية تشكيل chunk graph*

* *كما نلاحظ قد تم إهمال جميع وصلات كل مكون مرتبط بقوة وإضافة جميع الوصلات وفي الكود قمنا بتمثيل كل عقدة من عقد ال chunck بعقد الجذر بالتالي يصبح البيان على الشكل التالي وبفرض أننا بدأنا بترتيب معين وكانت جذور المكونات المرتبطة بقوة هي} 0,3,4,5,8 {*

*شكل 15يوضح ال chunk graph*

* *Adjacency matrix representation for chunk graph:*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 0 | 3 | 4 | 5 | 8 |
| 0 | 0 | 1 | 1 | 0 | 0 |
| 3 | 0 | 0 | 1 | 1 | 0 |
| 4 | 0 | 0 | 0 | 0 | 1 |
| 5 | 0 | 0 | 0 | 0 | 1 |
| 8 | 0 | 0 | 0 | 0 | 0 |

*وثم سنقوم بتطبيق ال DFS على كل عقدة من عقد هذا البيان فنحصل على المصفوفة*

*Tc-meta-matrix[ ][ ] على الشكل التالي :*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 0 | 3 | 4 | 5 | 8 |
| 0 | 1 | 1 | 1 | 1 | 1 |
| 3 | 0 | 1 | 1 | 1 | 1 |
| 4 | 0 | 0 | 1 | 0 | 1 |
| 5 | 0 | 0 | 0 | 1 | 1 |
| 8 | 0 | 0 | 0 | 0 | 0 |

*فيصبح ال chunk بعد إيجاد ال Transitive closureكما يلي :*

*شكل 16 يوضح بيان الخرج النهائي*

* *نلاحظ أنه كل عقدة قابلة للوصول من نفسها وإذا كانت عقدة u قابلة للوصول إلى عقدة v فهذا يعني أنه كل العقد المشكلة للمكون الذي تعبر عنه العقد في U قابلة للوصول لكل العقد المشكلة للمكون الذي تعبر عنه العقدة V.*

الفصل الثاني

* *نوع المعالج المستخدم : Intel(R)core(TM)i5-5200u*
* *نوع نظام التشغيل :Windows 8*
* *الأزمنة مقاسة بالثانية .*

*: TC\_dfsإيجاد ال transitive clousre باستخدام خوارزمية ال dfs.*

*Sh\_Tarjan:إيجاد Shrunken graph بدون مصفوفة ال transitiveمع استخدام خوارزمية Tarjan.*

*Sh\_ImTarjan:إيجاد Shrunken graph بدون مصفوفة ال transitiveمع استخدام Improved Tarjan.*

*ShM\_Tarjan:إيجاد shrunken graph مع مصفوفة ال transitive مع استخدام خوارزمية . Tarjan*

*ShM\_ImTarjan:إيجاد shrunken graph مع مصفوفة ال transitive مع استخدام خوارزمية Improved Tarjan.*

*Warshall:إيجاد ال transitive closure باستخدام خوارزمية Warshall.*

*TScc :إيجاد Strongly Connected Component باستخدام خوارزمية Tarjan.*

*ImTScc :إيجاد Strongly Connected Component باستخدام خوارزمية Improved Tarjan.*

*MRep\_dfs :تطبيق خوارزمية dfs باستخدام adjancency matrix.*

*ARep\_dfs :تطبيق خوارزمية dfs باستخدام adjancency array.*

*تم تمثيل البيان باستخدام adjancency matrix*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***TC\_dfs*** | ***ShM\_Tarjan*** | ***True***  ***TC\_dfs*** | ***True***  ***ShM\_Tarjan*** | ***False***  ***TC\_dfs*** | ***False ShM\_Tarjan*** |
| 250 | 1250 | 0.048 | 0.009 | 2.333E-6 | 3.732E-6 | 4.66E-7 | 4.67E-7 |
| 250 | 2500 | 0.063 | 0.031 | 4.665E-6 | 8.864E-6 | - | - |
| 250 | 3750 | 0.072 | 0.01 | 2.799E-6 | 2.799E-6 | - | - |
| 250 | 5000 | 0.073 | 0.014 | 2.799E-6 | 3.266E-6 | - | - |
| 500 | 2500 | 0.385 | 0.0168 | 3.266E-6 | 5.131E-6 | 9.33E-7 | 4.67E-7 |
| 500 | 5000 | 0.426 | 0.018 | 2.333E-6 | 3.732E-6 | - | - |
| 500 | 7500 | 0.442 | 0.016 | 2.333E-6 | 4.199E-6 | - | - |
| 500 | 10000 | 0.466 | 0.015 | 2.333E-6 | 2.8E-6 | - | - |
| 1000 | 5000 | 2.779 | 0.08 | 4.66E-7 | 4.199E-6 | 2.799E-6 | 1.353E-5 |
| 1000 | 10000 | 2.947 | 0.037 | 2.8E-6. | 4.199E-6 | - | - |
| 1000 | 15000 | 2.985 | 0.038 | 2.799E-6 | 4.199E-6 | - | - |
| 1000 | 20000 | 3.036 | 0.043 | 2.333E-6 | 5.131E-6 | - | - |
| 2000 | 10000 | 21.785 | 0.364 | 2.333E-6 | 3.732E-6 | 9.33E-7 | 4.67E-7 |
| 2000 | 20000 | 22.610 | 0.12 | 2.799E-6 | 3.266E-6 | - | - |
| 2000 | 30000 | 22.637 | 0.153 | 2.333E-6 | 6.065E-6 | - | - |
| 2000 | 40000 | 22.99 | 0.139 | 3.265E-6 | 4.666E-6 | - | - |
| 2500 | 12500 | 42.323 | 0. 453 | 3.265E-6 | 3.266E-6 | 9.33E-7 | 4.66E-7 |
| 2500 | 25000 | 43.299 | 0.277 | 3.266E-6 | 4.199E-6 | - | - |
| 2500 | 37500 | 43.690 | 0.189 | 3.266E-6 | 4.199E-6 | - | - |
| 2500 | 50000 | 43.970 | 0.206 | 2.799E-6 | 4.199E-6 | - | - |

الجدول رقم (1)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز 2500

نلاحظ ان*ShM\_Tarjan* هي أسرع في زمن التنفيذ من *TC\_df*s وذلك من اجل نفس عدد العقد والوصلات

ولكن *TC\_dfs* أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من *ShM\_Tarja*n.

*تم تمثيل البيان باستخدام adjancency matrix*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***TC\_dfs*** | ***Sh\_Tarjan*** | ***True***  ***TC\_dfs*** | ***True***  ***Sh\_Tarjan*** | ***False***  ***TC\_dfs*** | ***False Sh\_Tarjan*** |
| 250 | 1250 | 0.04 | 0.003 | 8.864E-6 | 1.306E-5 | 9.34E-7 | 9.34E-7 |
| 250 | 2500 | 0.051 | 0.007 | 2.799E-6 | 6.624E-5 | - | - |
| 250 | 3750 | 0.058 | 0.002 | 2.335E-6 | 5.598E-5 | - | - |
| 250 | 5000 | 0.06 | 0.005 | 3.265E-6 | 1.663E-5 | - | - |
| 500 | 2500 | 0.352 | 0.0128 | 3.733E-6 | 1.702E-5 | 4.66E-7 | 9.33E-7 |
| 500 | 5000 | 0.376 | 0.011 | 2.799E-6 | 1.026E-5 | - | - |
| 500 | 7500 | 0.44 | 0.008 | 3.266E-6 | 7.465E-6 | - | - |
| 500 | 10000 | 0.512 | 0.013 | 3.733E-6 | 1.632E-5 | - | - |
| 1000 | 5000 | 2.884 | 0.076 | 9.34E-7 | 1.072E-5 | 3.266E-6 | 6.531E-6 |
| 1000 | 10000 | 2.923 | 0.026 | 3.266E-6 | 1.306E-5 | - | - |
| 1000 | 15000 | 3.006 | 0.026 | 2.8E-6 | 9.797E-6 | - | - |
| 1000 | 20000 | 3.055 | 0.035 | 2.333E-6 | 1.026E-5 | - | - |
| 2000 | 10000 | 2.862 | 0.301 | 3.733E-6 | 5.598E-6 | 9.33E-7 | 2.333E-6 |
| 2000 | 20000 | 22.482 | 0.102 | 2.799E-6 | 1.663E-5 | - | - |
| 2000 | 30000 | 23.064 | 0.091 | 2.799E-6 | 4.666E-6 | - | - |
| 2000 | 40000 | 23.284 | 0.121 | 3.265E-6 | 1.121E-5 | - | - |
| 2500 | 12500 | 43.217 | 1.311 | 2.333E-6 | 1.586E-5 | 2.333E-6 | 1.362E-6 |
| 2500 | 25000 | 44.544 | 0.165 | 2.333E-6 | 1.026E-5 | - | - |
| 2500 | 37500 | 43.271 | 0.165 | 2.8E-6 | 1.119E-5 | - | - |
| 2500 | 50000 | 45.475 | 0.124 | 2.799E-6 | 1.073E-5 | - | - |

الجدول رقم (2)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز2500

نلاحظ أن ***Sh\_Tarjan*** هي أسرع في زمن التنفيذ ***TC\_dfs*** وذلك من اجل نفس عدد العقد والوصلات

ولكن ***TC\_dfs*** أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من ***Sh\_Tarjan***.

