# Büyük Veri Analitiği
# (Big Data Analytics)

M. Ali Akcayol
Gazi Üniversitesi
Bilgisayar Mühendisliği Bölümü

Bu dersin sunumları, "Mining of Massive Datasets, Jure Leskovec, Anand Rajaraman, Jeffrey David Ullman, Stanford University, 2011." kitabı kullanılarak hazırlanmıştır.

## Konular

- Stream Veri Modeli
  - Data stream yönetim sistemi
  - Stream veri kaynakları
  - Stream sorguları
  - Stream işleme sorunları
- Stream'de Veri Örnekleme
- Stream'lerde Filtreleme
  - Bloom filtresi
- Stream'de Farklı Eleman Sayısının Bulunması
  - Flajolet-Martin Algoritması

## Stream Veri Modeli

- **Stream veri geldiği anda işlem yapılmazsa** (depolama, data process vs.) **kalıcı şekilde kaybedilebilir.**
- Veriyi işleme hızından daha hızlı veri gelmesi durumunda da kaybedilebilir.
- **Stream veride işlem yapan algoritmalar stream veriyi bir şekilde özetlerler.**
- Stream mining algoritmaları, faydalı örnekleri seçer ve istenmeyen örnekleri filtrelerler.
- Başka bir özetleme yaklaşımında ise, **sabit boyutlu bir pencere içerisindeki elemanlarla (belirli bir süre için geçmiş veri) özetleme** yapılmaktadır.
- Stream verinin özetlenmesiyle birlikte daha küçük alanda saklanması da sağlanmış olur.

3

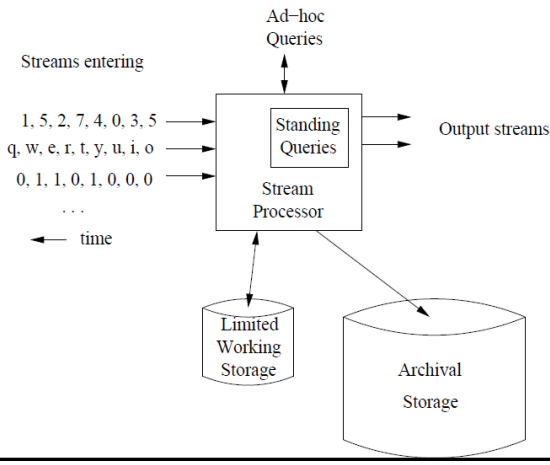
## Konular

- Stream Veri Modeli
  - Data stream yönetim sistemi
  - Stream veri kaynakları
  - Stream sorguları
  - Stream işleme sorunları
- Stream'de Veri Örnekleme
- Stream'lerde Filtreleme
  - Bloom filtresi
- Stream'de Farklı Eleman Sayısının Bulunması
  - Flajolet-Martin Algoritması

4

## Data stream yönetim sistemi

- **Stream işlemcisi** bir tür **veri yönetim sistemi** olarak görülebilir.
- **Sisteme çok sayıda farklı stream'den veri gelebilir.**
- **Veri türleri**, **veri oranları** ve **veri gelme aralıklarının dağılımları farklı olabilir.**




5

## Data stream yönetim sistemi

- Stream'lerden gelen veriler büyük bir depolama biriminde **(archival storage)** saklanabilir.
- Bu depolama birimindeki veri üzerinde **uzun zaman alan işlemlerin ardından sorgulama yapılabilir.**
- **Working storage** depolama birimi ise **stream verinin özetini** veya **bir parçasını saklar.**
- **Working storage** birimi, işlem hızı gereksinimine göre **disk** veya **ana hafıza** olabilir.
- Working storage birimi sınırlı kapasiteye sahiptir ve **stream verinin tamamını saklayamaz.**

6

## Konular

- Stream Veri Modeli
  - Data stream yönetim sistemi
  - Stream veri kaynakları
  - Stream sorguları
  - Stream işleme sorunları
- Stream'de Veri Örnekleme
- Stream'lerde Filtreleme
  - Bloom filtresi
- Stream'de Farklı Eleman Sayısının Bulunması
  - Flajolet-Martin Algoritması

