패킷 교환망에서 흐름과 에러 제어과정에 관한 성능분석†

이창훈* · 홍정완* · 홍정식** · 이강원**

Performance Analysis of Flow and Error Control Procedures in a Packet-Switching Network

Chang-Hoon Lie*, Jeong-Wan Hong*, Jung-Sik Hong** and Kang-Won Lee**

Abstract

In this paper, the Go-Back-N ARQ protocol with decoding in communication network is considered. The time delay and throughput are respectively analyzed as a function of window size and decoding time out.

Packets arrive continuously at the decoder, and are stored in a buffer if the decoder is busy upon its arrival. The decoder devotes no more than a time-out period of predetermined length to the decoding of any single packet. If packet decoding is completed within that period, the packet leaves the system. Otherwise, it is retransimitted and its decoding starts anew.

The time delay and throughput are obtained using recursive formula and difference equation. An appropriate time out and window size that satisfies the grade of service can be determined.

1. 서 론

통신망(communication network)은 어떠한 정보를 한 지점에서 다른 지점으로 보내는 데 사용된다. 보내는 정보가 메시지(message)의 형태로 되어 있을 때, 이것을 전송하기 쉬운 작은 단위로 쪼개어 보내게 되는데 이 작은 단위를 패킷(packet)이라고 한다.

이러한 패킷 전송 시스템에서는 전송선(channel)의 상태, 잡음(noise) 등의 요인에 의해 전송된 패킷에 다양한 에러가 발생하게 된다. 따라서 정확한 메시지의 전송을 위해서는 이러한 에러를 제어(control)하는 기법이 필요하게 되었다. 이러한 에러 제어기법으로는 순방향 에러 정정(forward error correction : FEC)과 자동 반복 요청(automatic repeat request : ARQ)의 두가지로 크게 나누어진다. 전자는 송신기로부터 전송된 패킷에 대해서 수신기 스스로 발생한 에러를 수정하는 기법이며, 후자는

† 이 논문은 1990년도 문교부 지원 한국학술진흥재단의 자유공모 과제 학술연구조성비에 의하여 연구되었음

* 서울대학교 산업공학과

** 서울산업대학 산업공학과

수신기가 에러가 발생한 패킷에 대하여 송신기측에 피드백(feedback)시켜 재 전송해 주도록 요구하는 방법이다.

본 연구에서는 위에서 말한 FEC 기법과 ARQ 기법을 결합시켰다. 즉 FEC 기법에서의 전송선의 효율면의 이익과 ARQ 기법의 신뢰성 측면에서의 이익을 결합하여 각각 시스템의 단점을 보완하였다. 특히 여러가지 ARQ 기법중에서 흐름제어(flow control)까지 동시에 수행하는 Go-Back-N ARQ 기법에 대하여 분석하였다. 그리고 FEC 기법으로는 가장 보편적인 디코딩(decoding) 기법을 사용하였다.

디코딩 기법이 결합된 ARQ에 대해서는 분석이 비교적 쉬운 stop and wait ARQ에 대해서 1980년대에 많이 연구되었다. 전송선과 디코더(decoder)의 용량을 고려하여 패킷의 전송비율을 최대로 하는 타임 아웃(time-out) 분석을 시도하였고[7], 수율을 최대로 하는 타임 아웃에 대해서도 분석하였다[8].

1-1. Go-Back-N ARQ

Go-Back-N ARQ 기법하에서 패킷은 처음의 고유 순서대로 수신기에 전달되며, 수신기에 전달된 패킷은 수신기에서 에러 검출을 하게 된다. 이때, 이 패킷에 에러가 있으면 NAK(non-acknowledgement)의 신호를 송신기측에 피드백시킨다. 따라서 송신기가 NAK을 수신하면 에러가 난 패킷과 그 이후에 이미 전송된 패킷을 모두 재전송하여야 한다. 그리고 수신자는 NAK을 보낸 패킷을 다시 받을 때까지 그동안 전송된 패킷을 모두 버린다. 왜냐하면 패킷은 항상 고유 순서대로 수신기에서 처리되어야 하기 때문이다. 또한 패킷에 에러가 없으면 송신기측에 ACK를 피드백 시킨다. Go-Back-N ARQ 기법을 알기 쉽게 다시 표현하면 다음의 그림 1과 같다.

