A Novel Scheduling Approach for Packet Delay in Wireless Ad Hoc Networks and Comparative Analysis with Other Scheduling Approaches

B. Sindhupiriyaa* and D. Maruthanayagam
PG and Research Department of Computer Science, Sri Vijay Vidyalaya College of Arts and Science, Dharmapuri – 636807, Tamil Nadu, India; sindhupiriyaamca2011@gmail.com, dr.d.maruthanayagam@gmail.com

Abstract

Objectives: The goal of the work is to design a network as well as a new scheduling approach that provides minimizing the delay, better operation of resources in the network and increasing optimality. Methods: In the wireless network and wire-line Network, the delay stability is one of the significant metrics considered. The delay analysis of packets plays an essential job in the network. Decomposition of packets partition into multiple paths, checks whether any two nodes assemble at same point as bottleneck occurred or not. To overcome these issues, it utilizes a new delay efficient scheduling policy. Findings: In this research, a new scheduling algorithm, Ant Colony Optimization Max Weight Scheduling (ACOMWS) for minimizing the delay, is proposed. For recognizing the behavior of each path in the network, lower bound analysis is used. Comparative result analyses of proposed scheduling algorithm (ACOMWS) are evaluated with other algorithms (MWS, BP, ACO, Greedy, and GMWS). Application/Improvements: The proposed hybrid scheduling algorithm (ACOMWS) improves the results to achieve better precision results compared to the previous related works.

Keywords: Adhoc, Backpressure, Greedy, Max-weight, Packet Delay, Scheduling, Throughput Optimal

1. Introduction

In ad hoc network, a mobile node may possibly inject similar traffic into the network at the proximal locations for the reason that their gathered information at these locations could be spatially correlated in the network traffic. Even if this correlation decays slowly as the time lag increases, then the traffic could exhibit heavy-tailed behavior or Long Range Dependent (LRD). Heavy tailed delay recognizes lower orders, for example, mean and variance. In that case, the network can show evidence of significant performance degradations including the noticeably system scalability, reduced network throughput and queue stability. In wireless domain, the tail behavior of transmission delay is still preformed with an under-explored area, by reason of the complex and dynamic network.

While transmission delay is a single-user performance metric, system stability is a network extensive one. A network is stable even if a feasible scheduling algorithm under which each network user has finite time-average expected queue length. The stability region is described as the closure of the set of all arrival rate vectors where the queues of all wireless users stabilize by a scheduling policy in network. Even though network stability areas and throughput-optimal scheduling algorithms have been explored in different network scenarios, they are generally derived under light tailed channel and traffic models. However, recent research on queuing theory shows that when two queues share the same server under the MWS algorithm, which is verified to be throughput optimal under light-tailed traffic arrivals, the queue with light-tailed traffic can experience unbounded average

*Author for correspondence
delay when the other queue has unbounded delay variance using heavy-tailed traffic.  
In this case, the nodes with parallel queue length will transmit with the same priority that is particularly problematic at the same time as the priority is high\textsuperscript{2,3}. Higher priorities control smaller contention windows and the probability of collision is much higher. Collisions save the performance of the MAC protocol. As a result, the benefits of “Throughput-optimal” scheduling in this case will be limited by enhancing collisions. The current queue state of any contending nodes manages at the same time of communication, so allocating nodes to adapt their priority depending on their neighbors by allocating nodes to exchange queue information. The proposed throughput optimal algorithms ACOMWS efficiently addresses these challenges in a trouble-free manner and to get better stability of the network by minimizing the delay.

2. Reviews on Various Scheduling Algorithms

2.1 Max-weight Scheduling
Max-Weight scheduling mainly solves the delay stability of traffic flows\textsuperscript{4,5}. The “Weight” of a feasible schedule is the sum of the lengths of the total queues comprised in it. MWS makes active a feasible schedule, at any particular time slot. The scheduling vector \( S(t) \) evaluates for Max-Weight policy as follows

\[
S(t) = \arg \max_{\{s_i\} \in S} \sum_{f=0}^{F} Q_{f(t)} \cdot S_f
\]

If the set on the right-hand side comprises multiple feasible schedules, after that one of them is uniformly selected at random.