7

## Stream veri kaynakları

**Sensor data**

- Bir okyanus yüzeyindeki **ısı sensörü her saat ölçtüğü ısı değerini** reel sayı olarak bir **istasyona göndersin.**
- Bu durumda **veri oranı çok düşük olduğundan** günümüz teknolojisinde **tüm veri ana hafızada tutulabilir.**
- **GPS birimindeki sensör** yüzeydeki **yükseklik değişimini** ölçüp bir istasyona göndersin.
- Bu durumda **veri oranı yüksektir** ve ancak **ana hafızada** veya **ayrı bir diskte tutulabilir.**
- Bir okyanusun **tüm davranışını ölçmek istersek**, **milyonlarca sensör kullanılır** ve **günlük birkaç terabyte veri** alınabilir.

8

## Stream veri kaynakları

**Image data**

- **Uydulardan sürekli dünyaya ilişkin görüntüler** alınıp **yeryüzündeki istasyonlara gönderilir.**
- Bu görüntü verilerinin boyutları **günlük birkaç terabyte düzeyinde olabilir.**
- **Şehirlerdeki güvenlik kameraları** uyduya göre düşük çözünürlüktedir, ancak her birisi **stream veri oluşturur.**
- Londra'da 6 milyon kamera olduğu belirtilmektedir ve her birisi stream veri oluşturur.

9

## Stream veri kaynakları

**İnternet ve Web trafiği**

- **İnternet anahtarlama düğümleri** (router) **IP paketlerinden oluşan stream'leri alır** ve **çıkış portlarına yönlendirme yapar.**
- Anahtarlama elemanlarının görevi verileri sorgulamak veya tutmak değildir.
- Ancak, günümüzde **anahtarlama elemanlarının kapasitesinin artırılmasına yönelik eğilim** (DOS ataklarının algılanması, tıkanıklık denetimi yapılması) vardır.
- **Web siteleri her gün milyonlarca sorgu almaktadır** (Google her gün yüzlerce milyon arama sorgusu almaktadır, Yahoo milyarlarca click almaktadır.).
- **Bu tür verilerden faydalı bilgiler elde edilebilir** (sorgulardaki ani yükselme, click sayısındaki ani yükselme veya düşme).

10

## Konular

- Stream Veri Modeli
  - Data stream yönetim sistemi
  - Stream veri kaynakları
  - Stream sorguları
  - Stream işleme sorunları
- Stream'de Veri Örnekleme
- Stream'lerde Filtreleme
  - Bloom filtresi
- Stream'de Farklı Eleman Sayısının Bulunması
  - Flajolet-Martin Algoritması

11

## Stream sorguları

- Stream veri üzerinde, **sürekli (standing query)** veya **anlık (ad-hoc query)** sorgulamalar yapılabilir.

**Sürekli sorgu**

- Okyanus yüzeyinde ısıyı ölçen sistemde **25° 'den büyük olduğunda alarm verilmesi istenebilir.**
- Bu durumda, sadece **son ölçülen değer** üzerinde işlem yapılır.
- Son **24 ölçüm değerinin ortalaması istenebilir.** Sürekli bir sorgu kullanılarak son 24 değer üzerinde işlem yapılabilir.
- **Tüm zamanların en yüksek ısı değeri istenebilir.** Sürekli bir sorgu ile her okumada maksimum değer ile karşılaştırma yapılabilir.
- **Tüm zamanların ortalama ısı değeri istenebilir.** Sürekli bir sorgu ile her okumada okuma sayısı artırılır ve tüm okunan değerlerin toplamına eklenir.

12

## Stream sorguları

**Anlık sorgu**

- Anlık sorgulamalar **stream'lerin mevcut durumlarına yöneliktir.**
- **Anlık sorguların ne şekilde olacağı önceden bilinemediğinden hazırlık yapılamaz** (son verilerin toplanması, geçmişin sürekli hesaplanması, …).
- Çok **farklı türlerdeki ad-hoc sorguları cevaplandırmak için** her stream için **sliding window** yaygın kullanılan yöntemdir.
- Sliding window, **working storage içinde oluşturulur.**
- Bir sliding window, stream'in **son *n* elemanını** veya ***t* süresindeki tüm elemanlarını içerebilir.**
- Her stream elemanı **bir tuple olarak alınırsa**, bir window ilişki kümesi olarak alınır ve **SQL sorgusu ile sorgulanabilir.**
- Sliding window içerisindeki elemanların güncel tutulması gereklidir.