*تم تمثيل البيان باستخدام adjancency matrix*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***ShM\_Tarjan*** | ***Sh\_Tarjan*** | ***True***  ***ShM\_Tarjan*** | ***True***  ***Sh\_Tarjan*** | ***False ShM\_Tarjan*** | ***False Sh\_Tarjan*** |
| 250 | 1250 | 0.009 | 0.003 | 3.732E-6 | 1.306E-5 | 9.34E-7 | 4.67E-7 |
| 250 | 2500 | 0.031 | 0.007 | 8.864E-6 | 6.624E-5 | - | - |
| 250 | 3750 | 0.01 | 0.002 | 2.799E-6 | 5.598E-5 | - | - |
| 250 | 5000 | 0.014 | 0.005 | 3.266E-6 | 1.663E-5 | - | - |
| 500 | 2500 | 0.0168 | 0.0128 | 5.131E-6 | 1.702E-5 | 9.34E-7 | 4.67E-7 |
| 500 | 5000 | 0.018 | 0.011 | 3.732E-6 | 1.026E-5 | - | - |
| 500 | 7500 | 0.016 | 0.008 | 4.199E-6 | 7.465E-6 | - | - |
| 500 | 10000 | 0.015 | 0.013 | 2.8E-6 | 1.632E-5 | - | - |
| 1000 | 5000 | 0.08 | 0.076 | 4.199E-6 | 1.072E-5 | 9.33E-7 | 4.67E-7 |
| 1000 | 10000 | 0.037 | 0.026 | 4.199E-6 | 1.306E-5 | - | - |
| 1000 | 15000 | 0.038 | 0.026 | 4.199E-6 | 9.797E-6 | - | - |
| 1000 | 20000 | 0.043 | 0.035 | 5.131E-6 | 1.026E-5 | - | - |
| 2000 | 10000 | 0.364 | 0.301 | 3.732E-6 | 5.598E-6 | 9.34E-7 | 4.67E-7 |
| 2000 | 20000 | 0.12 | 0.102 | 3.266E-6 | 1.663E-5 | - | - |
| 2000 | 30000 | 0.153 | 0.091 | 6.065E-6 | 4.666E-6 | - | - |
| 2000 | 40000 | 0.139 | 0.121 | 4.666E-6 | 1.121E-5 | - | - |
| 2500 | 12500 | 0. 453 | 1.311 | 3.266E-6 | 1.586E-5 | 1.399E-6 | 9.33E-7 |
| 2500 | 25000 | 0.277 | 0.165 | 4.199E-6 | 1.026E-5 | - | - |
| 2500 | 37500 | 0.189 | 0.165 | 4.199E-6 | 1.119E-5 | - | - |
| 2500 | 50000 | 0.206 | 0.124 | 4.199E-6 | 1.073E-5 | - | - |

الجدول رقم (3)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز2500

نلاحظ أن *Sh\_Tarjan* هي أسرع في زمن التنفيذ من *ShM\_Tarjan* وذلك من اجل نفس عدد العقد والوصلات.

وكما أن *Sh\_Tarjan* أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من *ShM\_Tarjan*.

تم تمثيل البيان من خلال ال *adjancancey array*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***TC\_dfs*** | ***ShM\_Tarjan*** | ***True***  ***TC\_dfs*** | ***True***  ***ShM\_Tarjan*** | ***False***  ***TC\_dfs*** | ***False***  ***ShM\_Tarjan*** |
| 250 | 1250 | 0.005 | 0.003 | 2.333E-6 | 3.733E-6 | 9.33E-7 | 9.33E-7 |
| 250 | 2500 | 0.006 | 0.0035 | 1.866E-6 | 3.265E-6 | - | - |
| 250 | 3750 | 0.008 | 0.0038 | 2.333E-6 | 2.799E-6 | - | - |
| 250 | 5000 | 0.01 | 0.0038 | 2.333E-6 | 3.732E-6 | - | - |
| 500 | 2500 | 0.018 | 0.006 | 2.332E-6 | 3.732E-6 | 4.67E-7 | 4.67E-7 |
| 500 | 5000 | 0.028 | 0.01 | 2.8E-6. | 6.532E-6 | - | - |
| 500 | 7500 | 0.035 | 0.01 | 3.732E-6 | 1.306E-6 | - | - |
| 500 | 10000 | 0.047 | 0.011 | 3.732E-6 | 6.998E-6 | - | - |
| 1000 | 5000 | 0.062 | 0.015 | 2.799E-6 | 1.073E-5 | 4.67E-7 | 4.67E-7 |
| 1000 | 10000 | 0.124 | 0.022 | 4.199E-6 | 1.586E-5 | - | - |
| 1000 | 15000 | 0.13 | 0.024 | 4.199E-6 | 1.772E-5 | - | - |
| 1000 | 20000 | 0.196 | 0.02 | 4.199E-6 | 1.539E-5 | - | - |
| 2000 | 10000 | 0.398 | 0.042 | 3.732E-6 | 1.306E-6 | 9.33E-7 | 9.33E-7 |
| 2000 | 20000 | 0.449 | 0.057 | 3.732E-6 | 1.166E-5 | - | - |
| 2000 | 30000 | 0.571 | 0.056 | 2.799E-6 | 1.539E-5 | - | - |
| 2000 | 40000 | 0.76 | 0.062 | 2.799E-6 | 1.399E-5 | - | - |
| 2500 | 12500 | 0.468 | 0.146 | 2.799E-6 | 1.399E-5 | 9.33E-7 | 4.67E-7 |
| 2500 | 25000 | 0.721 | 0.047 | 2.8E-6. | 2.099E-5 | - | - |
| 2500 | 37500 | 0.913 | 0.006 | 2.333E-6 | 1.586E-5 | - | - |
| 2500 | 50000 | 1.195 | 0.051 | 2.799E-6 | 1.446E-5 | - | - |

الجدول رقم (4)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز 2500

نلاحظ أن *ShM\_Tarjan* هي أسرع في زمن التنفيذ من *TC\_dfs* وذلك من اجل نفس عدد العقد والوصلات

ولكن *TC\_dfs* أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من *ShM\_Tarjan*.

تم تمثيل البيان باستخدام adjancency array

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***TC\_dfs*** | ***Sh\_Tarjan*** | ***True***  ***TC\_dfs*** | ***True***  ***Sh\_Tarjan*** | ***False***  ***TC\_dfs*** | ***False***  ***Sh\_Tarjan*** |
| 250 | 1250 | 0.005 | 0.003 | 1.866E-6 | 3.265E-6 | 4.6E-7 | 9.33E-7 |
| 250 | 2500 | 0.006 | 0.0031 | 2.333E-6 | 4.199E-6 | - | - |
| 250 | 3750 | 0.007 | 0.0035 | 2.8E-6 | 4.199E-6 | - | - |
| 250 | 5000 | 0.009 | 0.0036 | 2.333E-6 | 3.732E-6 | - | - |
| 500 | 2500 | 0.015 | 0.005 | 2.332E-6 | 3.266E-7 | 9.33E-7 | 9.34E-7 |
| 500 | 5000 | 0.029 | 0.007 | 2.799E-6 | 2.8E-7 | - | - |
| 500 | 7500 | 0.035 | 0.008 | 9.797E-6 | 5.598E-7 | - | - |
| 500 | 10000 | 0.047 | 0.009 | 3.732E-6 | 6.065E-7 | - | - |
| 1000 | 5000 | 0.061 | 0.015 | 2.799E-6 | 9.331E-6 | 9.33E-7 | 1.399E-6 |
| 1000 | 10000 | 0.111 | 0.019 | 3.733E-6 | 9.33E-6 | - | - |
| 1000 | 15000 | 0.134 | 0.019 | 2.799E-6 | 7.932E-6 | - | - |
| 1000 | 20000 | 0.159 | 0.021 | 4.199E-6 | 1.586E-5 | - | - |
| 2000 | 10000 | 0.262 | 0.054 | 2.8E-6 | 1.259E-5 | 1.399E-6 | 2.333E-6 |
| 2000 | 20000 | 0.447 | 0.058 | 3.732E-6 | 1.119E-5 | - | - |
| 2000 | 30000 | 0.573 | 0.059 | 2.8E-6. | 1.399E-5 | - | - |
| 2000 | 40000 | 0.632 | 0.416 | 2.8E-6 | 1.352E-5 | - | - |
| 2500 | 12500 | 0.45 | 0.078 | 2.799E-6 | 1.772E-5 | 9.33E-7 | 1.866E-6 |
| 2500 | 25000 | 0.645 | 0.043 | 3.732E-6 | 9.797E-6 | - | - |
| 2500 | 37500 | 0.846 | 0.069 | 2.799E-6 | 1.259E-5 | - | - |
| 2500 | 50000 | 1.195 | 0.093 | 3.733E-6 | 1.446E-5 | - | - |

الجدول رقم (5)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز2500

نلاحظ أن *Sh\_Tarjan* هي أسرع في زمن التنفيذ من *TC\_dfs* وذلك من اجل نفس عدد العقد والوصلات

ولكن *TC\_dfs* أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من *Sh\_Tarjan*.

