



(19) 대한민국특허청(KR)  
(12) 등록특허공보(B1)

(45) 공고일자 2008년03월24일  
(11) 등록번호 10-0815866  
(24) 등록일자 2008년03월17일

(51) Int. Cl.

H03M 1/00 (2006.01) H03M 1/12 (2006.01)

(21) 출원번호 10-2007-0008026

(22) 출원일자 2007년01월25일

심사청구일자 2007년01월25일

(56) 선행기술조사문헌

EP1594248 A1

JP09219645 A

KR1020050095933 A

(73) 특허권자

인하대학교 산학협력단

인천 남구 용현동 253 인하대학교

(72) 발명자

배해영

인천 연수구 송도동 2-3 이안아파트 101동 1703호

이재동

경기 성남시 분당구 서현동 시범단지우성아파트  
225동 1902호

(뒷면에 계속)

(74) 대리인

이은철

전체 청구항 수 : 총 6 항

심사관 : 손희수

(54) 데이터 스트림의 타임 스탬프 기반 샘플링 방법

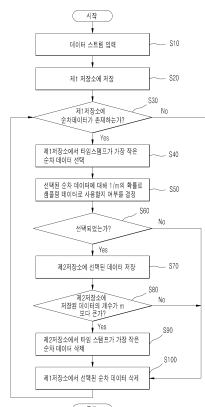
(57) 요약

본 발명은 데이터 스트림의 샘플링 방법에 관한 것으로서, 특히 순차패턴분석의 속도를 높이기 위한 샘플링 방법에 관한 것이다.

본 발명은 데이터 스트림 형태로 입력되는 데이터를 샘플링하기 위한 방법으로서, 데이터 스트림을 시간 기반 윈도우로 나눈 이벤트 순차들이 입력되는 단계; 상기 입력되는 데이터 스트림의 이벤트 순차들을 제 1 저장소에 저장하는 단계; 상기 제 1저장소에서 타임 스탬프가 가장 작은 순차 데이터를 선택하는 단계; 및 상기 선택된 순차 데이터를 필요한 순차 데이터 개수의 역수의 확률로 샘플링하는 단계를 포함하는 것을 특징으로 한다.

본 발명에 의하면, 타임스탬프 기반으로 데이터 스트림을 샘플링할 수 있다.

대표도 - 도3



(72) 발명자

**오영환**

충남 천안시 쌍용동 1538 월봉 벽산 태웅아파트  
205동 204호

**김경배**

대전 서구 관저2동 느리울마을 1106동 106호

**김호석**

경기 부천시 오정구 원종1동 229-2 8/1

**장용일**

인천 부평구 십정동 316-12

**유병섭**

경기 남양주시 도농동 부영아파트 103-1702

**어상훈**

인천 연수구 청학동 531-14 현암연구소

**이동욱**

서울 강남구 삼성2동 125-26 401호

**신재진**

인천 남구 용현4동 인하대학교 하이테크관 1008호  
데이터베이스실

**조숙경**

인천 남구 주안4동 1537-46 3층

## 특허청구의 범위

### 청구항 1

데이터 스트림을 시간 기반 윈도우로 나눈 이벤트 순차들이 입력되는 단계;

상기 입력되는 데이터 스트림의 이벤트 순차들을 제 1 저장소에 저장하는 단계;

상기 제 1저장소에서 타임 스탬프가 가장 작은 순차 데이터를 선택하는 단계; 및

상기 선택된 순차 데이터를 샘플링 데이터로 사용할지 여부를 결정하되, 상기 결정은 필요한 순차 데이터 개수의 역수의 확률로 샘플링되도록 하는 단계를 포함하는 것을 특징으로 하는 데이터 스트림의 타임 스탬프 기반 샘플링 방법.

### 청구항 2

제 1항에 있어서, 상기 필요한 순차 데이터 개수는 사용자에게 의해 미리 결정 되어진 값인 것을 특징으로 하는 데이터 스트림의 타임 스탬프 기반 샘플링 방법.

### 청구항 3

제 1항에 있어서, 상기 입력되는 데이터 스트림에는 타임 스탬프가 미리 설정되어진 것을 특징으로 하는 데이터 스트림의 타임 스탬프 기반 샘플링 방법.