1-2. 디코딩 기법

디코딩 기법은 말 그대로 부호화(encode)된 패킷(흔히 프레임(frame)이라고 부른다)을 풀어내면서 에러를 수정하는 기법이다. 본 연구에서는 디코딩 기법의 여러가지 알고리즘 중 sequential decoding 기법[4]을 대상으로 하였다.

이 기법의 특징은 다른 기법과는 달리 디코딩 시간이 확률적이라는 것이다. 다른 디코딩 기법은 프레임의 형태에 상관없이 항상 똑같은 디코딩 시간이 소요된다. 이것은 모든 가능한 경로(path)를 다 검색하기 때문이다. 따라서 전송선의 상태가 양호한 경우(에러율이 낮은 경우)에 많은 시간 낭

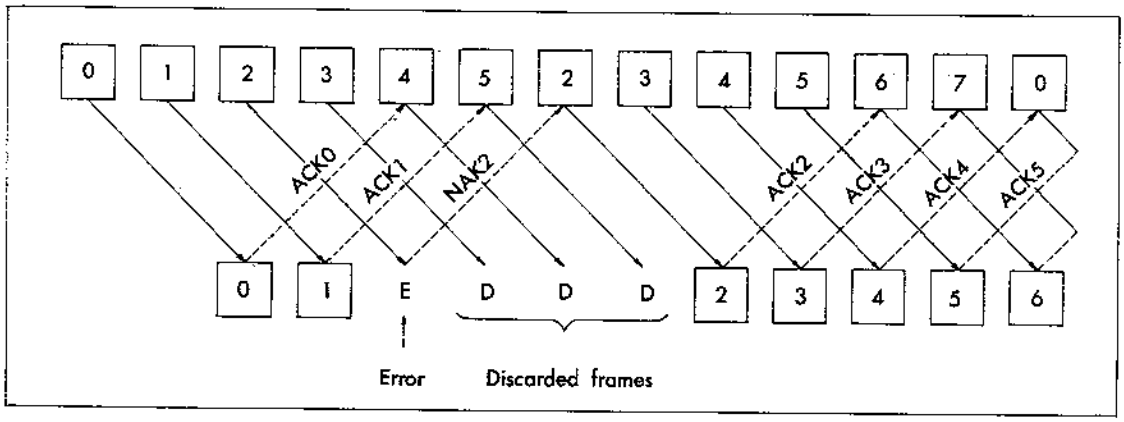

그림 1. Go-Back-N ARQ의 작동기법

비가 생긴다.

위와 같은 결점을 보완하기 위해서 디코딩 타임을 패킷의 에러율에 비례하는 디코딩 기법이 개발되었다. 즉 패킷에 에러가 작으면 디코딩이 빨리 끝나며, 에러가 많으면 오래 소요된다. 이러한 기법을 sequential decoding이라고 한다.

1-3. 분석모형

디코딩이 결합된 Go-Back-N ARQ 기법과 순수한 Go-Back-N ARQ 기법 사이에는 약간의 차이가 있다. 즉, 단순히 에러 검출만 하는 것이 아니고 에러 수정까지 하므로 존재하는 차이이다. 따라서 우리는 디코딩의 타임아웃(time-out)이라는 개념을 도입하였다. 디코딩 시간이 타임아웃보다 작을 경우에는 수신기가 송신기에게 ACK 신호를 보내고, 큰 경우에는 디코딩을 중단하며 송신기는 수신기에게 패킷을 재전송한다.

