부호 컴퓨팅 환경에서의 CRC 를 연접한 극 부호의 적용

김재화, 노종선
서울대학교

woghk9307@ccl.snu.ac.kr, jsno@snu.ac.kr

Application of CRC-aided Polar Codes on coded computation

Jaewha Kim, Jong-Seon No
Seoul National Univ.

요 약

본 논문에서는 부호를 활용한 컴퓨팅 상황에서 CRC 를 연접한 극 부호를 활용했을 때의 성능을 shifted exponential 분포를 활용하여 구해보았다. 대수적 부호를 통한 분석과는 달리 극 부호의 경우 (n,k) 부호에 대하여 성공 여부를 BLER 을 통해 구하였고 컴퓨팅 연산 시간에 따른 실패 확률에 대한 그래프를 시뮬레이션을 통해 구하였다. $(64,32)$ CRC 가 연접된 부호에서 목표 블록 오류율 10^{-3}을 만족하기 위해서는 적어도 50 개의 노드로부터 연산 결과를 받으면 된다는 사실을 확인하였고 shifted exponential distribution 모델에서 코드를 사용하지 않는 경우보다 연산 시간이 단축된다는 것을 확인하였다. 본 논문에서는 대수적 부호와 더불어 노드의 수가 많을 때 부호 및 복호의 복잡도가 비교적 낮은 극 부호도 연산 시간과 정확도를 고려 했을 때 부호 컴퓨팅에 사용될 수 있음을 확인할 수 있다.

Ⅰ. 서 론

특정 연산을 여러 노드들이 나누어서 하는 부호 컴퓨팅 (Coded computing) 환경에서 MDS(Maximum Distance Separable) 부호를 사용할 경우 연산 시간이 감소한다는 연구결과가 알려져 있다. MDS 부호를 활용하여 정해진 시간 내에 연산을 완료하지 않은 노드를 소실(erasure)로 생각하고 연산이 완료된 노드와 MDF 복호를 활용하여 전체 계산을 계산하게 된다. 본 논문의 가정은 우선 노드 들간의 통신은 고려하지 않고 노드들의 결과를 한번에 모은 후 적절할 부호 및 복호 과정을 거쳐 결과를 얻게 된다. 하지만, 부호 및 복호 복잡도를 고려한다면 노드가 많은 경우에는 MDS 부호의 경우 전체 컴퓨팅 시간에 영향을 미치게 되므로 연산 시간의 감소에 한계가 있다. 또한, 주어진 노드의 수와 연산을 나누게 되는 수가 정해져 있지 않으므로 임의의 (n,k) 파라미터에 대하여 부호 및 복호 과정이 정해져 있지 않기에 본 논문에서는 극 부호를 활용한 분석이 이루어져 있다.

본 논문에서는 섀넌의 채널 용량에 근접하면서 부호 및 복호의 복잡도가 낮은 극 부호(Polar Code)를 분산 컴퓨팅에 적용했을 때의 확률적 성공률을 분석하였고 기존의 MDS 부호를 사용했을 때와 비교하여 연산 시간이 얼마나 차이가 나는지 비교하였다.

컴퓨팅 할 수 있는 노드의 수가 n 개라 가정해보자. 우선 부호를 사용하지 않은 경우에는 해야 할 연산을 n등분하여 각각의 노드에 나누어 주면 된다. 그렇게 될 경우 모든 노드가 연산이 끝난 후 각각의 연산 값들을 합치게 되면 전체 연산이 성공하게 되고 그렇지 않으면 실패하게 된다.

MDS 부호의 경우 전체 연산을 k 개로 나누었다고 가정해보자. 그렇게 될 경우, 그 후 (n,k) MDS 부호가 있다고 가정할 경우 MDS 부호를 이용하여 n 개의 연산으로 만든 후 각 노드에 분배한다. 그렇게 될 경우, 전체 노드 중 k 개의 노드만 연산을 완료한다면 MDS 복호를 통해 전체 연산이 완료된다.

