PERFORMANCE COMPARISON OF ROUTING ALGORITHMS IN PACKET SWITCHED NETWORKS

B.P. Mohanty, C.G. Cassandras
and D. Towsley
University of Massachusetts
Amherst, MA 01003

COINS Technical Report 89-100

PERFORMANCE COMPARISON OF ROUTING ALGORITHMS IN PACKET SWITCHED NETWORKS*

B. P. Mohanty, C. G. Cassandras

Department of Electrical and Computer Engineering
University of Massachusetts
Amherst, MA 01003

D. Towsley

Department of Computer and Information Science
University of Massachusetts
Amherst, MA 01003

ABSTRACT

This paper presents a comparative performance study of three representative routing algorithms for packet switched networks: (1) the CODEX algorithm, (2) Gallager's distributed algorithm, and (3) the well-known ARPANET algorithm. The essential characteristics and critical issues in these algorithms are identified through experimentation under a variety of traffic and network operating conditions. The sensitivity of the mean delay performance of the algorithms to operating parameters such as the routing update period and the traffic adjustment parameters are examined. The comparison of mean delay behavior indicates that under stationary traffic conditions, gradient-based algorithms, such as Gallager's algorithm, outperform the other algorithms, whereas the CODEX algorithm, which is a shortest-path-based virtual circuit routing algorithm does better in quasistatic traffic conditions. The ARPANET algorithm performs poorly compared to the above two algorithms under moderate to heavy loading conditions. Some of the factors which could be contributing to the poor performance of the ARPANET algorithm are examined.

* This work is supported in part by the Rome Air Development Center under contract F 30602-88-D-0027 and by the Office of Naval Research under contract N 00014-87-K-0304.

1. INTRODUCTION

Routing is one of the principal functions of the Network Layer in the OSI reference model, and deals primarily with determining the path (i.e. a sequence of links) that a packet originating at one node of a network takes in order to reach its destination. Most often the paths are chosen so as to minimize the mean network delay. Various routing algorithms which perform the above function can be classified according to the packet switching scheme (*Datagram* (DG) vs. *Virtual Circuits* (VC)), the method of information dissemination (*Centralized* vs. *Distributed* algorithms) and the route selection mechanism (*shortest path* vs. *gradient-based* algorithms).

Most of these algorithms are too complex to be studied analytically under realistic operating conditions. The few theoretical studies available in the literature make simplifying assumptions such as stationarity of traffic inputs, convexity of network wide cost functions e.g. [3],[11] and M/M/1 models of individual links. In practice, one may have to resort to experimental methods to obtain insight into the operations of these algorithms. More important, experimenting may be the only means for determining suitable algorithmic parameters such as the length of the *routing update period* and the values of the *traffic adjustment parameters* if any. In this paper we examine and compare the mean delay performances of three representative routing algorithms, namely the CODEX algorithm [1],[2], Gallager's minimum delay algorithm [3] and the ARPANET algorithm [4], through simulations with a wide range of traffic and parametric variations. The choice of these algorithms is largely motivated by their diverse characteristics and operating conditions, as will be further discussed below. These algorithms are different in their packet switching and/or route selection mechanisms.

Of the three algorithms chosen for the performance comparison reported here, the ARPANET algorithm is datagram type. The same is true for Gallager's algorithm, although the route selection mechanism is different. On the other hand, the CODEX algorithm employs virtual circuits, except that control packets are still treated as datagram-based traffic. Although the method of information dissemination is of great importance for a routing algorithm, for the purposes of our study, it is assumed that an appropriate mechanism is in place which provides all nodes with all information required to implement routing decisions.

The ARPANET algorithm employs a shortest path algorithm to determine the minimum delay path from a source node to the destination node. On the other hand, Gallager's algorithm is a *gradient-based* algorithm which attempts to optimize a network-wide performance measure by adjusting routing probabilities for all outgoing links at each node. The CODEX algorithm is based on shortest paths where the "length" assigned to a link is an incremental function of link capacity and traffic rate. Shortest path algorithms are generally conceptually simple; however they are heuristic, and do not guarantee optimality of performance. Moreover, they are known to be highly susceptible to instabilities manifested in the form of oscillatory behavior in the observed mean delay [5],[6]. On the other hand, gradient-based algorithms are more sophisticated and can achieve optimal network-wide performance under stationary traffic conditions [3]. However, they too are susceptible to instabilities. Moreover, successful implementation of these algorithms relies on the development of efficient methods for estimating the required gradient information. The approach employed here is the direct estimation of the derivatives on line by using Perturbation Analysis (PA) techniques [7],[9].

The paper identifies the critical parameters such as the length of the routing update period in all the algorithms, the values of the rerouting probability in the CODEX algorithm and the step size in Gallager's algorithm. The sensitivity of the mean delay performances to each of these parameters is examined. This study, based solely on experimentation, brings out some of the inherent operational limitations of the algorithms and some of the tradeoffs involved in choosing suitable operational parameters. Some of the merits and demerits of using datagram (Gallager's) or virtual circuit (CODEX) routing algorithms in stationary as well as quasistatic traffic environments are examined. Moreover, the mean delay performances of the CODEX and Gallager's algorithms under suitably tuned parameters are compared with that of the standard ARPANET algorithm. Some of the possible causes of the oscillations in the ARPANET algorithm such as the traffic load are investigated. Finally, in the context of the comparison of datagram and virtual circuit routing algorithms, the effect of resequencing delays on Gallager's algorithm is investigated.

Our simulation-based comparative study indicates that in stationary traffic conditions, while the gradient-based Gallager's algorithm leads (under suitably chosen parameters) to oscillation-free mean delay characteristics at steady state, shortest-path-based algorithms suffer from instability problems. However, the CODEX algorithm, which is a shortest-path-based algorithm, is found to have better adaptivity to bursty traffic environments. This suggests that in an environment expected to feature abrupt traffic changes, a combination of approaches is desirable: an initial setup procedure based on shortest path algorithms, combined with a gradient-based scheme seeking to optimize the steady-state performance of the network. In the case of the ARPANET algorithm, the observed oscillatory behavior is due to lack of any control mechanism to regulate the traffic adjustments at each update instant. On the other hand, the CODEX algorithm displays oscillatory behavior due to its mechanism of rerouting of virtual circuits. These oscillations are considerably reduced under light load conditions. Besides oscillations, the shortest-path-based algorithms result in comparatively higher mean network delay due to their heuristic approaches to routing.

The remainder of this paper is organized as follows. Section 2 describes the basic operations of the three algorithms. This discussion is limited to those aspects which are important to our comparative study. The network configuration and the traffic environments considered, as well as some of the implementation details of the simulations we performed are described in Section 3. Section 4 discusses the results of the experiments for different parametric and operational variations. Finally in Section 5, we present the main conclusions of our experimental study.

2. THE ALGORITHMS

In this section, we briefly describe the basic operation of the three algorithms to be compared.

2.1. CODEX Algorithm

The CODEX routing algorithm actually consists of two routing algorithms executing concurrently, one of which performs datagram routing (DG) for network control traffic while the other performs virtual circuit routing (VC) for network user traffic. Their operation relies on the existence of a broadcast-type algorithm [1], where every node broadcasts all relevant information, i.e. the estimates of flow rates on each of its outgoing links, to every other node along a minimum spanning tree rooted at itself.

The datagram routing algorithm employs a shortest path algorithm to find the best routes between all the node pairs. Each node obtains estimates of the average flow rates on all its outgoing links over an *observation interval* which are used to compute the mean link delays assuming M/M/1 link models. Routing updates are done at the end of such an observation interval. The above mean delays on links are input as link *lengths* in computing the shortest paths.

Virtual circuit routing is again based on a shortest path algorithm; however, the link length assignment process is different. The algorithm allows different classes of traffic to be routed differently by assigning to them priority numbers which are incorporated in the calculation of link lengths. Flow rates on links are estimated separately for virtual circuits with different priority numbers.

There are two aspects to the VC routing algorithm: establishment of a VC upon initiation of a session, and periodic rerouting of VC's to account for changes in the network operating conditions. These two aspects are considered below.

***Routing Policies for VC Setup*:** The *cost* of a link (i,j), denoted by D_{ij}, is defined to be a weighted sum of mean delays experienced by packets of different priorities traversing the link. As in the case of datagram routing, an M/M/1 queueing model [12] is assumed for all links. D_{ij} is given by

$$D_{ij} = \left(\sum_{k=1}^{M} p_k F_{ij}^k \right) \left\{ \frac{1}{C_{ij} - \sum_{k=1}^{M} F_{ij}^k} + d_{ij}^k \right\}$$

where C_{ij} is the link capacity, F_{ij}^k is the mean flow rate estimated over an observation interval for a VC with priority k, d_{ij} is the propagation delay on the link, and p_k is a positive weighting factor representing the priority, with $k=1,\ldots,M$.

Suppose a new VC with estimated traffic rate Δ and priority k is to be set up. The length of link (i,j) is the estimated incremental link cost given by

$$l_{ij} = D_{ij}(F_{ij}^1, \ \ldots\ldots, \ F_{ij}^k + \Delta, \ \ldots, \ F_{ij}^M) - D_{ij}(F_{ij}^1, \ \ldots\ldots, \ F_{ij}^M)$$

The first term in the above expression corresponds to the projected link cost (as a result of the new VC) and the second term to the current cost of the link. Shortest paths are computed from the source to the destination using the above link lengths. The computation is done by Pape's version of the Bellman-Moore algorithm [9].

***Rerouting Policies for Active VC's*:** At the end of every observation interval, a fraction of the active VC's is chosen for rerouting. The selection is done randomly, by simply assigning each VC a certain probability for rerouting; we shall refer to this as the *rerouting probability*.

If a VC is selected for rerouting, it is viewed as a new VC. First, link costs are computed for all links based on the traffic conditions that would result if this VC were removed; then, link lengths are evaluated by treating the VC as a newly arriving one. Using these lengths, a new shortest path is computed for the VC. If the resulting cost is lower than the current one, then the VC is rerouted along the new shortest path. Details of this implementation can be found in [2].

Based on the discussion above, it is clear that there are two critical aspects which affect this algorithm's performance:

• the choice of the length of the observation interval on which routing updates are based: too short an interval results in poor traffic rate estimates, whereas too long an interval delays possible performance improvements

• the choice of rerouting probabilities for VC's: too small a probability limits adaptivity, while too large a probability may cause overloading of some links and lead to oscillations in performance

There is generally no well-defined way of choosing these parameters and one must resort to trial and error through experimentation.

