ALGORITHMS TO EVALUATE CONNECTED DETOUR NUMBER AND FORCING CONNECTED DETOUR NUMBER IN GRAPHS

S.Albert Antony Raj

Department of Computer Applications, SRM Institute of Science and Technology
Kattankulathur – 603 203, India, rsalberrtprakash@gmail.com

Abstract: In a graph G = (V, E) of order n, any longest path joining any two vertices u and v is called a u-v detour. A set S of vertices of G is a detour set if each vertex w of G lies on a u - v detour in G. The minimum cardinality of a detour set S of G is defined as the detour number of G, denoted by **dn(G)**. A set $S \subseteq V$ is called a connected detour set of G if S is detour set of G and the subgraph G[S] induced by S is connected. The connected detour number ***cdn(G)*** of G is the minimum order of its connected detour sets and any connected detour set of order ***cdn(G)*** is called a connected detour basis of G. The forcing connected detour number ***fcdn(S)*** of S is the minimum cardinality of a forcing subset for S. The forcing connected detour number ***fcdn(G)*** of G is ***min{fcdn(S)}***, where the minimum is taken over all connected detour bases S in G. This paper presents the sequential algorithm for evaluating connected detour number and forcing connected detour number of a graph and analysis of its efficiency.

Keywords: Distance, Detour, Connected Detour Set, Connected Detour Basis, Connected Detour Number, Forcing Connected Detour Set, Forcing Connected Detour Number.

Introduction: Theoretic terminology and definition for basic graph is referred from [1, 2]. A graph G is a finite non-empty set of objects called vertices together with a set of unordered pairs of distinct vertices of G, called edges. The vertex set and the edge set of G are denoted by V(G) or simply V and E(G) or E respectively. A graph G is said to be connected if any two distinct vertices of G are joined by a path. Detour distance D(x, y) is the length of a longest x - y path in G. For any vertex u of G, the detour eccentricity of u is $E_D(u) = \max\{D(u, v) : v \in V\}$. The detour radius R is defined by $R_D = \min\{e(v) : v \in V(G)\}$ and detour diameter D of G is defined by $D_D = \max\{e(v)\ v \in V(G)\}$. An x - y path of length D(x, y) is called an x - y detour. For a set S of vertices of G, let the closed interval $I_D[S]$ of S be the union of the closed intervals $I_D[u, v]$ over all pairs of vertices u and v in S lying on some u - v detour of G. If $I_D[S] = V(G)$ then S is called a detour set and the minimum cardinality of a detour set is the detour number ***dn(G)*** and this detour set is called a minimum detour set. The detour number of a graph was introduced in [3] and further studied in [4].

Connected Detour Set and Connected Detour Number: The concepts of connected detour number was introduced in [5]. A set $S \subseteq V$ is called a connected detour set of G if S is a detour set of G and the subgraph G[S] induced by S is connected. A subgraph H of G is induced, if for any two vertices x and y in H x and y are adjacent in H if and only if they are adjacent in G. In other words, H has the same edges as G between the vertices in H. A general subgraph can have less edges between the same vertices than the original one. An induced subgraph can be constructed by deleting all the incident edges. The connected detour number ***cdn(G)*** of G is the minimum order of its connected detour sets and any connected detour set of order ***cdn(G)*** is called a connected detour basis of G. A vertex v in a graph G is a connected detour vertex if v belongs to every connected detour basis of G. If G has a unique connected detour basis S, then every vertex in S is a connected detour vertex of G.

