스트림 환경에서 높은 균일신뢰도를 지원하는 가변 크기 샘플링

김하진*, 이상훈*, 길명선*, 문양세*
*강원대학교 컴퓨터과학과
e-mail : {hajinkim, sanghun, gils, ysmoon}@kangwon.ac.kr

Variable Size Sampling to Support Uniformity Confidence in Stream Environments

Hajin Kim*, Sanghun Lee*, Myeong-Seon Gil, Yang-Sae Moon*
*Dept. of Computer Science, Kangwon National University

요 약

본 논문에서는 스트림 환경에서의 샘플링 기법 중 랜덤 샘플링과 유사한 특성을 갖는 KSample 의 균일신뢰도 향상에 초점을 맞춘다. 이를 위해, 먼저 KSample 의 균일신뢰도 문제점을 분석하여 KSample 의 균일신뢰도가 초기에 많이 감소하는 현상을 **초기 균일신뢰도 저하** 문제로, KSample 의 균일신뢰도가 지속적으로 감소하는 현상을 **지속 균일신뢰도 저하** 문제라 정의한다. 그리고 초기 균일신뢰도 저하 문제를 발생시키는 성질을 과거 샘플 불변으로, 지속 균일신뢰도 저하 문제를 발생시키는 성질을 샘플 추출 범위 증가로 정의하고 이를 해결하여 균일신뢰도를 향상시킨 UC KSample 을 제안한다. 실험 결과, 샘플링 비율 p=0.01, 균일신뢰도 하한 ε=0.7 일 때, UC KSample 의 균일신뢰도가 기존 KSample 보다 약 2.2 배 증가하였고, 균일신뢰도는 항상 하한 이상으로 유지되었다. 본 연구는 스트림 환경에서 중요한 척도인 균일신뢰도를 KSample 에 결합시킨 최초의 시도로서, 샘플링 비율을 유지하며 동적으로 샘플링하는 KSample 의 장점은 유지하면서도 균일신뢰도를 증가시킨 우수한 연구라 사료된다.

1. 서론

데이터 스트림은 끊임없이 생성되는 연속된 형태의 데이터를 의미한다[1, 2]. 데이터 스트림이 대용량인 경우, 이를 실시간 처리하는 것은 매우 많은 비용을 발생시킨다[3, 4]. 따라서 데이터 스트림의 특징과 패턴을 잘 반영하는 샘플을 추출하여 사용하는 것이 효과적이다[5, 6]. 이러한 스트림 환경에서 사용되는 샘플링 기법은 샘플 양을 지정하는 방식에 따라 (1) 샘플 크기 고정 방법과 (2) 샘플링 비율 고정 방법으로 나뉜다[3]. 샘플 크기 고정 방법은 스트림의 양에 관계없이 샘플 크기를 고정하는 방법이고, 샘플링 비율 고정 방법은 스트림 양에 따라 샘플 크기를 변화시켜 샘플링 비율을 고정하는 방법이다. 스트림 환경에서 데이터가 무한히 생성된다고 가정할 때, 샘플 크기를 고정하는 방법보다는 입력되는 스트림 양에 따라 샘플의 양도 변하는 비율 고정 방법이 효과적이다. 본 논문에서는 스트림 환경에서의 샘플링 비율 고정 방법 중 랜덤 샘플링 특성을 갖는 KSample 의 균일신뢰도 향상에 초점을 맞춘다.

* 이 논문은 2017 년도 정부(미래창조과학부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임(No. R7117-17-0214, 데이터 스트림 정제를 위한 지능형 샘플링 및 필터링 기술 개발, No.2017-0-00253, 국제표준 기반 오픈 데이터 유통 플랫폼 확장 기술 개발)