2.2. Gallager's Distributed Algorithm

Gallager's distributed routing algorithm [3] is designed for datagram networks. The critical steps involved in this algorithm may be summarized as follows. Over an *observation interval*, each node estimates the traffic rate on each outgoing link as well as the sensitivity (*gradient*) of the mean packet delay on each outgoing link with respect to the traffic rate through that link - this is also referred to as the link *marginal delay*. At the end of an observation interval, each node identifies the "best" outgoing link, i.e. the link with the smallest marginal delay (gradient estimate). The node decreases the traffic rate into all other outgoing links and adds the total traffic rate removed from these links into the "best" link where the amount of traffic rate adjustments is regulated by a *step size* parameter. The process continues until equilibrium is reached, i.e. all outgoing links have the same marginal delays.

In theory, Gallager's algorithm achieves the minimum possible network-wide mean packet delay under stationary traffic conditions [3], provided step sizes are chosen sufficiently small. In practice, however, there are three critical aspects which determine the algorithm's performance:

• the choice of observation interval length: too short an interval results in poor estimates, whereas too long an interval delays improvements

• the choice of step size: too small a value results in slow convergence, too large a value may lead to instabilities

• the method of gradient estimation used.

Clearly, the first two choices are the analogs of the observation interval and rerouting probabilities required in the CODEX algorithm. As for gradient estimation in our simulations, we employ the PA technique used in [10].

2.3. ARPANET Algorithm

A detailed description of the current ARPANET algorithm can be found in [4]. Here, we shall discuss only the features relevant to our performance study. The main distinction between the datagram routing performed by the ARPANET algorithm and that used in Gallager's algorithm lies in the number of paths available between any source-destination pair. Whereas in Gallager's algorithm a packet between a node pair may take any of several available paths depending on the routing assignments at each intermediate node to the outgoing links, in the ARPANET algorithm it is constrained to the single best path available between this pair of nodes. This is sometimes referred to as *single path* datagram routing.

The path selection is done by employing a shortest path algorithm with "lengths" assigned to links. Each node estimates the mean packet delay on each outgoing link over an *observation interval* . If the current estimate of mean delay differs from the previous update by more than a certain *threshold*, then it is passed on to other nodes in the network. At the end of an observation interval, each node uses a shortest path algorithm to find the minimum delay path between itself and all other nodes.

It is worthwhile to note at this point that the basic philosophy of operation in the ARPANET algorithm makes it prone to oscillations in the observed performance. The policy of diverting *all* the flows to a currently less congested region of the network without any regulatory mechanism (like the step size in Gallager's algorithm and rerouting probabilities in the CODEX algorithm) may cause this area to be heavily congested at the next update time. Conversely, the region which is currently congested becomes lightly loaded at the next update, thus making it attractive for routing. This shifting back and forth of traffic flows gives rise to oscillations which may be damped by adding a *bias factor* to the link length.

The main parameter here is the observation interval, which is set to 10 seconds in the currently operating version of the algorithm. In addition, a threshold of 64 msec is used to trigger reporting a new mean delay estimate. This threshold decreases at the rate of 12.8 msec with every observation interval (10 sec). Thus, one update is guaranteed every 6 observation intervals. After every update reporting, the threshold is set back to 64 msec. This mechanism ensures reporting small changes slowly and large changes more rapidly.

3. EXPERIMENTAL SETUP

In this section the experimental setup used to carry out the simulations of the chosen routing algorithms is described. Most of the experiments were conducted under similar network conditions as those presented in [10], where Gallager's distributed routing algorithm employing PA techniques was considered. This was done to enable us to compare results in that work with some of our results here.

3.1. Network Configuration and Traffic Characteristics

A 6-node network, similar to that used in [10], was used for our experimentation. This is shown in Figure 1, where link capacities are shown in Kbps. We consider a traffic environment with 19 traffic-generating source-destination pairs. The packet interarrival time distributions are arbitrarily chosen from a set of exponential and uniform distributions. A list of these distributions is presented in Table 1, where EX(α) denotes an exponential distribution with mean α msec and UN(α,β) denotes a uniform distribution over the interval [α,β] msec. The same traffic interarrival distributions were used in the performance study reported in [10] as well. When Gallager's algorithm is employed in this network with this traffic load, the highest link utilization at steady state is 0.89.

In the case of datagram algorithms, each of the packets is treated independently. But from a virtual circuit routing point of view, each of the 19 traffic generating source-destination pairs may have one or more (10 or 20 in some experiments) virtual circuits active between them.

Of all the packets generated, 80% are long (1 Kbit) and 20% are short (128 bits). Long packets may be viewed as user data packets, while short ones represent system or control traffic. This distinction does not affect the handling of packets in the datagram routing setting (ARPANET and Gallager's Algorithms). However, as already described in section 2.1, the CODEX algorithm routes short packets on a datagram basis and long packets through virtual circuits.

3.2. Traffic Generation

Several traffic generation processes are considered in our experiments. The first is a renewal process where all traffic streams are switched ON at t = 0, and remain active for the entire simulation run length. This type of traffic generation has been used in performance studies such as [10]. However, it poses certain practical problems as described below:

1. If the initial routing assignments are not properly chosen, some of the links may be assigned packet flows beyond their capacities, thus making them unstable. Under these conditions, most of the theory based on modeling links as stable queueing systems becomes invalid, and the behavior of an algorithm may be unpredictable. This issue is addressed in [3], where it is assumed that a flow control mechanism is present, which ensures that no link flow ever exceeds the corresponding link capacity.
2. The method of stationary traffic inputs is not suitable for a performance study of the CODEX algorithm. Being a VC-based algorithm, its slow rerouting mechanism is not designed to clear heavy traffic congestions that would arise due to the simultaneous activation of several virtual circuits upon initialization. This congestion buildup may in fact take an extraordinarily long time to be cleared before steady state conditions can be established. Therefore, this would be an unrealistic model of the actual CODEX algorithm operation, and is likely to lead to erroneous conclusions (for example: it may appear that large value of rerouting probabilities improve the adaptivity of the CODEX algorithm; as we shall see in the next section, this may not be always true). In contrast, datagram routing algorithms such as Gallager's assume stationary traffic inputs over reasonably long periods, and may not be able to adapt to very frequent variations in network traffic resulting from switching virtual circuits ON and OFF.

To accommodate the above considerations, we adopted a scheme where VC's are gradually switched ON at time instants offset by random intervals, assumed to be exponentially distributed. Once a VC is ON, it remains active for the entire simulation run. The only transient phenomena in this traffic generation model are the arrivals of new VC's. To introduce more fluctuations into the traffic input process, in some experiments an active VC is turned OFF after generating a geometrically distributed number of packets and reactivated after an exponentially distributed time interval. The other implementation-related point worth mentioning is that, for simplicity, the priorities of all virtual circuits have been assumed to be the same.

4. COMPARATIVE PERFORMANCE RESULTS

In this section, we present the results of our comparative performance study based on the experimental setup described in section 3. The performance comparison of the three algorithms (CODEX, Gallager's, and ARPANET) reported here is on the basis of the mean packet delay, which includes queueing delays at intermediate nodes, as well as transmission and propagation delays on the links traversed.

The experimental results reported in this section are organized as follows. First, we examine the effect of the observation interval on the mean delay for each of the three algorithms. Second, the effects of the traffic adjustment parameters employed in the CODEX and Gallager's algorithms are investigated. The parameters of the CODEX and Gallager's algorithms are tuned as well as possible, and their mean delay performances are compared under these conditions. The mean delay behavior of the standard ARPANET algorithm is studied under similar traffic conditions and compared to that of the CODEX and Gallager's algorithms. Last, we investigate the effect of packet resequencing, manifested in datagram networks as additional end-to-end delay. This allows us to make truly fair comparisons between Gallager's algorithm and the CODEX algorithm.

4.1. Effect of Observation Interval Length on Mean Delay Performance.

As already described in section 2, the operation of all three algorithms requires some feedback information from the network. The CODEX algorithm requires estimates of individual link flows and traffic rates on each of the active virtual circuits. Gallager's algorithm requires estimates of the sensitivities of individual link delays with respect to corresponding flow rates. The ARPANET algorithm measures average mean delays over all links, which are then used in shortest path computations. This estimation is normally done over a period of time which we refer to as the *observation interval.* Routing updates are normally done at the end of such an an observation interval.

Assuming stationary distributions, a sufficiently long observation interval is required to obtain good estimates. In practice, however, the traffic input may not be stationary. Under these conditions, the speed of response of a routing algorithm as traffic conditions change is important, and the algorithm may respond faster if a short observation interval is used. A proper choice of observation interval length must take this factor into consideration.

Figure 2 shows the effect of the observation interval on the mean delay performance of the CODEX algorithm. In this experiment all virtual circuits were initially activated and kept active for the entire simulation run duration. A rerouting probability of 0.45 is considered here, which appears to be large enough to bring the network out of the initial congestion in approximately 20 iterations. The observation interval was then varied from 6 sec, which corresponds to successful departure of approximately 1000 packets over the network, to 56 sec, corresponding to approximately 10000 packet departures. The mean packet delay in msecs observed over an observation interval is plotted over time in secs. The results shown are averaged over 30 independent replications for each value of the observation interval length. The typical 95% confidence intervals observed for times beyond 1500 secs are ±10% for the 6 sec case and ±4% for the 30 sec and 56 sec cases. We can see that the network starts with an extremely large mean delay due to the initial congestion, and eventually clears out. This transient behavior is not of critical importance here, since in practice rarely would a virtual circuit network see this much traffic applied at the same instant.

The behavior of the algorithm after clearing the initial congestion is of importance to our study. It can be seen that a small observation interval length of 6 sec produces a highly oscillatory mean delay profile. This is also manifested in a larger 95% confidence interval as compared to the same for the longer observation interval length values of 30 secs and 56 secs. This is attributed to the large amount of noise in the estimation of the link flows, resulting in erroneous reroutings, as well as the large variance of the mean network delay itself. Clearly, as the observation interval is increased to 30 seconds and then to 56 seconds, the oscillations subside considerably, although the speed of convergence is relatively slow.

The effect of the observation interval on the mean delay performance of the ARPANET algorithm is presented in Figure 3. Contrary to our observations for the CODEX algorithm above, here the oscillations decrease with decreasing observation interval length. This behavior is explained by the fact that the inherent oscillations in the ARPANET are so significant that the effect of any mean delay estimation errors becomes negligible in comparison. The ARPANET algorithm is heuristic in nature and there is no guarantee that it can ever find a fair allocation of traffic over all the links in the network. This problem is exacerbated under moderate to high load conditions; in such circumstances there are always some links which are heavily congested while some others are lightly loaded. The switching of traffic back and forth between these two sets of links gives rise to the oscillations observed. By taking a larger observation interval we only allow the congestion to build up before switching to the other set of links where again the congestion builds up over the interval.

The effect of the observation interval on the performance of Gallager's algorithm was examined in [10]. The observations there were similar to those applicable to the CODEX algorithm, i.e. a longer observation interval produces mean delay results with smaller variance. For the sake of completeness, some results have been reproduced in Figure 4. The observation interval corresponding to 10000 packets/iteration is equivalent to approximately 56 secs. Other observation intervals can similarly be obtained by considering the fact that they would be proportional to the number of packets/iteration. A reasonably small step size of 0.01 E -05 was used to ensure convergence of the algorithm in this case.