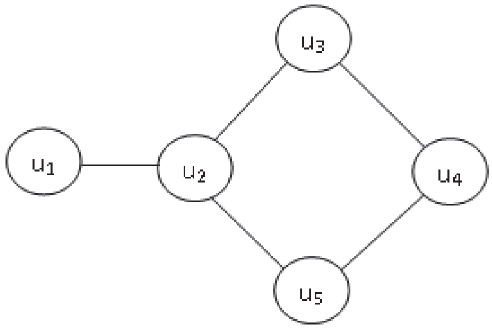

Figure (a)

For the graph G given in Figure (a), the sets S_1= {u_1, u_3}, S_2= {u_1, u_5} and S_3= {u_1, u_4} are the three detour bases of G so that dn(G) = 2. It is clear that no two element subset of vertices V is a connected detour set of G. However the set S_4 = {u_1, u_2, u_3} is a connected detour basis of G so that cdn(G) = 3. Also the set S_5 = {u_1, u_2, u_5} is another connected detour basis of G. Thus there can be more than one connected detour basis for a graph G.

The calculation of the detour number ***dn(G)*** and connected detour number ***cdn(G)*** of a graph G is described. Consider a graph G in Figure (a). Let us take S_1= {u_1, u_3}, S_2= {u_1, u_5} and S_3= {u_1, u_4} and the distance between u_1 and u_3 can be measured as $D(u_1,u_3)$ = 4. The closed interval of S_1, S_2 and S_3 can be evaluated as, $\mathbf{I_D[u_1,u_3] = I_D[u_1,u_4] = I_D[u_1,u_5] = \{u_1, u_2,u_5, u_4, u_3\}}$. $I_D[S]$ =V(G) and hence S_1, S_2 and S_3 is a detour set or detour basis and detour number dn(G) = 2. None of the sets S_1, S_2 and S_3 is a connected detour set. Consider S_4 = {u_1, u_2, u_3}, $\mathbf{I_D[u_1,u_2,u_3] = \{u_1, u_2,u_5, u_4, u_3\}}$ which a detour basis and ***dn(G) = 3***. The subgraph induced by S_4 is connected; hence S_4 is a connected detour set and ***cdn(G)***=3. Similarly S_4 = {u_1, u_2, u_5} is also a connected detour set with ***cdn(G)***=3.

Forcing Connected Detour Set and Forcing Connected Detour Number: Let G be a connected graph and S a connected detour basis of G. A subset T ⊆ S is called a forcing subset for S if S is the unique connected detour basis containing T. A forcing subset for S of minimum cardinality is a minimum forcing subset of S. The forcing connected detour number of S, denoted by ***fcdn(S)***, is the cardinality of a minimum forcing subset for S. The forcing connected detour number of G, denoted by ***fcdn(G)***, is ***fcdn(G) = min{fcdn(S)}***, where the minimum is taken over all connected detour bases S in G. The Forcing Connected Detour Set of a Graph has been studied in [5].

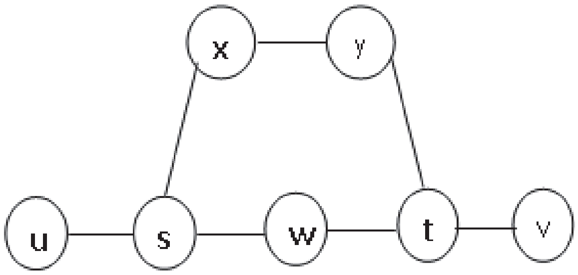

Figure (b)

For the graph G given in Figure (b) , S_1 = {u, s,w, t, v} is the unique connected detour basis of G so that fcdn(G) = 0 and for the graph G given in Figure (a) , S_2 = {u_1, u_2, u_3} and S_3 = {u_1, u_2, u_5} are the only connected detour bases of G so that cdn(G) = 3 and fcdn(G) = 3.

Algorithm For Computing Connected Detour Number: An algorithm using dynamic programming approach is used for evaluating the Connected Detour number of a connected graph G as described in [10] to evaluate Detour number.. The overall process is divided into two stages in the algorithm. The first stage of the algorithm records the longest path between each vertex to all other vertices for further process. The second stage of the algorithm uses the recorded longest path and finds the closed interval

$I_D[S]$ which is the union of the intervals I_D [u, v] over all pairs of vertices u and v in S and thereby determines the connected detour number of the given graph G.