Weight function: The matching policy is defined similar to a specific weight. For the definition of the weights, will make use of a family of real positive functions \( f_{k(x)} \) \( N \mapsto R, 1 \leq k \leq K \) that satisfy the following property\textsuperscript{6}:

\[
\lim_{n \to \infty} \frac{f_k(n)}{n} = \frac{1}{W_k}
\]

Computation of Lower Bound: Lower bound on the expected queue length is determined the sum of the contribution of each link toward the lower bound. Consider \( X \) is the set of \( N \) number of Systems established in the Network.

a. Step 1: Initially set a bound value to 0. This bound value is utilized for storing the Lower bound value in the extreme subset \( X_1 \) from the set \( X \).
b. Step 2: Obtain exclusive subset \( X_1 \) from the set \( X \) that maximize the lower bound value.
c. Step 3: Calculate the bound value by adding the bound and lower bound value of the exclusive subset \( X_1 \).
d. Step 4: Divide the entire number of system connected in the network by the subset of the system.
e. Step 5: Repeat the steps 3, 4 and 5, until the \( X \) is equal to null.
f. Step 6: Return the Bound value.

Minimization of Upper Bound: The Algorithm to minimize the upper bound by computing the network utility maximization using convex optimization techniques is stated below. Let \( a \) be the set of prices and \( \varsigma \) be a variable.

a. Step 1: Initialize first to the power of \( h \).
b. Step 2: Optimal value \( \mu \) computes by using the price of the set \( a \) and the queue length value \( l \).
c. Step 3: The sum of optimal value for each subset from the entire set is calculated and the each subset upper bound value is minimized.
d. Step 4: To evaluate the upper bound value to increment the price value.
e. Step 5: In the steps 3, 4 and 5, repeat till the optimal value to be obtained.
f. Step 6: To return the boundary value.

Evaluation of the Delay: The algorithm estimates the delay analyzes of the wireless network even if it is operated with a stationary randomized scheduler and then the service process can be explored as follows. The scheduler is unaware of the backlog and selects to schedule link independent of the queue is blank or not. In each slot check whether the link is scheduled, exactly one packet is served, otherwise the packets present in the queue wait for the next available slot. The following parameters should be identified.

a. Length of the queue process at given time slot.
b. Number of arrivals at link during the time slot.
c. Random variable that is 1 if link is scheduled and is 0 otherwise.
d. In the system has been evaluated the average delay.
Finally, the Max-Weight algorithm determines the maximum sum weight among all matching's, so that it has known as Max-Weight scheduling.

**Algorithm:**
For each time slot
\[ S \leftarrow \emptyset \]
For each layer from top to bottom
While (true)
\[ n \leftarrow \text{maximum weighted schedulable node} \]
if \( n \) is NULL then break
Add \( n \) to \( S \)
End while
End for
Transmit packets from all nodes in \( S \)
End for

### 2.2 Backpressure Algorithm

In Backpressure routing algorithm, it is dynamically utilized to perform routing traffic over a multi-hop network by using congestion gradients. Backpressure scheduling technique is chosen the “Most excellent solution set” for non-interfering links at every slot transmission. A differential backlog routing and a max-weight link selection stages are the two stages of backpressure algorithm shown in Figure 1.

![Figure 1. Information exchange and transmission scheduling in the backpressure algorithm](image)

The following actions are performed for scheduling and routing decisions in every time slot \( t \).