디코딩의 타임아웃을 설정하는 이유는 일반적으로 감지하지 못하는 에러를 줄이기 위해서 이지만 (디코딩 시간이 너무 길어지면 이 확률이 커진다) 본 연구에서는 다음과 같은 trade-off까지 고려하였다. 타임아웃이 적정 시간 이하일 경우에는 에러 수정시간은 줄어들지만 패킷 전송 횟수는 늘어나게 되고, 반대로 적정 시간 이상일 경우에는 패킷 전송 횟수는 줄어드는 반면 에러 수정시간은 늘어난다. 따라서 우리는 위의 trade-off를 이용하여 적정 타임아웃을 정할 수 있을 것이다.

또한 Go-Back-N ARQ에서 일반적으로 N을 윈도우 크기(window size)라고 하는데, 이는 송신기에서 수신기의 ACK를 수신하거나 타임아웃이 발생하기 전에 연속적으로 보낼 수 있는 패킷의 수를 의미한다. 지금까지 이 윈도우 크기는 주로 링크의 용량이나 디코더의 용량을 고려하여 결정되어 왔다. 또한 기존의 연구에서는 Go-Back-N ARQ의 수율을 윈도우 크기를 고려하지 않고 계산하였다[6]. 그러나 본 연구에서는 이 윈도우 크기와 수율과의 관계를 고려하여 수율을 윈도우 크기의 함수로 유도해 보았다.

기 호

X : 디코딩 시간 확률변수
디코딩 시간 x는 파레토 분포를 따르는 확률변수이다.
$f(x)=ab^a/x^{b+1}$, $x \geq a$

t : 디코딩 타임아웃
s : 전파시간(propagation delay)
a : 최소 디코딩 시간
b : 파레토 분포의 모수
d : 전송시간(transmission time)
N : 윈도우 크기

2. 순환식(Recursive formula)을 이용한 분석

Go-Back-N ARQ의 가장 큰 특징은 한 패킷이 자기 자신의 에러 여부뿐만 아니라, 그 이전 패킷의 에러 여부에 따라 재전송이 이루어질 수 있다는 것이다. 위의 특징을 가장 잘 나타낼 수 있는 것이 순환식이다. 본 장에서는 이 순환식을 이용하여 패킷의 시간 지연과 수율을 분석하고자 한다.

2-1. 시간 지연(time delay) 분석

패킷의 시간 지연이란 임의의 패킷이 윈도우내에 들어왔을 때부터 에러없이 수신기에서 처리될 때까지의 시간을 의미하는 것으로 이는 패킷이 전송 시스템내에 머물러 있을 시간을 의미한다.

1) 윈도우내의 패킷에 대한 시간 지연분석 ($n \leq N$)

이를 구하기 위해 먼저 윈도우 크기를 3으로 가정하고 전송되는 패킷의 형태를 패킷의 에러 형태에 따라 나누면 다음의 그림 2와 같다.

따라서 다음과 같은 순환식으로부터 시간 지연을

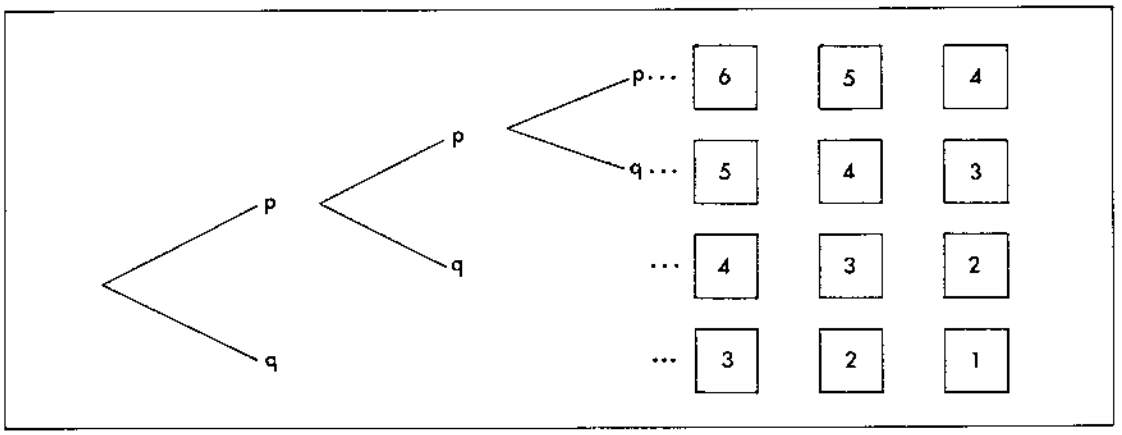

그림 2. 에러에 따른 패킷의 위치도(q=1−p).