13

## Stream sorguları

**Örnek**

- **Web sitelerinde** genellikle geçmiş bir zaman dilimindeki **unique kullanıcılara yönelik** veya buna benzer **raporlar istenir.**
- Her login, **stream içindeki bir eleman olarak düşünülebilir.**
- Sliding window, **Logins(name, time)** ilişkisi olarak görülebilir.
- Kullanılacak SQL aşağıdaki gibi olabilir:

```
SELECT COUNT(DISTINCT(name))
FROM Logins
WHERE time >= t;
```

- Geçmiş $t$ zaman aralığındaki tüm login'lerin **working storage** içinde tutulması gereklidir.
- Çok büyük ölçekli Web siteleri için bu veri birkaç terabyte'tır ve ancak disk üzerinde tutulabilir.

14

## Konular

- Stream Veri Modeli
  - Data stream yönetim sistemi
  - Stream veri kaynakları
  - Stream sorguları
  - Stream işleme sorunları
- Stream'de Veri Örnekleme
- Stream'lerde Filtreleme
  - Bloom filtresi
- Stream'de Farklı Eleman Sayısının Bulunması
  - Flajolet-Martin Algoritması

15

## Stream işleme sorunları

- **Stream'ler verileri genellikle çok hızlı gönderirler.**
- **Verilerin gerçek zamanlı işlenmesi zorunludur.** Aksi durumda, archival storage üzerinde işlem yapılması gerekir.
- **Stream-processing algoritmasının ana hafızada işlem yapması önemlidir.**
- Secondary storage birimini kullanmaması veya nadiren kullanması gereklidir.
- Stream algoritmalarının iki karakteristik özelliği vardır:
  - Kesin sonucu bulma ile karşılaştırıldığında, **yaklaşık sonucu bulmada daha etkindirler.**
  - Gerçek sonuca çok yakın yaklaşık sonucu elde etmek için **faydalı rastgelelik (useful randomness) yöntemleri** kullanılır.

16

## Konular

- Stream Veri Modeli
  - Data stream yönetim sistemi
  - Stream veri kaynakları
  - Stream sorguları
  - Stream işleme sorunları
- Stream'de Veri Örnekleme
- Stream'lerde Filtreleme
  - Bloom filtresi
- Stream'de Farklı Eleman Sayısının Bulunması
  - Flajolet-Martin Algoritması

17

## Stream'de Veri Örnekleme

- Örnekleme, **tüm stream'i ifade eden bir alt kümenin seçilmesini** ve sorguların bu alt küme üzerinde yapılmasını sağlar.
- Yapılacak **sorguların türü önceden bilinirse**, çok sayıda farklı yöntem kullanılabilir.
- **Ad-hoc sorgularda örnek oluşturmak** ve yaklaşık cevabı oluşturmak **daha zordur.**

18

## Stream'de Veri Örnekleme

**Örnek**

- **Bir arama motoru kullanıcı sorgularından oluşan stream almaktadır** ve stream veri kullanılarak **kullanıcı davranış analizinin yapılması istenmektedir.**
- Stream elemanlarının (user, query, time) üçlüsü şeklinde ifade edildiği varsayılsın.
- Son bir ayda **tekrarlı sorguya sahip kullanıcıların oranını bulmak istiyoruz.**
- Tüm stream elemanlarının sadece **1/10 uncu elemanlarını saklamak istiyoruz.**
- Her sorgu için 0-9 arasında artan bir sayı tutulmakta ve **her 0 geldiğinde sorgu saklanmaktadır.**