**Resource allocation:** For each link \((n, m)\) allocate a temporary weight in proportion to the differential backlog of every commodity (destination) in the network:
\[
w_{nm}(t) = \max_{d \in D} \left( Q_n^d - Q_m^d, 0 \right) \]

After that, term the maximum difference of queue backlogs according to:
\[
\frac{\sum_{(n, m) \in E} \mu_{nm} w_{nm}(t)}{t} = \mu(t) \quad \text{arg max} \quad \sum_{(n, m) \in E} \mu_{nm} w_{nm}(t) \]

Let \( [t] \) be the commodity with maximum backpressure for link \((n,m)\) in time slot \( t \).

**Scheduling:** The network controller selects the control action that resolves the following optimization issue:
\[
\mu^*(t) = \arg \max_{\mu \in \Gamma} \sum_{(n, m) \in E} \mu_{nm} w_{nm}(t) \quad \text{where}, \quad \text{specifies the group of all schedules subject to} \]

1. for \( t=0,1,2,\ldots \) do
2. observe local queue lengths \( \{q_k^i(t)\}_{k} \) for all flows \( k \)
3. for all neighbors \( j \in n_i \) do
4. send queue lengths \( \{q_k^i(t)\}_{k} \) – receive \( \{q_k^j(t)\}_{k} \)
5. determine flow with largest pressure:
6. \( k^*_{ij} = \arg \max_k \left[ q_k^i(t) - q_k^j(t) \right] + \]
7. Set routing variables to \( r_{ij}^k(t) = 0 \) for all \( k \neq k^*_{ij} \) and
8. \( r_{ij}^{k^*_{ij}}(t) = c_{ij} \Pi \{ q_k^i(t) - q_k^j(t) \} > 0 \)
9. Transmit \( r_{ij}^{k^*_{ij}}(t) \) packets for flow \( k^*_{ij} \)
10. End
12. End

The backpressure algorithm is used as throughput-optimal and sending to congested nodes, utilizing the entire possible ways between destination and source. To solve unnecessary delay even if the traffic loads is light. Moreover, using longer paths in cases of light or moderate traffic wastes network resources (node energy).

### 2.3 Ant Colony Optimization

Ant Colony Optimization (ACO) is defined as a common search method for solving the solution of difficult combinatorial issues that has inspired by the pheromone trail laying behavior of ant colonies to be effectively per-
A Novel Scheduling Approach for Packet Delay in Wireless Ad Hoc Networks and Comparative Analysis with Other Scheduling Approaches

The basic principle of the ACO algorithm works as follows: initially, a finite set \( \mathcal{C} \) of solution components has to be derived to construct solutions to the Combinatorial Optimization (CO) problem. Subsequently, the pheromone model is a set of pheromone values \( \mathcal{T} \), has to be defined. The probabilistic model is parameterized using this set of values. The pheromone values \( \tau_i \in \mathcal{T} \) are correlated with solution components. ACO algorithm rectifies the optimization problems in two steps as follows:

- Candidate solutions are mainly used to bias the sampling by modifying the pheromone value to obtain first-rate solutions.
- Candidate solutions are constructed using probability distribution from a pheromone model.

Mainly, the ACO meta-heuristic performs using three main phases\(^2\): Update Pheromone Phase, Solution Construction Phase, and Daemon Actions. Every phase briefly described as follows: Solution Construction Phase controls the movement of the ants to one adjacent state permitted in their vicinity. Formerly the ants have completed their solutions; they compute the quality of their solution that will be used in the Pheromone Update Phase. Pheromone update phase, the pheromone trails are adjusted based on the latest experience of the colony. The update phase consists of decreasing and increasing the pheromone intensity on the trails. Daemon actions phase - is an optional phase where there is extra enhancement to the original solutions or a centralized action is implemented, which cannot be done by a single ant. Given below explain the ACO algorithm.

