This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2009 proceedings

Coding-Aware Scheduling for Reliable Many-to-One flows

Osameh M. Al-Kofahi Ahmed E. Kamal
Department of Electrical and Computer Engineering, Iowa State University
e-mails:{osameh, kamal}@iastate.edu

***Abstract*—We revisit the problem of scheduling the sources transmissions in a many-to-one flow to provide reliable communication between n sources and a single destination. The performance of coding-aware scheduling is studied based on both digital network coding (DNC) and analog network coding (ANC). We discuss some special cases in which an optimal ANC-based schedule can be constructed efficiently. Finally, we show that the maximum gain from using ANC is theoretically bounded by n, where n is the number of sources.**

I. Introduction

The problem of providing proactive protection for many-to-one flows was studied in [1]. The problem considers a set of wireless mesh routers and the set of wireless mesh clients (or users) associated with them. It is assumed that the users data units will be relayed to a common gateway through multihop wireless communication. The set of users (U_s) contains n users, and the set of routers (L_s) contains at least $n+1$ routers. The routers are to create $n+1$ linear combinations from the user's data units, such that any n of them are linearly independent. These $n+1$ combinations are then forwarded to the sink on $n+1$ edge-disjoint paths, which means that a single link failure will at most affect one path, and thus the sink will be able to recover the original data units if at most one failure took place, by using the remaining n linear combinations.

In this paper we are interested in utilizing network coding to schedule the users transmissions to the routers. In [1], it was shown that network coding-based protection takes some extra time slots (1 slot on average) compared to 1:N protection. In this paper we show how to enhance the transmissions schedule for the network coding case by using analog network coding (ANC) [2] instead of digital network coding (DNC).

Analog network coding (ANC) was proposed in [2] to enhance the capacity of the two-way relay channel, where there are 2 terminal nodes that communicate with each other through a relay node. If the 2 terminal nodes transmit together, the relay node receives their added analog signals, and then forwards the result to the two terminal nodes. After that, each terminal can recover the data destined to it by subtracting its data from the received signal. That is, with ANC 2 packets can be exchanged through the relay node and received by the terminals in only 2 time slots (compared to 3 with digital network coding or 4 without coding). This is clarified in the example shown in Figure 1, where nodes A and B are exchanging their data units through the relay node R.

This work was supported in part by the National Science Foundation under grants CNS-0626822, CNS-0626741, CNS-0721453 and ECS-0601570 and by a gift from Cisco Systems.

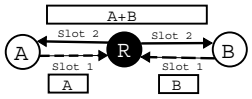

Fig. 1. 2-way relay channel

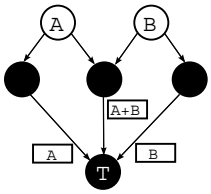

Fig. 2. Protecting 2 data units that are destined to node T against a single link failure

Note that the scheduling scenario considered in this paper is not similar to the 2-way relay channel. This is because all the data units and the sum of the analog signals will be forwarded to a single destination, through multihop wireless communication. When receiving all this information the common destination will be able to decode the data units. This is illustrated in Figure 2 that shows the 2 users A and B, and their 1-hop neighboring routers (the unlabeled black nodes). The edges represent the available wireless links between the users and the routers. Using ANC, both users transmit in the same time slot, which results in the combinations shown in the figure at the router nodes. One should note that the same combinations can be constructed at the routers using DNC with $\{0,1\}$ coefficients. Note also that ANC is used for the transmissions from the users to the routers, but not for the transmissions from the routers to the common destination.

This paper is organized as follows. The problem description is in Section II. We discuss the problem of scheduling based on digital network coding in Section III. Scheduling based on analog network coding is discussed in Section IV. The performance of the two schemes is evaluated and compared in Section V. Finally, the paper is concluded in Section VI.

II. Problem Description

In this paper we consider applying analog network coding to the scheduling problem presented in [1]. We summarize the problem for the reader's convenience. The problem considers a set of wireless mesh routers, and we refer to it by L_s, where $|L_s| \geq n+1$. Associated with this set of routers is a set of wireless mesh clients, U_s, of size n. The users in U_s need to send their data units to the routers in L_s to create $n+1$ linear combinations, such that any n combinations of them are linearly independent. We make the following assumptions:

978-1-4244-3435-0/09/$25.00 ©2009 IEEE

1) The nodes in U_s and L_s, and the available connections between them are modeled as a bipartite graph.
2) $|U_s| = n$ and $|L_s| = m \geq n + 1$.
3) To be able to provide protection [1], we need any k users in U_s to have a set of neighboring routers in L_s of size $k + 1$ at least, where $1 \leq k \leq n$.