19

## Stream'de Veri Örnekleme

**Örnek - devam**

- Her kullanıcı sorgusunun **ortalama %10'luk kısmı** saklanır.
- Bir kullanıcının **çift sorgularının** ortalama **sayısı yanlış elde edilir.**
- Bir kullanıcı son ay içinde, $s$ adet sorguyu **tek**, $d$ adet sorguyu **çift** kez girmiş olsun. **İkiden fazla tekrarlı sorgu olmadığını varsayalım.**
- Örnekte, $s/10$ **tek sorgu** yer alır.
- Ancak, $d/100$ **çift sorgu** yer alır.
- Çiftlerden birer tanesinin **iki 10 uncu sorguda** olma olasılığı $d/100$ olur $(1/10*1/10)$ .
- **Birisi 10 uncuda olan, diğeri seçilmeyen 9 içinde olanlar** tek görünür.
- Stream'deki **çiftlerden** $18d/100$ kadarı **tek görünür.** $(1/10*9/10+9/10*1/10)$

20

## Stream'de Veri Örnekleme

**Örnek - devam**

- **Tek** görünen sayısı, $s/10+18d/100$ olur.
- **Çift** görünen sayısı ise $d/100$ olur.
- Örnek içerisinde **çift görünenlerin oranı**, $(d/100)/[(d/100)+s/10+(18d/100)] = d/(10s+19d)$ olur.
- **Gerçekte** olması gereken $d/(s+d)$ oranıdır.
- Hiçbir $s$ ve $d$ pozitif değerleri için $d/(s+d) = d/(10s+19d)$ eşitliği sağlanamaz.

21

## Stream'de Veri Örnekleme

- Çok sayıda kullanıcıya ait sorgulardan örnek elde etmek için **her kullanıcının 1/10 oranında sorgusu elde edilebilir.**
- **Örnekte yer alması istenen kullanıcıların** isimleri de ayrı bir **listede tutulabilir.**
- Her gelen sorgu için **kullanıcı örnek listesinde yer alıyorsa**, **0-9 arasında rastgele bir sayı üretilir.**
- **Rastgele sayı 0 ise**, sorgu çiftiyle birlikte örnek kümesine **kullanıcı ismiyle birlikte eklenir,** değilse eklenmez.
- 0-9 arası rastgele sayı üreten fonksiyon bir hash fonksiyonudur ve 10 bucket kullanır.
- **Bucket 0'a eşleştiğinde sorgu örneğe eklenir**, değilse eklenmez.

22

## Konular

- Stream Veri Modeli
  - Data stream yönetim sistemi
  - Stream veri kaynakları
  - Stream sorguları
  - Stream işleme sorunları
- Stream'de Veri Örnekleme
- Stream'lerde Filtreleme
  - Bloom filtresi
- Stream'de Farklı Eleman Sayısının Bulunması
  - Flajolet-Martin Algoritması

23

## Stream'lerde Filtreleme

- **Stream veri üzerinde seçme veya filtreleme uygulamaları yaygın olarak kullanılmaktadır.**
- Veriler genellikle tuple şeklinde ifade edilebilir.
- Eğer **veri seçme kriteri tuple içerisinde bir özellik ise** (Yaş > 10 gibi) **seçme işlemi kolaylıkla yapılabilir.**
- Eğer **geçmiş veriler ve gelecek verilerle ilgili** bir kümeye yönelik işlem **yapılacaksa seçme daha zordur.**
- **Kümenin boyutu hafızaya sığmayacak kadar büyükse** yapılacak işlem **daha zor hale gelmektedir.**

24

## Stream'lerde Filtreleme

**Örnek**

- Bir $S$ kümesi içinde **spam olmadığı bilinen** geçerli **1 milyar e-posta adresi olsun.**
- Stream ikililerden oluşmaktadır: **e-posta adresi** ve **e-posta içeriği.**
- Tipik bir **e-posta adresinin boyutu 20 byte** veya daha fazla olduğundan **tümünü hafızada tutmak uygun değildir (>20GB).**
- **Disk üzerinden erişim yapılması gerekir** veya kullanılabilir hafızadan daha **fazla hafıza alanı gerektirmeyen yöntem kullanmak gerekir.**
- **E-posta adresleri için** ana hafızada **1GB** alan ayrıldığını varsayalım.
- Bloom filtresi hafızayı **bit dizileri** şeklinde kullanır ve toplam **8 milyar bit (bucket)** vardır.
- **E-posta adresleri** ile **8 milyar bit (bucket) arasında** eşleştirme için bir **hash fonksiyonu oluşturulabilir.**