**Algorithm:**

**Input:** An instance \( p \) of a CO problem model \( p = (S, f, \Omega) \)

1. Initialize Pheromone Values \( (T) \)
2. \( S_{b_0} \leftarrow \text{null} \)
3. While termination conditions not met do
   1. \( S \leftarrow \text{local search} (S) \) \{optional\}
   2. If \( S \) is a valid solution then
      1. \( S \leftarrow S \cup \{ S \} \)
      2. If \( f(S) < f(S_{b_0}) \) or \( S_{b_0} = \text{null} \) then \( S_{b_0} \leftarrow S \)

4. End while

**Output:** the best so far solution \( S_{b_0} \)

**2.4 Greedy Algorithm**

In Greedy algorithm, it has a set of algorithms which have one general characteristic, making the best option locally at every step without considering any future plans. Therefore, a set of options to select the current best option is provided by the essence of greedy algorithm. Due to the myopic nature of greedy algorithm, it is (as expected) not correct for many issues\(^3\). On the other hand, some certain problems know how to easily be solved by using greedy algorithm like Interval Coloring, Knapsack Problem, Huffman Code and Interval Scheduling (Activity Selection) Problem.

In greedy technique, a node formulates a result to the transmission pathway on the position of its neighbors; the sender node recognizes the spot of the receiver node of its neighbors and then it transmits the data to neighbor check whether if it is closest to the destination node. The process is repeated until the whole packet arrives at the deliberate destination. During the closeness of the idea is similar to several metrics in that area and the popular metrics are provided the projected line and the Euclidean distance connects the relay node and the destination node\(^4\). Intended for the retransmissions process, the untrustworthy neighbors are not taken into action. In the multi hop network area, the main use of greedy policy is consecutively admitting links to a schedule in a greedy fashion\(^5\). In this algorithm empirically perform in addition to an optimal scheduling policy.

Additionally, the greedy partitioning algorithm makes use of dividing the entire wireless network into many single - queue systems and obtain bound on expected delay performance\(^6\). Throughput maximization on multi hop can be mathematically states as below.

\[
\text{Maximize} \sum_{j=1}^{T} f_{C_j}
\]

Let \( T \) and \( C \) be the number of tasks and cores in the system. A greedy algorithm intended for the difficulty of run-time multi hops scheduling. This algorithm is com-
posed of following sequential steps performed by the greedy scheduler before every scheduling epoch.

- Assume that $C_i = 0 \forall i \in T$.
- All tasks in $T$ sort arrange in ascending order by using comparator $[I_{C_i} + 1 - I_{C_i}]$ and store in a queue.
- Virtually assign a core to Task $j$ in front of the queue and update the corresponding using performance prediction models from $\text{14}$.
- Reposition the Task $I_{C_j}$ according to the updated in the sorted queue using a binary search insertion.
- Repeat Step 3 and Step 4 till all cores are allocated.
- Read just real core allocations from last scheduling epoch to reflect the new optimal core allocations.
- Execute tasks with the optimal core allocations.

The greedily searching algorithm proceeds as below for a $(K, X)$ bottleneck which yields the maximum lower bound for computing operation in the multi-hop networks. The flows transferring through the selected bottleneck are removed and that process is repeated until the entire flows are utilized. Therefore, the wireless network decomposes into numerous single queue systems $\text{18}$. The system average delay can be effortlessly computed. Remind that the decomposition process obtained by the greedy algorithm but it is not the optimal decomposition. Finally, the optimal decomposition can be alternately obtained by using a dynamic programming technique with the cost of increased computation complexity $\text{19}$. The negative aspect of the greedy algorithms is the transmission may possibly be unsuccessful in that case if the current holder of the message has no neighbors closer to the receiver apart from itself. Even if there is a possible way between the two extremes, for example, when an obstacle is coming into existence.