4.2. Effect of Traffic Adjustment Parameters on Mean Delay.

The traffic adjustment mechanism appears in the form of VC *rerouting probabilities* in the CODEX algorithm, and as a *step size* in Gallager's algorithm, which scales the amount of outgoing traffic flow at a node transferred from "bad" links to a "good" link (i.e. from links with larger marginal delays to those with smaller marginal delays as explained in section 2.2.). The choice of these parameters must take the stability and adaptivity aspects of the algorithm into account. It may also depend on other factors, such as the network size and loading conditions. Gallager's algorithm was shown [3] to require a "small" value of step size to guarantee convergence; the proper choice in practice has to be obtained through experimentation on the specific network under consideration. Similarly, in the case of the CODEX algorithm, one must resort to such experimental means, since there are no theoretical guidelines for the choice of rerouting probabilities. It may be noted that the ARPANET algorithm has no such traffic adjustment parameter.

The experimental results depicting the effect of VC rerouting probabilities on the mean packet delay in the CODEX algorithm are presented in Figure 5. In this experiment, we activate the virtual circuits gradually, each with an exponentially distributed offset period with mean 4 minutes. Once activated, a virtual circuit remains active for the entire duration of simulation. This was done to avoid the initial congestion from which it may be difficult to recover in the case of a small rerouting probability. The observation interval was chosen to be 56 sec which is a reasonably long period for obtaining good estimates as observed in section 4.1. The results shown are averaged over 30 independent replications for each choice of the rerouting probability. A typical 95% confidence interval for the mean network delay observed beyond 50 iterations in the case of rerouting probability values of 0.15, 0.3 and 0.45 is ± 4% of the point estimate. The algorithm appears to be unstable for the rerouting probability value of 0.6. To maintain clarity in figures we do not plot the confidence intervals.

A small value of rerouting probability such as 0.15 results in slow adaptivity which is evident from the high mean network delays seen in the initial 40 iterations. This results in the inability of the algorithm to clear the initial congestion and may cause a poor performance of the algorithm in quasistatic traffic conditions. The abruptness of transition in the mean delay profile is due to erroneous initial routing of some high traffic rate virtual circuits in one or more of the sample paths. This creates heavy congestion in some of the links which is cleared only when these potential VC's are rerouted. A very large value of rerouting probability results in extremely large oscillations, as seen for the value of 0.6 in Figure 5. These large oscillations are due to the uncoordinated rerouting of too many virtual circuits at the same time which leads to instabilities.

Similar experiments were performed for different values of step sizes for Gallager's algorithm [10]. The results are shown in Figure 6. The same trend was observed as in the case of rerouting probabilities for the CODEX algorithm, i.e. a small step size results in a slow but smooth convergence to the optimal mean delay value, whereas a large value causes rapid convergence at the expense of oscillations at steady state. Moreover, an extremely small value of the step size is considerably slow in bringing the network out of an initial congestion.

4.3. Comparison of Mean Delay Performance of the CODEX and Gallager's Algorithms with Tuned Parameters.

In this section, we compare the transient and steady state mean delay performances of the CODEX and Gallager's algorithms under suitably tuned parameters. The experiments are conducted for two different traffic environments :

1. *Long-run stationary traffic* environment: the traffic generating source-destination pairs are gradually activated with exponentially distributed random offset periods and they stay ON forever. Thus, the traffic is initially not stationary, but becomes stationary after a sufficiently long period of time.

2. *Quasistatic traffic* environment: the traffic generating source-destination pairs are activated and deactivated at random instants of time as was described in section 3.2.

For both algorithms, a reasonably large observation interval of 56 sec was chosen. The rerouting probability and step size were tuned (by trial and error) to obtain the best possible achievable performance in terms of the smoothness of the mean delay convergence profile. Accordingly, a rerouting probability of 0.3 in the CODEX algorithm, and a step size of 0.01 E - 05 in Gallager's algorithm were chosen respectively. The results for both the algorithms are averaged over 30 replications. Typical 95% confidence intervals for the mean network delays observed beyond 50 iterations are ±4% for CODEX and ±2% for Gallager's algorithms respectively.

4.3.1. Long-run Stationary Traffic Environment Results

The results for this case are presented in Figure 7. The mean network delays for the algorithms are plotted against time measured as the number of iterations implemented (each iteration corresponding to a 56 sec observation interval).

It can be seen in Figure 7 that Gallager's algorithm is slower to converge (up to 25 iterations approximately) compared to the CODEX algorithm (up to 15 iterations approximately). This is because the CODEX algorithm is better suited to handle new source-destination sessions by finding the paths of least incremental delays for the new virtual circuits. On the other hand, Gallager's algorithm adapts slowly to the increased traffic due to the small step size typically used.

Once the initial transients have died out and stationary traffic conditions have been established, Gallager's algorithm converges to a steady state mean delay value with a tighter 95% confidence interval as mentioned earlier, while the CODEX algorithm displays an oscillating behavior within a relatively wider interval. Moreover, it is seen that the steady state nominal mean delay is somewhat higher in the case of the CODEX algorithm.

This difference in mean delay at steady state is attributed to the fact that Gallager's algorithm attempts to optimize the mean delay under stationary traffic inputs (which is the case here after all the traffic generating pairs have been activated). In contrast, the CODEX algorithm is heuristic in nature.

The difference in the degree of oscillations for the two algorithms is attributed to the fundamental difference in the method of traffic adjustments. At any link, Gallager's algorithm can shift a fraction of the current traffic on the link taking any desired value between 0% to 100% (depending on the estimated derivatives and the step size); the CODEX algorithm is only allowed to do so in units of virtual circuits. Therefore, the amount of traffic shifted depends on the traffic rates of the virtual circuits being rerouted. If the cumulative traffic rate of the VC's being rerouted is significant, their shifting can give rise to considerable oscillations.

The fact that rerouting of virtual circuits is indeed the reason for the observed oscillatory behavior of the CODEX algorithm is corroborated by the results of the following experiment. Instead of one virtual circuit per source-destination pair, now the same traffic is carried through 10 and 20 virtual circuits respectively. Each VC is set up and routed independently. Thus, given the initial 19 source destination pairs, we have 190 and 380 virtual circuits each with 1/10 and 1/20th traffic rates of the original VC's respectively. Each of these VC's is activated after an exponentially distributed offset interval and remains active for the entire simulation duration. The results are again averaged over 30 independent replications. As expected, the oscillations in the mean delay were considerably reduced (and a tighter 95% confidence interval of ±2% was observed) as shown in Figure 8. The somewhat larger values of mean delays in the case of larger number of virtual circuits are probably due to improper choices of rerouting probabilities which may need to be more than the values used here (0.03 and 0.015 respectively).

4.3.2. Quasistatic Traffic Environment Results

In this case, a traffic generating source-destination pair can be in any one of the two possible states: active (ON) or inactive (OFF). The duration of time spent in the inactive state is an exponentially distributed random variable with a mean of 4 minutes. The duration of time spent in the active state is a random variable determined by the time required to generate a geometrically distributed random number of packets (with a mean of 200). The active period obviously depends on the traffic generation rates of the individual VC's. The mean active periods for all the source destination pairs are either comparable to or shorter than the length of the observation interval. The results are presented in Figure 9A, and provide additional insight on the adaptivity of the algorithms to burstier traffic conditions. All the results are averaged over 30 independent replications.

Even though Gallager's algorithm performed better than the CODEX algorithm under stationary traffic conditions as shown earlier, it is clear from Figure 9A that it has poorer adaptivity to fast-changing traffic conditions. The sharp peaks observed in the mean delay response of Gallager's algorithm correspond to arrivals of new VC's. As was noted in the previous subsection, the CODEX algorithm is suited to adapt better to arriving VC's through its explicit VC setup routing mechanism, whereas Gallager's algorithm tries to adapt to the situation rather slowly. Moreover, incorrect gradient estimates at the end of an observation interval tend to cause instability in Gallager's algorithm which no longer converges to a steady state mean network delay. Figure 9B presents similar results for the case when there are 190 VC's operating. The active period of each of the VC's is the time required to generate a geometrically distributed number of packets with mean 20. Since the VC traffic rates are smaller, the peaks observed in the case of Gallager's algorithm are less sharp; however, the algorithm is unstable because of inaccurate estimates.

Finally, note that the nominal mean delays observed in Figure 9A and 9B are lower than those seen in previous cases. This is simply due to the fact that the overall network load is now lower, since traffic generating source-destination pairs are periodically inactive.

4.4. Mean Delay Performance of the Standard ARPANET Algorithm.

The results of applying the standard ARPANET algorithm as described in Section 2.3 to the network under consideration are reported in Figure 10. It can be seen that ARPANET displays large high-frequency oscillations. Moreover, the value of the mean delay is much higher compared to the CODEX and Gallager's algorithms. The main reasons for this behavior were discussed in earlier sections: the basic operating philosophy of the ARPANET algorithm causes a far from optimal performance and high susceptibility to oscillations. However, we were able to identify a number of factors such as the traffic intensity and nonuniformity of link capacities which may be contributing to the oscillatory performance. The observed effects of these factors are described below.

In reality, the ARPANET algorithm may not be performing as poorly as it appears in Figure 10. One of the reasons may be that it is designed to operate in environments with a light traffic load on the network. To verify the effect of light load on the observed oscillations, the traffic intensities of each of the 19 source-destination pairs in our experiment were reduced by 50% and 20% respectively thus reducing the overall traffic intensity by the same factors as well.

The mean delay behavior of the ARPANET algorithm under these light load conditions is presented in Figure 11. It can be seen that a 50% load reduction results in a drastic improvement in performance, both in terms of nominal mean delay and the extent of oscillations. A further reduction to 20% of the original load does not substantially affect the oscillation characteristics (as expected, however, the mean delay is reduced).

It was also speculated that the nonuniform link capacities in our network (ranging from 9.6 Kbps to 72 Kbps) are contributing to the observed oscillations in the mean delay. In some experiments with uniform link capacities of different denominations such a trend was observed [10].

4.5. Effect of Packet Resequencing on Mean End-to-End Delay in Datagram Networks.

One advantage of using virtual circuits over datagrams is the sequential delivery of packets at the destination. In the case of datagram networks, the packets have to be resequenced at the destination nodes before being passed on to the host. A simple way of accomplishing this is to assign a sequence number to each of the packets for a source-destination pair. At any time, the destination node is expecting a packet with a particular sequence number. Any arriving packet with a sequence number less than this is stored in a buffer. When the expected packet arrives, all the packets forming a sequence are removed from the buffer and passed on to the host. [illegible] "resequencing delays" are the queueing delays experienced by packets stored in this buffer.

