비트 그룹화 기술을 사용한 정보재건기법 Cascade 프로토콜의 일반화

이은상*, 노종선*

*서울대학교

*eslee3209@ccl.snu.ac.kr, *jsno@snu.ac.kr

Generalization of the information reconciliation protocol Cascade using bit grouping technique according to conditional probability

Eunsang Lee*, Jong-Seon No*

*Seoul National Univ.

요 약

본 논문에서는 현재까지 가장 잘 알려진 정보재건기법(information reconciliation protocol) 프로토콜인 Cascade 프로토콜에 관해 최근 연구된 비트 그룹화 기술을 일반화하였다. 이전 논문에서는 BER이 낮은 경우에 단순히 조건부확률이 같은 비트들 끼리만 그룹화하여 블록을 형성하였는데 본 논문에서는 다른 조건부확률 값의 비트들도 그룹화하는 것을 허용함으로써 이 기술을 일반화하였고 프로토콜의 성능을 더 개선하게 되었다.

Ⅰ. 서 론

정보재건기법 프로토콜 중 가장 잘 알려진 논문이 Cascade 프로토콜이다[1]. 최근 정보재건기법 관련 논문들의 상당수는 이 Cascade 프로토콜을 최적화시킨 논문들이다. 현재까지 Cascade 프로토콜을 가장 잘 최적화시킨 논문은 [2]인데, 여기서는 조건부확률에 따라 비트들을 그룹화하는 기술을 사용한다.

본 논문에서는 BER이 낮은 경우에는 조건부확률이 다른 비트들끼리도 두 번째 패스에서 같이 블록을 형성하는 것을 허용함으로써 비트 그룹화 기술을 일반화하였다. BER이 낮은 경우 조건부확률이 같은 비트들끼리만 그룹화를 하면 그룹의 비트 수 자체가 적어 최적의 블록 길이와 많이 차이가 나는 경우가 생긴다. 그러나 조건부확률 값이 다른 비트들도 그룹화하는 것을 허용함으로써 프로토콜의 성능을 더 개선할 수 있다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 Cascade 정보재건기법 프로토콜을 소개한다. Ⅲ에서는 Cascade 프로토콜 상에서 조건부확률에 따라 비트들을 그룹화하는 기술을 소개한다. Ⅳ장에서는 BER이 낮은 경우 조건부확률 값이 다른 비트들도 그룹화하는 것을 허용하는 아이디어를 제시하고 마지막으로 결론을 맺는다.

Ⅱ. Cascade 프로토콜

먼저 정보재건기법 프로토콜의 상황은 다음과 같다. 앨리스가 $\Pr[X=0]=\Pr[X=1]=\frac{1}{2}$인 확률변수를 비트 오류 확률(Bit error rate, BER)이 p인 2진 대칭 채널(binary symmetric channel, BSC)을 통해 밥에게 보내고 밥은 이 값을 수신하는데 그 확률변수를 Y라 하자. 앨리스는 총 n번 비트를 전송하여 결과적으로 앨리스와 밥은 벡터 $\mathbf{x},\mathbf{y}\in\{0,1\}^n$를 얻게 된다. $\mathbf{x}$값을 기준으로 $\mathbf{y}$값은 오류 비트들을 갖게 된다. 앨리스와 밥은 공개 채널을 통해 자유롭게 메시지를 주고받을 수 있다고 하자. 이 때, 공개 채널 상에서 최대한 적은 정보를 공격자 이브에게 누출하면서 밥이 $\mathbf{y}$의 오류들을 고쳐서 $\mathbf{x}$와 거의 같은 값을 얻는 과정이 바로 정보재건기법 프로토콜이다.

이 정보재건기법의 대표적인 프로토콜은 Cascade 프로토콜이다[1]. Cascade 프로토콜은 여러 패스(pass)로 구성되는데 각 패스에서 앨리스와 밥이 각자의 벡터 $\mathbf{x},\mathbf{y}$를 길이 k_i인 블록들로 랜덤하게 쪼갠다. 앨리스와 밥은 각 블록의 패리티(비트들의 modulo 2 덧셈)를 공개채널을 통해 서로 교환하며, 패리티가 같으면 넘어가지만 다르면 그 블록에 적어도 하나의 오류가 있는 것이므로 이진 검색(binary search)를 통해 오류 하나를 찾아낸다. 여기서 이진 검색은 블록을 반으로 쪼갠 후, 두 반쪽의 패리티를 교환하고, 패리티가 다른 블록을 다시 반으로 쪼개고, 두 반쪽의 패리티를 교환하며 이 과정을 끝까지 반복하여 오류 하나를 찾아내는 것이다. 오류를 찾아내면 밥은 그 오류 비트를 반전시켜 오류를 정정한다.