In this paper we investigate the possible benefits that could be gained from using analog network coding (ANC) in the scheduling process. We show how ANC can make a scheduling more efficient. We also discuss some special cases, in which the optimal (i.e., minimum) number of time slots can be achieved for an ANC-based schedule. Moreover, we show that, theoretically, ANC can outperform DNC by a factor of n if the bipartite graph G had a certain structure, which will be discussed later in Section IV-C.

III. Scheduling Based on Digital Network Coding

In a feasible schedule for DNC, users in U_s that have a common router in L_s (i.e., connected to the same router) cannot be scheduled in the same time slot (this is called the *feasibility condition* for a DNC-based schedule, and will be referred to later in Section V). This is because the router needs to know the actual bit representation for each of the data units from the two users to be able to perform digital network coding. That is, we say that there is a conflict between two users in U_s if they have a common neighboring router in L_s. In this case there is a direct relation with the vertex coloring problem. The DNC-based scheduling problem can be shown to be NP-Complete by the following reduction from the vertex-coloring problem on a given conflict graph $H(V', E')$:

- The nodes in V' are mapped to the nodes in U_s.
- An edge (u, v) in H is transformed to a node n_{uv} that belongs to L_s, and is connected to nodes u and v in U_s.
- For each node in U_s add a neighbor in L_s and connect them together to satisfy assumptions 3 and 4, if needed.

Thus an optimal and feasible schedule of k time slots exists if and only if H is k-chromatic. This mapping is not the only possible mapping from the vertex coloring problem to the DNC-based scheduling problem, however this does not invalidate the reduction. Also note that using this transformation all the nodes in L_s will have degree 2. Nevertheless, this does not mean that we can construct a conflict graph for G only if all of its L_s nodes had degree 2. For example consider the graph in Figure 3, applying the transformation stated above will result in the graph shown in Figure 5. However, this is not the only graph that has a conflict graph similar to H in Figure 3. For instance the graph shown in Figure 6 will have exactly the same conflict graph H, and thus solving the vertex-coloring on H provides a feasible and optimal schedule for both the graphs in Figures 5 and 6. Although in our coming discussion we use examples where all the nodes in L_s has degree 2, the analysis is valid and applies for any graph G.

We use this relation with the vertex-coloring problem to bound the number of needed time slots for a DNC-based schedule. Let Δ_{L_s} be the maximum node degree in L_s in G, let T_{schd} be the minimum number of time slots needed for a schedule (or equivalently the number of colors in H), and let Δ_H be the maximum node degree in the conflict graph H corresponding to G. We have the following bounds on the number of needed time slots.

$$\Delta_{L_s} \leq T_{schd} \leq \Delta_H$$

The lower bound follows since no two sources in U_s can be scheduled together if they have a common neighbor in L_s. The upper bound follows from Brook's theorem [3], which says that *"for a connected graph H that is neither a full graph nor an odd cycle the chromatic number $\chi(H)$ is less than or equal to the maximum node degree Δ in H"*. If H is a full graph or an odd cycle then $\chi(H) = \Delta + 1$.

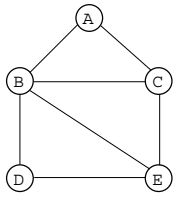

Fig. 3. Graph H

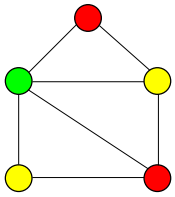

Fig. 4. 3-coloring of H

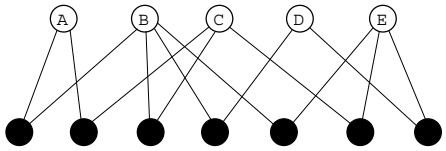

Fig. 5. Corresponding Graph G

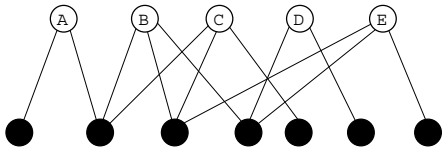

Fig. 6. An alternative corresponding Graph G

It easy to solve the coloring problem for graph H in Figure 3 by just looking at it. The minimum number of needed colors for H is 3 and the solution is shown in Figure 4. This indicates that an optimal schedule for the transmissions of the sources in G needs 3 time slots. For this example note that $\Delta_{L_s} = 2$, $T_{schd} = 3$, and $\Delta_H = 4$.