25

## Stream'lerde Filtreleme

- **Hash fonksiyonu** $S$ kümesindeki her bir **e-posta adresini bir bite eşleştirir ve o biti 1 yapar**, diğerleri üzerinde işlem yapmaz 0 kalır.
- $S$ kümesinde 1 milyar eleman (e-posta) olduğu için **hafızadaki bitlerin 1/8** tanesi **1 olur. Gerçekte 1/8'den biraz daha az bit 1 olur**, çünkü hash fonksiyonu ile **birden fazla epostayı aynı bit** ile eşleştirilebilir.
- **Yeni gelen bir e-posta adresinin** hash fonksiyonu ile eşleştirildiği yerdeki **bit 1** ise e-posta adresi **geçerli** kabul edilir, **0** ise **geçersiz** kabul edilir.
- Bu durumda **bazı spam e-postalarda geçerli kabul edilebilir.**
- Yaklaşık olarak **1/8 yeni stream elemanına ($S$ kümesinde yer almayan)** hash fonksiyonu ile **1 değeri atanır** ve **spam olarak algılanmaz.**
- E-postaların **7/8'i spam** olarak algılanacaktır. **Gerçekte e-postaların %80'i spam'dir.**
- Tümünü seçmek için $S$ kümesindeki tüm elemanlara bakmak zorunludur.

26

## Konular

- Stream Veri Modeli
  - Data stream yönetim sistemi
  - Stream veri kaynakları
  - Stream sorguları
  - Stream işleme sorunları
- Stream'de Veri Örnekleme
- Stream'lerde Filtreleme
  - Bloom filtresi
- Stream'de Farklı Eleman Sayısının Bulunması
  - Flajolet-Martin Algoritması

27

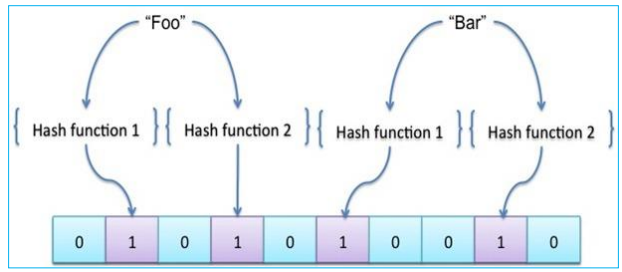
## Bloom filtresi

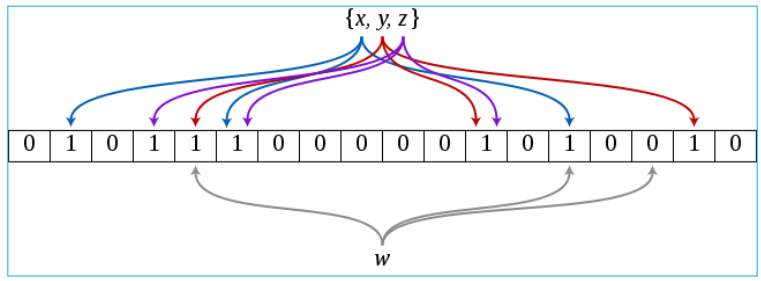
- **Bloom filtresi aşağıdaki bileşenlerden oluşur:**
  - ***n* bit dizi**, başlangıçta tümü 0 değerine sahiptir.
  - **Hash fonksiyon kümesi** $h_1, h_2, \ldots, h_k$.
    Her hash fonksiyonu bir **anahtar** değerini ***n* bucket'tan birisine eşleştirir.**
  - ***m* adet anahtar** değere sahip ***S* kümesi.**
- Bloom filtresindeki amaç ***S* kümesinde olanların pozitif**, **olmayanların negatif olarak seçilmesidir.**
- *S* kümesindeki **her anahtar değer hash fonksiyonlarına girilir** ve elde edilen **ilgili bitler 1 yapılır.**
- Yeni gelen bir anahtar değer,
  - **Tüm hash fonksiyonlarında 1 değerine eşleşiyorsa *S* kümesinde vardır (pozitif)**,
  - **Bir hash fonksiyonuyla bile 0 değerine atanıyorsa *S* kümesinde yoktur (negatif)**.