Algorithm for Connected Detour Number:
Algorithm ConnectedDetourNumber(A,V)

```
// Input: A-Adjacency matrix and V-Set of Vertices of a graph G
// Output: Connected Detour Number of G
1.  Record the edges in the longest path between each vertex to every other vertices in a set called
    DetourPath
2.  Found = false
3.   While (SSize ≤ |V|  and Found≠ true)
4.         SSize = SSize+1
5.         Generate SS subsets of V such that ∀ S∈ SS and |S|=SSize
6.          While SS≠Ø and Found≠ true
7.          Get next subset S from SS
8.          For every vertex u and v in S
9.            Find the closed interval I_D[S] union of the intervals I_D [u, v] over all pairs of vertices u and v
    in S from set DetourPath[u, v] computed above.
10.                Loop
11.      If I_D[S] =E Then
// Finding if a subgraph G[S] is induced by S
12.     Collect all possible subgraphs say H_1(U,F_1), H_2(U,F_2)..... H_n(U,F_n) of the graph S fixing the set of
    vertices U in H_i where F_1, F_2, F_3, ........, F_n are subset of E.
13.       Find F=max{F_1,F_2,....F_n}
14.       If all edges in G are in H then  Found=true
15. Loop
16. Loop
17. Return SSize
```

Theorem 2.1: Let G be an undirected connected graph such that $|V|\geq 2$, the Algorithm 2.1 finds the connected detour number **cdn(G)** and $\mathbf{2\leq cdn(G)\leq |V|}$.

Proof: Let G be a connected graph with $V=\{v_1,v_2,v_3,......,v_n\}$. Step 1 of the Algorithm 2.1 evaluates the longest path between each vertex to all other vertices. The vertices lying in the identified longest path between each vertex and all other vertices is counted and the related edges are recorded in a set data structure called DetourPath for the later reference related to the process of finding detour number. Step 5 of the Algorithm 2.1 generates subsets of V, SS = $\{v_1, v_2, v_3,......,v_k \mid 2\leq k\leq n\}$. Cardinality of generated subset starts with minimum value 2 and it increases by one during every iteration until detour set is found. Step 7 of the Algorithm 2.1 chooses a subset S = $\{v_1, v_2, v_3,......,v_k \mid 2\leq k\leq n \}$ from SS and Step 9 of the Algorithm 2.1 and executes the main the process of finding the detour number by evaluating closed interval $I_D[S]$ which has the set of vertices of G that forms the longest path between vertices in S from the Path set that contains the longest path from each vertex to every other vertex. Step 11 of the Algorithm 2.1 checks if $I_D[S]=E(G)$ then S is considered as the detour set. When the subgraph G[S] induced by S is connected , then S is the connected detour set and the connected detour number is **|S|** and **|S|**≥ 2 to be the connected detour number. Steps 3 to step 16 of the Algorithm 2.1 are repeated until a minimum connected detour set is identified.

If $I_D[S]$ = E(G) in the first iteration itself, the connected detour number is 2 otherwise it linearly increases by one for each iteration. For the worst case S contains all the vertices of G and so the detour number **cdn(G)** is **|V|.** Hence it is proved that $\mathbf{2\leq cdn\ (G)\leq |V|}$.

Theorem 2.2: Connected Detour number of an undirected graph G can be computed using Algorithm 2.1 in $\mathbf{O(|V|^3+2^{|V|})}$.

Proof: Let G be a connected undirected graph with V= $\{v_1, v_2, v_{3,\ \dots}\ v_n\}$. Algorithm 2.1 initiates the task of finding detour number by evaluating longest path from each vertex to all other vertices. All the vertices lying in the identified longest path are recorded in a set data structure called DetourPath for the later reference related to the process of finding detour number. The efficiency of this process is $O(|\mathbf{V}|^3)$. The algorithm continues by constructing the subset of V(G), chooses each set S (as per Step 5 and Step 7 of the Algorithm 2.1) and evaluates the closed interval $I_D[S]$ of S be the union of the intervals $I_D[u, v]$ over all pairs of u-v detour of vertices u and v in S. Computing of $I_D[S]$ which is the detour of a set of vertices is done by referring the DetourPath set constructed in Step 1 of the Algorithm 2.1. For each subset S, computing of $I_D[S]$ is to be performed repeatedly until the detour set is finalized. There are maximum $\mathbf{2}^{|V|}$ subsets constructed and the closed interval evaluation is performed for the worst case for all the subsets and so efficiency of this process is $O(\mathbf{2}^{|V|})$. Hence the computing of connected detour number **cdn(G)** using Algorithm 2.1 can be completed in $\mathbf{O(|V|^3+2^{|V|})}$.