For a fair comparison of mean delay performance one must consider the end-to-end delay, which includes the resequencing delay, in the case of datagram routing. It should be noted that even though the CODEX algorithm is a VC-type algorithm, resequencing delays do occur when a VC is rerouted: when a new path is established, there are still some packets propagating on the old path; consequently, the destination node must be able to do resequencing for a brief period of time, until the old path has been completely drained of all the packets. One would, however, expect that this resequencing delay is very small when averaged over all the packets in an observation interval.

Experiments were performed incorporating the resequencing mechanism described above for Gallager's algorithm. The increase in the mean delay at steady state was found to be very small, amounting to approximately 5% of the mean delay without resequencing. This mean end-to-end delay performance of Gallager's algorithm with resequencing is compared with that of the CODEX algorithm in Figure 12. These results are obtained by averaging over 30 replications. It is clear that under stationary traffic conditions Gallager's algorithm outperforms the CODEX algorithm with respect to both mean delay and steady state oscillations. Thus, resequencing delays have little effect on the relative performances of the two algorithms.

5. CONCLUSIONS

The main objective of this experimental performance study was to compare the gradient-based routing algorithms with the shortest-path-based ones. Under stationary traffic conditions, the gradient based algorithms perform better in terms of both mean network delay and steady state oscillations. Among the two shortest path algorithms, the CODEX algorithm produces lower mean delays because it computes shortest paths based on incremental link delays rather than the absolute link delays as in the case of the ARPANET. The steady state oscillations are particularly significant in case of the ARPANET algorithm. However, we have found that these oscillations are considerably reduced under light load conditions. Under these conditions, the simplicity of shortest path algorithms becomes their main attractive feature. This is probably why the ARPANET algorithm is still a favorite in light traffic datagram environments.

Under quasistatic conditions, where traffic is essentially bursty, a gradient-based algorithm may not be able to adapt to the changing conditions as rapidly as a shortest path algorithm. The adaptivity in this case depends on the step size chosen. There is a tradeoff involved in the choice of this parameter. While a larger value provides greater adaptivity, a smaller one ensures stability. Consequently, one has to select an appropriate value depending on the specific requirements, leading to the strong argument for mechanisms to dynamically adjust this step size parameter.

Despite their many attractive features, gradient-based routing algorithms have not yet become popular in practice. One of the reasons has been the lack of simple and efficient gradient estimation techniques not requiring restrictive modelling assumptions. However, with the use of techniques such as Perturbation Analysis this problem may be overcome [9]. Thus future research may be directed at further exploring the implementation-related issues for such algorithms.

Another objective of our work was the performance comparison of virtual circuit routing and datagram routing. The nature of traffic conditions in the network is an important factor that must be considered before drawing any conclusion. As was mentioned earlier, under stationary traffic conditions the steady state performance of the VC-based CODEX algorithm displays oscillatory behavior which is due to the VC rerouting mechanism. On the other hand, it was seen that the rerouting mechanism is essential to provide adaptivity. However, we have observed that oscillations can be limited by imposing an upper bound on the traffic rates of individual VC's. Whether such a constraint is desirable or not is a high level issue, and a subject for further research. In bursty traffic conditions, however, the virtual circuit routing algorithms perform better.

While comparing virtual circuits to datagrams, a naturally arising issue is that of resequencing. This is a potential drawback for datagram networks, since (a) additional processing at the end nodes is required, and (b) packets experience additional buffering delays at the end nodes. We have found this additional delay to be insignificant compared to the overall mean system delay. However, the extra processing requirement may be a more crucial factor, depending on the network under consideration.

The main limitation of our simulation-based study is that all our experiments were conducted on a network (6 node, 16 links) which is relatively small compared to most of the existing networks. Although with a proper design of experiments a small network can bring out the essential characteristics of a routing algorithm, some of the trends observed in a small network may not carry over to a larger network. For example, in the case of CODEX algorithm a large network with a large number of virtual circuits each with a smaller traffic rate may be relatively less sensitive to rerouting. Similarly the effect of resequencing in datagram algorithms may be significant in a larger network. These are some of the issues which are currently under investigation [10] with a 19 node, 33 full duplex links network (an earlier model of the ARPANET).

REFERENCES

[1] P.A. Humblet, S.R. Soloway, "Algorithms for Data Communication Networks - Part 1", Codex Corp., 1986.

[2] P.A. Humblet, S.R. Soloway, B. Steinka, "Algorithms for Data Communication Networks - Part 2", Codex Corp., 1986.

[3] R.G. Gallager, "A Minimum Delay Routing Algorithm Using Distributed Computation", *IEEE Trans. Commun.*, COM-23, pp. 73-85,1977.

[4] J.M. McQuillan, I. Richer, E.C. Rosen, "The New Routing Algorithm for the ARPANET", *IEEE Trans. Commun.*, COM-28, pp. 711-719,1980.

[5] D.P. Berstsekas, "Dynamic Models of Shortest Path Routing Algorithms for Communication Networks with Multiple Destinations", *Proc. 1979 IEEE Conf Decision and Contr*, Ft. Lauderdale, FL, pp. 127-133.

[6] D.P. Berstsekas, "Dynamic Behaviour of Shortest Path Routing Algorithms for Communication Networks", *IEEE Trans Auto Contr.*, AC-27, 1982, pp. 60-74.

[7] Y.C. Ho, C. Cassandras, "A New Approach to the Analysis of Discrete Event Dynamic Systems", *Automatica*, 19, pp. 149-167, 1983.

[8] U. Pape, "Implementation and Efficiency of Moore - Algorithms for the Shortest Route Problem", *Mathematical Programming*, Vol. 7, pp. 212-222, 1974.

[9] C.G. Cassandras, M.V. Abidi, D.F. Towsley, "Distributed Routing With On-line Marginal Delay Estimation", to appear in *IEEE Trans. Commun.*, 1989.

[10] B.P. Mohanty, C. Cassandras, D. Towsley, "Issues in Gradient based Adaptive and Distributed Schemes for Dynamic Network Management", Technical Report CCS-, Dept. of Electrical and Computer Engineering, Univ. of Massachusetts, Amherst, 1989.

[11] Fu Chang and Lancelot Wu, "An Optimal Adaptive Routing Algorithm", *IEEE Trans Auto Contr.*, AC-31, 1986, pp. 690-700.

[12] L. Kleinrock, "Communication Nets: Stochastic Message Flow and Delay", McGraw Hill, New York, 1964, (Reprinted by Dover Publications, 1972).

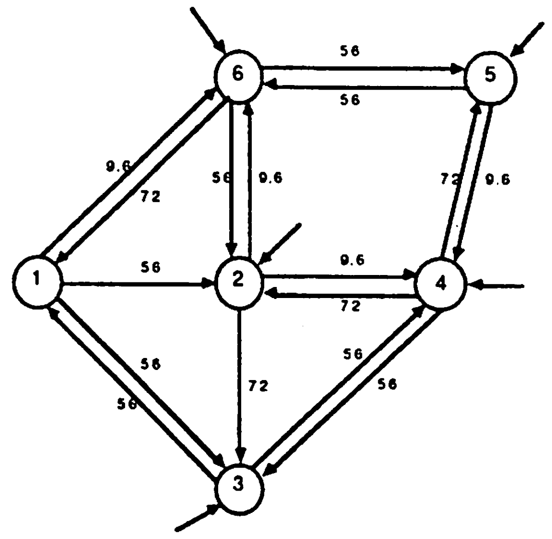

FIGURE 1: The Network Configuration

SOURCE-DESTINATION PAIR	INTERARRIVAL TIME DISTRIBUTION
1-3	EX(60)
1-4	EX(200)
1-5	EX(300)
1-6	EX(300)
2-1	UN(80,100)
2-5	UN(80,100)
2-6	UN(60,80)
3-1	EX(70)
3-2	EX(300)
3-5	EX(300)
4-1	EX(200)
4-3	EX(300)
4-6	EX(90)
5-1	EX(90)
5-2	EX(90)
5-3	EX(90)
6-1	UN(60,80)
6-2	UN(80,100)
6-4	EX(70)

TABLE 1. List of External Traffic Distributions Between Node Pairs

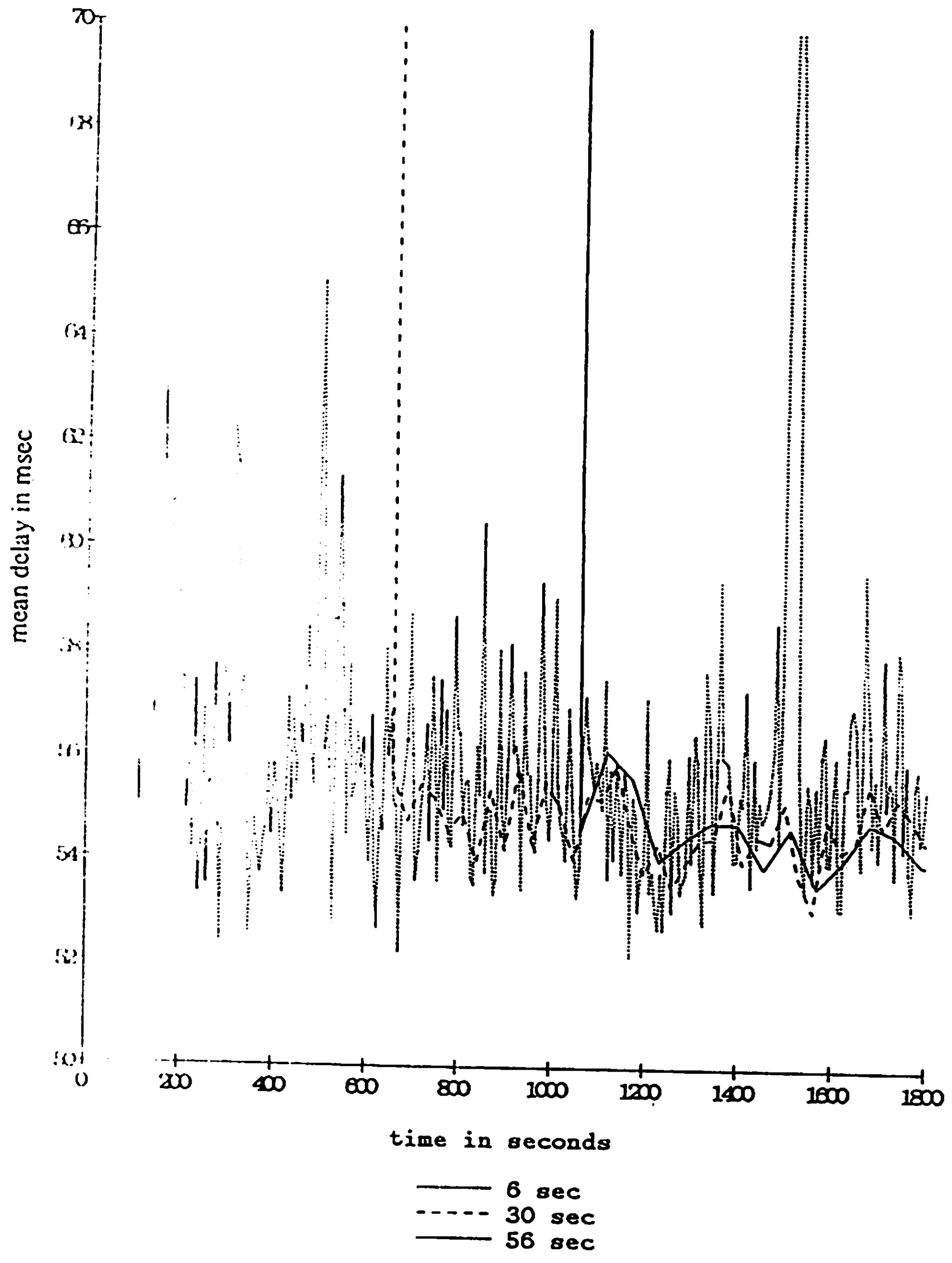

Figure 2: **Effect of Observation Interval Length on the Mean Delay Performance of the CODEX Algorithm**
Rerouting Probability = 0.45