Finally, one should note that creating the combinations in the L_s nodes is independent from the transmission schedule of the users. That is, after all users have transmitted, a coding tree [1] can be constructed to decide on the linear combinations. A coding tree is simply a tree on the bipartite graph that contains all the nodes in U_s and all of its leaves are router nodes in L_s. Assigning a coding coefficient of 1 to the links on the tree and 0 to those not on the tree will give the needed $n + 1$ combinations. Any coding tree will have at least 2 leaf nodes corresponding to 2 *leaf combinations*, which are basically trivial combinations containing a single data unit. These leaf combinations are critical in the decoding process since the destination needs at least one of them to be able to

recover the original data units. But since we assume there can be at most one failure the sink is guaranteed to receive at least 1 leaf combination. One possible coding tree for the graph G in Figure 5 is shown in Figure 7. The solid links represent the links on the tree, which will decide on the combinations as shown. For more details the reader is referred to [1].

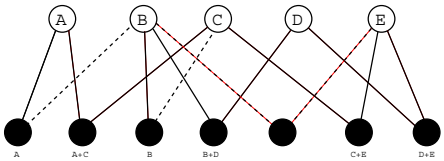

Fig. 7. Coding tree on graph G

IV. Scheduling based on Analog Network Coding

Using ANC enables a node in L_s to receive from at most two sources at the same time. This means we can schedule transmissions from 2 nodes in U_s to the same node in L_s in the same time slot. Therefore, the lower bound on the minimum number of time slots needed, T_{schd}, is as follows:

$$\lceil \frac{\Delta_{L_s}}{2} \rceil \leq T_{schd}$$

Unfortunately, there is no clear relation between scheduling in this case and vertex coloring. However, we can still take Δ_H as a pessimistic upper bound. Note that unlike DNC-based scheduling, a node in L_s can receive from 2 nodes in U_s. Therefore, the transmissions from the users in a time slot will decide the combinations that will be sent to the common destination. For example, if in a certain time slot both users A and B transmitted to a common router, the router will receive the sum of the two analog signals. This sum will be treated as a degree-2 combination (i.e., a combination of 2 data units).

In addition to not hearing from more than two sources, there is one more constraint that must be satisfied in each time slot of an ANC-based schedule, and that is to have the suitable number of leaf combinations that will enable the decoding at the sink. Specifically, every set of sources that transmit in a certain time slot must produce at least two leaf combinations. For example, Figure 8 shows a feasible schedule based on analog network coding that takes 2 time slots for graph G shown previously in Figure 5. In the first time slot, sources A and B transmit and produce 4 leaf combinations. The remaining sources transmit in the second slot and produce 4 leaf combinations also.

If duplicate or excess combinations are created by the transmissions in a time slot of an ANC-based schedule we assume that the duplicates are eliminated to reduce the number of packets sent to the destination. This can be done by finding a coding tree for the sources in each time slot and then discarding the combinations created at nodes not on the tree. For example in Figure 8 we only need the combinations inside the boxes, and all other combinations are not discarded.

To summarize, in a feasible ANC-based schedule the following conditions must hold:

1) In any time slot a node in L_s does not receive from more than 2 nodes in U_s.
2) A set of sources that transmit in a certain time slot must produce at least 2 leaf combinations.

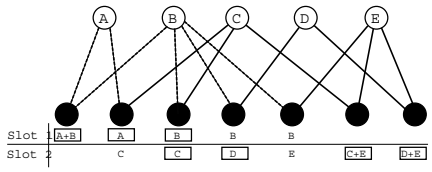

Fig. 8. ANC Schedule

A. Special Case: When $\Delta_{L_s} = 2$

In this case, the first feasibility condition is satisfied by the construction of the bipartite graph G. However, we still need to satisfy the second feasibility condition for an ANC-based schedule. Practically, there are two possibilities:

1) There are 2 nodes in L_s with degree 1: in this case the second condition will also be satisfied and all sources in U_s can transmit in a single time slot.
2) All nodes in L_s have degree 2: in this case we can schedule any single node in U_s that has at least 2 neighbors in L_s in one time slot, and schedule all remaining users in U_s in the second time slot. Scheduling the node in the first time slot creates a sufficient number of leaf combinations in the second time slot to make it feasible.