Algorithm For Computing Forcing Connected Detour Number: An algorithm using dynamic programming approach is introduced for evaluating Forcing connected detour number of a connected graph G. This algorithm follows every step as we discussed in Algorithm 2.1 and additionally includes the steps to check the forcing connected detour set to determine the forcing detour number.

Algorithm for Forcing Connected Detour Number:
Algorithm ForcingConnectedDetourNumber(A)
// Input: Adjacency matrix and Set of Vertices of a graph G
// Output: Forcing Connected Detour Number of G

1. Record the edges in the longest path between each vertex to every other vertices in a set called DetourPath
2. Found = false
3. While (SSize ≤ |V| and Found≠ true)
4. SSize = SSize+1
5. Generate SS subsets of V such that ∀ S∈ SS and |S|=SSize
6. While SS≠∅ and Found≠ true
7. Get next subset S from SS
8. For every vertex u and v in S
9. Find the closed interval $I_D[S]$ union of the intervals I_D [u, v] over all pairs of vertices u and v in S from set **DetourPath[u, v]** computed above.
10. Loop
11. If $I_D[S]$ =E Then

// Finding if a subgraph G[S] is induced by S

12. Collect all possible subgraphs say $H_1(U,F_1)$, $H_2(U,F_2)$.....$H_n(U,F_n)$ of the graph S fixing the set of vertices U in H_i where F_1, F_2, F_3, ..., F_n are subset of E.
13. Find $F=\max\{F_1,F_2,....F_n\}$
14. If all edges in G are in H then
15. Found=true
16. Record all SSize that constitute the connected detour set that is cdn(S).
17. Loop
18. Loop
19. fcdn= min{cdn(S)}
20. Return fcdn

Theorem 3.1: Let G be an undirected connected graph such that |V|≥2, the Algorithm 3.1 finds the forcing connected detour number **cdn(G)** and **2≤ cdn(G) ≤|V|.**
Proof: Let G be a connected graph with $V=\{v_1,v_2,v_3,......,v_n\}$. Step 1 of the Algorithm 3.1 evaluates the longest path between each vertex to all other vertices. The vertices lying in the identified longest path between each vertex and all other vertices is counted and the related edges are recorded in a set data structure called DetourPath for the later reference related to the process of finding detour number. Step 5 of the Algorithm 3.1 generates subsets of V, SS = $\{v_1, v_2, v_3,......,v_k \mid 2\le k\le n\}$. Cardinality of generated subset starts with minimum value 2 and it increases by one during every iteration until detour set is found. Step 7 of the Algorithm 3.1 chooses a subset S = $\{v_1, v_2, v_3,......,v_k \mid 2\le k\le n \}$ from SS and Step 9 of the

Algorithm 3.1 and executes the main the process of finding the detour number by evaluating closed interval $I_D[S]$ which has the set of vertices of G that forms the longest path between vertices in S from the Path set that contains the longest path from each vertex to every other vertex. Step 11 of the Algorithm 3.1 checks if $I_D[S]=E(G)$ then S is considered as the detour set. When the subgraph G[S] induced by S is connected , then S is the connected detour set and the connected detour number is $|\mathbf{S}|$ and $|\mathbf{S}|\geq 2$ to be the connected detour number. Steps 3 to step 16 of the Algorithm 3.1 are repeated until a minimum connected detour set is identified.