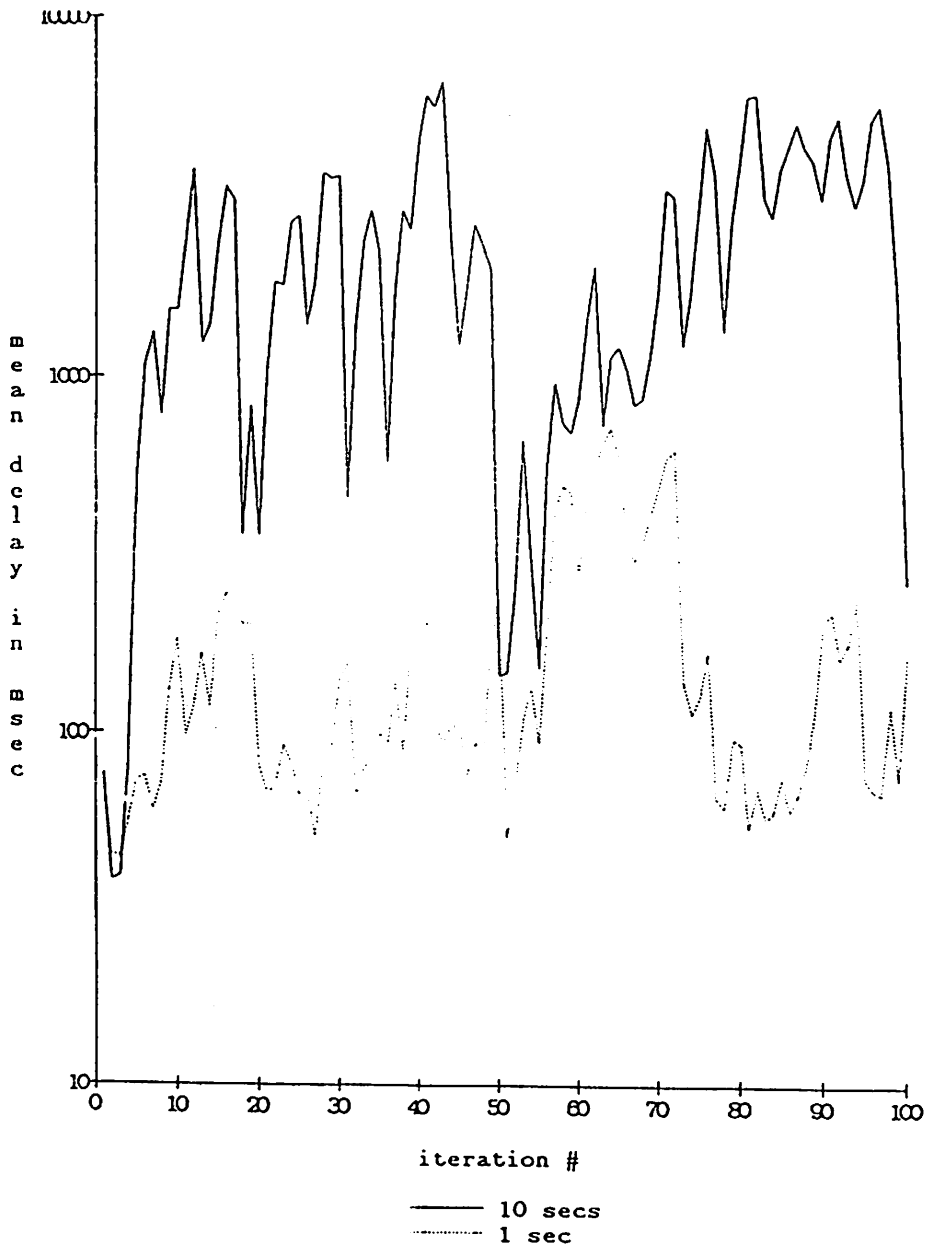

Figure 3: Effect of Observation Interval Length on the Mean Delay Performance of the ARPANET Algorithm

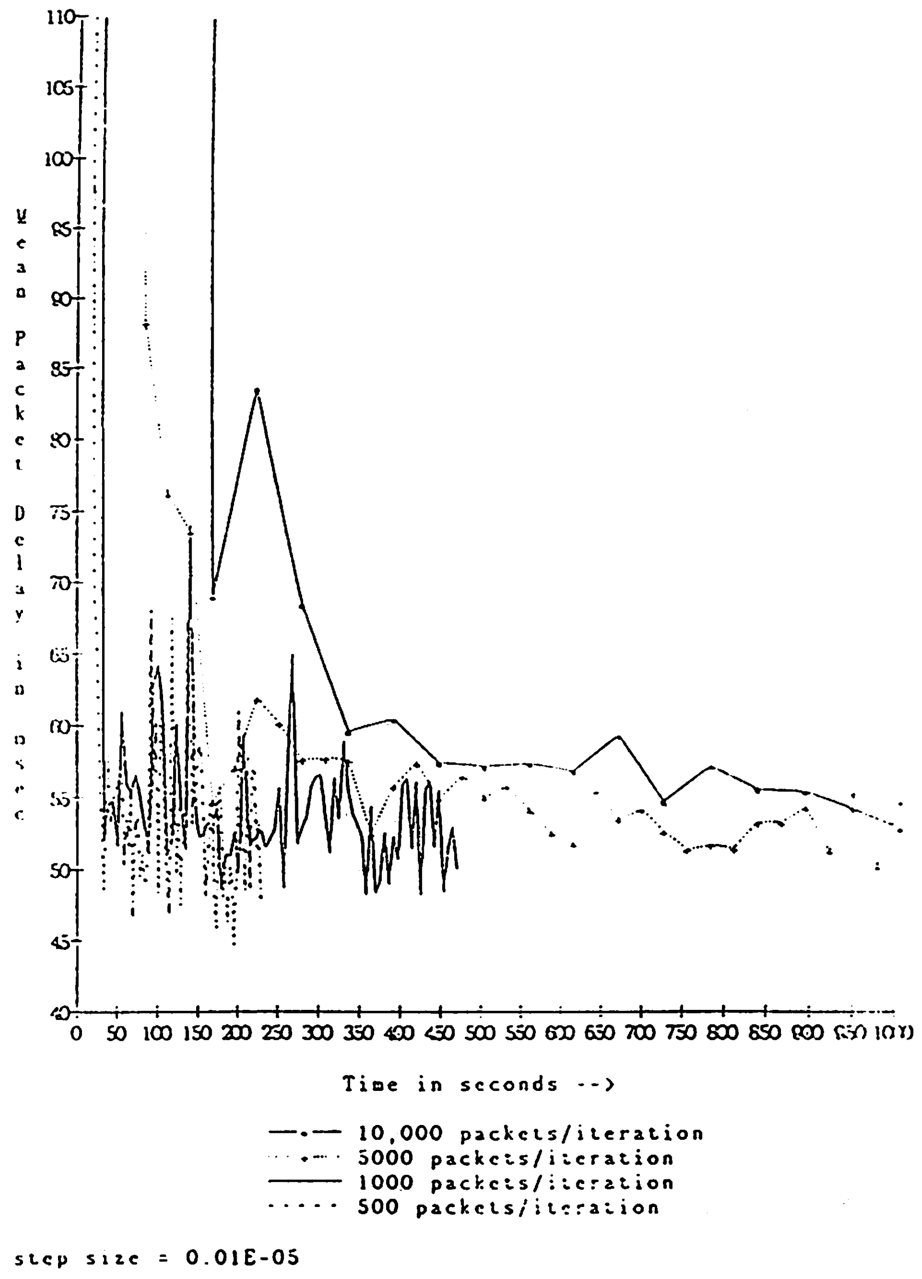

Figure 4: Effect of Observation Interval Length on the Mean Delay Performance of Gallager's Algorithm
Step Size = 0.01 E -05

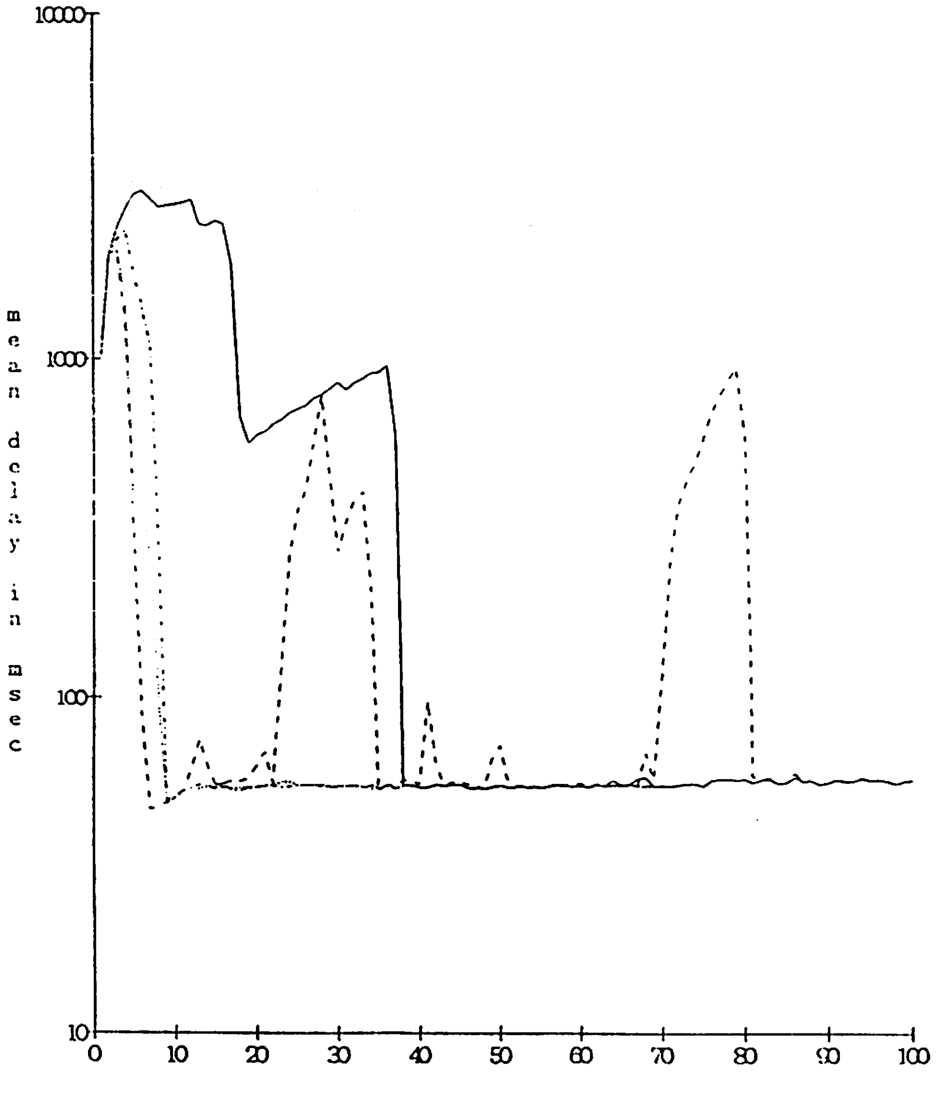

Figure 5: Effect of Rerouting Probabilities on the Mean Delay Performance of the CODEX Algorithm
Observation Interval Length = 56 sec