### 2.5 Greedy Max-weight Scheduling (GMWS)

The combination of greedy techniques with Ma-Weight Scheduling, it accomplishes more throughputs at low complexity. The GMWS algorithm is naturally low-complexity different to MWS. The scheduling difficulty of multi-hop wireless networks find out set of links such that

$$r = \arg \max_{r \in \mathcal{C}_p(r)} \sum_{l=1}^{L} r_i q_l$$

whereas, $q_i$ is the implicit price of link $l$ and $r_l$ is the rate of the link $l$. This scheduling problem can be interpreted to the matching issue in node-exclusive interference model. We note that the weight of link $l$ is $w_l = q_i c_i$. Then rewriting above equation as

$$x = \arg \max_{r \in \mathcal{C}_p(x)} \sum_{i=1}^{L} r_i w_i$$

Let be $\mathcal{C}$ contains a feasible set of matches performing under node-exclusive interference model. Greedy Max-weight can be viewed an approximation to Max-weight scheduling. A well-known sub-optimal algorithm is GMWS $\text{20}$ and it is also known as Longest Queue First (LQF) $\text{21}$. The operation of GMWS is moderately natural; in this scheduling the opportunity to transmit is specified to a link with higher weight prior to the links with lower weight (i.e. shorter queue length).

In this model, a wireless network performs by using a graph $G = (V, E)$, where a group of nodes denote as $V$ and $E$ the set of links. Nodes representing wireless transmitters and receivers, and a link between two nodes directly communicate with each other. For any link, the set of its interfering links define as

$I(l) = \{ l' \in E | l' \text{ interferes with } l \}$

Assume that if $l'$ interferes with $l$, after that $l$ also interferes with $l'$.

A schedule of graph symbolizes $G = (V, E)$, a subset of links $(M \subseteq E)$ are simultaneously activated/ scheduled and it is no two links $M$ interfere with each other using the interference constraint. For instance, all links have unit capacity; a scheduled link sends one packet in one time slot. A schedule performs to be maximal even if no link can be added to it without violate the interference constraint $\text{22}$. The capacity region of the network contains the set of arrival rates for performing with pre-exist scheduling algorithm that stabilize the queues, whereas the queues are bounded in some deterministic sense depending on the arrival model utilized $\text{23}$. The GMWS algorithms

---

**Algorithm: Computing the Lower Bound**

1: $Z \leftarrow \{1, 2 \ldots N\}$
2: $\text{BOUND} \leftarrow 0$
3: repeat
4: Find the $(K, X)$-bottleneck which maximizes $E[DX]$
5: $\text{BOUND} \leftarrow \text{BOUND} + E[DX]$
6: $Z \leftarrow Z \setminus i : i \in X$
7: until $Z = \Phi$
8: return $\text{BOUND}$
controls as follows: First select the link \( l \) with the largest backlog; add \( l \) into the schedule, and disable other links in \( I(l) \); subsequently, choose the link \( l' \) with the largest backlog from the remaining links, add \( l' \) into the schedule, and disable other links \( I(l') \); this process continues until the entire links are either disabled or selected. The entire chosen links \( \{l, l', ...\} \) is scheduled during time slot.

**Algorithm:**

\[
\text{GMS}(E) := \emptyset \\
E' := E \\
\text{WHILE} (E' \neq \emptyset)
\]

Pick a globally heaviest link \( l \in \arg \max_{l' \in E'} w(l') \)

\[
\text{GMS}(E) := \text{GMS}(E) \cup \{l\} \\
E' := E \setminus \{l\} \setminus I\{l\}
\]

ENDWHILE

This scheduling algorithm achieves the throughput or throughput optimal solution only if the network remains stable condition for the complete arrival rates in the capacity area if not it is unsuccessful\(^{[24]}\).

So, another efficient scheduling approach is required to attain minimizing the end-to-end delay, packet delivery ratio and maximum throughput optimality, and also minimizes the routing overheads in wireless ad-hoc networks.