28

## Bloom filtresi

- Bloom filtresi örnekleri




29

## Bloom filtresi

- Eğer **yeni gelen bir anahtar** değer $S$ **kümesinde varsa, kesinlikle Bloom filtresinden geçer** (True Positive).
- Ancak, $S$ **kümesinde olmayan bir anahtar değerinin** de **Bloom filtresinden geçme olasılığı vardır** (False Positive).

30

## Bloom filtresi

- ***x*, toplam hedef sayısını** (bit sayısını) göstersin (8 milyar = $8.10^9$).
- ***y*, *S* kümesindeki** toplam **eleman sayısını** göstersin (1 milyar = $1.10^9$).
- ***S*'deki bir elemanın** belirli bir hedef bite **eşleşmeme olasılığı** $(x\text{-}1)/x$'dir.
- **Hiçbir *y* elemanın** belirlenen bite **eşleşmeme olasılığı** $\left(\frac{x-1}{x}\right)^y$ olur.

$$\left(\frac{x-1}{x}\right)^y = \left(1-\frac{1}{x}\right)^{x\left(\frac{y}{x}\right)} \Rightarrow \lim_{x\to\infty}\left(1-\frac{1}{x}\right)^{x\left(\frac{y}{x}\right)} = e^{-y/x}$$

- Örnek için $e^{-1/8}$ şeklinde hesaplanır
- **Yeni** gelen ***y* tane elemanın (S kümesinde olmayan)** belirlenen bite (1 değeri) **eşleşme olasılığı** $1-e^{-1/8} = 0{,}1175$ **olur (%11,75 FP).**
- $k = 2$ hash fonksiyonu kullanılırsa, $(1-e^{-ky/x})^k = (1-e^{-2(1/8)})^2 = 0{,}0493$ olur **(%4,93 FP).**

31

## Bloom filtresi

- ***m* hedef bit sayısı**, ***n* stream içerisindeki eleman sayısı** ve ***k* hash fonksiyonu sayısı** olsun.
- **Optimal hash fonksiyonu sayısı** aşağıdaki eşitlikle hesaplanır.

$$k = \frac{m}{n}\ln 2$$

- **İstenen False Positive oranına (*p*) göre** optimal **m hedef bit sayısı** aşağıdaki eşitlikle hesaplanır.

$$m = -\frac{n \ln p}{(\ln 2)^2}$$

32

## Konular

- Stream Veri Modeli
  - Data stream yönetim sistemi
  - Stream veri kaynakları
  - Stream sorguları
  - Stream işleme sorunları
- Stream'de Veri Örnekleme
- Stream'lerde Filtreleme
  - Bloom filtresi
- Stream'de Farklı Eleman Sayısının Bulunması
  - Flajolet-Martin Algoritması

33

## Stream'de Farklı Eleman Sayısının Bulunması

- **Bir** evrensel küme elemanlarını içeren **stream için**, **başlangıçtan itibaren** veya **belirli bir süre içindeki farklı eleman sayısını bulmak** istenebilir.
- Bir Web sitesi için **son bir ay içerisindeki unique ziyaretçilerin sayısı** bulunmak istenebilir (Amazon, Google, vs.).
- **Evrensel küme tüm login'leri içerir**, **stream elemanları** ise **her bir login'den oluşur.**
- Google, kullanıcıları **IP adresi ile ayırt edebilir.** Ancak, yaklaşık 4 milyar IP adresi kullanımdadır (IPv4 için $2^{32}$).
- **Eleman sayısı az ise**, bir **hash tablosu** veya **arama ağacı** ile bulunabilir.
- **Çok sayıda stream veri varsa** veya **her Web sayfası için** belirli süredeki **unique kullanıcıyı belirlemek gerekiyorsa** (Yahoo aylık her sayfa) hafızada işlem yapılamaz.
- **Daha az hafıza kullanarak** farklı eleman sayısı tahmin edilebilir.