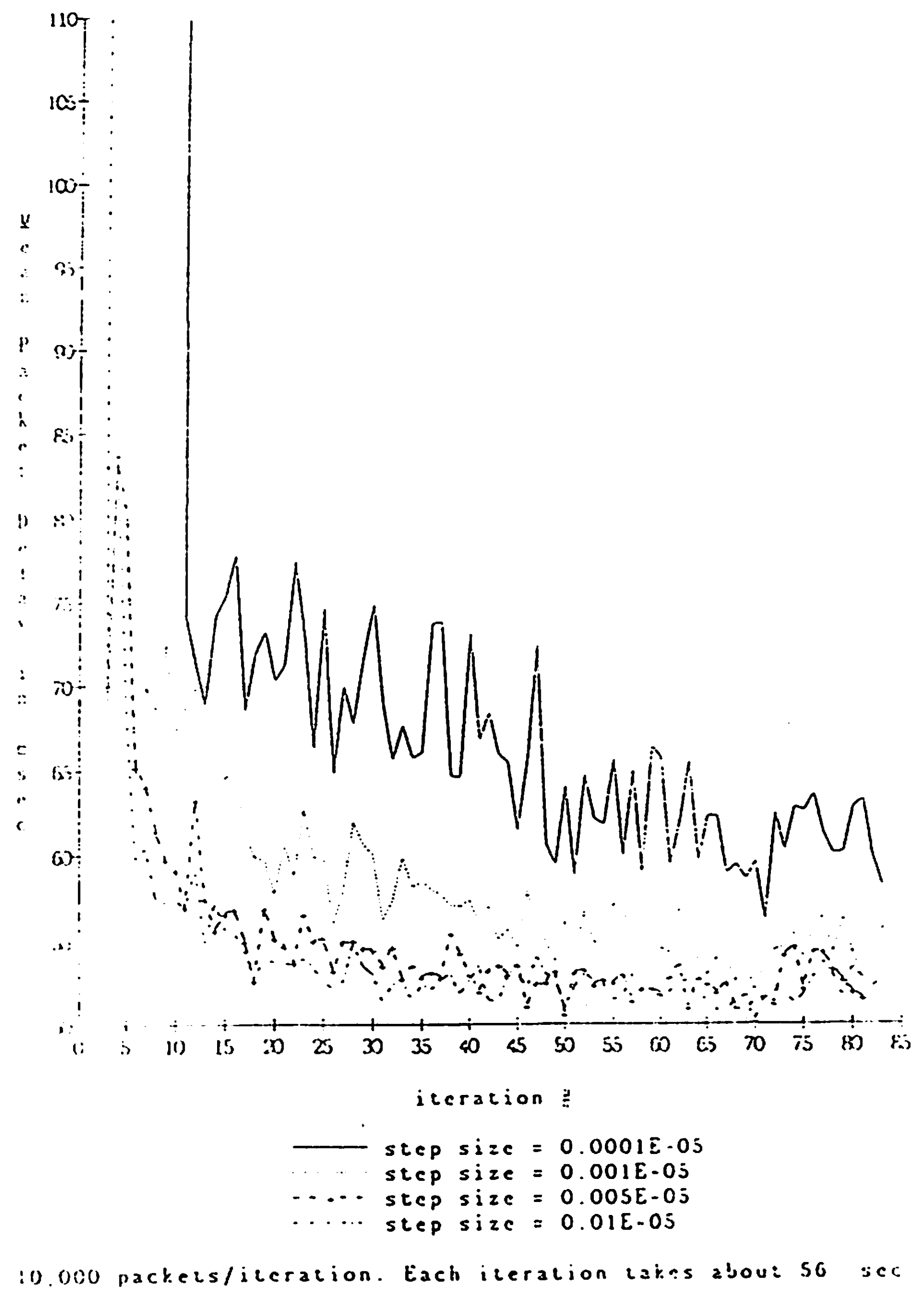

Figure 6: Effect of Step Sizes on the Mean Delay Performance of Gallager's Algorithm
Observation Interval Length = 56 sec

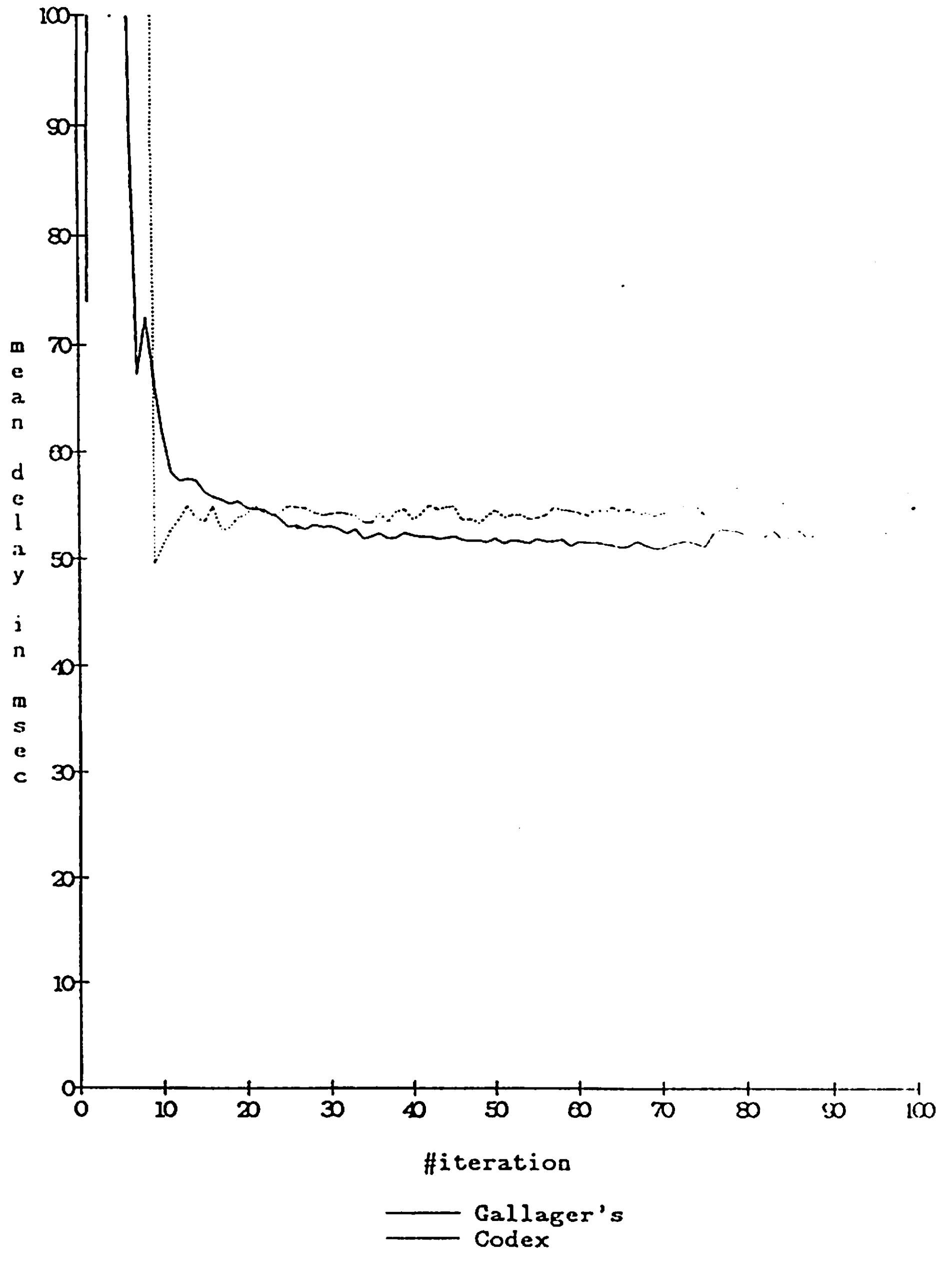

Figure 7: Comparison of Mean Delay Performances of the CODEX and Gallager's Algorithms with Tuned Parameters
Observation Interval Length = 56 sec, Rerouting Prob. = 0.3
Step Size = 0.01 E -05