구할 수 있다.

$$E[T_1]=p(d+s+E[X \mid X<\tau])+q(d+s+\tau+E[T_1])$$

$$E[T_2]=p^2(d+s+2E[X \mid X<\tau])+pq(d+s+E[X \mid X<\tau]+\tau+E[T_1])+q(d+s+\tau+E[T_2])$$

$$E[T_3]=p^3(d+s+3E[X \mid X<\tau])+p^2q(d+s+2E[X \mid X<\tau]+\tau+E[T_1])+pq(d+s+E[X \mid X<\tau]+\tau+E[T_2])+q(d+s+\tau+E[T_3])$$

윗 식을 연립하여 풀면,

$$E[T_1]=\frac{1}{p}(d+s)+E[X \mid X<\tau]+\frac{q}{p}\tau$$

$$E[T_2]=\frac{1+q}{p}(d+s)+2E[X \mid X<\tau]+\frac{2q}{p}\tau$$

$$E[T_3]=\frac{1+2q}{p}(d+s)+3E[X \mid X<\tau]+\frac{3q}{p}\tau$$

가 된다. 따라서 임의의 윈도우 크기(N)에 대해 위의 결과를 확장하면

$$E[T_n]=\frac{1+(n-1)q}{p}(d+s)+nE[X \mid X<\tau]+\frac{nq}{p}\tau \quad (n\leq N) \quad \cdots\cdots (1)$$

가 된다.

2) 윈도우 크기 이후의 패킷의 시간 지연분석 (n>N)

윈도우 크기 이후의 패킷의 시간 지연은 이 패킷이 윈도우 안으로 들어왔을 때 몇번째 위치에 있는지에 따라 좌우된다. 예를 들어 그림 2에서 보면, 4번째 패킷은 1,2,3 패킷이 모두 전송에 성공하면 첫번째 위치에 오게 되고 1,2 패킷만 성공하면 2번째 위치에 오게 되고, 1 패킷만 성공하면 세번째 위치에 오게 된다. 따라서 윈도우 크기 이후의 임의의 패킷의 시간 지연은 다음과 같은 조건부 평균값으로 나타내어 질 수 있다.

$$E[T]=\sum_{i=1}^{N} f_i E[T_i] \quad \cdots\cdots (2)$$

위에서 f_i는 임의의 패킷이 윈도우내에서 i번째에 위치할 극한 확률을 나타낸다. 그렇다면 이 위치 확률에 대해 알아보자.

$f_{n,i}$를 n번째 패킷이 윈도우 내에서 i번째에 위치할 확률이라 하면, N=3인 경우 $f_{4,i}$는 그림 2로부터

다음과 같은 순환식으로 나타낼 수 있다.

$$f_{4,1}=q\cdot f_{4,1}+p\cdot q\cdot 0+p^2\cdot q\cdot 0+p^3\cdot 1=p^2$$

$$f_{4,2}=q\cdot f_{4,2}+p\cdot q\cdot 0+p^2\cdot q\cdot 1+p^3\cdot 0=pq$$

$$f_{4,3}=q\cdot f_{4,3}+p\cdot q\cdot 1+p^2\cdot q\cdot 0+p^3\cdot 0=q$$

다음, 임의의 n번째 패킷의 위치확률을 직전 패킷(n−1번째)의 조건부 확률로 나타내면 아래와 같다.