B. Special Case: When G is a Tree

In this case the second feasibility condition is always satisfied for any set of sources, and we need to guarantee the first feasibility condition if the max-degree in L_s is larger than 2. We show that for this special case the needed number of time slots is $\lceil \frac{\Delta_{L_s}}{2} \rceil$.

Let us consider Algorithm 1. In a certain time slot, the algorithm starts by considering the node in L_s with the highest degree (node v), it marks 2 of its neighbors (which are users in U_s), and deletes all other unmarked neighbors. After that, it considers other nodes in L_s with marked or deleted U_s neighbors. If a node , u, has a marked neighbor it can mark another one (since it can receive from at 2 users). If u has a deleted neighbor, and at least 2 remaining neighbors it marks 2 for transmission. Finally, if u has a deleted neighbor, and a single remaining neighbor it just deletes this remaining neighbor so that it will be scheduled later. When there are no more nodes to consider, time slot x is filled with the nodes in set S. To find a feasible schedule, the process is repeated until all users are scheduled.

Lemma 1. *If G is a tree and Algorithm 1 is used, then any node in L_s with a degree > 2 will have its degree reduced by 2 after each iteration of Algorithm 1.*

Proof. First of all, note that for an L_s node to have its degree reduced by 2, two of its neighbors in U_s must be marked. After marking 2 neighbors of node v (the one with the max degree in L_s) and deleting the remaining neighbors, node v

Algorithm 1 ANC-Based Scheduling on a Tree

Input: Graph $G(V, E)$, $V = \{U_s \cup L_s\}$. G is bipartite, and a tree.
Output: Feasible ANC-based schedule that takes $\lceil \frac{\Delta_{L_s}}{2} \rceil$ time slots.

$S = \emptyset$, $N = \emptyset$, $T = \emptyset$, $x = 0$
while ($|T| < |U_s|$) **do**
  v = Node in L_s with max degree.
  Mark any two U_s neighbors of v for transmission.
  $S = S \cup$ {Marked neighbors of v}
  Delete all other U_s neighbors of v. //Temporarily
  $N = N \cup$ {All 2-hop neighbors of v}
  while ($N \neq \emptyset$) **do**
    Remove u from N
    if (u has a marked neighbor) **then**
      Mark an extra neighbor of u, & delete all others.
    else if (u has a deleted neighbor & remaining degree ≥ 2) **then**
      Mark two neighbors of u, & delete all others neighbors.
    else if (u has a deleted neighbor & remaining degree == 1) **then**
      Delete remaining neighbor
    end if
    $S = S \cup$ {Marked neighbors of u}
    $N = N \cup$ {All 2-hop neighbors of u not in S}
  end while
  Put all nodes in S in slot number x.
  Remove the nodes in S and their incident edges from G
  $T = T \cup S$, & $S = \emptyset$
  $x++$
end while

starts reaching-out to its 2-hop neighbors (a 2-hop neighbor is reached when it has a marked or a deleted U_s neighbor), and in the next iteration the 2-hop neighbors will reach-out to the 4-hop neighbors, and so on.

We now prove the implication by contradiction. Assume Algorithm 1 was used, but some node, u, in L_s with degree > 2 did not have its degree reduced by 2 after running a single iteration of Algorithm 1. For this to happen, node u must have had all of its neighbors deleted in the reaching-out process, which means that all of its neighbors can be reached from node v (i.e., there are multiple paths from v to u). However, this contradicts the assumption that G is a tree. Therefore, every node in L_s with degree > 2 must have its degree reduced by 2 after Algorithm 1 is executed ■.