### 청구항 4

제 1항에 있어서, 상기 샘플링하는 단계 이후, 샘플링된 데이터를 패턴 분석을 위해 제 2저장소에 저장하는 단계;

상기 저장된 샘플링 데이터가 상기 필요한 순차 데이터의 개수보다 많은 경우에는 가장 작은 타임 스탬프를 가지는 순차 데이터를 제 2 저장소에서 삭제하는 단계; 및

상기 가장 작은 타임 스탬프를 제 1저장소에서 삭제하는 단계를 더 포함하는 것을 특징으로 하는 데이터 스트림의 타임 스탬프 기반 샘플링 방법.

### 청구항 5

제 1항에 있어서, 상기 입력되는 데이터 스트림은 단일 데이터 스트림이거나 다중 데이터 스트림인 것을 특징으로 하는 데이터 스트림의 타임 스탬프 기반 샘플링 방법.

### 청구항 6

데이터 스트림을 샘플링하는 방법에 있어서,

데이터 스트림의 타임 스탬프가 가장 작은 순차를 선택하는 단계; 및

상기 선택된 데이터를 필요한 샘플의 개수의 역수의 확률로 샘플링하는 단계를 포함하는 것을 특징으로 하는 타임 스탬프 기반 샘플링 방법.

## 명세서

### 발명의 상세한 설명

#### 발명의 목적

#### 발명이 속하는 기술 및 그 분야의 종래기술

<5> 본 발명은 데이터 스트림의 샘플링 방법에 관한 것으로서, 특히 순차패턴분석의 속도를 높이기 위한 샘플링 방법에 관한 것이다.

<6> 데이터 스트림은 네트워크 모니터링에서부터 센서 네트워크로까지 다양한 분야에서 나타나고 있다. 데이터 스트림은 데이터가 지속적으로 입력이 되므로 미래에 입력되는 데이터의 크기를 알 수 없고, 또 다중 소스에서 데이

터가 들어올 수 있는 특징을 가진다. 이러한 데이터 스트림의 새로운 특징으로 인해 트랜잭션 중심으로 데이터를 처리하던 기존의 데이터베이스 관리 시스템(Database Management System, DBMS)의 연산들은 새롭게 정의되어야 한다. DBMS는 컴퓨터에 수록한 수많은 자료들을 쉽고 빠르게 추가, 수정, 또는 삭제할 수 있도록 해주는 소프트웨어이다.

