Đề cương luận văn thạc sĩ

Nghiên cứu phát triển kỹ thuật đếm số phần tử trên dòng dữ liệu

Học viên: Lê Anh Quốc ID: 2070428

Người hướng dẫn khoa học: PGS. TS. THOẠI NAM

Outline

Giới thiệu

Giới thiệu

Ngày nay, các ứng dụng và dịch vụ trực tuyến đóng vai trò ngày càng quan trọng trong cuộc sống của con người. Chúng ta sử dụng mạng xã hội để kết nối với bạn bè và chia sẻ thông tin, mua sắm trực tuyến để tiết kiệm thời gian và tiền bạc, hay xem phim và chơi game trực tuyến để giải trí. Để đánh giá hiệu quả hoạt động của các ứng dụng và dịch vụ này, một trọng những chỉ số quan trọng nhất là số lượng người dùng hoạt động. Việc theo dõi số lượng người dùng hoạt động trong một khoảng thời gian nhất định trên một dòng dữ liệu (data stream) là một yêu cầu quan trọng đối với nhiều ứng dụng và dịch vụ trực tuyến, hiệu quả của các chiến dịch marketing, và hỗ trợ ra quyết định kinh doanh. Ví dụ, trong các ứng dụng mạng xã hội, số lượng người dùng hoạt động cho thấy mức độ tương tác và sự quan tâm của người dùng đối với nền tảng. Trong các dịch vụ thương mại điện tử, số lượng người dùng hoạt động cho thấy hiệu quả và các chiến dịch quảng cáo và khuyến mãi. Tuy nhiên, việc đếm số lượng người dùng không phải là một nhiệm vụ đơn giản, đặc biệt là khi dữ liệu lớn và tốc độ truy cập cao. Các phương pháp truyền thống như lưu trữ và truy vấn trực tiếp vào cơ sở dữ liệu có thể

Giới thiệu

Trong nhiều trường hợp, cần phải tổng hợp số lượng người dùng trên nhiều dòng dữ liệu khác nhau. Việc này giúp có được bức tranh toàn cảnh về hoạt động của người dùng trên toàn hệ thống, từ đó đưa ra các phân tích và đánh giá chính xác hơn. Ví dụ, trong hệ thống thương mại điện tử, cần tổng hợp số lượng người dùng từ các trang web, ứng dụng di động và API khác nhau để có được số lượng người dùng hoạt động thực tế trên toàn hệ thống. Tuy nhiên, việc tổng hợp dữ liệu từ nhiều nguồn khác nhau có thể gặp thách thức về đồng bộ hóa dữ liệu, xử lý dữ liệu bị thiếu hoặc lỗi, và đảm bảo tính nhất quán của kết quả. Ngoài ra, có thể cần phải đếm số lượng người dùng trên nhiều khoảng thời gian khác nhau trên một hoặc nhiều dòng dữ liệu khác nhau. Việc này giúp phân tích chi tiết hơn hoạt động của người dùng theo thời gian, theo khu vực hoặc theo tiêu chí khác.

Giới thiệu

Ví dụ, trong một ứng dụng phát trực tiếp, cần đếm số lượng người dùng hoạt động theo giờ hoặc từng phân đoạn chương trình để đánh giá mức độ quan tâm của người xem. Tuy nhiên, việc phân chia và xử lý dữ liệu theo nhiều đoạn có thể làm tăng độ phức tạp của thuật toán và ảnh hưởng đến hiệu suất của hệ thống. Do đó, cần phải có một giải pháp đếm số lượng phần tử trên dòng dữ liệu đạt hiệu suất cao và tin cậy, từ đó có thể ứng dụng rộng rãi trong các hệ thống khác nhau như mạng xã hội, thương mại điện tử, chương trình phát trực tiếp, hệ thống giám sát và hệ thống giao thông thông minh.

Sections

Sections group slides of the same topic

```
\section{Elements}
```

for which **metropolis** provides a nice progress indicator ...

Các công trình nghiên cứu liên quan

LogLog [2]

Thuật toán LogLog cho phép ước lượng số lượng từ vựng khác nhau trong toàn bộ tác phẩm của Shakespeare chỉ trong một lần quét và với độ chính xác cỡ vài phần trăm, sử dụng một lượng bộ nhớ phụ nhỏ. Phiên bản cơ bản đã được xác minh qua phân tích toàn diện và có phiên bản tối ưu hóa có khả năng song song.