Theorem 1. *If the bipartite graph G is a tree, a feasible ANC-based schedule can be achieved in exactly* $\lceil \frac{\Delta_{L_s}}{2} \rceil$ *time slots.*

Proof. A direct proof is used. We rely on Lemma 1 and the special case in Section IV-A to prove this theorem. Basically, we need to count the number of time slots needed to reduce the max-degree in L_s to 2, let us call this number x. When the max degree in L_s is 2 all nodes in U_s will be marked by Algorithm 1, and non of them will be deleted. Therefore, the total number of needed time slots is $x + 1$. From the previous lemma the maximum degree after x time slots, d_x, is:

$$d_x = \Delta_{L_s} - 2x \tag{1}$$

Now let $d_x = 2$, and solve for x

$$x = \frac{\Delta_{L_s}}{2} - 1$$

Adding the last time slot for the remaining sources, and taking the ceiling of $\frac{\Delta_{L_s}}{2}$ if Δ_{L_s} was odd, we have:

$$T_{schd} = \lceil \frac{\Delta_{L_s}}{2} \rceil \qquad \blacksquare$$

C. Maximum Gain of ANC-Based Scheduling

The gain of ANC-based scheduling is maximized when it requires the minimum number of slots, while DNC-based scheduling requires the maximum number of slots. This happens when the following is true: 1) the maximum degree in L_s is 2 and there are at least two nodes in L_s with degree 1, i.e., only one time slot is needed for ANC. 2) the conflict graph (H) corresponding to G, is fully connected. In this case the gain is n. However, if all the nodes in L_s had degree 2, the gain will be $\frac{n}{2}$. An example is shown in Figure 9, where the maximum degree in L_s in the graph is 2. An ANC schedule is shown in the figure, where node A transmits in the first time slot, and all remaining nodes transmit in the second time slot. The digital network coding based schedule is better shown on the corresponding conflict graph H, where each color on H represents a time slot. This is shown in Figure 10.

Recall that this scheduling problem has originated from the need to provide protection against a single failure for n data units. Therefore, practically, we do not need more than $2n$ neighboring routers. Therefor, although this case is theoretically possible and illustrates the maximum gain of ANC-based scheduling, it is practically unlikely to happen. This is because we need $\binom{n}{2}$ nodes in L_s to get a fully connected conflict graph

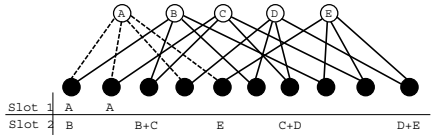

Fig. 9. 2-slot ANC Schedule

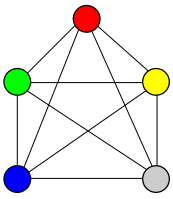

Fig. 10. n-slot DNC Schedule

V. Performance Evaluation and Comparison

In this section we are interested in evaluating and comparing the performance of both scheduling schemes on large and random graphs. For this purpose, we use the simple greedy algorithm, shown in Algorithm 2.

In a certain time slot, the algorithm starts by selecting the user in U_s with the least degree (to reduce future conflicts with other users) that is not scheduled for transmission in any previous time slot. After that, all the users that will invalidate the schedule feasibility, are removed by putting them in the set of forbidden sources F. This process is repeated until no more sources can be added to the current time slot. Then the

Algorithm 2 Scheduling Algorithm

Input: Graph $G(V, E)$, $V = \{U_s \cup L_s\}$. G is bipartite.
Output: Feasible ANC-based or DNC-based schedule //depending on the feasibility conditions checked in Step 8

1: $S = \emptyset$, $F = \emptyset$, $N = U_s$
2: $Slots = 0$
3: **while** ($|S| < |U_s|$) **do**
4: Select user, u, with minimum degree in N, and $u \notin S$.
5: $S = S \cup \{u\}$
6: $F = F \cup \{All\ sources\ infeasible\ with\ any\ source\ in\ S\}$
7: $N = N \backslash F$
8: **if** ($N == \emptyset$) **then**
9: $Slots++$
10: $N = U_s$, & $F = \emptyset$
11: **end if**
12: **end while**
13: **return** $Slots$

algorithm starts filling a new time slot with users that are not scheduled previously, and so on. The infeasible sources in step 8 in algorithm 2, are the ones that will result in violating the feasibility conditions defined previously in Section III for DNC, and in Section IV for ANC.