- <7> 데이터 스트림의 입력은 빠르고 지속적으로 이루어지기 때문에 입력되었던 데이터를 처리하고 있을 때도 새로운 데이터가 끊임없이 입력될 수 있다. 따라서 입력되는 모든 데이터를 처리하려고 하면, 이 후에 들어오는 데이터들이 지연되는 상황이 발생할 수 있다. 또한, 엄청난 양의 입력 데이터 크기는 연산의 부담과 메모리 부족을 초래한다. 따라서 데이터 스트림을 효과적으로 처리하기 위해서는 일부 데이터만을 선택하여 처리하는 샘플링 방법을 고려해야만 한다.
- <8> 그러나 앞에서 언급한 대로 데이터 스트림의 새로운 특징으로 인하여 기존의 샘플링 방법 또한 데이터 스트림에 맞추어 새롭게 정의되어야 한다. 데이터 스트림을 샘플링을 정의할 때 고려사항은 다음과 같다.
- <9> 첫 번째로 알 수 없는 크기에서의 샘플링 방법이 고려되어야 한다.
- <10> 기존의 샘플링 방법들은 정해진 크기의 데이터에서 일정한 양의 샘플들을 임의로 추출해 내었다. 그러나 스트림 데이터는 지속적으로 데이터가 입력되고 전체 크기를 알 수 없기 때문에 알 수 없는 크기에서도 샘플링을 할 수 있어야 한다.
- <11> 두 번째로는 샘플링 알고리즘이 빠른 속도로 처리되어야 한다.
- <12> 데이터 스트림은 빠른 속도로 입력이 되기 때문에 샘플링 알고리즘도 빠른 처리로 이루어져야 하며 특히 1 패스(pass)로 이루어져야 한다. 1패스란 알고리즘을 실행할 때, 소스가 되는 데이터를 단 한 번만 읽고 처리한다는 뜻이다. 방대한 양의 소스 데이터를 한번 읽고 처리하는 데는 많은 시간이 소요되므로 데이터 스트림에 대한 연산은 1패스로 데이터를 처리하여야 한다.
- <13> 세 번째 고려해야 할 데이터 스트림의 특징은 다중 소스에서 입력이 이루어진다는 점이다. 따라서 시간의 흐름에 따라 소스들의 입력 크기가 크게 다를 수 있기 때문에 매번 데이터 스트림을 같은 확률로 샘플링을 한다면 옹지 않은 결과가 나올 것이다.
- <14> 도 1은 개울을 건너는 모습을 이벤트로 나타낸 것이다.
- <15> 개울을 건너려면 A, B, C, D의 이벤트가 차례대로 발생되어야만 한다. 이것은 순차 <A, B, C, D>가 발생한다는 것을 의미한다. 돌 1로 간 후 돌 2로 건너가는 이벤트 <A, B>는 발생할 수 있다. 그러나 돌 1에서 바로 돌 3으로 갈 수는 없다. 즉 이벤트 A가 발생한 후 이벤트 B를 거치지 않고서는 이벤트 C가 일어나는 순차 <A, C>는 발생 할 수 없다는 것을 의미한다. 이 예제는 발생된 순차를 임의로 데이터별로 샘플링을 하면 의미 없는 부분 순차가 된다는 것을 표현한다. 따라서 이벤트 <A, B, C>는 임의로 샘플링이 돼서는 안 되는 일련의 이벤트 집합이며 그 순서는 지켜져야만 한다.
- <16> 이때 이벤트 순차 <A, B, C, D> 사이에 잡음이 섞일 수 있다. 원래의 중요한 순차는 <A, B, C, D>이지만 <A, B, F, C, D>가 될 수도 있고, <A, B, C, X, D>가 될 수도 있다. 이 경우에는 이벤트 순차에 섞인 잡음들은 의미가 없다. 하지만 샘플링을 할 경우 어떤 것이 의미가 없는 데이터인지 판별하기는 불가능하다. 만약 잡음 섞인 이벤트 순차를 데이터별로 샘플링을 한다면 실제 패턴인 <A, B, C, D>중에서 한 데이터가 선택이 되지 않아 본래 패턴을 잃을 수도 있다. 따라서 이벤트 순차에서 어떤 것이 패턴인지, 잡음인지는 다수의 순차들을 이용한 패턴 분석을 통하여 이루어져야 하고 데이터를 중심 샘플링을 적용해서는 안 된다.
- <17> 데이터 스트림에 대한 샘플링 방법으로 2002년에 B. 밥콕(Bobcock)에 의해 제안된 샘플링 방법이 있다. 밥콕(Bobcock)은 데이터 스트림에 어떠한 종류의 윈도우 연산이 쓰였는가에 따라서 다른 샘플링 방법은 제안하였다. 우선샘플(Priority-Sample)은 시간 기반 윈도우를 가지는 데이터 스트림에 대한 샘플링 방법이고, 연관샘플(Chain-Sample)은 튜플 혹은 순차 기반 윈도우를 가지는 데이터 스트림에서의 샘플링 방법이다.
- <18> 그러나 이러한 샘플링 알고리즘들은 개별적인 데이터별로 샘플링을 진행한다. 따라서 샘플링 후 순차패턴분석을 한다면, 한 트랜잭션으로 의미되는 한 순차를 잘게 나누는 방법이 되어 한 트랜잭션의 의미가 깨지게 된다. 따라서 기존의 데이터 스트림의 순차패턴을 분석할 때 기존의 방법으로 샘플링을 하면 잘못된 패턴분석의 결과를 초래하게 된다.

**발명이 이루고자 하는 기술적 과제**

- <19> 본 발명에서는 데이터 스트림에 대한 순차패턴분석을 실행할 때 메모리의 사용율을 낮추고 분석의 속도를 높이기 위한 샘플링 알고리즘인 타임스탬프 기반(Timestamp based Sampling, TB-Sampling) 샘플링 방법을 제공하는 것을 목적으로 한다.
- <20> 또한, 본 발명은 타임스탬프 기반 샘플링 방법에 의해 샘플링시 연산의 부하와 메모리 사용량을 줄이는 것을 목적으로 한다.
- <21> 또한, 본 발명은 샘플링을 효율적으로 하여 샘플링 후 순차패턴분석의 결과의 정확도를 높이는 것을 목적으로 한다.