KSample[7]은 샘플 크기를 동적으로 증가시켜 데이터 스트림에 대한 샘플링 비율을 유지하는 랜덤 샘플링 방법이며, 균일신뢰도(UC: uniformity confidence)[8]는 샘플링 알고리즘이 얼마나 균일하게 샘플을 생성하는지를 나타내는 척도이다. 스트림은 끊임없이 생성되지만 시스템의 메모리는 제한되어 있기 때문에 모든 데이터를 고려한 샘플링은 불가능하다. 따라서 샘플의 신뢰도 측정을 위해 샘플링 알고리즘이 얼마나 균일하게 샘플을 생성하는지 판단하는 것이 중요하다. 그러나, 일정 크기의 스트림 단위로 샘플을 추출하는 KSample 의 경우, 이미 샘플로 추출된 데이터는 변하지 않는 특징 때문에 균일신뢰도가 매우 낮아지는 문제가 있다. 또한, 데이터 스트림이 끊임없이 유입됨에 따라 "통계적으로 생성 가능한 샘플 경우의 수" 증가량보다 "KSample 에서 생성 가능한 샘플 경우의 수" 증가량이 더 적어 두 경우의 수의 비율이 점점 감소하고, 결국 균일신뢰도가 지속적으로 감소하는 문제가 있다. 본 논문에서는 이를 **초기 균일신뢰도 저하** 문제와 **지속 균일신뢰도 저하** 문제라 부르고, 이의 해결책을 제시하여 KSample 의 균일신뢰도를 높인 *UC KSample* 을 제안한다.

먼저 KSample 에서 균일신뢰도가 낮아지는 원인을 분석한다. KSample 의 두 가지 문제에 대한 자세한 설명 및 해결 방안은 다음과 같다. 먼저, 초기 균일신

뢰도 저하는 이미 슬롯에 저장된 원소는 2 차 저장소로 보내지기 때문에 변경이 불가능한 과거 샘플 불변성질 때문에 발생한다. 이를 해결하기 위해, UC KSample 에서는 이미 선택된 샘플이라도 변경 가능하도록 하여 샘플로 추출 가능한 경우의 수를 증가시킨다. 균일신뢰도는 "통계적으로 생성 가능한 모든 샘플의 경우의 수"와 "현재 시점에서 특정 알고리즘을 통해 생성 가능한 샘플의 경우의 수"의 비율이기 때문에, 생성 가능한 샘플의 경우의 수를 증가시키면 균일신뢰도를 향상시킬 수 있다.

다음으로, 지속 균일신뢰도 저하는 입력 스트림이 증가하면 샘플로 추출될 스트림의 범위가 실제 고려할 수 있는 스트림의 범위보다 더 많이 증가하는 샘플 추출 범위 증가 때문에 발생한다. 이를 해결하기 위해, UC KSample 에서는 균일신뢰도 기반 윈도우 개념을 제시한다. 즉, 샘플링을 진행할 때, 샘플링 알고리즘의 균일신뢰도가 지정한 균일신뢰도의 하한보다 낮아지기 전에 새로운 윈도우를 시작하여 균일신뢰도가 항상 하한 이상으로 유지되도록 하는 방법이다. 균일신뢰도 실험 결과, 샘플링 비율 p=0.01, 균일신뢰도 하한 ε=0.7 일 때, UC KSample 의 균일신뢰도가 기존 KSample 보다 약 2.2 배 증가하였고, UC KSample 의 균일신뢰도는 항상 하한보다 높게 유지되었다.

2. 관련 연구

균일신뢰도는 특정 샘플링 알고리즘이 얼마나 많은 경우의 수를 고려하여 샘플을 생성하는지를 나타내는 성능 지표이다. 균일신뢰도는 식 (1)과 같이 "통계적으로 생성 가능한 모든 샘플의 경우의 수"와 "특정 알고리즘을 통해 생성 가능한 샘플의 경우의 수"의 비율로 계산된다.

$$UC = \frac{the\ number\ of\ different\ samples\ of\ the\ same\ size\ possible\ with\ the\ algorithm}{the\ number\ of\ different\ samples\ of\ the\ same\ size\ possible\ statistically} \times 100 \quad (1)$$

예를 들어, 10 개의 데이터에서 크기 3 인 샘플을 무작위로 추출할 때, 10 개의 모든 데이터를 고려하는 샘플링의 균일신뢰도는 100%(=((10¦3))/((10¦3))×100)이다. 반면, 10 개의 데이터 중 앞의 3 개 데이터를 고려하지 못한다면, 즉 7 개만 샘플 추출 범위에 포함시킨다면, 이 샘플링의 균일신뢰도는 29.17%(=((7¦3))/((10¦3))×100)가 된다. 스트림 환경과 같이 모든 데이터가 샘플 추출 범위가 될 수 없는 경우, 균일신뢰도의 향상을 위해 샘플링 알고리즘이 얼마나 많은 데이터를 고려하는지가 중요한 성능 요소이다.