تم تمثيل البيان من خلال ال *adjancency array*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***Sh\_Tarjan*** | ***ShM\_Tarjan*** | ***True***  ***Sh\_Tarjan*** | ***True***  ***ShM\_Tarjan*** | ***False***  ***Sh\_Tarjan*** | ***False***  ***ShM\_Tarjan*** |
| 250 | 1250 | 0.003 | 0.003 | 2.333E-6 | 3.265E-6 | 9.33E-7 | 9.33E-7 |
| 250 | 2500 | 0.0031 | 0.0035 | 1.866E-6 | 4.199E-6 | - | - |
| 250 | 3750 | 0.0035 | 0.0038 | 2.333E-6 | 4.199E-6 | - | - |
| 250 | 5000 | 0.0036 | 0.0038 | 2.333E-6 | 3.732E-6 | - | - |
| 500 | 2500 | 0.005 | 0.006 | 2.332E-6 | 3.266E-7 | 4.67E-7 | 9.34E-7 |
| 500 | 5000 | 0.007 | 0.01 | 2.8E-6. | 2.8E-7 | - | - |
| 500 | 7500 | 0.008 | 0.01 | 3.732E-6 | 5.598E-7 | - | - |
| 500 | 10000 | 0.009 | 0.011 | 3.732E-6 | 6.065E-7 | - | - |
| 1000 | 5000 | 0.015 | 0.015 | 2.799E-6 | 9.331E-6 | 4.67E-7 | 1.399E-6 |
| 1000 | 10000 | 0.019 | 0.022 | 4.199E-6 | 9.33E-6 | - | - |
| 1000 | 15000 | 0.019 | 0.024 | 4.199E-6 | 7.932E-6 | - | - |
| 1000 | 20000 | 0.021 | 0.02 | 4.199E-6 | 1.586E-5 | - | - |
| 2000 | 10000 | 0.054 | 0.042 | 3.732E-6 | 1.259E-5 | 9.33E-7 | 2.333E-6 |
| 2000 | 20000 | 0.058 | 0.057 | 3.732E-6 | 1.119E-5 | - | - |
| 2000 | 30000 | 0.059 | 0.056 | 2.799E-6 | 1.399E-5 | - | - |
| 2000 | 40000 | 0.416 | 0.062 | 2.799E-6 | 1.352E-5 | - | - |
| 2500 | 12500 | 0.078 | 0.146 | 2.799E-6 | 1.772E-5 | 9.33E-7 | 1.866E-6 |
| 2500 | 25000 | 0.043 | 0.047 | 2.8E-6. | 9.797E-6 | - | - |
| 2500 | 37500 | 0.069 | 0.006 | 2.333E-6 | 1.259E-5 | - | - |
| 2500 | 50000 | 0.093 | 0.051 | 2.799E-6 | 1.446E-5 | - | - |

الجدول رقم (6)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز 2500

نلاحظ أن *Sh\_Tarjan* أسرع في زمن التنفيذ من *ShM\_Tarjan* وذلك من اجل نفس عدد العقد والوصلات

وكما أن *Sh\_Tarjan* أسرع في التحقق من ارتباط عقدتين او عدم ارتباطهما مع بعضهما أكثر من *ShM\_Tarjan*.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***TC\_dfs*** | ***ShM\_ImTarjan*** | ***True***  ***TC\_dfs*** | ***True***  ***ShM\_ImTarjan*** | ***False***  ***TC\_dfs*** | ***False***  ***ShM\_ImTarjan*** |
| 250 | 1250 | 0.05 | 0.009 | 2.799E-6 | 2.799E-6 | 9.34E-7 | 9.33E-7 |
| 250 | 2500 | 0.06 | 0.006 | 2.333E-6 | 2.799E-6 | - | - |
| 250 | 3750 | 0.064 | 0.007 | 3.266E-6 | 2.799E-6 | - | - |
| 250 | 5000 | 0.069 | 0.009 | 2.8E-6 | 5.132E-6 | - | - |
| 500 | 2500 | 0.368 | 0.011 | 3.266E-6 | 7.931E-6 | 4.66E-7 | 9.34E-7 |
| 500 | 5000 | 0.385 | 0.012 | 3.733E-6 | 4.199E-6 | - | - |
| 500 | 7500 | 0.442 | 0.017 | 2.799E-6 | 3.732E-6 | - | - |
| 500 | 10000 | 0.448 | 0.018 | 2.799E-6 | 3.732E-6 | - | - |
| 1000 | 5000 | 2.83 | 0.067 | 3.266E-6 | 1.166E-6 | 4.67E-7 | 9.33E-7 |
| 1000 | 10000 | 2.972 | 0.033 | 3.733E-6 | 9.33E-6 | - | - |
| 1000 | 15000 | 3.051 | 0.0331 | 2.799E-6 | 7.465E-6 | - | - |
| 1000 | 20000 | 3.139 | 0.034 | 2.8E-6 | 8.864E-6 | - | - |
| 2000 | 10000 | 21.898 | 0.315 | 2.799E-6 | 1.912E-6 | 4.66E-7 | 9.33E-7 |
| 2000 | 20000 | 22.241 | 0.123 | 3.732E-6 | 1.119E-6 | - | - |
| 2000 | 30000 | 23.074 | 0.127 | 3.733E-6 | 1.073E-5 | - | - |
| 2000 | 40000 | 23.483 | 0.129 | 2.799E-6 | 1.119E-6 | - | - |
| 2500 | 12500 | 44.016 | 0.92 | 2.8E-6 | 1.772E-5 | 9.33E-7 | 1.866E-6 |
| 2500 | 25000 | 44.949 | 0.158 | 2.799E-6 | 1.258E-5 | - | - |
| 2500 | 37500 | 54.375 | 0.163 | 3.263E-6 | 7.465E-6 | - | - |
| 2500 | 50000 | 54.340 | 0.214 | 2.799E-6 | 1.632E-6 | - | - |

تم تمثيل البيان من خلال ال *adjancency matrix*

الجدول رقم (7)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز2500

نلاحظ أن *ShM\_ImTarjan* هي أسرع في زمن التنفيذ من *TC\_dfs* وذلك من اجل نفس عدد العقد والوصلات

ولكن *TC\_dfs* أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من *ShM\_ImTarjan*.

تم تمثيل البيان من خلال ال *adjancency matrix*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***TC\_dfs*** | ***Sh\_ImTarjan*** | ***True***  ***TC\_dfs*** | ***True***  ***Sh\_ImTarjan*** | ***False***  ***TC\_dfs*** | ***False***  ***Sh\_ImTarjan*** |
| 250 | 1250 | 0.05 | 0.005 | 2.799E-6 | 5.132E-6 | 4.66E-7 | 1.866E-6 |
| 250 | 2500 | 0.06 | 0.0054 | 2.332E-6 | 3.266E-6 | - | - |
| 250 | 3750 | 0.064 | 0.0059 | 2.8E-6 | 4.665E-6 | - | - |
| 250 | 5000 | 0.069 | 0.0085 | 3.732E-6 | 6.998E-6 | - | - |
| 50 | 2500 | 0.368 | 0.099 | 2.799E-6 | 3.266E-6 | 4.67E-7 | 9.33E-7 |
| 500 | 5000 | 0.415 | 0.01 | 2.799E-6 | 3.732E-6 | - | - |
| 500 | 7500 | 0.442 | 0.013 | 3.732E-6 | 3.265E-6 | - | - |
| 500 | 10000 | 0.448 | 0.0135 | 2.799E-6 | 3.733E-6 | - | - |
| 1000 | 5000 | 2.83 | 0.05 | 3.266E-6 | 4.198E-6 | 4.66E-7 | 1.866E-6 |
| 1000 | 10000 | 2.972 | 0.027 | 2.8E-6 | 4.199E-6 | - | - |
| 1000 | 15000 | 3.051 | 0.0272 | 2.333E-6 | 6.065E-6 | - | - |
| 1000 | 20000 | 3.139 | 0.028 | 2.8E-6 | 4.199E-6 | - | - |
| 2000 | 10000 | 21.898 | 0.681 | 2.799E-6 | 7.931E-6 | 4.673E-7 | 3.266E-6 |
| 2000 | 20000 | 22.241 | 0.08 | 2.332E-6 | 5.599E-6 | - | - |
| 2000 | 30000 | 23.074 | 0.086 | 2.799E-6 | 4.665E-6 | - | - |
| 2000 | 40000 | 23.483 | 0.0999 | 2.332E-6 | 5.598E-6 | - | - |
| 2500 | 12500 | 44.016 | 0.712 | 2.799E-6 | 7.465E-6 | 9.33E-7 | 2.799E-6 |
| 2500 | 25000 | 44.949 | 0.127 | 3.732E-6 | 5.132E-6 | - | - |
| 2500 | 37500 | 54.375 | 0.151 | 2.799E-6 | 7.464E-6 | - | - |
| 2500 | 50000 | 54.340 | 0.153 | 2.8E-6 | 6.998E-6 | - | - |

الجدول رقم (8)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز2500

نلاحظ ان *Sh\_ImTarjan* هي أسرع في زمن التنفيذ من *TC\_dfs* وذلك من اجل نفس عدد العقد والوصلات