### 발명의 구성 및 작용

- <22> 본 발명은 데이터 스트림 형태로 입력되는 데이터를 샘플링하기 위한 방법으로서, 데이터 스트림을 시간 기반 윈도우로 나눈 이벤트 순차들이 입력되는 단계; 상기 입력되는 데이터 스트림의 이벤트 순차들을 제 1 저장소에 저장하는 단계; 상기 제 1저장소에서 타임 스탬프가 가장 작은 순차 데이터를 선택하는 단계; 및 상기 선택된 순차 데이터를 필요한 순차 데이터 개수의 역수의 확률로 샘플링하는 단계를 포함하는 것을 특징으로 한다.
- <23> 바람직하게는, 상기 필요한 순차 데이터 개수는 사용자에게 의해 미리 결정 되어진 값인 것을 특징으로 한다.
- <24> 상기 입력되는 데이터 스트림에는 타임 스탬프가 미리 설정된 것을 특징으로 한다.
- <25> 또한, 본 발명은 상기 샘플링하는 단계 이후, 샘플링된 데이터를 패턴 분석을 위해 제 2저장소에 저장하는 단계; 상기 저장된 샘플링 데이터가 상기 필요한 순차 데이터의 개수보다 많은 경우에는 가장 작은 타임 스탬프를 가지는 순차 데이터를 제 2 저장소에서 삭제하는 단계; 및 상기 가장 작은 타임 스탬프를 제 1저장소에서 삭제하는 단계를 더 포함하는 것을 특징으로 한다.
- <26> 상기 입력되는 데이터 스트림은 단일 데이터 스트림이거나 다중 데이터 스트림인 것을 특징으로 한다.
- <27> 도 2는 단일 데이터 스트림 환경에서의 TB-Sampling의 개념을 나타낸 것이다.
- <28> 먼저, 데이터 스트림을 시간 기반 윈도우로 나눈 이벤트 순차들이 입력된다. 이들 순차들은 타임스탬프를 가지고 있어서, 이 타임스탬프를 기반으로 패턴 분석에 필요한 m개의 순차를 샘플링한다.
- <29> 타임스탬프는 시간을 나타내는 인덱스로서 입력되는 데이터 스트림에 표시되어 있는 것으로 가정한다. 도 2에서 T1 내지 T6이 타임스탬프를 표시한 것이다 순차란 일련의 순서를 가지고 있는 데이터를 의미한다.
- <30> 타임스탬프 T1에서 순차는 B A D E G F C가 된다. 본 발명에서 순차와 순차 데이터는 동일한 의미이다.
- <31> 여기서 m의 크기는 순차패턴분석에 쓰이는 알고리즘에 따라 사용자가 결정한다. 샘플링을 통해 얻어진 m개의 순차들로 순차패턴분석을 수행한다. 그러나 데이터 스트림은 데이터의 입력이 지속적으로 이루어지므로 이 과정에서 몇 개의 이벤트 순차에서 m개의 순차를 샘플링해야 할 것인가 미리 알 수 없다.
- <32> 따라서 이미 전체 데이터 크기를 아는 상태에서 진행되는 기존의 샘플링 알고리즘은 데이터 스트림에 적용할 수 없다. 도 3은 지속적으로 입력되는 시간 기반 윈도우로 나누어진 이벤트 순차를 샘플링하는 과정을 보이고 있다.
- <33> 도 3은 단일 데이터 스트림 환경에서의 TB-Sampling 과정을 나타낸 흐름도이다.
- <34> 먼저 데이터 스트림이 입력되고(S10), 입력된 데이터들을 메모리 장치인 제 1저장소에 저장된다(S20). 상기 제 1 저장소는 버퍼와 유사한 개념으로 샘플링을 위해 입력되는 데이터들을 임시 저장해 두는 공간이다. 외부로부터 입력된 순차 데이터들을 가지고 있는 제 1저장소를 입력 값으로 쓰는 것이다.
- <35> 순차들에 타임스탬프를 설정하는 곳은 원본 데이터를 생성하는 곳일 수도 있고, 반대로 데이터 스트림을 수집하는 곳에서 타임스탬프를 설정해 줄 수도 있다. TB-Sampling에서는 한 트랜잭션은 한 타임스탬프 안에서 발생한다고 가정하였다. 그러나 만약 데이터 스트림을 수집하는 곳에서 타임스탬프를 부여하면 지속적으로 입력되는 이벤트들이 어디서부터 어디까지가 한 트랜잭션에서 이루어진 순차인지 모르므로 타임스탬프를 부여할 수 없다. 따라서 본 발명에서는 타임스탬프는 데이터 스트림을 생성하는 곳에서 부여한다고 가정한다.
- <36> 데이터가 입력되면 제 1저장소에 순차 데이터가 존재하는지 여부를 확인하여(S30) 저장소에 아무런 데이터가 없는 경우에는 절차를 종료한다.