Ⅱ. 본론

본 논문에서는 전체 노드 수가 $n=64(\text{bit})$ 이고 CRC 는 $6(\text{bit}), 8(\text{bit})$에 대해 각각 시뮬레이션 해보았다. 전체적인 시스템은 다음 Figure1 과 같다.

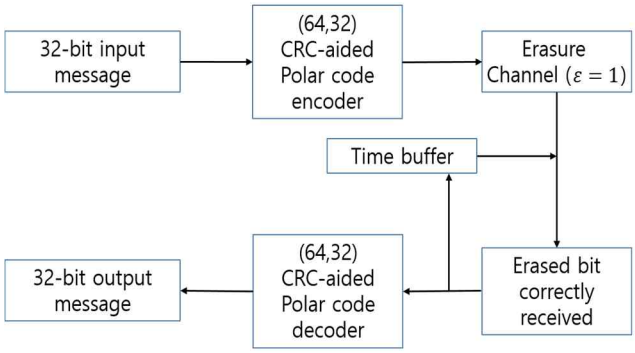

Figure 1. 극 부호를 이용한 부호 컴퓨팅 시스템 모델

우선 전체 연산을 1 개의 노드가 연산을 했다고 가정하고 그 연산이 완료되는 시간에 대한 확률 변수를 T_0 라 하고 t 초 안에 주어진 연산을 완료하는 확률인 누적분포함수 $F(t) = Pr(T_0 \le t)$ 를 아래와 같이 shifted exponential distribution 을 이용하였다.

$$F(t) = 1 - e^{-(t-1)} \; for \; t \ge 1$$

입력 비트 수(나누어서 일하는 노드 수)가 $k = 32$ 이므로 각각의 노드는 F(kt)의 분포를 가지게 되고 모든 노드들의 연산 능력이 동일하다는 전제하에 같은 분포를 독립적으로 가진다. 이 때, F(kt)의 확률분포에서 64개의 임의의 시간을 뽑아낸 후 그 시간에 따라 시간 버퍼가 작용하여 각 노드로부터 연산이 도착한다고 가정하고 소실된 비트를 복원하게 된다. 모든 노드들의 연산 능력이 동일하므로 극 부호의 복호 과정을 실행할 때마다 소실된 비트의 인덱스는 임의로 바뀌가면서 실행하였다. 복원되는 비트는 처음에 $t = 0$일 때는 모든 노드가 연산을 시작하지 않았기에 $n = 64$ 비트 모두 채널에서 소실된다고 가정한다. 그 후 순차적으로 시간 버퍼에 따라 한 비트씩 복원하면서 그때마다 극 부호 복호화 과정을 거쳐 블록 오류율 (Block Error Rate: BLER)을 구하게 된다.

극 부호의 부호 및 복호과정에서 리스트 크기는 16 으로 고정하여서 시뮬레이션이 이루어졌고 순환 중복 검사 (Cyclic Redundancy Check) 비트를 연접하여 복호를 하였다. 이때 CRC 비트는 6,8 비트로 나누어서 시뮬레이션을 하였다. 64 비트 중 임의의 인덱스로 소실을 하게 되므로 모든 조합에 대하여 시뮬레이션 하기에는 복잡도가 기하급수적으로 증가하기에 주어진 (n, k) 극 부호와 주어진 소실된 비트수에 대하여 각각 10,0000 번씩 임의의 소실 조합에 대하여 BLER 을 계산하였다.

CRC-6 비트와 CRC-8 비트에 대해서 각각 Matlab 시뮬레이션이 이루어졌고 결과는 아래 Figure 2 와 Figure 3 과 같다. 시뮬레이션에서의 x 축은 시간을 나타내고 각각의 노드의 연산 결과가 도착한 순간의 BLER 들을 그래프로 나타낸 것이다.