ولكن *TC\_dfs* أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من *Sh\_ImTarjan*.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***Sh\_ImTarjan*** | ***ShM\_ImTarjan*** | | ***True***  ***Sh\_ImTarjan*** | ***True***  ***ShM\_ImTarjan*** | ***False Sh\_ImTarjan*** | ***False ShM\_ImTarjan*** |
| 250 | 1250 | 0.005 | | 0.009 | 5.132E-6 | 2.799E-6 | 1.866E-6 | 9.33E-7 |
| 250 | 2500 | 0.0054 | | 0.006 | 3.266E-6 | 2.799E-6 | - | - |
| 250 | 3750 | 0.0059 | | 0.007 | 4.665E-6 | 2.799E-6 | - | - |
| 250 | 5000 | 0.0085 | | 0.009 | 6.998E-6 | 5.132E-6 | - | - |
| 500 | 2500 | 0.099 | | 0.011 | 3.266E-6 | 7.931E-6 | 9.33E-7 | 9.34E-7 |
| 500 | 5000 | 0.01 | | 0.012 | 3.732E-6 | 4.199E-6 | - | - |
| 500 | 7500 | 0.013 | | 0.017 | 3.265E-6 | 3.732E-6 | - | - |
| 500 | 10000 | 0.0135 | | 0.018 | 3.733E-6 | 3.732E-6 | - | - |
| 1000 | 5000 | 0.05 | | 0.067 | 4.198E-6 | 1.166E-6 | 1.866E-6 | 9.33E-7 |
| 1000 | 10000 | 0.027 | | 0.033 | 4.199E-6 | 9.33E-6 | - | - |
| 1000 | 15000 | 0.0272 | | 0.0331 | 6.065E-6 | 7.465E-6 | - | - |
| 1000 | 20000 | 0.028 | | 0.034 | 4.199E-6 | 8.864E-6 | - | - |
| 2000 | 10000 | 0.681 | | 0.315 | 7.931E-6 | 1.912E-6 | 3.266E-6 | 9.33E-7 |
| 2000 | 20000 | 0.08 | | 0.123 | 5.599E-6 | 1.119E-6 | - | - |
| 2000 | 30000 | 0.086 | | 0.127 | 4.665E-6 | 1.073E-5 | - | - |
| 2000 | 40000 | 0.0999 | | 0.129 | 5.598E-6 | 1.119E-6 | - | - |
| 2500 | 12500 | 0.712 | | 0.92 | 7.465E-6 | 1.772E-5 | 2.799E-6 | 1.866E-6 |
| 2500 | 25000 | 0.127 | | 0.158 | 5.132E-6 | 1.258E-5 | - | - |
| 2500 | 37500 | 0.151 | | 0.163 | 7.464E-6 | 7.465E-6 | - | - |
| 2500 | 50000 | 0.153 | | 0.214 | 6.998E-6 | 1.632E-6 | - | - |

تم تمثيل البيان من خلال ال *adjancency matrix*

الجدول رقم (9)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز2500

نلاحظ ان *Sh\_ImTarjan* هي أسرع في زمن التنفيذ من *ShM\_ImTarjan* وذلك من اجل نفس عدد العقد والوصلات

ولكن *ShM\_ImTarjan* أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من *Sh\_ImTarjan*.

تم تمثيل البيان من خلال ال *adjancency array*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***TC\_dfs*** | ***ShM\_ImTarjan*** | ***True***  ***TC\_dfs*** | ***True***  ***ShM\_ImTarjan*** | ***False***  ***TC\_dfs*** | ***False***  ***ShM\_ImTarjan*** |
| 250 | 1250 | 0.005 | 0.004 | 2.335E-6 | 3.733E-6 | 4.66E-7 | 4.67E-7 |
| 250 | 2500 | 0.006 | 0.003 | 2.799E-6 | 3.266E-6 | - | - |
| 250 | 3750 | 0.008 | 0.005 | 2.335E-6 | 6.998E-6 | - | - |
| 250 | 5000 | 0.01 | 0.007 | 4.66E-6 | 6.065E-6 | - | - |
| 500 | 2500 | 0.025 | 0.008 | 2.332E-6 | 4.199E-6 | 9.33E-7 | 9.33E-7 |
| 500 | 5000 | 0.029 | 0.0081 | 2.799E-6 | 1.166E-5 | - | - |
| 500 | 7500 | 0.035 | 0.0084 | 4.199E-6 | 1.352E-5 | - | - |
| 500 | 10000 | 0.043 | 0.01 | 3.732E-6 | 5.132E-6 | - | - |
| 1000 | 5000 | 0.061 | 0.021 | 3.266E-6 | 1.213E-6 | 1.4E-6 | 9.33E-7 |
| 1000 | 10000 | 0.111 | 0.025 | 3.733E-6 | 1.166E-5 | - | - |
| 1000 | 15000 | 0.183 | 0.018 | 5.132E-6 | 1.353E-5 | - | - |
| 1000 | 20000 | 0.212 | 0.028 | 6.065E-6 | 1.306E-5 | - | - |
| 2000 | 10000 | 0.331 | 0.055 | 5.132E-6 | 1.446E-5 | 9.33E-7 | 1.399E-6 |
| 2000 | 20000 | 0.585 | 0.051 | 2.8E-6 | 1.259E-5 | - | - |
| 2000 | 30000 | 0.573 | 0.051 | 2.799E-6 | 9.33E-6 | - | - |
| 2000 | 40000 | 0.632 | 0.056 | 2.799E-6 | 1.353E-5 | - | - |
| 2500 | 12500 | 0.45 | 0.091 | 5.131E-6 | 1.306E-5 | 9.33E-7 | 4.67E-7 |
| 2500 | 25000 | 0.645 | 0.054 | 5.985E-6 | 1.259E-6 | - | - |
| 2500 | 37500 | 0.846 | 0.061 | 3.266E-6 | 9.797E-6 | - | - |
| 2500 | 50000 | 1.195 | 0.081 | 4.199E-6 | 1.166E-5 | - | - |

الجدول رقم (10)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز2500

نلاحظ أن *ShM\_ImTarjan* هي أسرع في زمن التنفيذ من *TC\_dfs* وذلك من اجل نفس عدد العقد والوصلات.

ولكن *TC\_dfs* أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من *ShM\_ImTarjan*.

تم تمثيل البيان باستخدام adjancency array

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***TC\_dfs*** | ***Sh\_ImTarjan*** | ***True***  ***TC\_dfs*** | ***True***  ***Sh\_ImTarjan*** | ***False***  ***TC\_dfs*** | ***False***  ***Sh\_ImTarjan*** |
| 250 | 1250 | 0.005 | 0.002 | 3.266E-6 | 4.199E-6 | 1.4E-6 | 4.99E-6 |
| 250 | 2500 | 0.006 | 0.0021 | 1.867E-6 | 4.665E-6 | - | - |
| 250 | 3750 | 0.007 | 0.0025 | 2.8E-6 | 5.132E-6 | - | - |
| 250 | 5000 | 0.009 | 0.003 | 2.333E-6 | 6.065E-6 | - | - |
| 500 | 2500 | 0.015 | 0.0027 | 2.332E-6 | 3.266E-6 | 9.33E-7 | 9.33E-7 |
| 500 | 5000 | 0.029 | 0.0058 | 2.798E-6 | 7.465E-6 | - | - |
| 500 | 7500 | 0.035 | 0.006 | 3.732E-6 | 4.665E-6 | - | - |
| 500 | 10000 | 0.047 | 0.0069 | 3.732E-6 | 7.464E-6 | - | - |
| 1000 | 5000 | 0.061 | 0.0075 | 2.799E-6 | 4.66E-6 | 9.33E-7 | 2.799E-6 |
| 1000 | 10000 | 0.111 | 0.008 | 4.665E-6 | 4.666E-6 | - | - |
| 1000 | 15000 | 0.134 | 0.011 | 3.732E-6 | 5.599E-6 | - | - |
| 1000 | 20000 | 0.159 | 0.014 | 4.199E-6 | 5.131E-6 | - | - |
| 2000 | 10000 | 0.262 | 0.019 | 2.8E-6 | 3.732E-6 | 9.33E-7 | 1.44E-6 |
| 2000 | 20000 | 0.447 | 0.02 | 3.732E-6 | 3.266E-6 | - | - |
| 2000 | 30000 | 0.573 | 0.021 | 2.8E-6. | 6.532E-6 | - | - |
| 2000 | 40000 | 0.632 | 0.0213 | 2.8E-6 | 6.065E-6 | - | - |
| 2500 | 12500 | 0.45 | 0.01 | 2.799E-6 | 3.266E-6 | 9.33E-7 | 2.332E-6 |
| 2500 | 25000 | 0.645 | 0.014 | 3.265E-6 | 4.665E-6 | - | - |
| 2500 | 37500 | 0.846 | 0.02 | 3.265E-6 | 3.733E-6 | - | - |
| 2500 | 50000 | 1.195 | 0.028 | 2.8E-6 | 3.266E-6 | - | - |

الجدول رقم (11)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز2500

نلاحظ أن *Sh\_ImTarjan* هي أسرع في زمن التنفيذ من *TC\_dfs* وذلك من اجل نفس عدد العقد والوصلات.

ولكن *TC\_dfs* أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من *Sh\_ImTarjan*.