- <37> 순차가 존재하는 경우에는 타임스탬프가 가장 작은 순차를 선택한다(S40). 예를 들어 도 2의 실시예를 예로 들면, T1을 임의로 선택하는 것이다.
- <38> 상기 선택된 T1을 순차패턴분석에 쓰이는 데이터로서 사용할지 여부, 즉 샘플링할지 여부를 결정한다(S50). 이때 샘플링 방법으로는 미리 설정된 순차패턴분석에 필요한 순차의 개수  $m$ 의 역수의 확률로 샘플링 여부를 결정한다. 예를 들어 도 2의 실시예에서와 같이 3개의 순차패턴분석이 필요한 경우에는  $1/3$ 의 확률로 샘플링을 한다.
- <39> 순차패턴분석에 쓰이는 순차의 개수  $m$ 은 임의로 사용자가 부여한 값이다. 그러나 이러한  $m$ 은 아무런 원칙 없이 정하는 것이 아니라 데이터의 입력 속도 패턴분석의 속도를 고려하여 정하여야 한다.
- <40> 만약  $m$ 이 크다면 패턴분석에 쓰이는 순차의 개수가 많은 것이므로 패턴분석의 속도는 느려지게 되어, 입력되는 데이터들이 지연되는 결과를 초래할 수도 있다. 또한,  $m$ 이 클수록 샘플링하는 확률  $1/m$ 은 작아지게 되어, 점점 더 실제 패턴과 다른 패턴 결과가 얻어질 수 있다. 반대로 너무 작은  $m$ 은 순차패턴분석의 속도는 빠를 수 있지만,  $1/m$ 이 커지게 되어 너무 빈번한 샘플링을 초래할 수 있다.
- <41> 이러한  $m$ 은 사용자가 어떠한 순차패턴 알고리즘을 쓰는가와 원하는 순차패턴분석의 결과 정확도에 따라 실험을 통하여 이끌어 내어야 한다. TB-Sampling은 일반적인 순차패턴분석 알고리즘에 쓰일 수 있기 때문에 사용자가 사용하는 순차패턴분석 알고리즘에 따라 분석 속도가 다를 수 있다. 만약 느린 순차패턴 분석 알고리즘을 사용한다면  $m$ 을 크게 하여 순차패턴분석의 주기를 길게 하여야 한다. 하지만  $m$ 은 샘플링 비율과 반비례하기 때문에 높은 정확도의 순차패턴분석을 원한다면  $m$ 을 작게 설정하여야 한다.
- <42> 상기와 같은 방법에 의해 순차가 샘플링 데이터로서 선택되면 제 2 저장소에 선택된 순차 데이터를 저장하고(S70), 선택되지 않은 경우에는 제 1저장소에서 상기 S40절차에서 선택된 순차 데이터를 삭제한다(S100).
- <43> 순차 데이터를 "선택한다"라는 용어가 중복되어 사용되어 혼동의 우려가 있으나 그 의미가 상이하다. 제 1 저장소에서 타임스탬프가 가장 작은 순차 데이터를 선택하는 단계(S40)는 샘플링을 위해 데이터를 뽑는 것이고, 그 다음 절차에서는 선택된 데이터를 샘플링 데이터, 즉 순차패턴분석에 이용할지 여부를 결정하고 선택하는 것이다. 따라서 제 1저장소에 선택된 순차 데이터를 삭제한다는 의미는 S40절차에서 선택된 데이터를 완전히 삭제한다는 의미이다.
- <44> 제 2 저장소에 저장한 후에는 저장된 순차의 개수가 사용자에게 의해 설정된  $m$ (패턴분석에 필요한 순차의 개수)보다 큰지 여부를 판단한다(S80). 더 큰 경우, 예를 들어 도 2에서와 같이 3개의 순차가 필요한데 4개가 저장된 경우에는 제 2저장소에서 타임스탬프가 가장 작은 순차를 삭제한다(S90). 타임 스탬프가 가장 작은 순차를 삭제하는 이유는 가장 오래된 데이터를 삭제하고 최근의 데이터를 패턴분석에 이용하기 위해서이다.
- <45> 마지막 절차로, 상기의 과정을 거친 순차 데이터는 제 1 저장소에서 삭제한다(S100). 동일한 데이터가 다시 선택되는 것을 방지하기 위한 것이다.
- <46> 삭제 후 다시 S30 절차로 되돌아가서 동일한 절차를 반복한다.
- <47> 도 4는 다중 데이터 스트림 환경에서의 TB-Sampling 과정을 나타낸다.
- <48> 데이터 스트림에서 순차패턴분석을 할 때, 위와 같이 한 데이터 스트림에서 순차패턴을 분석할 수 있다. 그러나 데이터 스트림은 입력이 이루어지는 소스가 여러 개일 수 있다는 특징을 가지고 있다. 따라서 여러 데이터 스트림간의 순차패턴분석도 고려할 수 있다. 다중 데이터 스트림간의 패턴 분석의 예를 들면 공장의 기계1과 기계2에서 생성되는 이벤트 스트림의 비교가 있다. 기계1에서의 이벤트들과 기계2에서의 이벤트들의 변화가 서로 어떤 영향을 끼치는지 순차패턴분석을 통하여 알아낼 수 있다.
- <49> 만약 데이터 스트림을 수집하는 곳에서 타임스탬프를 부여하면 다른 트랜잭션들끼리 같은 타임스탬프를 부여받을 수 있다. 다중 데이터 스트림간의 순차패턴분석은 단일 데이터 스트림에서와 마찬가지로 데이터 스트림을 수집하는 곳에서 타임스탬프를 부여한다고 가정한다. 따라서 데이터 스트림 소스가 여러 개이면 데이터를 수집하는 곳에서는 같은 타임스탬프를 가진 순차를 여러 개 볼 수 있다. 도 4는 이러한 모습을 나타낸 것이다.
- <50> 도 4에서는 여러 데이터 스트림에서 입력되는 이벤트 순차들을 볼 수 있다. S는 소스를 표시한 것이다.
- <51> 타임스탬프 T1, T3에서는 세 군데 소스 모두에서 데이터가 왔다. 그러나 T2에서는 두 번째 소스(S2)에서 데이터가 오지 않았다. 이는 데이터를 전송하는 소스쪽에서 네트워크 불량과 같은 문제로 인해 데이터 유실과 같은 문제 발생을 표현한 것이다. 이럴 경우 만약 데이터 스트림을 수집하는 곳에서 세 군데 모두에서의 데이터 입력을