두 번째 패스부터는 어떤 오류를 정정하면 그 오류 비트를 갖고 있는 이미 완료된 패스의 블록의 패리티는 원래 0이었는데 정정하는 순간 1이 되므로 그 블록의 오류를 이진 검색을 통해 추가로 찾아낼 수 있다.

Ⅲ. 조건부확률에 따라 비트 그룹화

위 Cascade 프로토콜을 최적화하는 논문들이 최근 많이 게재가 되었는데 그 중 가장 잘 최적화한 논문은 [2]이다. 여기서 사용하는 기법 중 하나는 조건부확률에 따라 비트를 그룹화하는 것이다. 첫 번째 패스가 끝난 시점에서 두 번째 패스에서의 각 비트가 오류일 조건부확률은 그 비트가 첫 번째 패스에서 어떤 길이의 블록에 있었느냐에 따라 다르다. BER이 p일 때, 길이 t블록에 있었던 비트가 두 번째 패스에서 오류일 조건부확률을 실제 계산하면 $p_{bit}=p\frac{1-(1-2p)^{t-1}}{1+(1-2p)^t}$로서 t에 따라 다르다[2].

최적의 블록 길이는 조건부확률에 따라 달라지는데 조건부확률이 다른 비트들로 블록을 만들게 되면 최적의 블록 길이가 아니므로 성능이 저하된다. 따라서 조건부확률이 같은 비트들을 그룹화하고 각 그룹 내에서 최적 길이의 블록을 형성하여 프로토콜을 진행함으로써 성능을 향상시킨다. 아래 그림은 최적화한 [2] 논문의 성능을 나타낸다.

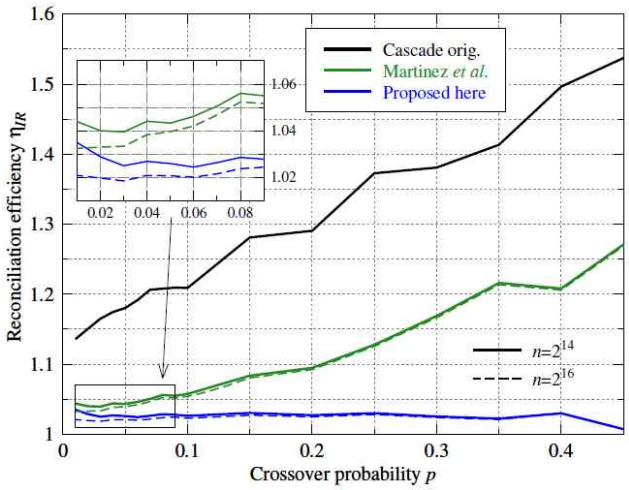

그림 1 비트 그룹화 기법을 사용한 프로토콜의 성능 개선

Ⅳ. 조건부확률에 따라 비트 그룹화 기법의 일반화

조건부확률에 따라 비트들을 그룹화하는 경우의 단점은 그룹을 지을 때 각 그룹 내의 비트 개수가 작아 최적의 블록 길이를 적용시키지 못할 수 있다는 점이다. 이런 경우에는 조건부확률이 조금 다르더라도 같이 그룹화하는 것을 통해 블록 길이를 크게 하는 것이 성능이 더 좋은 경우가 발생한다. 따라서 기본적으로는 조건부확률이 같은 비트들끼리 그룹을 짓되 최적 블록 길이에 비해 비트 수가 적은 경우 조건부확률 값이 다른 비트들도 같이 그룹화하는 것을 허용한다.