$$f_{n,1}=0\cdot f_{n-1,1}+0\cdot f_{n-1,2}+f_{4,1}\cdot f_{n-1,3}$$

$$f_{n,2}=1\cdot f_{n-1,1}+0\cdot f_{n-1,2}+f_{4,2}\cdot f_{n-1,3}$$

$$f_{n,3}=0\cdot f_{n-1,1}+1\cdot f_{n-1,2}+f_{4,3}\cdot f_{n-1,3}$$

따라서, 패킷 위치확률은 위와 같은 차분 방정식으로 나타낼 수 있다. 위의 차분 방정식을 행렬로 나타내면,

$$\begin{bmatrix} f_{n,1} \\ f_{n,2} \\ f_{n,3} \end{bmatrix} = \begin{bmatrix} 0 & 0 & p^2 \\ 1 & 0 & pq \\ 0 & 1 & q \end{bmatrix} \begin{bmatrix} f_{n-1,1} \\ f_{n-1,2} \\ f_{n-1,3} \end{bmatrix} \quad \cdots\cdots (3)$$

단, $\begin{bmatrix} f_{1,1} \\ f_{1,2} \\ f_{1,3} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$

이 되고, n이 충분히 큰 경우 위의 극한 확률은 다음과 같다[2].

$$\begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} = \begin{bmatrix} p^2/(1+p+p^2) \\ p/(1+p+p^2) \\ 1/(1+p+p^2) \end{bmatrix} \quad \cdots\cdots (4)$$

위의 결과를 N에 대해 확장하면 n>N인 임의의 패킷의 극한 위치확률은 다음과 같다.

$$f_i=\frac{p^{N-i}}{1+p+\cdots+p^{N-2}+p^{N-1}} \quad \cdots\cdots (5)$$

따라서 이 패킷에 대한 시간 지연은 식(2)에 의해 다음과 같이 나타내어진다.

$$E[T]=\frac{N(1-p)}{p(1-p^N)}(d+s)+\frac{N(1-p)-p(1-p^N)}{(1-p)(1-p^N)}$$

$$E[X\mid X<\tau]+\frac{N(1-p)-p(1-p^N)}{p(1-p^N)}\tau \quad \cdots\cdots (6)$$

2-2. 수율($E[T_{inter}]$)의 분석

$E[T'_n]$을 n개의 패킷이 에러없이 모두 전송될 때까지의 시간으로 정의하면, $E[T_{inter}]$는 다음의 식으로 나타낼 수 있다.

$$E[T_{inter}]=\lim_{n\to\infty}(E[T'_n]-E[T'_{n-1}]) \quad \cdots\cdots (7)$$

패킷의 총 전송시간 ($E[T_{inter}]$)은 그림에서 알 수 있듯이 같은 윈도우내의 패킷끼리는 시간 중복이 존재하지만 윈도우끼리는 시간중복이 없다. 즉 n번째 패킷이 윈도우내에 i번째 위치에 있을 경우 총 전송시간은 [이전 윈도우 안의 마지막 패킷의 총 전송시간]+[윈도우내의 i번째 패킷의 전송시간]으로 나타낼 수 있다.

따라서 $E[T'_n]$은 다음과 같이 쓸수 있다(N=3인 경우).

$$E[T'_n]=f_1(E[T'_{n-1}]+E[T_1])$$
$$+f_2(E[T'_{n-2}]+E[T_2])$$
$$+f_3(E[T'_{n-3}]+E[T_3]) \quad \cdots\cdots (8)$$

식(8)은 3차 차분 방정식이므로 이를 상태 공간형(state space form)으로 바꾸면 다음과 같다. 즉,

$X(n)=\begin{bmatrix} E[T'_{n-2}] \\ E[T'_{n-1}] \\ E[T'_n] \end{bmatrix}$ 으로 정의하면,

$$X(n)=A\cdot X(n-1)+b$$

$$=\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ f_3 & f_2 & f_1 \end{bmatrix}\cdot X(n-1)+\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}\cdot u$$

단, $u=f_1E[T_1]+f_2E[T_2]+f_3E[T_3]$

가 된다. 위 차분 방정식의 일반해는 다음과 같이 주어진다[2].