If $I_D[S] = E(G)$ in the first iteration itself, the connected detour number is 2 otherwise it linearly increases by one for each iteration. For the worst case S contains all the vertices of G and so the detour number **cdn(G)** is $|\mathbf{V}|$. Hence it is proved that $\mathbf{2\leq}$ **cdn (G)** $\leq|\mathbf{V}|$. Among the connected detour number the minimum number is selected so that it is the forcing detour number such that $\mathbf{2\leq}$ **fcdn (G)** $\leq|\mathbf{V}|$**.**

Theorem 3.2: Connected Detour number of an undirected graph G can be computed using Algorithm 3.1 in $\mathbf{O(|V|^3+2^{|V|})}$.

Proof: Let G be a connected undirected graph with $V= \{v_1, v_2, v_{3, \dots} v_n\}$. Algorithm 3.1 initiates the task of finding detour number by evaluating longest path from each vertex to all other vertices. All the vertices lying in the identified longest path are recorded in a set data structure called DetourPath for the later reference related to the process of finding detour number. The efficiency of this process is $O(|\mathbf{V}|^3)$. The algorithm continues by constructing the subset of V(G), chooses each set S (as per Step 5 and Step 7 of the Algorithm 3.1) and evaluates the closed interval $I_D[S]$ of S be the union of the intervals $I_D[u, v]$ over all pairs of u-v detour of vertices u and v in S. Computing of $I_D[S]$ which is the detour of a set of vertices is done by referring the DetourPath set constructed in Step 1 of the Algorithm 3.1. For each subset S, computing of $I_D[S]$ is to be performed repeatedly until the detour set is finalized. There are maximum $\mathbf{2}^{|V|}$ subsets constructed and the closed interval evaluation is performed for the worst case for all the subsets and so efficiency of this process is $O(\mathbf{2}^{|V|})$. Hence the computing of forcing connected detour number f**cdn(G)** using Algorithm 3.1 can be completed in $\mathbf{O(|V|^3+2^{|V|})}$**.**

Conclusion: The concept of connected detour set and forcing connected detour set and there by determining the number is discussed. Algorithms for Connected Detour set identification and there by the evaluation of connected Detour numbers and forcing connected detour set numbers has been developed. The performance of those algorithms is also analyzed. These concepts have interesting applications in the Channel Assignment Problem in radio technologies.

References:

1. F. Harary, Graph Theory, Addison-Wesley, Reading, Mass, 1972.
2. Gary Chartrand and Ping Zhang, Introduction to Graph Theory (Tata Mc-Graw-Hill, 2006).
3. G. Chartrand, G.L. Johns, P. Zhang, The Detour Number of a Graph, Utilitas Mathematica, 64 (2003), 97-113.
4. G. Chartrand, G.L. Johns, P. Zhang, On the Detour Number and Geodetic Number of a Graph, Ars Combinatoria, 72 (2004), 3-15.
5. A.P.Santhakumaran and S.Athisayanathan, On the Connected Detour Number of a Graph, Journal of Prime Research in Mathematics Vol. 5 (2009), 149-170.
6. A.P. Santhakumaran and P.Titus, The Vertex Detour Number of a Graph, AKCE J. Graphs. Combin., 4, No. 1 (2007), pp. 99-112
7. A.P. Santhakumaran and S. Athisayanathan, On Edge Detour Graphs, Discussiones Mathematicae Graph Theory 30 (2010) 155-174.
8. A.P.Santhakumaran and S.Athisayanathan, The Forcing Weak Edge Detour Number of a Graph, International J.Math. Combin. Vol.2 (2010), 22-29.
9. A. P. Santhakumaran and S. Athisayanathan: Weak Edge Detour Number of a Graph. Ars Comb. 98: 33-61 (2011).
10. Albert Antony Raj.S and Victor S.P , Algorithms to find detour numbers, vertex and edge detour numbers in graphs, International Journal of Applied Engineering Research, Vol 9 (2014) , I 24, B 30323-E 30332.