KSample 은 스트림 환경에서 동적으로 샘플 크기를 증가시켜 입력 스트림에 대한 샘플링 비율을 일정하게 유지하는 랜덤 샘플링 방법이다[7]. KSample 은 샘플링 비율 p(∈[0,1])를 사용자로부터 입력 받아 샘플 크기가 항상 데이터 스트림의 P%(=p×100) 이상으로 유지되도록 샘플링한다. 즉, 끊임없이 유입되는 데이터 스트림의 P%를 샘플로 유지하기 위해 샘플 크기를 동적으로 증가시킨다. 예를 들어, p=0.1 로 주어졌을 때, KSample 은 스트림의 유입에 따라 동적으로 샘플 크기를 증가시켜 적어도 입력 데이터 스트림의 10%를 샘플로 유지한다.

KSample 은 샘플링 비율에 따라 동적으로 샘플 크기가 증가하기 때문에 샘플 크기를 사전에 정의할 필요가 없고, 이미 샘플로 추출한 스트림 원소를 바꾸지 않기 때문에 메모리를 효율적으로 사용할 수 있다는 장점이 있다. 하지만 여기서 주목할 점은 KSample 의 장점인 메모리 효율성이 균일신뢰도를 크게 낮추는 원인이 된다는 것이다. 즉, 이전 슬롯에 샘플로 추출된 데이터가 다음 슬롯의 샘플링 범위에 포함되지 않기 때문에 샘플의 경우의 수가 줄어들고 균일신뢰도 또한 감소하게 된다. 따라서, 본 논문에서는 KSample 의 장점인 샘플 크기의 사전 결정 불필요와 메모리 효율성을 유지하면서도 균일신뢰도를 향상시키는 UC KSample 을 제안한다.

3. UC KSample

3.1. 문제 정의 및 원인 분석

KSample 은 각 슬롯에 추출될 수 있는 데이터 스트림의 범위가 제한되어 있고, 이미 샘플로 추출된 데이터는 변경되지 않는 특징 때문에 균일신뢰도가 매우 낮은 문제가 있다. 또한, 데이터 스트림이 끊임없이 유입됨에 따라 "통계적으로 생성 가능한 샘플 경우의 수" 증가량보다 "KSample 에서 생성 가능한 샘플 경우의 수" 증가량이 더 작아 두 경우의 수의 비율이 점점 감소하고, 결국 균일신뢰도가 지속적으로 감소하는 문제가 있다. 본 논문에서는 KSample 의 균일신뢰도가 초기에 많이 감소하는 현상을 **초기 균일신뢰도 저하** 문제, KSample 의 균일신뢰도가 지속적으로 감소하는 현상을 **지속 균일신뢰도 저하** 문제라 부른다.

KSample 에서 초기에 균일신뢰도가 현저하게 낮아지는 초기 균일신뢰도 저하 문제는 샘플로 선택된 데이터가 2 차 저장소로 이동한 후 변경되지 않기 때문에 발생한다. 샘플 슬롯의 추출 범위에 포함된 스트림 데이터들은 해당 슬롯에 샘플로 선택되기 위해 경쟁한다. 하지만 특정 슬롯으로 선택될 수 있는 데이터 스트림의 범위가 끝나고 새로운 슬롯이 생성되면, 이전 슬롯에 샘플로 선택된 데이터는 2 차 저장소로 이동하여 변경이 어려워진다. 이와 같이 특정 슬롯에 샘플로 선택된 데이터가 변경될 수 없기 때문에 생성될 수 있는 샘플의 경우의 수가 감소하게 된다. 특히 KSample 은 샘플 크기가 2 이상으로 증가하는 샘플링 초기에 생성 가능한 샘플의 경우의 수가 현저히 감소하여 균일신뢰도가 크게 감소한다.