기다린다면 전체 시스템은 블록킹이 발생 될 수 있다. 따라서 한 타임스탬프가 처음으로 들어온 때부터 어느 정도 일정 시간  $t$ 만큼 기다린 다음, 타임스탬프를 받는 것을 중단해야 한다.  $t$ 는 타임스탬프 단위와 동일하게 설정될 수 있다. 블록킹을 방지하기 위해서 입력이 이루어지지 않은 데이터 스트림이 있다면 입력을 기다리지 않고 비워둔다.

<52> 일단 각 소스에서 데이터가 수집되면 수집된 데이터로 샘플링을 하여야 한다. 샘플링하는 과정은 도 3의 단일 소스 샘플링 과정과 동일하다.

### 발명의 효과

<53> 본 발명에 의하면, 타임스탬프 기반으로 데이터 스트림을 샘플링할 수 있다.

<54> 또한, 본 발명에 의하면, 데이터 스트림의 순차패턴분석을 위한 샘플링을 보다 효율적으로 할 수 있다.

<55> 또한, 본 발명에 의하면, 데이터 스트림이 발생하는 네트워크의 패킷 분석 및 센서 데이터 분야 등 많은 분야의 순차패턴분석을 더 빠르고 정확하게 실행할 수 있다.

<56> 또한, 본 발명에 의하면, 1 패스로 샘플링 연산이 가능하다.