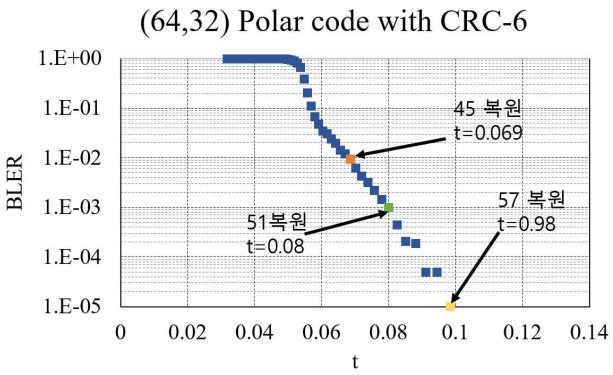

Figure 2 - CRC 6 비트를 연접한 연산 성공률

우선 CRC-6 비트의 경우 목표 블록 오류율 10^{-3} 를 만족하기 위해서는 51 비트가 복원되어야 하고 그때의 시간은 $t = 0.08$ (본 논문의 경우 shifted exponential distribution 시간의 변수이므로 단위가 없다)이고 CRC-8 비트의 경우 목표 블록 오류율 10^{-3} 를 만족하기 위해서는 전체 소실된 64 비트 중 50 비트가 복원될 경우(50 개의 노드에서 연산을 받은 것과 동일하다) 즉, 14비트만 소실되는 경우이고 그때의 시간은 $t = 0.078$이 소요된다.

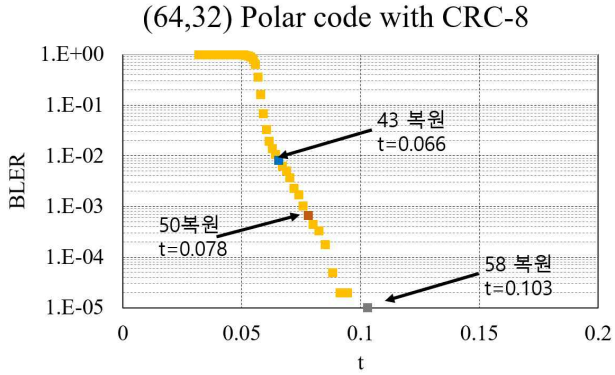

Figure 3 - CRC 8 비트를 연접한 연산 성공률

부호를 사용하지 않은 경우에 대해서는 $n = 64$ 비트로 동일하게 적용하되 각각의 노드가 전체 연산을 n 등분하여 하기 때문에 F(nt)의 확률분포를 따르게 되고 n개의 노드의 연산 결과를 모두 받아야 하기 때문에 극 부호의 경우와 동일하게 F(nt) 의 분포에 따른 64 개의 임의의 시간을 뽑은 후 가장 늦게 도착한 노드에 대한 시간이 곧 연산이 성공적으로 끝나게 되는 시간이라고 볼 수 있다. 본 논문에서는 임의의 시간 데이터를 1000 번 추출하여 평균을 취하였고 $n = 64$의 경우 $t = 0.09$ 만큼 소요되었다.

Ⅲ. 결론

본 논문에서는 CRC 를 연접한 극 부호를 부호 컴퓨팅에 적용하여 각 노드들의 연산 능력이 shifted exponential distribution 이라고 가정한 상황에서 연산의 지연 시간에 따른 연산 성공확률을 그래프로 그려보았다. 그 결과 부호를 사용하지 않는 경우보다 극 부호를 사용하는 경우 연산 시간이 단축되었고 MDS 부호 같은 대수적 부호에 비해 부호 및 복호 과정의 복잡도가 낮아 효율적이다. 또한 CRC 의 연접의 경우 목표 블록 오류율에 따라 연산 시간의 성능이 다르게 나왔다.

.

참 고 문 헌

[1] K. Lee, et al. "Speeding up distributed machine learning using codes." Information Theory (ISIT), 2016 IEEE International Symposium on. IEEE, 2016..

[2] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos and K. Ramchandran, "Speeding Up Distributed Machine Learning Using Codes," in IEEE Transactions on Information Theory, vol. PP, no. 99, pp. 1-1.

[2] Niu, Kai, and Kai Chen. "CRC-aided decoding of polar codes." IEEE Communications Letters 16.10 (2012): 1668-1671.