34

## Konular

- Stream Veri Modeli
  - Data stream yönetim sistemi
  - Stream veri kaynakları
  - Stream sorguları
  - Stream işleme sorunları
- Stream'de Veri Örnekleme
- Stream'lerde Filtreleme
  - Bloom filtresi
- Stream'de Farklı Eleman Sayısının Bulunması
  - Flajolet-Martin Algoritması

35

## Flajolet-Martin Algoritması

- **Evrensel küme elemanları uzun bir bit string'ine hash yapılarak** farklı eleman sayısı tahmin edilebilir.
- **Bit dizisi eleman sayısının evrensel küme eleman sayısından fazla olması** yeterlidir (URL için 64-bit yeterlidir.).
- **Birden fazla hash fonksiyonu** ile stream elemanları hash yapılabilir.
- **Bir hash fonksiyonu stream'deki aynı eleman için hep aynı sonucu vermelidir.**
- **Flajolet-Martin algoritmasına göre,** stream içinde ne kadar çok farklı elemanla karşılaşılırsa, o kadar çok farklı hash-value ikilisi görülür.
- Stream'deki herhangi bir $a$ elemanı için **$h(a)$ değerinde sondaki ardışık sıfırların o ana kadar karşılaşılan maksimum sayısı ($R$) tutulur.**
- Stream içerisindeki **farklı eleman sayısı** $2^R$ olarak tahmin edilir.

36

## Flajolet-Martin Algoritması

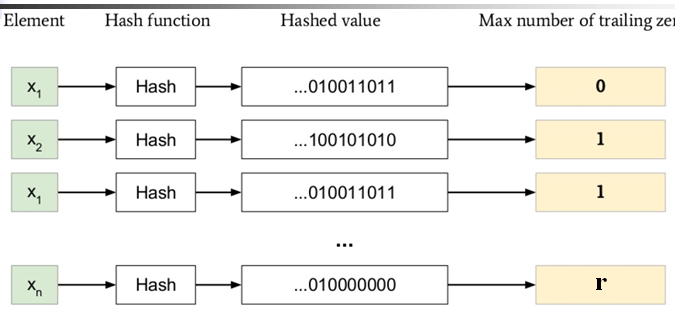
- Stream'deki bir ***a*** **elemanı için** hesaplanan ***h*(*a*) değerinde sondaki *r* ardışık bitin 0 olma olasılığı** $(1/2)^r = 2^{-r}$ **dir.**
- Stream'de ***m*** **farklı eleman** olsun. **Hiçbirinin hash değerinin en az *r* boyutunda 0 kuyruğa sahip olmama olasılığı** $(1-2^{-r})^m$ **olur.**

$$(1-2^{-r})^m = ((1-2^{-r})^{2^r})^{m2^{-r}} = (1/e)^{m2^{-r}} = \frac{1}{e^{m/2^r}}$$

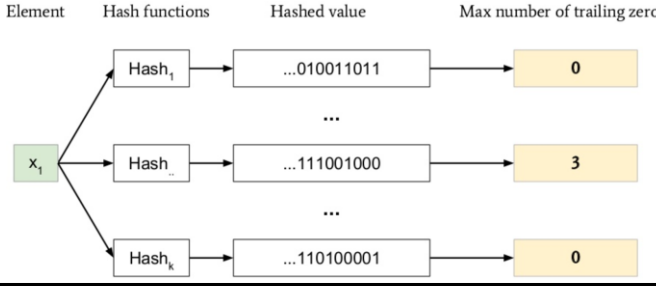
- $m >> 2^r$ **ise**, en az ***r*** **boyutunda 0 olmama olasılığı 0'a, olma olasılığı 1'e yaklaşır.**
- $m << 2^r$ **ise**, en az ***r*** **boyutunda 0 olmama olasılığı 1'e, olma olasılığı 0'a yaklaşır.**
- Stream'deki **farklı eleman sayısı (m),** genellikle $2^r$ **'den çok küçük** veya **çok büyük değildir.**

37

## Flajolet-Martin Algoritması




Birden çok hash fonksiyonu kullanılabilir.




38

## Ödev

- Stream içerisindeki farklı olayların (elemanların) **dağılım sıklıklarının bulunması** ve **bir sonraki olayın gerçekleşme anının tahmini** için kullanılan yöntemlere yönelik bir araştırma ödevi hazırlayınız.

39