### 도면의 간단한 설명

<1> 도 1은 데이터 스트림의 샘플링이 필요한 실시예를 나타낸 도면

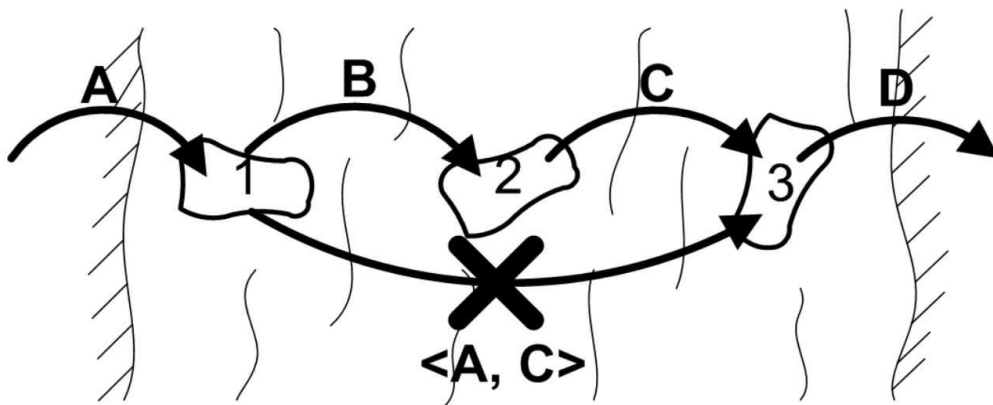
<2> 도 2는 본 발명의 실시예에 따른 단일 데이터 스트림 환경에서 TB 샘플링 개념을 나타낸 도면

<3> 도 3은 본 발명의 실시예에 따른 샘플링 과정을 나타낸 흐름도

<4> 도 4는 본 발명의 실시예에 따른 다중 데이터 스트림 환경에서 TB 샘플링 개념을 나타낸 도면

### 도면

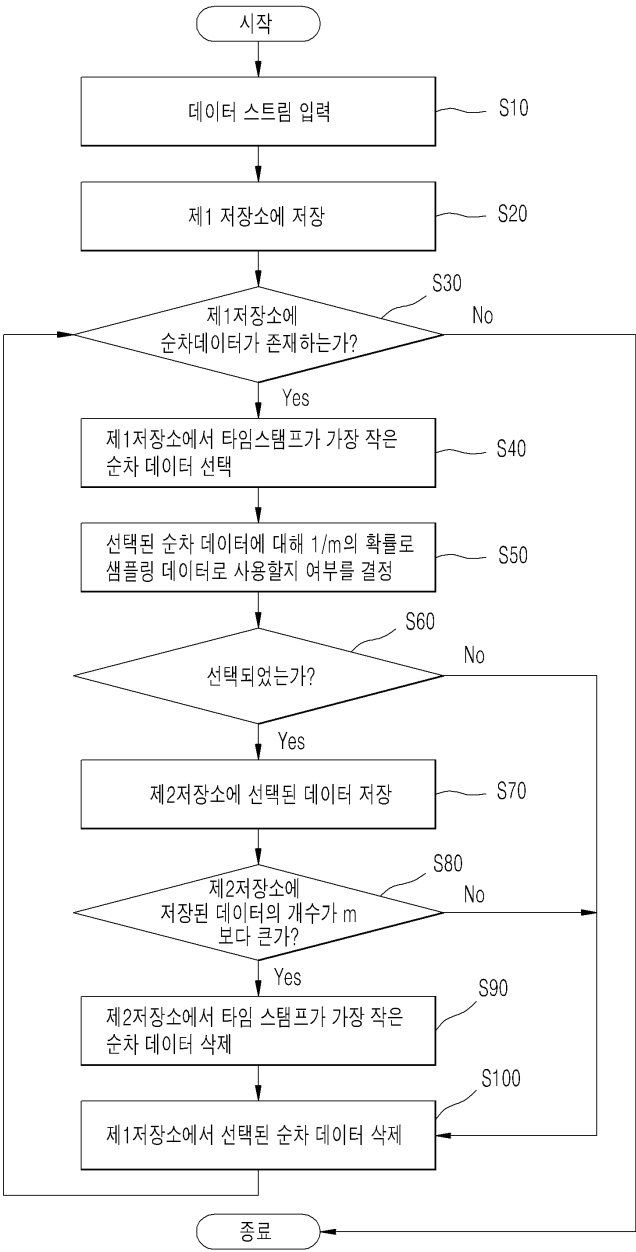
#### 도면1



#### 도면2



도면3



도면4