다음으로, KSample 에서 입력 스트림의 증가에 따라 균일신뢰도가 지속적으로 감소하는 지속 균일신뢰도 저하는 통계적으로 생성 가능한 샘플 경우의 수의 증가량보다 KSample 에서 생성 가능한 샘플 경우의 수의 증가량이 더 작기 때문에 발생한다. 입력 스트림이 증가하면 모집단도 함께 증가하는데, 메모리 손실로 인해 모든 입력 스트림을 샘플 추출 범위로 할 수 없다. 따라서, 시간이 지남에 따라 통계적으로 생성 가능한 샘플 경우의 수와 KSample 로 생성 가능한 샘플 경우의 수의 비율이 지속적으로 감소하게 된다. 본 논문에서는 초기 균일신뢰도 저하와 지속 균일신뢰도 저하의 원인을 다음과 같이 구체적으로 기술한다.

- **과거 샘플 불변**: 특정 샘플 슬롯으로 선택될 수 있는 스트림 데이터의 범위가 끝나면 샘플로 선택된

데이터가 변경되지 않아 균일신뢰도가 저하되는 성질이다.

- **샘플 추출 범위 증가**: 통계적으로 생성 가능한 샘플 경우의 수의 증가량보다 KSample 에서 생성 가능한 샘플 경우의 수의 증가량이 더 작기 때문에 두 경우의 수의 비율이 점점 감소하는 성질이다.

3.2. UC KSample 의 요구사항 정의

과거 샘플 불변 성질의 완화를 위한 요구사항은 과거 샘플 변경인데, 이는 샘플의 특정 원소로 추출될 수 있는 데이터 스트림의 범위가 끝나도 이미 샘플로 선택된 데이터가 변경 가능하도록 하는 것이다. 이미 샘플로 선택된 데이터가 다른 데이터로 변경될 수 있다는 것은 추출 가능한 샘플의 경우의 수가 증가하는 것이므로, 과거 샘플 변경을 허용하면 균일신뢰도를 향상시킬 수 있다. 그림 1 은 p=0.3 이고 스트림 데이터가 여섯 개 유입되었을 때 KSample 과 UC KSample 의 동작 절차를 나타낸다. KSample 에서는 그림 1(a)와 같이 두 번째 슬롯이 생성되면 첫 번째 슬롯에 이미 선택된 원소 1 은 2 차 저장소로 이동하여 변경되지 않는다. 그러나, UC KSample 에서는 그림 1(b)와 같이 첫 번째 슬롯에 선택된 원소 1 이 2 차 저장소로 이동하지 않고 일정 요건이 충족될 때까지 메인 메모리에 유지되어 다른 스트림 원소로 변경되는 것을 허용한다.

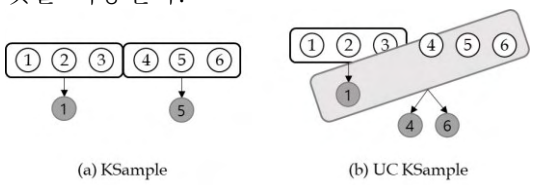


그림 1. p=0.3 이고 스트림 데이터가 여섯 개 유입됐을 때, KSample 과 UC KSample 의 동작 절차.

지속 균일신뢰도 저하를 야기하는 샘플 추출 범위 증가를 해결하기 위한 요구사항은 **균일신뢰도 기반 윈도우 사용**이다[1]. 균일신뢰도 기반 윈도우 사용은 샘플의 균일신뢰도가 주어진 하한(ε) 이상으로 항상 유지되는 윈도우 크기를 계산하고, 입력 스트림을 윈도우 단위로 나누어 샘플링을 수행하는 것이다. 샘플링 비율 p 와 균일신뢰도 하한 ε 를 만족하는 최대 윈도우 크기 계산 방법은 식 (2)와 같다.

$$\varepsilon \le \frac{\sum_{x=\max\{0,(\lfloor kp \rfloor+1)-m\}}^{\lfloor kp \rfloor} \binom{k}{x}\binom{m}{\lfloor kp \rfloor+1-x}}{\binom{k+m}{\lfloor kp \rfloor+1}} \times 100 \qquad (2)$$

식 (2)에서 ε 은 균일신뢰도의 하한, p 는 샘플링 비율, k 는 현재까지 들어온 스트림 크기, m 은 샘플 크기가 1 증가할 때 들어올 수 있는 최대 입력 스트림 크기이다.