$$X(n)=A^n \cdot X(0)+\sum_{i=0}^{n-1} A^{n-1-i} \cdot b \quad \cdots\cdots (9)$$

윗 식에서 행렬 A의 최대 고유치(dominant eigenvalue)는 1이므로 행렬 A^n은 수렴하며, 따라서 $E[T_{inter}]$는 다음의 식으로 주어질 수 있다.

$$E[T_{inter}]=X(n)-X(n-1)$$

$$=\bar{A} \cdot b \quad (\bar{A}=\lim_{n\to\infty} A^n)$$

$$=\frac{\sum_{i=1}^{N} f_i E[T_i]}{\sum_{i=1}^{N} i f_i}$$

$$=\frac{N(1-p)^2}{p(p^{N+1}-(N+1)p+N)}(d+s)+E[X \mid X<\tau]$$

$$+\frac{q}{p}\tau \quad \cdots\cdots (10)$$

3. 예 제

본 예제에서는 거리가 100km 정도 떨어져 있는 장거리 통신 시스템을 고려해 보았다. 왜냐하면 Go-Back-N ARQ는 이러한 장거리 통신 시스템에 효율적이기 때문이다. 그리고 전송선의 상태가 좋지 않아 에러가 상당히 많은 경우를 생각해 보았다. 본 예제에서 다루는 시스템은 아래와 같다.

$r=0.5$, $B=1000$(bit), $R=10^7$(bps),
$a=0.2$(msec)

$$s=\frac{100\times10^3}{2\times10^8}=5\times10^{-4}(\text{sec})=0.5(\text{msec})$$

(일반적인 통신 시스템의 전파속도는 2×10^8 m/sec이다.)

$$d=\frac{10^3}{0.5\times10^7}=2\times10^{-4}(\text{sec})=0.2(\text{msec})$$

그리고 파레토 분포의 모수인 b는 전송선의 비트당 에러율의 함수이다. 또 다음과 같은 근사적인 함수를 따른다[8].

$$b=-16\times\frac{r}{1-\log_2[1+2\sqrt{e(1-e)}]}+17$$

단, e : 비트당 에러율

본 예제에서는 비트당 에러율을 3가지 경우로 변화시켰을 때 타임아웃과 $E[T_{inter}]$와의 관계를 살펴봄으로써, 적정 타임아웃을 알아볼 것이다. 또한, 윈도우 크기와 $E[T_{inter}]$와의 관계도 알아보려고 한다. 표 1은 윈도우 크기가 5인 경우에 타임아웃의 변화에 따라 수율이 어떻게 변하는가를 나타낸 것이며, 또한 표 2는 타임아웃이 0.5인 경우에 윈도우 크기가 변함에 따라 수율이 어떻게 변해가는 가를 나타낸 것이다.

4. 결 론

지금까지의 Go-Back-N ARQ의 시간지연과

표 1. 타임아웃과 수율과의 관계

Time out	Inter packet time(E[Tinter])		
	e=0.001	e=0.01	e=0.05
0.25	0.5720	0.6812	15.7183
0.30	0.4803	0.5196	8.9659
0.35	0.4661	0.4884	6.7359
0.40	0.4627	0.4788	4.6376
0.45	0.4616	0.4752	4.9925
0.50	0.4613	0.4737	4.5741
0.55	0.4611	0.4729	4.2855
0.60	0.4611	0.4726	4.0779
0.65	0.4610	0.4724	3.9243
0.70	0.4610	0.4723	3.8084
0.75	0.4619	0.4722	3.7200
0.80	0.4610	0.4722	3.6521
0.85	0.4610	0.4721	3.5999
0.90	0.4610	0.4721	3.5600
0.95	0.4610	0.4721	3.5299
1.00	0.4610	0.4721	3.5078
1.05	0.4610	0.4721	3.4921
1.10	0.4610	0.4721	3.4819
1.15	0.4610	0.4721	3.4761
1.20	0.4610	0.4721	3.4741
1.25	0.4610	0.4721	3.4753
1.30	0.4611	0.4721	3.4792