### 3. Proposed Work

#### 3.1 The Proposed ACO-MWS Algorithm

The Max-Weight type algorithms stabilize the system and to compare better performance buffer-usage than the other algorithm, in multi-hop networks. The Max-Weight algorithm schedules a set of links that uses the total weight (max-weight independent set), and assigns a weight of queue-length \( X \) channel-rate. A rate vector \( \lambda \) interior is specified to randomized, a stationary, the capacity region, queue-independent policy is designed to stabilize the system, even though the channel state probabilities and traffic rates were analyzed. Each slot \( t \), current channel state probabilities and queue backlogs \( S(t) \) and \( Q(t) \) for each link \( i \), and select to serve the link \( i^*(t) \in \{1, \ldots, N\} \) with the largest \( Q(t) S_i(t) \) product. This is also called the Longest Connected Queue policy (LCQ), as it serves the queue with the largest backlog between all that are currently ON. In Max-Weight policy, the scheduling vector \( S(t) \) belongs to the set:

\[
S(t) \in \arg \max_{S} \left\{ \sum_{I_{S} \in S} Q_s(t) \mu_i(t) \right\}.
\]

In support of arrival rate vectors tightly in the interior the stability region of the system that assure several additional constraints, if the system scale is large sufficient, the algorithm remains the system stable in multi-hop networks. MWS requires statistical knowledge to execute in the following the proposed work that aims to combine Max-Weight with ACO algorithms that decide how much data is to be injected into the network. The aim of this research is to maximize the total utility even though in traffic injected into the network, and obtains higher throughput optimality then minimizes the delay performance in multi-hop heavy tailed networks.

The proposed ACOMWS has achieves the above objectives in successful manner. This research work proofs all these things in our real time experimental results.

In ACO algorithm has been efficiently applied to numerous optimization combinatorial problems. It uses the pheromone mechanism in ad hoc networks. In ACO approach, the pheromone mechanism of evaporation updates with the entire nodes and that can rapidly adapt to the dynamic modifications. It is responsible for maintaining local traffic statistics and routing table in ad hoc network. Local traffic statistics explains for traffic distribution over the network and it has a simple parametric statistical model by node \( k \). It maintains track of the amount of traffic flows towards all probable destinations For each source node \( n \) and for each possible destination \( d \) symbolizes in routing table, a probability value \( P_{nd} \) that communicates the selecting \( n \) as the next node. The amount of pheromone deposits on the link \( (k, n) \), a destination node \( d \) heads toward by an ant at node \( k \), it chooses the neighbor node \( n \) with the probability \( P_{nd} \) equation as follows:

\[
P_{nd} = \frac{P_{nd} + a \times l_n}{1 + a \times (|N_k| - 1)} \text{ where } l_n = 1 - \frac{q_n}{\sum_{n' \in \mathbb{N}_k} q_n'} (7)
\]

where, \( |N_k| \) controls the number of the neighbors of node \( k \), \( q_n \) is the length of the queue related with the link connecting \( k \) to \( n \) and \( a \) is the weight of the heuristic function to the pheromone deposit. It estimates the honesty of the path before that if an ant reaches the destination node. The honesty of the path can be defined according to an application’s requirement.
Algorithm:
Step 1:
The wireless graph is designed by the network topology model, \((V, E)\), where, \(V\) is a network node and \(E\) is the link between two nodes. Simultaneously, there are considered as \((t)\) ants. The number of ants in the network is; \((t)\) is the probability of selecting link \(E_{ij}\) for ant \(k\) at time \(t\).

\[
P_y(t) = \frac{\tau_y(t)^\alpha \eta_y^\beta}{\sum_{j \in \text{allowed}_y} \tau_j(t)^\alpha \eta_j^\beta}, j \in \text{allowed}_y, 0 \text{ else}
\]

where, \(\tau_{ij}(t)\) is the strength of the pheromone in the link \(E_{ij}\); \(\alpha\) is a parameter to evaluate the trajectory of pheromones; \(\eta_{ij}\) is provided between node \(j\) and node \(i\), and that it has defined as \(1/d_{ij}\) (\(d_{ij}\) is the distance between node \(j\) and node \(i\)); \(\beta\) is a parameter that measures visibility; and \(\text{allowed}_k\) is a collection of nodes that have not been visited.