최대 윈도우 크기 계산을 위한 식 (2)를 자세히 설명하면 다음과 같다. 현재까지 유입된 입력 스트림의 크기가 k 개라면 샘플링 비율 p 를 유지하며 $\lfloor kp \rfloor$개를 샘플로 추출하고, 이후 입력 스트림이 m 개 더 유입되어 샘플 크기가 1 증가할 때, UC KSample 의 동작 절차는 그림 2 와 같다. 그림을 보면, $k+m$ 개의 전체 스트림 데이터에서 $\lfloor kp \rfloor$+1 개의 데이터를 샘플로 추출하기 때문에 통계적으로 생성 가능한 샘플의 경우의 수는 $((k+m)\,|\,(\lfloor kp \rfloor+1))$이다. 이미 샘플로 추출한 $\lfloor kp \rfloor$개의 데이터 중 x 개를 변경할 때, 만약 새로 유입된 스트림 데이터 개수 m 이 $\lfloor kp \rfloor$+1 보다 작다면 x 는 $\lfloor kp \rfloor+1-m$ 이하여야 한다. 따라서 x 는 max{0, ($\lfloor kp \rfloor$+1)-m} 이상 r 이하의 범위를 갖고, 이를 정리하면, UC KSample 을 통해 생성 가능한 샘플의 경우의 수는 $\sum_{x=\max\{0,\lfloor kp \rfloor+1-m\}}^{\lfloor kp \rfloor} \binom{k}{x}\binom{m}{\lfloor kp \rfloor+1-x}$ 가 된다. 샘플 크기가 증가할 때 UC KSample 의 균일신뢰도는 항상 하한보다 커야 하므로 식 (2)를 만족하는 최대 k 와 m 을 구한 후, k 와 m 을 더한 값을 윈도우 크기로 설정한다.

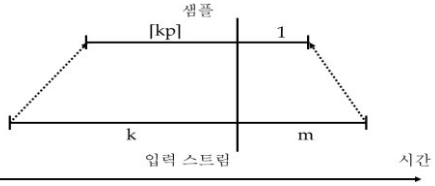


그림 2. 샘플 크기가 1 증가할 때 UC KSample 의 동작 절차.

표 1 은 지금까지 설명한 KSample 의 두 가지 문제와 그 원인, 그리고 이를 해결하기 위한 요구사항을 정리한 것이다.

표 1. KSample 의 문제 원인, 해결을 위한 요구사항.

요구사항	설명	원인	관련문제
과거 샘플 변경	이미 추출된 샘플의 원소라 하더라도 새로 유입되는 데이터 스트림에 의해 대체될 수 있도록 허용	과거 샘플 불변	초기 균일신뢰도 저하
균일신뢰도 기반 윈도우사용	균일신뢰도 기반 윈도우를 사용하여 균일신뢰도를 항상 하한 이상으로 유지	샘플 범위 증가	지속 균일신뢰도 저하

3.3. UC KSample 알고리즘

그림 3 은 제안하는 UC KSample 알고리즘이다. UC KSample 은 샘플링 비율 p, 스트림 *stream*, 균일신뢰도의 하한 ε 을 입력으로 받고, 비어있는 샘플 *reservoir* 로 샘플링을 시작한다. 먼저, 균일신뢰도의 하한을 만족하는 최대 윈도우 크기 w 를 식 (2)로 계산한다(라인 3). 만약 현재까지 유입된 스트림의 길이 *sLength* 가 1 이거나 현재 윈도우에 들어온 스트림 길이 *wLength* 가 윈도우 크기보다 크다면 현재까지 생성된 샘플을 2 차 저장소에 저장하고, 새로운 윈도우를 생성하여 샘플링을 다시 시작한다(라인 7-9). 데이터 스트림 원소가 유입되면 각 원소에 대해 난수를 생성한다(라인 10). 만약 현재 샘플 크기가 (p×*wLength*)보다 작다면 샘플 크기를 하나 증가시키고 현재 들어온 데이터를 난수와 함께 샘플에 추가한다(라인 11-13). 그러나 현재 샘플 크기가 (p×*wLength*)보다 크지 않다면, 현재 원소가 기존 샘플에 들어갈 수 있는지 검사하기 위해 샘플에서 제일 작은 난수 값을 갖는 원소와 현재 들어온 원소의 난수 값을 비교한다(라인 16). 샘