표 2. Window Size와 수율과의 관계

Window size	Inter packet time(E[Tinter])		
	e=0.001	e=0.01	e=0.05
1	2.7289	2.7473	13.4610
2	1.8955	1.9129	12.3328
3	1.4788	1.4958	11.8684
4	1.2287	1.2455	11.6306
5	1.0620	1.0786	11.4899
6	0.9430	0.9594	11.3977
7	0.8537	0.8700	11.3328
8	0.7842	0.8005	11.2846
9	0.7286	0.7449	11.2475
10	0.6831	0.6994	11.2181
11	0.6453	0.6615	11.1941
12	0.6132	0.6294	11.1742
13	0.5857	0.6019	11.1575
14	0.5619	0.5781	11.1431
15	0.5410	0.5572	11.1308
16	0.5227	0.5388	11.1200
17	0.5064	0.5224	11.1105
18	0.4917	0.5078	11.1020
19	0.4786	0.4946	11.0945
20	0.4668	0.4827	11.0877

수율의 분석은 윈도우 크기(N)와 무관한 방향으로 진행되어 왔다. 그러나 본 연구를 통하여 윈도우 크기의 함수로 나타내어질 수 있다는 것을 보여주었다.

본 연구에서 수율의 척도로 제시된 $E[T_{inter}]$는 예제에서 보다시피 윈도우 크기(N)가 증가함에 따라 감소됨을 알 수 있다. 따라서 Go-Back-N ARQ에서는 전송선의 용량과 디코더의 용량을 고려하여 가능한 한 N을 최대로 하는 것이 좋은 수율을 제공하여 줄 것이다. 그러나 윈도우 크기의 증가에 따라 수율의 증가폭은 감소한다.

또한, 디코더의 타임아웃 측면에서는 적정 타임아웃이 존재함을 알 수 있었다. 특히, 에러율이 큰 경우에는 최적 타임아웃이 유일하게 존재하며, 타임아웃을 정하는 것이 매우 중요함을 알 수 있었다.

끝으로, 본 연구에서 수율의 척도로 구한 $E[T_{inter}]$는 수신기에서의 패킷 도착률을 정확하게 제시하여 주므로, 통신망에서 정확한 서비스의 기대 수준을 제공하는 자료가 될 수 있다.

참고문헌

[1] Drukarev, A., Costello, D.J., "Hybrid ARQ Error Control Using Sequential Decoding," IEEE Trans. Inform. Theory, Vol. IT-29, pp. 521-535, July, 1983.

[2] Feller, W., "An Introduction to Probability Theory and Its Applications," John Wiley & Sons, 1971.

[3] Halsall, F., "Data Communications, Computer Networks and OSI," Second ed., Addison Wesley Publishing Company, 1988.

[4] Lin, S., Costello, D.J., "Error Control Coding: Fundamentals and Applications," Prentice Hall, 1983.

[5] Molloy, M.K., "Performance Analysis Using Stochastic Pertri Nets," IEEE Trans. Computers, Vol. C-31, pp.913-917, Sep., 1982.

[6] Schwartz, M., "Telecommunication Networks: Protocols, Modeling and Analysis," Addison Wesley Publishing Company, 1986.

[7] Shacham, N., "Performance of ARQ with Sequential Decoding over One Hop and Two Hop Radio Links," IEEE Trans. Commun., Vol. COM-31, pp.1172-1180, Oct., 1983.

[8] Shacham, N., "ARQ with Sequential Decoding of Packetized Data: Queueing Analysis," IEEE Trans. Commun., Vol. COM-32, pp.1118-1126, Oct., 1984.

[9] Stalling, W., "데이타 통신 및 컴퓨터 통신," 희중각, 1987.

[10] Yamamoto, H., Itoh, K., "Viterbi Decoding Algorithm for Convolutional Codes with Repeat Request," IEEE Trans. Inform. Theory, Vol. IT-26, pp.540-547, Sep., 1980.