Step 2:
In ad hoc networks, the pheromones update formula on each path as follows:

\[
\tau_y(t+1) = (1-\rho) \tau_y(t) + \Delta \tau_y,
\]

where, \(\rho\) is the pheromone volatilization coefficient, which is a constant between 0 and 1, and \(\Delta \tau_{ij}\) is the increase of the pheromone of ants passing through links \(i\) and \(j\).

\[
\Delta \tau_{ij} = \sum_{k=1}^j \Delta \tau_{ij}^k.
\]

The movement of nodes on the communication path, and the other one is the nodes withdrawn from the network due to energy depletion (reduction). By using two major reasons for path breaking perform in ad hoc networks, therefore to choose reliable links and nodes to be effectively performed.

Step 3:
After that, the Path Stability (PS) factor is to judge the stability of the path. The max-weight policy is the most important because of its simplicity and its general stability properties.

Step 4:
**Stability Region:** In the stability region, an arrival rate vector \(\lambda = (\lambda_1, \ldots, \lambda_F)\) process of the multi-hop switched queuing network explained above if there pre-existing \(\zeta_{f,i,j} \geq 0, f \in F, i, j \in N\) such that the following set of constraints is satisfied:

- **Flow efficiency constraints**
  \(\zeta_{f,i,j} = \zeta_{f,i,j} = \zeta_{f,j,i} = 0, \forall i \in N, \forall f \in F\);

- **Routing constraints**
  \(\zeta_{f,i,j} = 0, \forall (i,j) \in L_f, \forall f \in F\);

- **Flow conservation constraints**
  \(\sum_{f \in F} \zeta_{f,i,j} + \lambda_{f,i,j} \cdot \mathbb{1}_{i=j} = \sum_{f \in F} \zeta_{f,i,j}, \forall i \neq d_f, \forall f \in F\);

- **Link capacity constraints**
  \(\sum_{f \in F} \zeta_{f,i,j} < 1, \forall (i,j) \in L\).

An arrival rate vector process in the stability region and subsequently pre-exists a scheduling policy in the sense of stability that become constant the network. The stability region depends on the routing constraints, the link capacities and the network topology; however, it does not on higher order statistics of the arriving traffic.

Step 5:
In single-hop networks, the two metrics are equivalent progress. A traffic flow is delay stable if and only if the queue buffering the traffic of that flow is delay stable. However, in multi-hop networks the situation could be more difficult.

Step 6:
For instance, a traffic flow can be delay unstable at the same time as some queues of that flow are delay stable.

**Lemma 1:** The multi-hop switched queuing network described above under a stabilizing policy. If queue \((f, i)\) is delay stable, for all \(i\), then traffic flow \(f\) is delay stable.

**Lemma 2:** Let \(f\) be a traffic flow with fixed routing. If queue \((f, i)\) is delay unstable, for some \(i \in N_f\), then traffic flow \(f\) is delay unstable.

**4. Experimental Results and Discussions**

The Experimental results present the performance measurements over the topologies shown in Figure 2 for the six algorithms (MWS, BP, ACO, Greedy, GMWS, and proposed ACOMWS). The real time Simulations are run...
A Novel Scheduling Approach for Packet Delay in Wireless Ad Hoc Networks and Comparative Analysis with Other Scheduling Approaches

in NS-2 with results averaged over $10^5$ time slots. The topology has five source-destination pairs, same Poisson arrival rates and $\mu_M = 2$. The required minimum data rates for the all flows are all set to 0.1. Each link can transmit one packet during a time slot. The topologies are intentionally kept simple, so that we can have a understanding on the minimum delay, however they need routing along multiple paths to attain the network capacity and induce implicit routing loops.

Figure 2. Network topology.

The end-to-end delay as the metric in the simulations since minimizing delay is used for several real-time applications. As a result, the end-to-end delays performances of six algorithms were compared. The delay performance verifies by either the busy time or the heavier tail. And also this work compares the throughput, packet delivery, ratio overhead, and queue length, between these six algorithms.