We ran Algorithm 2 on four different configurations, where the degree, D, of all nodes in U_s took the values 2, 3, 5 and 10. Because of space limitations we only show the detailed results for $D = 3$ and $D = 10$. The results for $D = 3$ are shown in Figure 11, where the number of time slots required under DNC is reduced by up to 44% on average if ANC is used on the same graph. The results for $D = 10$ are shown in Figure 12, the gain is less (around 41%) for this case because when the node degree is large and the number of U_s nodes is small, ANC and DNC will have the same performance (for $|U_s| \leq 6$). To compare the four configurations against each other, we have plotted the gain for all configurations in Figure 13. When $|U_s| \leq 5$ the largest ANC gain is for the configuration with smallest degree. This is because when the number of nodes is small, having a small degree will help in satisfying the second feasibility condition for ANC-based scheduling, and thus more users can be grouped together. However, when $|U_s| \geq 12$, the best gain is for the configuration with the largest degree. This is because, if the number of nodes is small and the node degree is small then DNC's performance will be very close to ANC. But, as the node degree increases, the performance of DNC will get worse faster than ANC because the conflicts will increase, and thus ANC starts to gain performance over DNC. Although we have shown that ANC can outperform DNC by a factor of n theoretically, it is clear that on randomly generated graphs the actual gain of ANC is around 2.

VI. Conclusions

We showed how ANC affects the scheduling problem for reliable many-to-one communication. Applying ANC reduces the number of needed time slots to schedule the transmissions of the data units from the sources to the coding nodes. We showed some cases in which the optimal ANC-based schedule can be found in polynomial time. In addition, we showed that using ANC can reduce the number of required time slots by at most a factor of n compared to using DNC, where

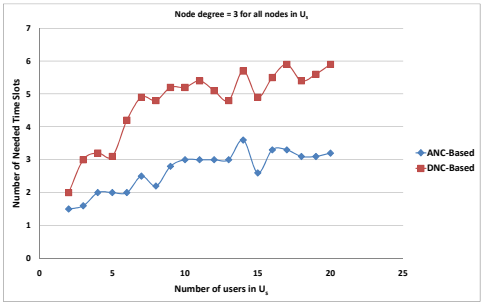

Fig. 11. Comparison between ANC and DNC. All U_s nodes have degree 3

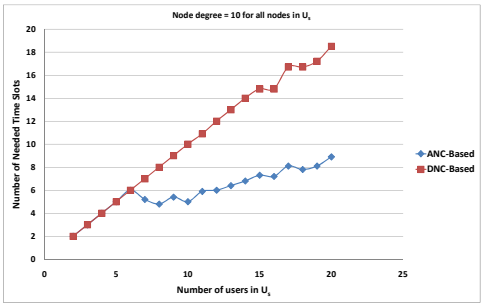

Fig. 12. Comparison between ANC and DNC. All U_s nodes have degree 10

n is the number of source nodes. This however, is highly unlikely to happen in practice. We evaluated and compared both scheduling schemes on randomly generated graphs. It was shown that the gain from ANC is usually around 2, and that the ANC gain depends on the node degree of the sources. Specifically, the more sources we have the more dense the network should be, and vice versa.

References

[1] O. Al-Kofahi and A. Kamal. Network coding-based protection of many-to-one wireless flows. Accepted in the IEEE Journal on Selected Areas in Communications (an earlier version appeared in the proceedings of the 4th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS 2007)).

[2] S. Katti, S. Gollakota, and D. Katabi. Embracing wireless interference: Analog network coding. In the proceedings of SIGCOMM 2007.

[3] R. Diestel. Graph theory (electronic edition 2005). Springer-Verlag Heidelberg, New York 1997, 2000, 2005.

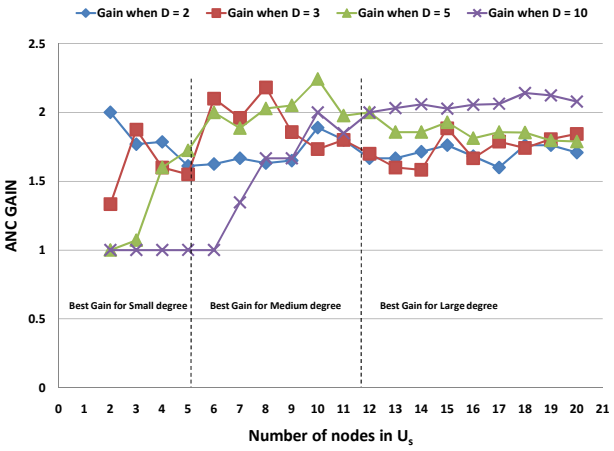

Fig. 13. ANC Gain comparison for different node degree