[1] 균일신뢰도의 하한을 0 으로 설정하면 윈도우 크기($k+m$)는 식 (2)에 따라 무한대가 된다. ($k+m$)이 무한대라면 UC KSample 을 통해 생성된 샘플의 균일신뢰도는 샘플 범위 확장과 과거 샘플 변경을 통해 기존 KSample 보다는 증가하지만 지속적으로는 감소하게 된다. 본 논문에서는 UC KSample 의 균일신뢰도의 하한을 없애 윈도우 크기를 무한대로 설정하여 샘플링하는 기법을 Naïve UC KSample[9]이라 하고, 실험에서 성능 비교를 위해 사용한다.

플에 포함된 가장 작은 난수 값보다 현재 데이터의 난수 값이 크다면 제일 작은 난수 값을 갖는 원소를 제거하고 현재 데이터를 샘플에 삽입한다(라인 16-17). 이와 같은 과정을 스트림 입력이 끝나거나 사용자가 샘플링을 멈출 때까지 반복한다(라인 4-19).

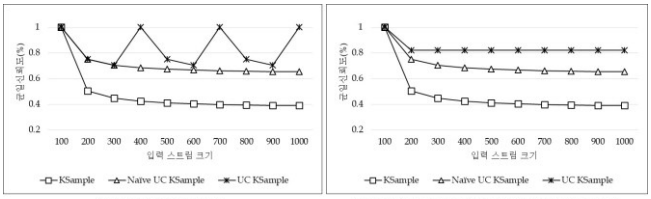
```
Input: p, sampling ratio (p ∈ [0,1])
Input: stream, data stream of undefined length
Input: ε, threshold of UC
Output: reservoir, priority queue
1.   sLength ← 0
2.   wLength ← 0
3.   w ← maximum window size that causes UC to exceed ε
4.   while stream != EOF do
5.       sLength++
6.       wLength++
7.       if sLength=1 or wLength > w then
8.           reservoir.flush()
9.           wLength ← 1
10.      rand = Random(0, 1)
11.      if reservoir.size() < (p × windowRound) then
12.          reservoir.newSlot()
13.          reservoir.insert(stream[sLength],rand)
14.      else
15.          minimum = reservoir.delete()
16.          if rand > minimum.rand then
17.              reservoir.insert(stream[sLength], rand)
18.          else
19.              reservoir.insert(minimum)
20.  return reserevoir
```

그림 3. UC KSample 알고리즘.

4. 실험 평가

본 절에서는 KSample 과 Naïve UC KSample, UC KSample 의 균일신뢰도를 비교 · 평가한다. 실험에서는 UC KSample 의 샘플링 비율 p=0.01, 균일신뢰도 하한 ε=0.7 일 때 윈도우 크기를 계산하고, 이 때의 균일신뢰도를 측정하여 나머지 두 알고리즘과 비교한다. 실험에 사용한 하드웨어 플랫폼은 Intel i7 3.60GHz cpu, 8.0GB RAM 을 장착한 HP 워크스테이션이며, 소프트웨어 플랫폼은 Ubuntu 16.04 LTS 운영체제이다.

그림 4 는 p=0.01, ε=0.7 일 때 KSample, Naïve UC KSample, UC KSample 의 균일신뢰도를 비교한 것이다. 먼저, 그림 4(a)는 세 알고리즘의 균일신뢰도 변화를 그래프로 나타낸 결과이다. 비교 결과, UC KSample 과 Naïve UC KSample 의 균일신뢰도가 기존 KSample 보다 약 2.2 배, 1.7 배 증가한 것으로 나타났다. 또한, UC KSample 은 균일신뢰도가 하한인 70%보다 낮아지기 전에 새로운 윈도우를 생성하여 균일신뢰도를 항상 하한 이상으로 유지하는 것을 확인할 수 있다. 그림 4(a)의 UC KSample 에서 급한 오르내림(fluctuation)이 발생하는 이유가 이러한 새 윈도우 생성에 따른 하한 유지에서 기인한다. 다음으로, 그림 4(b)는 그림 4(a)에서 UC KSample 의 오르내림 현상을 구간 평균으로 완화해 나타낸 것이다. 즉, KSample 과 Naïve UC KSample 의 균일신뢰도를 그대로 쓰되, UC KSample 균일신뢰도의 경우 윈도우 크기 기준 구간 평균을 나타낸 것이다. 그림을 보면, UC KSample 균일신뢰도의 구간 평균이 항상 KSample 과 Naïve UC KSample 의 균일신뢰도보다 높고, 하한인 70%보다 항시 높은 것을 알 수 있다.