تم تمثيل البيان باستخدام adjancency array

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***Sh\_ImTarjan*** | ***ShM\_ImTarjan*** | ***True***  ***Sh\_ImTarjan*** | ***True***  ***ShM\_ImTarjan*** | ***False***  ***Sh\_ImTarjan*** | ***False***  ***ShM\_ImTarjan*** |
| 250 | 1250 | 0.002 | 0.004 | 4.199E-6 | 2.799E-6 | 4.99E-6 | 9.33E-7 |
| 250 | 2500 | 0.0021 | 0.003 | 4.665E-6 | 2.799E-6 | - | - |
| 250 | 3750 | 0.0025 | 0.005 | 5.132E-6 | 2.799E-6 | - | - |
| 250 | 5000 | 0.003 | 0.007 | 6.065E-6 | 5.132E-6 | - | - |
| 500 | 2500 | 0.0027 | 0.008 | 3.266E-6 | 7.931E-6 | 9.33E-7 | 9.34E-7 |
| 500 | 5000 | 0.0058 | 0.0081 | 7.465E-6 | 4.199E-6 | - | - |
| 500 | 7500 | 0.006 | 0.0084 | 4.665E-6 | 3.732E-6 | - | - |
| 500 | 10000 | 0.0069 | 0.01 | 7.464E-6 | 3.732E-6 | - | - |
| 1000 | 5000 | 0.0075 | 0.021 | 4.66E-6 | 1.166E-6 | 2.799E-6 | 9.33E-7 |
| 1000 | 10000 | 0.008 | 0.025 | 4.666E-6 | 9.33E-6 | - | - |
| 1000 | 15000 | 0.011 | 0.018 | 5.599E-6 | 7.465E-6 | - | - |
| 1000 | 20000 | 0.014 | 0.028 | 5.131E-6 | 8.864E-6 | - | - |
| 2000 | 10000 | 0.019 | 0.055 | 3.732E-6 | 1.912E-6 | 1.44E-6 | 9.33E-7 |
| 2000 | 20000 | 0.02 | 0.051 | 3.266E-6 | 1.119E-6 | - | - |
| 2000 | 30000 | 0.021 | 0.051 | 6.532E-6 | 1.073E-5 | - | - |
| 2000 | 40000 | 0.0213 | 0.056 | 6.065E-6 | 1.119E-6 | - | - |
| 2500 | 12500 | 0.01 | 0.091 | 3.266E-6 | 1.772E-5 | 2.332E-6 | 1.866E-6 |
| 2500 | 25000 | 0.014 | 0.054 | 4.665E-6 | 1.258E-5 | - | - |
| 2500 | 37500 | 0.02 | 0.061 | 3.733E-6 | 7.465E-6 | - | - |
| 2500 | 50000 | 0.028 | 0.081 | 3.266E-6 | 1.632E-6 | - | - |

الجدول رقم (12)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز2500

نلاحظ ان *Sh\_ImTarjan* هي أسرع في زمن التنفيذ من *ShM\_ImTarjan* وذلك من اجل نفس عدد العقد والوصلات

ولكن *ShM\_ImTarjan* أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من *Sh\_ImTarjan*.

تم تمثيل البيان باستخدام *adjancency matrix*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***TC\_dfs*** | ***Warshall*** | ***True***  ***TC\_dfs*** | ***True***  ***Warshall*** | ***False***  ***TC\_dfs*** | ***False***  ***Warshall*** |
| 250 | 1250 | 0.065 | 0.067 | 5.132E-6 | 2.799E-6 | 9.34E-7 | 4.67E-7 |
| 250 | 2500 | 0.072 | 0.048 | 7.465E-6 | 4.199E-6 | - | - |
| 250 | 3750 | 0.088 | 0.061 | 9.331E-6 | 2.799E-6 | - | - |
| 250 | 5000 | 0.087 | 0.043 | 8.398E-6 | 3.266E-6 | - | - |
| 500 | 2500 | 0.383 | 0.018 | 9.797E-6 | 3.733E-6 | 9.34E-7 | 4.67E-7 |
| 500 | 5000 | 0.44 | 0.355 | 1.539E-5 | 4.665E-6 | - | - |
| 500 | 7500 | 0.441 | 0.344 | 4.665E-6 | 4.198E-6 | - | - |
| 500 | 10000 | 0.598 | 0.331 | 3.266E-6 | 2.799E-6 | - | - |
| 1000 | 5000 | 2.832 | 0.0279 | 2.052E-5 | 3.733E-6 | 9.33E-7 | 4.67E-7 |
| 1000 | 10000 | 2.98 | 2.809 | 2.052E-5 | 3.266E-6 | - | - |
| 1000 | 15000 | 3.07 | 2.667 | 1.213E-5 | 2.8E-6 | - | - |
| 1000 | 20000 | 3.167 | 2.674 | 7.932E-6 | 2.799E-6 | - | - |
| 2000 | 10000 | 22.022 | 23.681 | 2.332E-5 | 3.733E-6 | 9.34E-7 | 4.67E-7 |
| 2000 | 20000 | 23.3 | 22.429 | 1.632E-5 | 2.799E-6 | - | - |
| 2000 | 30000 | 23.003 | 21.701 | 1.632E-5 | 4.199E-6 | - | - |
| 2000 | 40000 | 23.222 | 21.358 | 1.772E-5 | 2.799E-6 | - | - |
| 2500 | 12500 | 41.159 | 47.889 | 2.006E-5 | 2.266E-6 | 1.399E-6 | 9.33E-7 |
| 2500 | 25000 | 45.024 | 43.881 | 1.819E-5 | 3.732E-6 | - | - |
| 2500 | 37500 | 44.911 | 42.257 | 1.866E-5 | 3.732E-6 | - | - |
| 2500 | 50000 | 45.432 | 40.908 | 1.866E-5 | 3.732E-6 | - | - |

الجدول رقم (13)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز2500

بشكل عام Warshall اسرع في التنفيذ من  *TC\_dfs*وكما أنها أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من*TC\_dfs*.

تم تمثيل البيان باستخدام *adjancency matrix*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***Sh\_Tarjan*** | ***Warshall*** | ***True***  ***Sh\_Tarjan*** | ***True***  ***Warshall*** | ***False***  ***Sh\_Tarjan*** | ***False***  ***Warshall*** |
| 250 | 1250 | 0.003 | 0.067 | 1.306E-5 | 2.799E-6 | 4.67E-7 | 4.67E-7 |
| 250 | 2500 | 0.007 | 0.048 | 6.624E-5 | 4.199E-6 | - | - |
| 250 | 3750 | 0.002 | 0.061 | 5.598E-5 | 2.799E-6 | - | - |
| 250 | 5000 | 0.005 | 0.043 | 1.663E-5 | 3.266E-6 | - | - |
| 500 | 2500 | 0.0128 | 0.018 | 1.702E-5 | 3.733E-6 | 4.67E-7 | 4.67E-7 |
| 500 | 5000 | 0.011 | 0.355 | 1.026E-5 | 4.665E-6 | - | - |
| 500 | 7500 | 0.008 | 0.344 | 7.465E-6 | 4.198E-6 | - | - |
| 500 | 10000 | 0.013 | 0.331 | 1.632E- 5 | 2.799E-6 | - | - |
| 1000 | 5000 | 0.076 | 0.0279 | 1.072E-5 | 3.733E-6 | 4.67E-7 | 4.67E-7 |
| 1000 | 10000 | 0.026 | 2.809 | 1.306E-5 | 3.266E-6 | - | - |
| 1000 | 15000 | 0.026 | 2.667 | 9.797E-6 | 2.8E-6 | - | - |
| 1000 | 20000 | 0.035 | 2.674 | 1.026E-5 | 2.799E-6 | - | - |
| 2000 | 10000 | 0.301 | 23.681 | 5.598E-6 | 3.733E-6 | 4.67E-7 | 4.67E-7 |
| 2000 | 20000 | 0.102 | 22.429 | 1.663E-5 | 2.799E-6 | - | - |
| 2000 | 30000 | 0.091 | 21.701 | 4.666E-6 | 4.199E-6 | - | - |
| 2000 | 40000 | 0.121 | 21.358 | 1.121E-5 | 2.799E-6 | - | - |
| 2500 | 12500 | 1.311 | 47.889 | 1.586E-5 | 2.266E-6 | 9.33E-7 | 9.33E-7 |
| 2500 | 25000 | 0.165 | 43.881 | 1.026E-5 | 3.732E-6 | - | - |
| 2500 | 37500 | 0.165 | 42.257 | 1.119E-5 | 3.732E-6 | - | - |
| 2500 | 50000 | 0.124 | 40.908 | 1.073E-5 | 3.732E-6 | - | - |

الجدول رقم (14)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز2500

بشكل عام *Sh\_Tarjan* أسرع في زمن التنفيذ من warshall وذلك من اجل نفس عدد العقد والوصلات

وكما أن *Sh\_Tarjan* أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من warshall.