4.1 End-to-End Delay

End-to-end delay is measured with regard to varying payload. In the below graph using six policies different variations on different time, when in the initial time the six policies are performing equally. After increasing the time, the delays of the policies are varying simultaneously. In the graph the GMWS have low amount of delay trade-off than the other policies, but compare GMWS with ACOMWS policy, ACOMWS is showing low amount of end-to-end delay is illustrating in Figure 3.

Figure 3. Comparison of end-end delay.

4.2 Average Queue Length

Run each algorithm for arrival rate vector for 10 random seeds and then take their average. The evolution of the queue lengths of nodes versus time is given in Figure 4. It can be seen that for the desired arrival rate vector, the queue sizes with BP and MWS rapidly increase as time evolves. On the contrary, the figure seems to suggest that GMWS and ACOMWS have the potential of stabilizing the queues. Especially the ACOMWS decrease than GMWS. It is worth mentioning that node 0, without the relaying nodes, can at most support an arrival rate of 0.4 packets per time slot. However, by enabling relaying from other nodes, it can support an arrival rate of 0.6 packets per time slot.

Figure 4. Comparisons of average queue length.
4.3 Throughput
The metric of throughput as the average amount of distributed data per time slot is normalized by the link rate. It is one of the dimensional parameters of the network which gives the fraction of the channel capacity used for useful transmission selects a destination at the beginning of the simulation i.e., information whether or not data packets correctly delivered to the destinations. Initially when the network simulation started the throughput is maximum as all data is transmitted properly. Little increase in amount of transmission of packets throughput remains same. But as the packet loss and packet delay started due to network congestion the throughput start decreasing gradually. As the amount of packet send is not received properly at the source. Thus the throughput is depended upon the packet loss, packet end-to-end delay, congestion etc. But we use the proposed approach it increases the throughput even if the difficult situation compares than MWS, BP, ACO, Greedy, GMWS. Figure 5 is shown that ACOMWS algorithm is designed to maximize throughput over all possible routing and scheduling strategies.

4.4 Packet Delivery Ratio
As the results the simulation are performed on long time simulation up to 3 hours to six algorithms shown in Figure 6. In that figure simulates the MWS, BP, ACO, GREEDY, GMWS and ACOMWS algorithms. The results the long simulation has a good delivery ratio. Compare to traditional algorithms (MWS, BP) the combined algorithms have better packet delivery ratio, because they run faster than others and have a better energy level. Depending upon that the GMWS contain better result illustrate in below graph. Although, finally the figure concludes that the performance of ACOMWS excellent results compare to other policies including GMWS.

4.5 Overhead
The routing overheads computed under varying number of nodes and varying pause times, the routing overhead
A Novel Scheduling Approach for Packet Delay in Wireless Ad Hoc Networks and Comparative Analysis with Other Scheduling Approaches

for ACOMWS was comparing with other MWS, BP ACO, Greedy and GMWS algorithms are shown in Figure 7. The routing overhead for ACOMWS is lower than other algorithms. The routing overhead is low for less number of nodes. Its value is approximate equal with less number of nodes. When the overhead increased with number of nodes also concurrently increasing in the value of routing overhead is more but at the moment also low overhead for our proposed algorithm than the other algorithms.

5. Conclusion

As a result, our proposed algorithm ACOMWS improves the throughput optimality, packet delivery ratio and minimizing the end-to-end delay in multi-hop networks and also reduces the routing overheads. In this study, greedy, Ant Colony Optimization (ACO), Max-Weight Scheduling (MWS), and Greedy Max-Weight Algorithms (GMWS) are effectively compared through the real time simulation environment that demonstrates the efficiency of ACOMWS. ACOMWS satisfies a tradeoff between the throughputs and minimize end-to-end delay bound while assuring the high data rate requirements for delay stability of traffic flows. Thus ACOMWS provides better performance as compare to other algorithms.

6. References