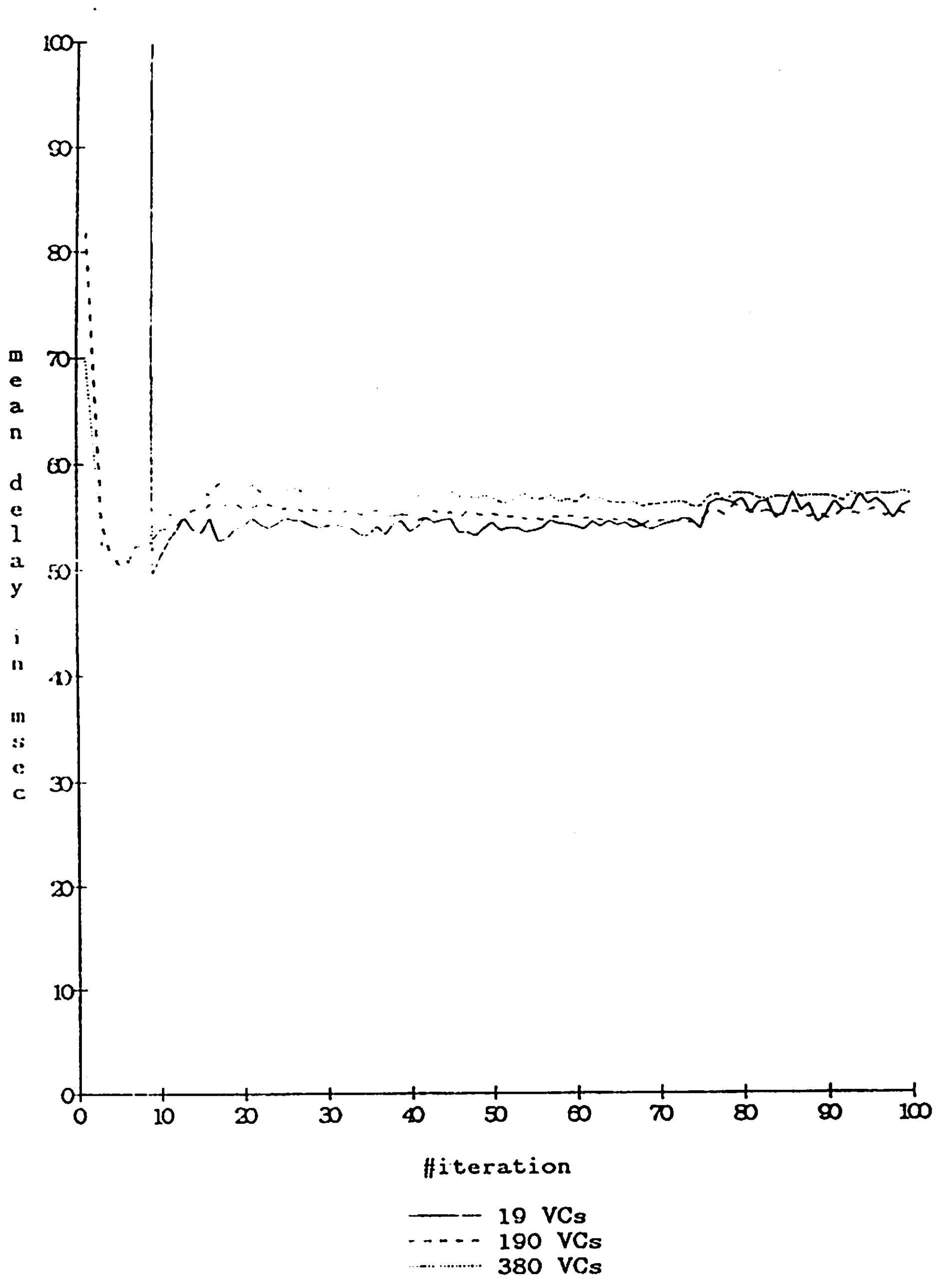

Figure 8: Effect of Increasing the Number of VC's while Decreasing the Traffic Rate of Each in the CODEX Algorithm
Observation Interval Length = 56 sec, Rerouting Prob = 0.3, 0.03, 0.015

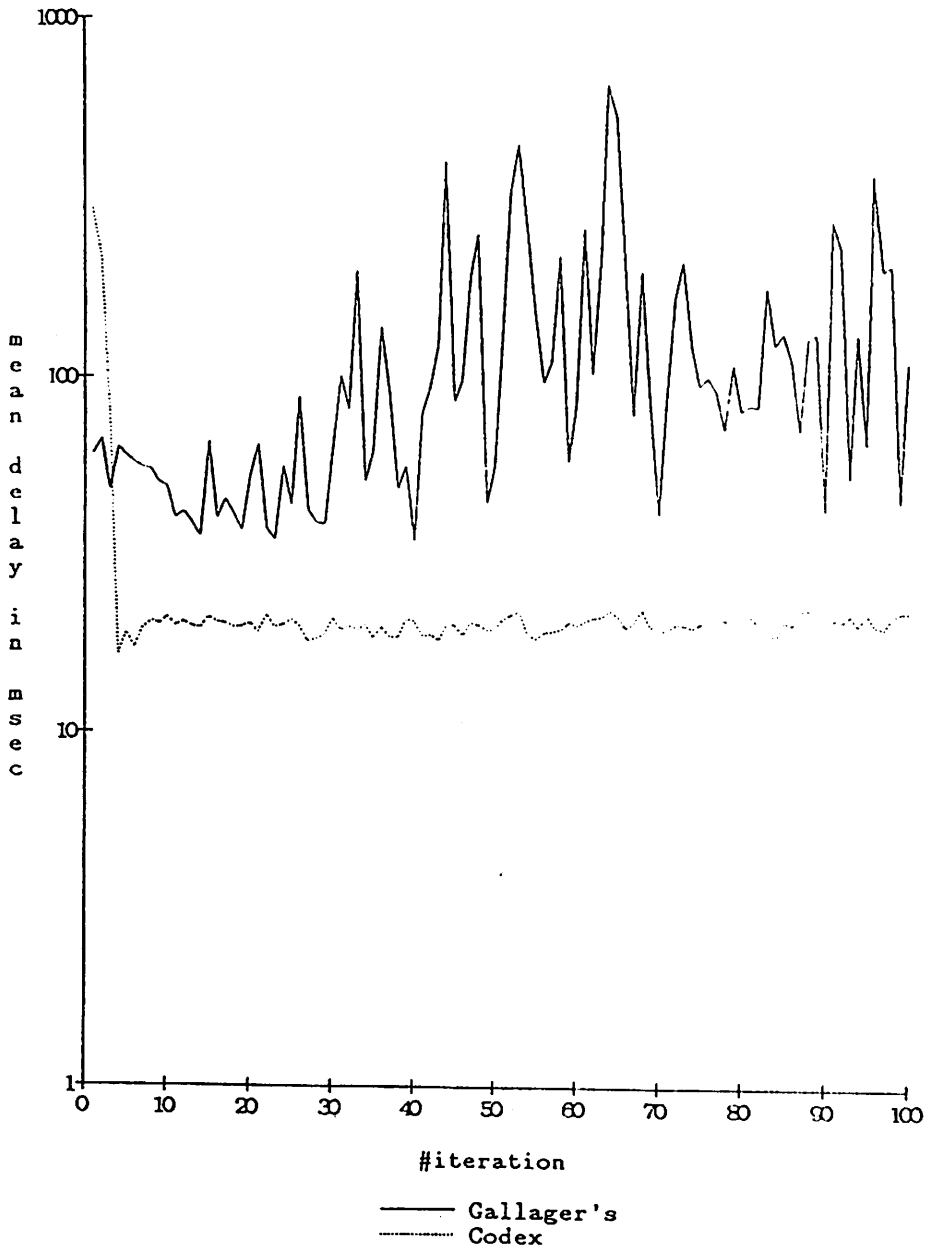

Figure 9A: Comparison of Mean Delay Performances of the CODEX and Gallager's Algorithm in Quasistatic Traffic Conditions (19 VC's)
Observation Interval Length = 56 sec, Rerouting Prob. = 0.3
Step Size = 0.01 E -05

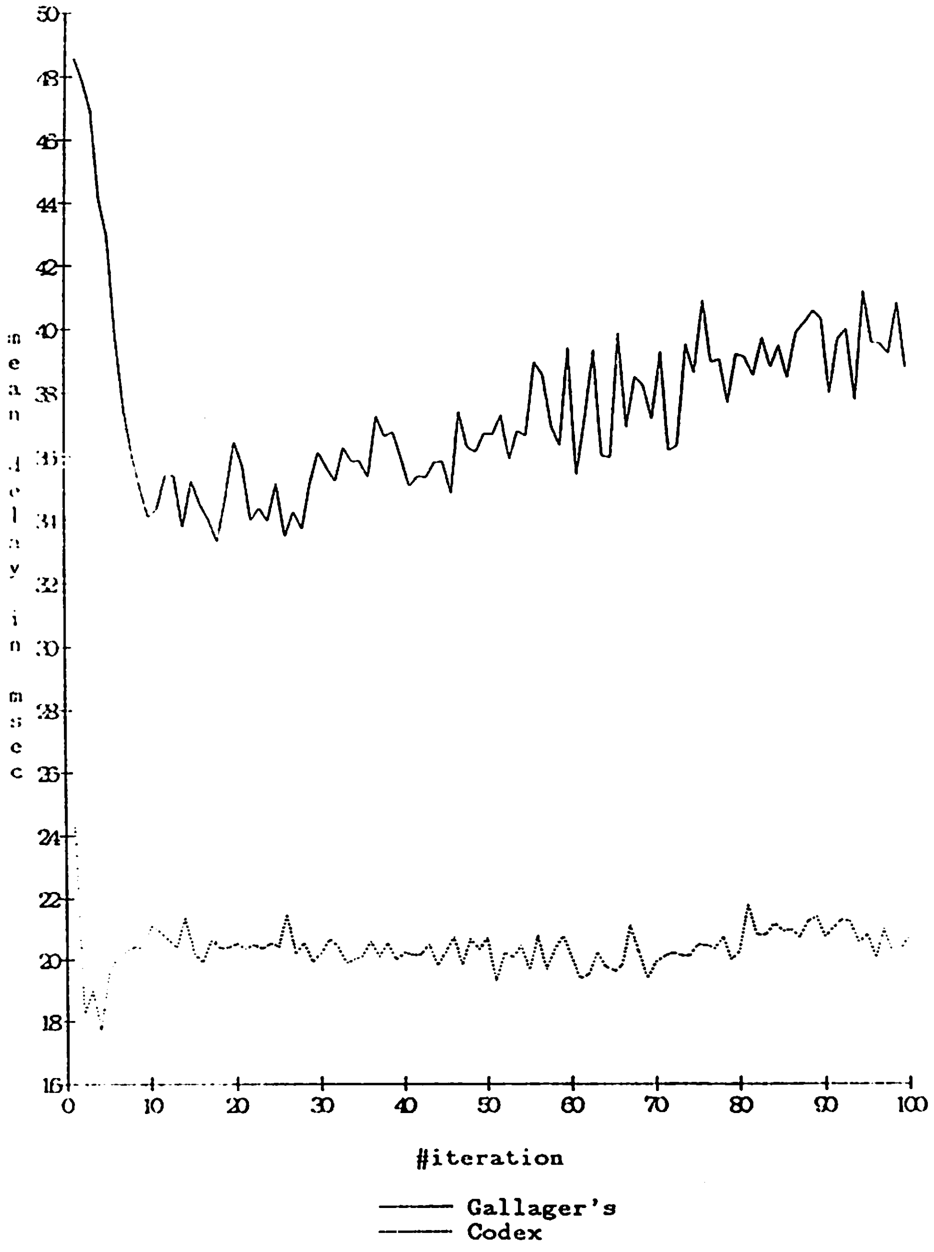

Figure 9B: Comparison of Mean Delay Performances of the CODEX and Gallager's Algorithm in Quasistatic Traffic Conditions (190 VC's)
Observation Interval Length = 56 sec, Rerouting Prob. = 0.03
Step Size = 0.01 E -05