تم تمثيل البيان باستخدام *adjancency matrix*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***ShM\_Tarjan*** | ***Warshall*** | ***True***  ***ShM\_Tarjan*** | ***True***  ***Warshall*** | ***False***  ***ShM\_Tarjan*** | ***False***  ***Warshall*** |
| 250 | 1250 | 0.009 | 0.067 | 3.732E-6 | 2.799E-6 | 9.34E-7 | 4.67E-7 |
| 250 | 2500 | 0.031 | 0.048 | 8.864E-6 | 4.199E-6 | - | - |
| 250 | 3750 | 0.01 | 0.061 | 2.799E-6 | 2.799E-6 | - | - |
| 250 | 5000 | 0.014 | 0.043 | 3.266E-6 | 3.266E-6 | - | - |
| 500 | 2500 | 0.0168 | 0.018 | 5.131E-6 | 3.733E-6 | 9.34E-7 | 4.67E-7 |
| 500 | 5000 | 0.018 | 0.355 | 3.732E-6 | 4.665E-6 | - | - |
| 500 | 7500 | 0.016 | 0.344 | 4.199E-6 | 4.198E-6 | - | - |
| 500 | 10000 | 0.015 | 0.331 | 2.8E-6 | 2.799E-6 | - | - |
| 1000 | 5000 | 0.08 | 0.0279 | 4.199E-6 | 3.733E-6 | 9.33E-7 | 4.67E-7 |
| 1000 | 10000 | 0.037 | 2.809 | 4.199E-6 | 3.266E-6 | - | - |
| 1000 | 15000 | 0.038 | 2.667 | 4.199E-6 | 2.8E-6 | - | - |
| 1000 | 20000 | 0.043 | 2.674 | 5.131E-6 | 2.799E-6 | - | - |
| 2000 | 10000 | 0.364 | 23.681 | 3.732E-6 | 3.733E-6 | 9.34E-7 | 4.67E-7 |
| 2000 | 20000 | 0.12 | 22.429 | 3.266E-6 | 2.799E-6 | - | - |
| 2000 | 30000 | 0.153 | 21.701 | 6.065E-6 | 4.199E-6 | - | - |
| 2000 | 40000 | 0.139 | 21.358 | 4.666E-6 | 2.799E-6 | - | - |
| 2500 | 12500 | 0. 453 | 47.889 | 3.266E-6 | 2.266E-6 | 1.399E-6 | 9.33E-7 |
| 2500 | 25000 | 0.277 | 43.881 | 4.199E-6 | 3.732E-6 | - | - |
| 2500 | 37500 | 0.189 | 42.257 | 4.199E-6 | 3.732E-6 | - | - |
| 2500 | 50000 | 0.206 | 40.908 | 4.199E-6 | 3.732E-6 | - | - |

الجدول رقم (15)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز2500

بشكل عام *ShM\_Tarjan* أسرع في زمن التنفيذ من warshall وذلك من اجل نفس عدد العقد والوصلات

ولكن warshall أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من *ShM\_Tarjan* .

تم تمثيل البيان باستخدام *adjancency matrix*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***Sh\_ImTarjan*** | ***Warshall*** | ***True***  ***Sh\_ImTarjan*** | ***True***  ***Warshall*** | ***False***  ***Sh\_ImTarjan*** | ***False***  ***Warshall*** |
| 250 | 1250 | 0.005 | 0.067 | 5.132E-6 | 2.799E-6 | 1.866E-6 | 4.67E-7 |
| 250 | 2500 | 0.0054 | 0.048 | 3.266E-6 | 4.199E-6 | - | - |
| 250 | 3750 | 0.0059 | 0.061 | 4.665E-6 | 2.799E-6 | - | - |
| 250 | 5000 | 0.0085 | 0.043 | 6.998E-6 | 3.266E-6 | - | - |
| 500 | 2500 | 0.099 | 0.018 | 3.266E-6 | 3.733E-6 | 9.33E-7 | 4.67E-7 |
| 500 | 5000 | 0.01 | 0.355 | 3.732E-6 | 4.665E-6 | - | - |
| 500 | 7500 | 0.013 | 0.344 | 3.265E-6 | 4.198E-6 | - | - |
| 500 | 10000 | 0.0135 | 0.331 | 3.733E-6 | 2.799E-6 | - | - |
| 1000 | 5000 | 0.05 | 0.0279 | 4.198E-6 | 3.733E-6 | 1.866E-6 | 4.67E-7 |
| 1000 | 10000 | 0.027 | 2.809 | 4.199E-6 | 3.266E-6 | - | - |
| 1000 | 15000 | 0.0272 | 2.667 | 6.065E-6 | 2.8E-6 | - | - |
| 1000 | 20000 | 0.028 | 2.674 | 4.199E-6 | 2.799E-6 | - | - |
| 2000 | 10000 | 0.681 | 23.681 | 7.931E-6 | 3.733E-6 | 3.266E-6 | 4.67E-7 |
| 2000 | 20000 | 0.08 | 22.429 | 5.599E-6 | 2.799E-6 | - | - |
| 2000 | 30000 | 0.086 | 21.701 | 4.665E-6 | 4.199E-6 | - | - |
| 2000 | 40000 | 0.0999 | 21.358 | 5.598E-6 | 2.799E-6 | - | - |
| 2500 | 12500 | 0.712 | 47.889 | 7.465E-6 | 2.266E-6 | 2.799E-6 | 9.33E-7 |
| 2500 | 25000 | 0.127 | 43.881 | 5.132E-6 | 3.732E-6 | - | - |
| 2500 | 37500 | 0.151 | 42.257 | 7.464E-6 | 3.732E-6 | - | - |
| 2500 | 50000 | 0.153 | 40.908 | 6.998E-6 | 3.732E-6 | - | - |

الجدول رقم (16)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز2500

بشكل عام *Sh\_ImTarjan* أسرع في زمن التنفيذ من warshall وذلك من اجل نفس عدد العقد والوصلات

ولكن warshall أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من *Sh\_ImTarjan*.

تم تمثيل البيان باستخدام *adjancency matrix*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***ShM\_ImTarjan*** | ***Warshall*** | ***True***  ***ShM\_ImTarjan*** | ***True***  ***Warshall*** | ***False***  ***ShM\_ImTarjan*** | ***False***  ***Warshall*** |
| 250 | 1250 | 0.009 | 0.067 | 2.799E-6 | 2.799E-6 | 9.33E-7 | 4.67E-7 |
| 250 | 2500 | 0.006 | 0.048 | 2.799E-6 | 4.199E-6 | - | - |
| 250 | 3750 | 0.007 | 0.061 | 2.799E-6 | 2.799E-6 | - | - |
| 250 | 5000 | 0.009 | 0.043 | 5.132E-6 | 3.266E-6 | - | - |
| 500 | 2500 | 0.011 | 0.018 | 7.931E-6 | 3.733E-6 | 9.34E-7 | 4.67E-7 |
| 500 | 5000 | 0.012 | 0.355 | 4.199E-6 | 4.665E-6 | - | - |
| 500 | 7500 | 0.017 | 0.344 | 3.732E-6 | 4.198E-6 | - | - |
| 500 | 10000 | 0.018 | 0.331 | 3.732E-6 | 2.799E-6 | - | - |
| 1000 | 5000 | 0.067 | 0.0279 | 1.166E-6 | 3.733E-6 | 9.33E-7 | 4.67E-7 |
| 1000 | 10000 | 0.033 | 2.809 | 9.33E-6 | 3.266E-6 | - | - |
| 1000 | 15000 | 0.0331 | 2.667 | 7.465E-6 | 2.8E-6 | - | - |
| 1000 | 20000 | 0.034 | 2.674 | 8.864E-6 | 2.799E-6 | - | - |
| 2000 | 10000 | 0.315 | 23.681 | 1.912E-6 | 3.733E-6 | 9.33E-7 | 4.67E-7 |
| 2000 | 20000 | 0.123 | 22.429 | 1.119E-6 | 2.799E-6 | - | - |
| 2000 | 30000 | 0.127 | 21.701 | 1.073E-5 | 4.199E-6 | - | - |
| 2000 | 40000 | 0.129 | 21.358 | 1.119E-6 | 2.799E-6 | - | - |
| 2500 | 12500 | 0.92 | 47.889 | 1.772E-5 | 2.266E-6 | 1.866E-6 | 9.33E-7 |
| 2500 | 25000 | 0.158 | 43.881 | 1.258E-5 | 3.732E-6 | - | - |
| 2500 | 37500 | 0.163 | 42.257 | 7.465E-6 | 3.732E-6 | - | - |
| 2500 | 50000 | 0.214 | 40.908 | 1.632E-6 | 3.732E-6 | - | - |

الجدول رقم (17)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز2500

بشكل عام *ShM\_ImTarjan* أسرع في زمن التنفيذ من warshall وذلك من اجل نفس عدد العقد والوصلات