예를 들어 BER이 0.01인 경우 길이 $2^{14}=16384$비트의 프레임에 대해서 Cascade 프로토콜을 진행한다고 하자. BER이 p인 경우 논문 [2]에서 시뮬레이션을 통해 제시한 첫 번째 패스의 최적의 블록길이는 다음과 같다.

$$k_1=\begin{cases}\min(2^{[\log_2(1/p)]},n/2) & \text{if } p\leq 0.25\\ \min(\max(1,2^{[\log_2(1/p)]-1}),n/2) & \text{if } p>0.25\end{cases}$$

여기서 $[x]$ 는 x와 가장 가까운 정수를 의미한다. $p=0.01$인 경우 $k_1=128$이 된다. 길이 128인 블록들로 쪼갰을 때, 각 블록의 패리티가 다를 확률(Alice와 Bob의 패리티)을 계산해보면 0.5에 가깝다. 패리티가 다른 경우 이진 검색이 진행된다. 따라서, 첫 번째 패스가 끝났을 때, 길이 128, 64, 32, 16, 8, 4, 2 인 블록의 개수가 각각 대략 $\frac{2^{14}}{128\times 2}=64$개 정도 있을 것이다. 이제 두 번째 패스에서의 최적의 블록 길이를 생각하자. 첫 번째 패스에서 길이 $2^{K'}$ 블록에 있었던 비트들끼리 그룹화한 그룹을 $B_{K'}$이라 하면 그 그룹에서 두 번째 패스 최적의 블록 길이는 다음과 같다[2].

$$k_2=\min(2^{[\log_2(4/p_{bit})]},|B_{K'}|/2)$$

(p_{bit}: 길이 $2^{K'}$ 블록에 있었던 비트가 두 번째 패스에서 오류가 될 확률)

기본적으로 $2^{[\log_2(4/p_{bit})]}$가 최적의 길이지만 그룹 크기의 반을 넘을 경우에는 그룹 크기의 반을 블록 길이로 하는 것이 낫다는 의미이고 그룹 크기의 반을 블록 길이로 하는 경우 다소 성능 저하가 발생할 수 있다. 두 번째 패스에서 예상 그룹 크기 및 최적 블록 길이를 표로 정리하면 다음과 같다.

첫 번째 패스에 속했던 블록의 길이	두 번째 패스에서 속한 그룹의 예상 크기	최적 블록 길이 $2^{[\log_2(4/p_{bit})]}$
128	8192	512
64	4096	512
32	2048	1024
16	1024	2048
8	512	4096
4	256	16384
2	128	32768

결과를 보면 첫 번째 패스에서 길이가 16, 8, 4, 2인 블록에 있었던 비트들은 두 번째 패스에서 그룹 크기에 비해 최적의 블록 길이가 훨씬 크기 때문에 최적 블록 길이를 적용시킬 수 없었고 이는 성능 저하로 이어진다. 그러나 이 비트들만 예외로 다른 그룹임에도 블록을 형성한다면 최적 블록 길이에 가깝게 되어 성능이 좋아질 수 있다.

첫 번째 패스에 속했던 블록의 길이	본 논문 아이디어를 적용한 두 번째 패스에서 속한 그룹의 예상 크기	최적 블록 길이 $2^{[\log_2(4/p_{bit})]}$
128	8192	512
64	4096	512
32	2048	1024
16, 8, 4, 2	2047	2048, 4096, 16384, 32768

일반적으로 BER p가 큰 경우 $2^{[\log_2(4/p_{bit})]}$값이 작게 나오기 때문에 문제 없지만 작은 경우에는 $2^{[\log_2(4/p_{bit})]}$값이 크게 나와 성능 저하가 발생한다. 따라서 기본적으로는 조건부확률이 같은 비트들 내에서 블록을 형성하지만 작은 p값에 대해서는 조건부확률 값이 다른 비트들도 같이 블록을 형성하는 것을 허용함으로써 프로토콜의 성능을 높일 수 있다.

Ⅴ. 결 론

본 논문에서는 기존에 조건부확률이 같은 비트들만 그룹화하여 Cascade 프로토콜의 성능을 높이는 방법의 경우, 작은 BER에서는 조건부확률이 같은 비트들끼리 그룹화함으로써 오히려 성능 저하가 발생할 수 있음을 지적하였다. BER이 낮은 경우 조건부확률이 다른 비트들도 그룹화할 수 있도록 이 기법을 일반화하였고, 이를 통해 기존의 Cascade 프로토콜의 성능을 더 높일 수 있음을 보였다.

참 고 문 헌

[1] G. Brassard and L. Salvail, "Secret-key reconciliation by public discussion," in *Proc. EUROCRYPT '93, Lecture Notes in Computer Science*, vol. 765, 1994, pp. 410-423.

[2] C. Pacher, Philipp Grabenweger, Jesus Martinez-Mateo, and Vicente Martin, "An information reconciliation protocol for secret-key agreement with small leakage," in *Proc. IEEE ISIT 2015*, June 2015, pp. 730-734.