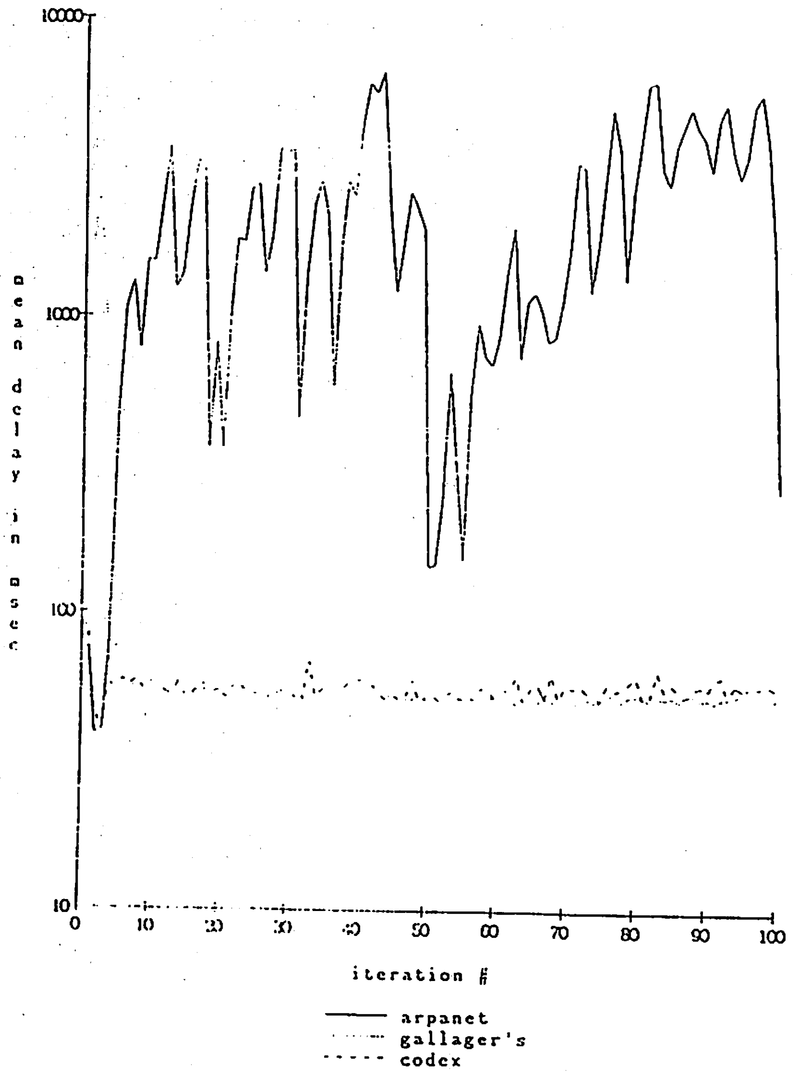

Figure 10: Comparison of Mean Delay Performances of the CODEX, Gallager's and the ARPANET Algorithm

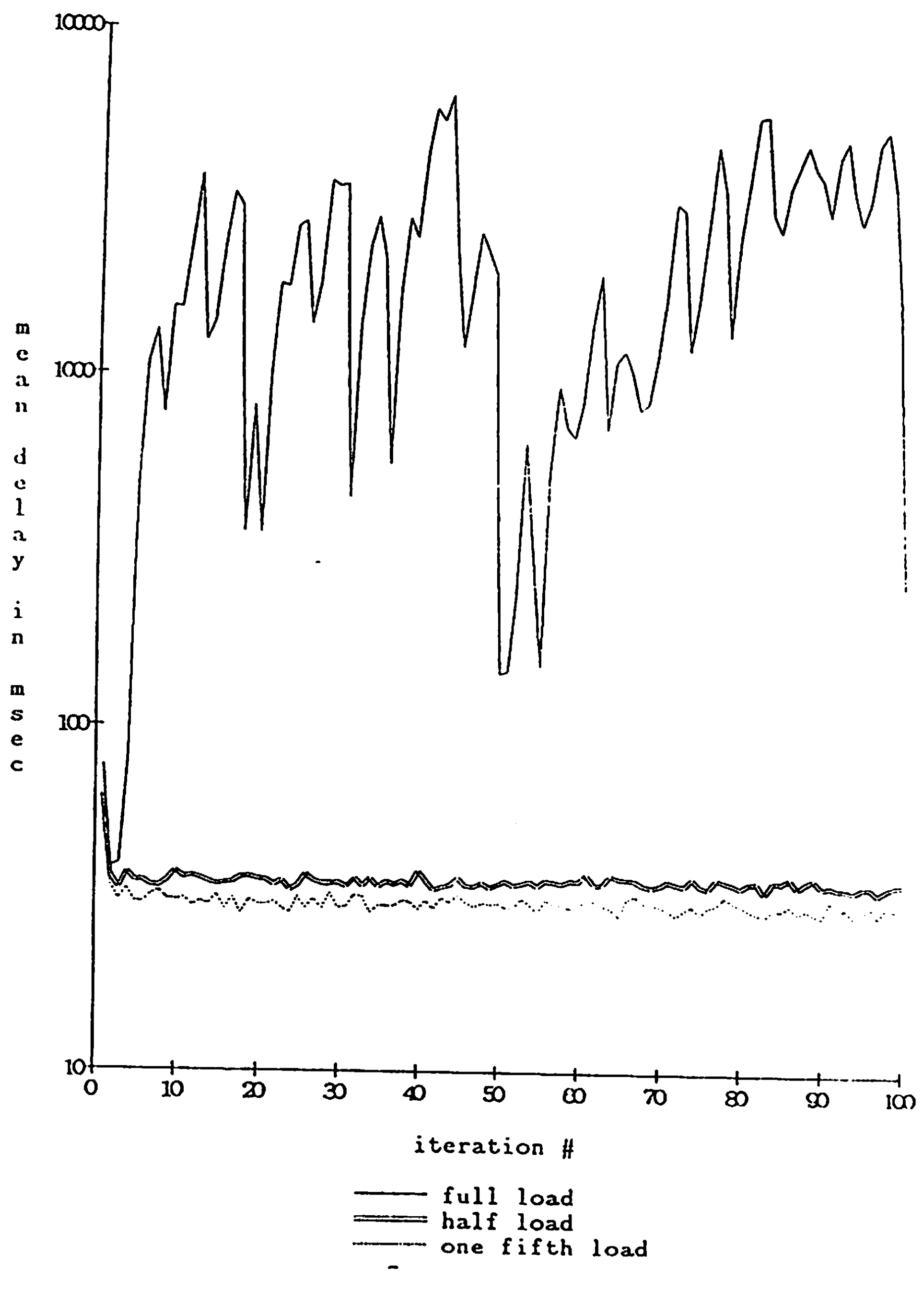

Figure 11: Mean Delay Performance of the ARPANET Algorithm under Light Traffic Conditions

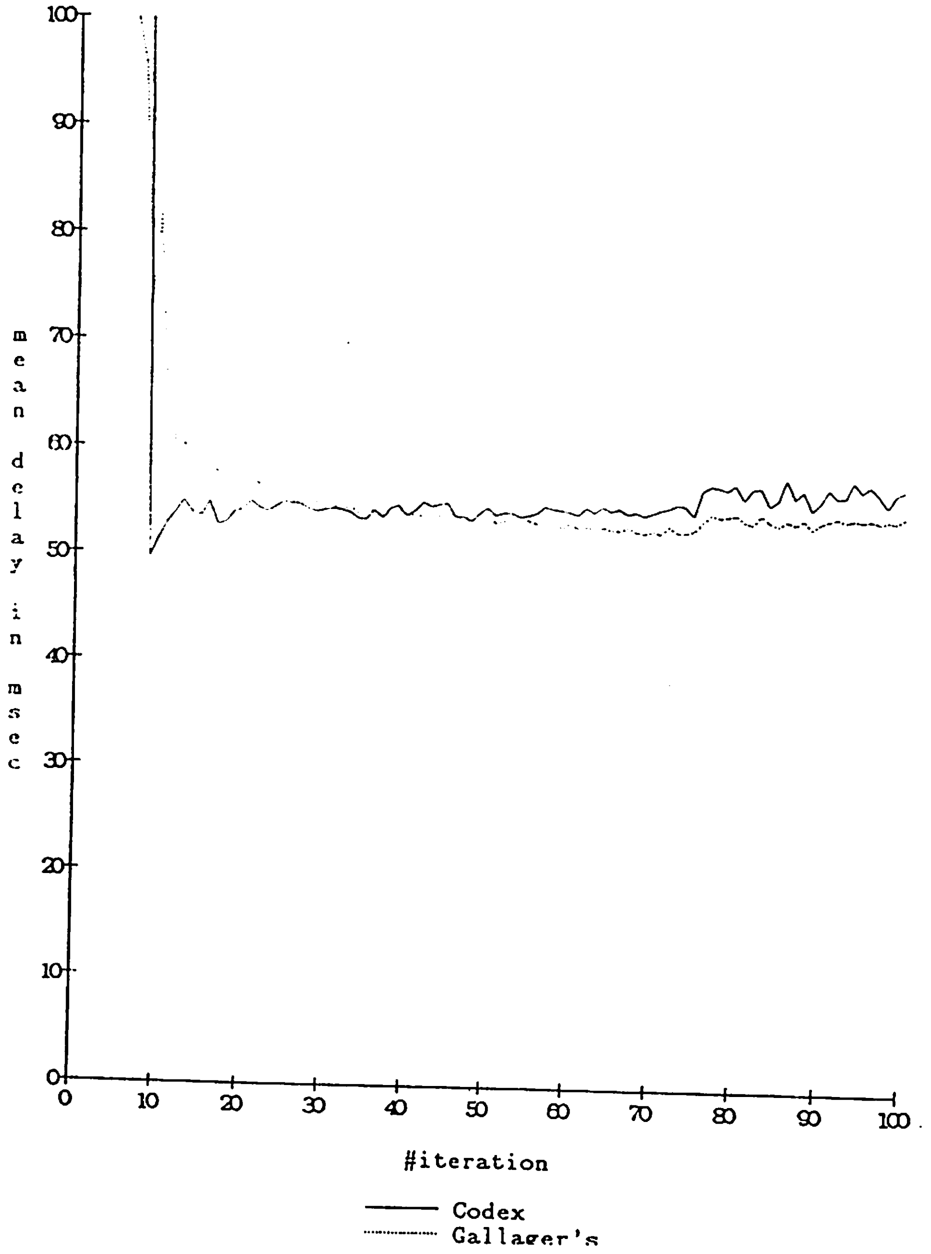

Figure 12: Comparison of Mean Delay Performances of the CODEX Algorithm to that of Gallager's Algorithm with Resequencing
Observation Interval Length = 56 sec, Rerouting Prob. = 0.3
Step Size = 0.01 E -05