وكما أن *ShM\_ImTarjan* أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من warshall.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***ShM\_Tarjan*** | ***ShM\_ImTarjan*** | ***True***  ***ShM\_Tarjan*** | ***True***  ***ShM\_ImTarjan*** | ***False***  ***ShM\_Tarjan*** | ***False***  ***ShM\_ImTarjan*** |
| 250 | 1250 | 0.009 | 0.009 | 3.732E-6 | 2.799E-6 | 9.34E-7 | 9.33E-7 |
| 250 | 2500 | 0.031 | 0.006 | 8.864E-6 | 2.799E-6 | - | - |
| 250 | 3750 | 0.01 | 0.007 | 2.799E-6 | 2.799E-6 | - | - |
| 250 | 5000 | 0.014 | 0.009 | 3.266E-6 | 5.132E-6 | - | - |
| 500 | 2500 | 0.0168 | 0.011 | 5.131E-6 | 7.931E-6 | 9.34E-7 | 9.34E-7 |
| 500 | 5000 | 0.018 | 0.012 | 3.732E-6 | 4.199E-6 | - | - |
| 500 | 7500 | 0.016 | 0.017 | 4.199E-6 | 3.732E-6 | - | - |
| 500 | 10000 | 0.015 | 0.018 | 2.8E-6 | 3.732E-6 | - | - |
| 1000 | 5000 | 0.08 | 0.067 | 4.199E-6 | 1.166E-6 | 9.33E-7 | 9.33E-7 |
| 1000 | 10000 | 0.037 | 0.033 | 4.199E-6 | 9.33E-6 | - | - |
| 1000 | 15000 | 0.038 | 0.0331 | 4.199E-6 | 7.465E-6 | - | - |
| 1000 | 20000 | 0.043 | 0.034 | 5.131E-6 | 8.864E-6 | - | - |
| 2000 | 10000 | 0.364 | 0.315 | 3.732E-6 | 1.912E-6 | 9.34E-7 | 9.33E-7 |
| 2000 | 20000 | 0.12 | 0.123 | 3.266E-6 | 1.119E-6 | - | - |
| 2000 | 30000 | 0.153 | 0.127 | 6.065E-6 | 1.073E-5 | - | - |
| 2000 | 40000 | 0.139 | 0.129 | 4.666E-6 | 1.119E-6 | - | - |
| 2500 | 12500 | 0. 453 | 0.92 | 3.266E-6 | 1.772E-5 | 1.399E-6 | 1.866E-6 |
| 2500 | 25000 | 0.277 | 0.158 | 4.199E-6 | 1.258E-5 | - | - |
| 2500 | 37500 | 0.189 | 0.163 | 4.199E-6 | 7.465E-6 | - | - |
| 2500 | 50000 | 0.206 | 0.214 | 4.199E-6 | 1.632E-6 | - | - |

تم تمثيل البيان باستخدام *adjancency matrix*

الجدول رقم (18)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز2500

نلاحظ ان *ShM\_ImTarjan* هي أسرع في زمن التنفيذ من *ShM\_Tarjan* وذلك من اجل نفس عدد العقد والوصلات

وكما ان *ShM\_ImTarjan* أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من *ShM\_Tarjan*.

تم تمثيل البيان باستخدام *adjancency matrix*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***Sh\_Tarjan*** | ***Sh\_ImTarjan*** | ***True***  ***Sh\_Tarjan*** | ***True***  ***Sh\_ImTarjan*** | ***False***  ***Sh\_Tarjan*** | ***False***  ***Sh\_ImTarjan*** |
| 250 | 1250 | 0.003 | 0.005 | 1.306E-5 | 2.799E-6 | 4.67E-7 | 4.67E-7 |
| 250 | 2500 | 0.007 | 0.0054 | 6.624E-5 | 4.199E-6 | - | - |
| 250 | 3750 | 0.002 | 0.0059 | 5.598E-5 | 2.799E-6 | - | - |
| 250 | 5000 | 0.005 | 0.0085 | 1.663E-5 | 3.266E-6 | - | - |
| 500 | 2500 | 0.0128 | 0.099 | 1.702E-5 | 3.733E-6 | 4.67E-7 | 4.67E-7 |
| 500 | 5000 | 0.011 | 0.01 | 1.026E-5 | 4.665E-6 | - | - |
| 500 | 7500 | 0.008 | 0.013 | 7.465E-6 | 4.198E-6 | - | - |
| 500 | 10000 | 0.013 | 0.0135 | 1.632E- 5 | 2.799E-6 | - | - |
| 1000 | 5000 | 0.076 | 0.05 | 1.072E-5 | 3.733E-6 | 4.67E-7 | 4.67E-7 |
| 1000 | 10000 | 0.026 | 0.027 | 1.306E-5 | 3.266E-6 | - | - |
| 1000 | 15000 | 0.026 | 0.0272 | 9.797E-6 | 2.8E-6 | - | - |
| 1000 | 20000 | 0.035 | 0.028 | 1.026E-5 | 2.799E-6 | - | - |
| 2000 | 10000 | 0.301 | 0.681 | 5.598E-6 | 3.733E-6 | 4.67E-7 | 4.67E-7 |
| 2000 | 20000 | 0.102 | 0.08 | 1.663E-5 | 2.799E-6 | - | - |
| 2000 | 30000 | 0.091 | 0.086 | 4.666E-6 | 4.199E-6 | - | - |
| 2000 | 40000 | 0.121 | 0.0999 | 1.121E-5 | 2.799E-6 | - | - |
| 2500 | 12500 | 1.311 | 0.712 | 1.586E-5 | 2.266E-6 | 9.33E-7 | 9.33E-7 |
| 2500 | 25000 | 0.165 | 0.127 | 1.026E-5 | 3.732E-6 | - | - |
| 2500 | 37500 | 0.165 | 0.151 | 1.119E-5 | 3.732E-6 | - | - |
| 2500 | 50000 | 0.124 | 0.153 | 1.073E-5 | 3.732E-6 | - | - |

الجدول رقم (19)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز2500

نلاحظ ان *Sh\_Tarjan* هي أسرع في زمن التنفيذ من *Sh\_ImTarjan* وذلك من اجل نفس عدد العقد والوصلات.

وكما أن *Sh\_Tarjan* أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من *Sh\_ImTarjan* .

تم تمثيل البيان باستخدام *adjancency array*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***ShM\_Tarjan*** | ***ShM\_ImTarjan*** | ***True***  ***ShM\_Tarjan*** | ***True***  ***ShM\_ImTarjan*** | ***False***  ***ShM\_Tarjan*** | ***False***  ***ShM\_ImTarjan*** |
| 250 | 1250 | 0.009 | 0.004 | 3.732E-6 | 2.799E-6 | 9.34E-7 | 9.33E-7 |
| 250 | 2500 | 0.031 | 0.003 | 8.864E-6 | 2.799E-6 | - | - |
| 250 | 3750 | 0.01 | 0.005 | 2.799E-6 | 2.799E-6 | - | - |
| 250 | 5000 | 0.014 | 0.007 | 3.266E-6 | 5.132E-6 | - | - |
| 500 | 2500 | 0.0168 | 0.008 | 5.131E-6 | 7.931E-6 | 9.34E-7 | 9.34E-7 |
| 500 | 5000 | 0.018 | 0.0081 | 3.732E-6 | 4.199E-6 | - | - |
| 500 | 7500 | 0.016 | 0.0084 | 4.199E-6 | 3.732E-6 | - | - |
| 500 | 10000 | 0.015 | 0.01 | 2.8E-6 | 3.732E-6 | - | - |
| 1000 | 5000 | 0.08 | 0.021 | 4.199E-6 | 1.166E-6 | 9.33E-7 | 9.33E-7 |
| 1000 | 10000 | 0.037 | 0.025 | 4.199E-6 | 9.33E-6 | - | - |
| 1000 | 15000 | 0.038 | 0.018 | 4.199E-6 | 7.465E-6 | - | - |
| 1000 | 20000 | 0.043 | 0.028 | 5.131E-6 | 8.864E-6 | - | - |
| 2000 | 10000 | 0.364 | 0.055 | 3.732E-6 | 1.912E-6 | 9.34E-7 | 9.33E-7 |
| 2000 | 20000 | 0.12 | 0.051 | 3.266E-6 | 1.119E-6 | - | - |
| 2000 | 30000 | 0.153 | 0.051 | 6.065E-6 | 1.073E-5 | - | - |
| 2000 | 40000 | 0.139 | 0.056 | 4.666E-6 | 1.119E-6 | - | - |
| 2500 | 12500 | 0. 453 | 0.091 | 3.266E-6 | 1.772E-5 | 1.399E-6 | 1.866E-6 |
| 2500 | 25000 | 0.277 | 0.054 | 4.199E-6 | 1.258E-5 | - | - |
| 2500 | 37500 | 0.189 | 0.061 | 4.199E-6 | 7.465E-6 | - | - |
| 2500 | 50000 | 0.206 | 0.081 | 4.199E-6 | 1.632E-6 | - | - |

الجدول رقم(2500)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز

نلاحظ ان *ShM\_ImTarjan* هي أسرع في زمن التنفيذ من *ShM\_Tarjan* وذلك من اجل نفس عدد العقد والوصلات

وكما ان *ShM\_ImTarjan* أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من *ShM\_Tarjan*.