(a) 세 방법의 균일신뢰도 비교. (b) UC KSample의 윈도우 기반 구간 평균의 균일신뢰도 비교.

그림 4. p=0.01, ε=0.7 일 때, KSample, Naïve UC KSample, UC KSample 균일신뢰도 비교 결과.

5. 결론

본 논문에서는 스트림 환경에서 KSample 의 균일신뢰도를 높인 UC KSample 알고리즘을 제안하였다. 먼저, KSample 의 초기 균일신뢰도 저하와 지속 균일신뢰도 저하의 두 가지 문제를 도출하였다. 그리고 초기 균일신뢰도 저하 문제를 발생시키는 성질을 과거 샘플 불변으로, 지속 균일신뢰도 저하 문제를 발생시키는 성질을 샘플 추출 범위 증가로 정의하였다. 또한 이를 완화시키기 위한 요구사항으로 과거 샘플 변경, 균일신뢰도 기반 윈도우 사용을 제시하고, 이를 적용한 새로운 KSample 알고리즘인 UC KSample 을 제안하였다.

균일신뢰도 실험 결과, 샘플링 비율 p=0.01, 균일신뢰도 하한 ε=0.7 일 때 UC KSample 의 균일신뢰도가 기존 KSample 보다 약 2.2 배 증가하였고 항상 하한 이상으로 유지됨을 확인하였다. 본 연구는 스트림 환경에서 중요한 척도인 균일신뢰도를 KSample 에 결합시킨 최초의 시도로서, 메모리를 효율적으로 사용하고, 샘플링 비율에 맞춰 동적으로 샘플링하는 KSample 의 장점은 유지하면서도 균일신뢰도를 증가시킨 우수한 연구라 사료된다.

참고문헌

[1] J. Leskovec, A. Rajaraman, and J. D. Ullman, Mining of Massive Datasets, Cambridge University Press, 2014.

[2] B. Babcock et al., "Models and Issues in Data Stream Systems," In *Proc. of the 21st ACM Symp. on Principles of Database Systems*, Madison, Wisconsin, pp. 1-16, June 2002.

[3] G. Cormode and N. Duffield, "Sampling for Big Data: a Tutorial," In *Proc. of the 20th Int'l Conf. on Knowledge Discovery and Data Mining*, New York, New York, p. 1975, Aug. 2014.

[4] S. Ramírez-Gallego et al., "A Survey on Data Preprocessing for Data Stream Mining: Current Status and Future Directions," *Neurocomputing*, Vol. 239, No. C, pp. 39-57, May 2017.

[5] P. J. Haas, Data-Stream Sampling: Basic Techniques and Results, Data Stream Management, Springer, 2016.

[6] C. H. Weiß, Sampling in Data Mining, Wiley StatsRef: Statistics Reference Online, Sept. 2014.

[7] T. R. Kepe et al., "KSample: Dynamic Sampling Over Unbounded Data Streams," *Journal of Information and Data Management*, Vol. 6, No. 32, pp.32-47, Feb. 2015.

[8] M. Al-Kateb, B. S. Lee, and X. S. Wang, "Adaptive-Size Reservoir Sampling over Data Streams," In *Proc. of the 19th Int'l Conf. on Scientific and Statistical Database Management*, Banff, Canada, pp. 22-33, July 2007.

[9] 김하진, 이상훈, 문양세, "스트림 환경에서 균일신뢰도를 유지하는 가변 크기 샘플링," *한국정보과학회 학술발표논문집*, pp. 215-217, 2017년 6월.