تم تمثيل البيان باستخدام *adjancency array*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***Sh\_Tarjan*** | ***Sh\_ImTarjan*** | ***True***  ***Sh\_Tarjan*** | ***True***  ***Sh\_ImTarjan*** | ***False***  ***Sh\_Tarjan*** | ***False***  ***Sh\_ImTarjan*** |
| 250 | 1250 | 0.003 | 0.002 | 2.333E-6 | 4.199E-6 | 9.33E-7 | 4.99E-6 |
| 250 | 2500 | 0.0031 | 0.0021 | 1.866E-6 | 4.665E-6 | - | - |
| 250 | 3750 | 0.0035 | 0.0025 | 2.333E-6 | 5.132E-6 | - | - |
| 250 | 5000 | 0.0036 | 0.003 | 2.333E-6 | 6.065E-6 | - | - |
| 500 | 2500 | 0.005 | 0.0027 | 2.332E-6 | 3.266E-6 | 4.67E-7 | 9.33E-7 |
| 500 | 5000 | 0.007 | 0.0058 | 2.8E-6. | 7.465E-6 | - | - |
| 500 | 7500 | 0.008 | 0.006 | 3.732E-6 | 4.665E-6 | - | - |
| 500 | 10000 | 0.009 | 0.0069 | 3.732E-6 | 7.464E-6 | - | - |
| 1000 | 5000 | 0.015 | 0.0075 | 2.799E-6 | 4.66E-6 | 4.67E-7 | 2.799E-6 |
| 1000 | 10000 | 0.019 | 0.008 | 4.199E-6 | 4.666E-6 | - | - |
| 1000 | 15000 | 0.019 | 0.011 | 4.199E-6 | 5.599E-6 | - | - |
| 1000 | 20000 | 0.021 | 0.014 | 4.199E-6 | 5.131E-6 | - | - |
| 2000 | 10000 | 0.054 | 0.019 | 3.732E-6 | 3.732E-6 | 9.33E-7 | 1.44E-6 |
| 2000 | 20000 | 0.058 | 0.02 | 3.732E-6 | 3.266E-6 | - | - |
| 2000 | 30000 | 0.059 | 0.021 | 2.799E-6 | 6.532E-6 | - | - |
| 2000 | 40000 | 0.416 | 0.0213 | 2.799E-6 | 6.065E-6 | - | - |
| 2500 | 12500 | 0.078 | 0.01 | 2.799E-6 | 3.266E-6 | 9.33E-7 | 2.332E-6 |
| 2500 | 25000 | 0.043 | 0.014 | 2.8E-6. | 4.665E-6 | - | - |
| 2500 | 37500 | 0.069 | 0.02 | 2.333E-6 | 3.733E-6 | - | - |
| 2500 | 50000 | 0.093 | 0.028 | 2.799E-6 | 3.266E-6 | - | - |

الجدول رقم (21)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز 2500

نلاحظ ان *Sh\_ImTarjan* هي أسرع في زمن التنفيذ من *Sh\_Tarjan* وذلك من اجل نفس عدد العقد والوصلات.

وكما أن *Sh\_ImTarjan* أسرع في التحقق من ارتباط عقدتين مع بعضهما أكثر من*Sh\_Tarjan* .

*تم تمثيل البيان باستخدام adjancency matrix*

|  |  |  |  |
| --- | --- | --- | --- |
| ***N*** | ***M*** | ***TScc*** | ***ImScc*** |
| 250 | 1250 | 0.0086 | 0.0085 |
| 250 | 2500 | 0.0038 | 0.0035 |
| 250 | 3750 | 0.0032 | 0.0033 |
| 250 | 5000 | 0.0037 | 0.0027 |
| 500 | 2500 | 0.012 | 0.01 |
| 500 | 5000 | 0.0068 | 0.0064 |
| 500 | 7500 | 0.006 | 0.004 |
| 500 | 10000 | 0.005 | 0.0044 |
| 1000 | 5000 | 0.0067 | 0.0066 |
| 1000 | 10000 | 0.006 | 0.008 |
| 1000 | 15000 | 0.0044 | 0.003 |
| 1000 | 20000 | 0.0094 | 0.0081 |
| 2000 | 10000 | 0.011 | 0.01 |
| 2000 | 20000 | 0.017 | 0.012 |
| 2000 | 30000 | 0.015 | 0.014 |
| 2000 | 40000 | 0.0118 | 0.0111 |
| 2500 | 12500 | 0.0152 | 0.0126 |
| 2500 | 25000 | 0.021 | 0.0164 |
| 2500 | 37500 | 0.019 | 0.017 |
| 2500 | 50000 | 0.016 | 0.015 |

الجدول رقم (22)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز 2500

بشكل عام *ImScc* افضل من *Scc* من حيث زمن التنفيذ مع وجود بعض الحالات الاستثنائية.

تم تمثيل البيان باستخدام *adjancency array*

|  |  |  |  |
| --- | --- | --- | --- |
| ***N*** | ***M*** | ***TScc*** | ***ImScc*** |
| 250 | 1250 | 4.936E-4 | 8.047E-4 |
| 250 | 2500 | 0.001 | 8.999E-4 |
| 250 | 3750 | 8.654E-4 | 9.979E-4 |
| 250 | 5000 | 0.0125 | 0.00126 |
| 500 | 2500 | 0.0011 | 0.0015 |
| 500 | 5000 | 0.002 | 0.001 |
| 500 | 7500 | 0.0021 | 0.002 |
| 500 | 10000 | 0.0021 | 0.001 |
| 1000 | 5000 | 0.0018 | 0.0027 |
| 1000 | 10000 | 0.003 | 0.002 |
| 1000 | 15000 | 0.0044 | 0.003 |
| 1000 | 20000 | 0.0046 | 0.0035 |
| 2000 | 10000 | 0.0098 | 0.0091 |
| 2000 | 20000 | 0.005 | 0.004 |
| 2000 | 30000 | 0.006 | 0.0051 |
| 2000 | 40000 | 0.0067 | 0.0056 |
| 2500 | 12500 | 0.005 | 0.003 |
| 2500 | 25000 | 0.006 | 0.005 |
| 2500 | 37500 | 0.008 | 0.006 |
| 2500 | 50000 | 0.005 | 0.006 |

الجدول رقم (23)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز 2500

بشكل عام *ImScc* افضل من *Scc* من حيث زمن التنفيذ مع وجود بعض الحالات الاستثنائية.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***N*** | ***M*** | ***MRep\_dfs*** | ***ARep\_dfs*** | ***True***  ***MRep\_dfs*** | ***True***  ***ARep\_dfs*** | ***False***  ***MRep\_dfs*** | ***False***  ***ARep\_dfs*** |
| 250 | 1250 | 0.05 | 0.005 | 2.799E-6 | 3.266E-6 | 4.66E-7 | 1.4E-6 |
| 250 | 2500 | 0.06 | 0.006 | 2.332E-6 | 1.867E-6 | - | - |
| 250 | 3750 | 0.064 | 0.007 | 2.8E-6 | 2.8E-6 | - | - |
| 250 | 5000 | 0.069 | 0.009 | 3.732E-6 | 2.333E-6 | - | - |
| 50 | 2500 | 0.368 | 0.015 | 2.799E-6 | 2.332E-6 | 4.67E-7 | 9.33E-7 |
| 500 | 5000 | 0.415 | 0.029 | 2.799E-6 | 2.798E-6 | - | - |
| 500 | 7500 | 0.442 | 0.035 | 3.732E-6 | 3.732E-6 | - | - |
| 500 | 10000 | 0.448 | 0.047 | 2.799E-6 | 3.732E-6 | - | - |
| 1000 | 5000 | 2.83 | 0.061 | 3.266E-6 | 2.799E-6 | 4.66E-7 | 9.33E-7 |
| 1000 | 10000 | 2.972 | 0.111 | 2.8E-6 | 4.665E-6 | - | - |
| 1000 | 15000 | 3.051 | 0.134 | 2.333E-6 | 3.732E-6 | - | - |
| 1000 | 20000 | 3.139 | 0.159 | 2.8E-6 | 4.199E-6 | - | - |
| 2000 | 10000 | 21.898 | 0.262 | 2.799E-6 | 2.8E-6 | 4.673E-7 | 9.33E-7 |
| 2000 | 20000 | 22.241 | 0.447 | 2.332E-6 | 3.732E-6 | - | - |
| 2000 | 30000 | 23.074 | 0.573 | 2.799E-6 | 2.8E-6. | - | - |
| 2000 | 40000 | 23.483 | 0.632 | 2.332E-6 | 2.8E-6 | - | - |
| 2500 | 12500 | 44.016 | 0.45 | 2.799E-6 | 2.799E-6 | 9.33E-7 | 9.33E-7 |
| 2500 | 25000 | 44.949 | 0.645 | 3.732E-6 | 3.265E-6 | - | - |
| 2500 | 37500 | 54.375 | 0.846 | 2.799E-6 | 3.265E-6 | - | - |
| 2500 | 50000 | 54.340 | 1.195 | 2.8E-6 | 2.8E-6 | - | - |

الجدول رقم (24)يوضح زمن تنفيذ الخوارزميات من أجل عدد عقد لا يتجاوز2500

نلاحظ أن *ARep\_dfs* أسرع في زمن التنفيذ من  *MRep\_dfs*ولكن *MRep\_dfs اسرع في التحقق من وجود**ارتباط بين عقدتين****.***

*References:*

*[1] Esko Nuutila: Efficient Transitive Closure Computation in Large Digraphs.*

*[2] Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser:*

*Data Structures and Algorithms in Java™.*

*[3] Mark Allen Weiss: data Structures and Algorithm Analysis in Java ( 2nd Edition).*

*[4]* [*Cormen, Thomas H.*](https://en.wikipedia.org/wiki/Thomas_H._Cormen), [*Leiserson, Charles E.*](https://en.wikipedia.org/wiki/Charles_E._Leiserson), [*Rivest, Ronald L.*](https://en.wikipedia.org/wiki/Ron_Rivest) (1990): [*Introduction to Algorithms*](https://en.wikipedia.org/wiki/Introduction_to_Algorithms) (1st Edition)

[5] Robert Tarjan (1979): Depth first Search and linear graph algorithms.

[6] Pearce, David: A Space Efficient Algorithm for detecting Scc .