HyperLogLog [4]

Thuật toán HYPERLOGLOG là một thuật toán xác suất gần tối ưu, được thiết kế để ước lượng số lượng các phần tử khác nhau trong các tập dữ liệu rất lớn. Sử dụng bộ nhớ phụ có kích thước m đơn vị, HYPERLOGLOG thực hiện một lần quét qua dữ liệu và tạo ra một ước lượng về số lượng phần tử khác nhau với độ chính xác tương đối là khoảng $\frac{1.04}{\sqrt{m}}$. Thuật toán này có khả năng ước lượng số lượng phần tử lớn hơn 10^9 với độ chính xác khoảng 2% chỉ sử dụng 1.5 kilobytes bộ nhớ, đồng thời có khả năng song song hoá tối ưu và thích nghi với mô hình cửa sổ trượt (sliding windown).

HyperLogLog++ [5]

Bài báo giới thiệu một thuật toán mới ước lượng số lượng luồng hoạt động trong dòng dữ liệu, sử dụng cơ chế cửa sổ trượt kết hợp với thuật toán HyperLogLog. Thuật toán này có độ chính xác cao, lỗi tiêu chuẩn khoảng $\frac{1.04}{\sqrt{m}}$, với m là số lượng thanh ghi trong bộ nhớ. Dù cần bộ nhớ bổ sung so với HyperLogLog, tổng bộ nhớ cần thiết không vượt quá $5m \ln(\frac{n}{m})$ byte, với n là số luồng thực sự trong cửa sổ trượt. Kết quả lý thuyết được xác minh trên cả dữ liệu thực và tổng hợp.

Sliding HyperLogLog [1]

Bài báo giới thiệu một thuật toán mới ước lượng số lượng luồng hoạt động trong dòng dữ liệu, sử dụng cơ chế cửa sổ trượt kết hợp với thuật toán HyperLogLog. Thuật toán này có độ chính xác cao, lỗi tiêu chuẩn khoảng $\frac{1.04}{\sqrt{m}}$, với m là số lượng thanh ghi trong bộ nhớ. Dù cần bộ nhớ bổ sung so với HyperLogLog, tổng bộ nhớ cần thiết không vượt quá $5m \ln(\frac{n}{m})$ byte, với n là số luồng thực sự trong cửa sổ trượt. Kết quả lý thuyết được xác minh trên cả dữ liệu thực và tổng hợp.

ExaLogLog [3]

ExaLogLog là một cấu trúc dữ liệu mới cho việc đếm độc lập xấp xỉ, tương tự như HyperLogLog, nhưng tiêu tốn ít hơn 43% không gian với cùng lỗi ước lượng.

Phát biểu bài toán

Phát biểu bài toán

Bài toán 1: Phát triển thuật toán để ước lượng số lượng phần tử (cardinality estimation) trong một khoảng thời gian trên một dòng dữ liệu (data stream).
Bài toán 2: Mở rộng thuật toán để ước lượng số lượng phần tử trong một khoảng thời gian trên nhiều dòng dữ liệu.

Mục tiêu, đối tượng và giới hạn nghiên cứu

Mục tiêu

Mục tiêu chính:

- Phát triển kỹ thuật đếm số lượng phần tử hiệu quả, có chính xác cao trên dòng dữ liệu.
- Nâng cao hiệu suất xử lý dữ liệu lớn, đáp ứng nhu cầu ngày càng tăng trong kỷ nguyên số.
- Đóng góp vào sự phát triển của công nghệ dữ liệu lớn, mở ra tiềm năng ứng dụng rộng lớn trong nhiều lĩnh vực.

Mục tiêu cụ thể:

- Phân tích và đánh giá các kỹ thuật đếm số lượng phần tử hiện có.
- Đề xuất và triển khai kỹ thuật đếm số lượng phần tử mới, tối ưu hóa hiệu suất và độ chính xác.
- Thực hiện thí nghiệm để chứng minh tính ưu việt của kỹ thuật mới so với các kỹ thuật hiện có.
- Phân tích kết quả thí nghiệm, rút ra kết luận và đề xuất hướng nghiên cứu tiếp theo.

Giới hạn

- Đề tài tập trung nghiên cứu kỹ thuật đếm số lượng phần tử trên dòng dữ liệu dạng văn bản.
- Các kỹ thuật được đề xuất và triển khai có thể chưa áp dụng được cho tất cả các loại dữ liệu.
- Nghiên cứu chỉ giới hạn trong thời gian cho phép.

Đối tượng nghiên cứu

Đối tượng nghiên cứu của đề tài "Nghiên cứu phát triển kỹ thuật đếm số lượng phần tử trên dòng dữ liệu" Đối tượng nghiên cứu chính:

- Dòng dữ liệu dạng văn bản có chứa nhiều phần tử cần đếm.
- Các kỹ thuật đếm số lượng phần tử hiện có và mới được đề xuất.

Đối tượng nghiên cứu cụ thể:

- Số phần tử có thể là userID, IP address hoặc bất kỳ đối tượng nào tương đương mà có thể đếm được nhờ định danh của nó.
- Các ứng dụng xử lý dữ liệu lớn có nhu cầu đếm số lượng phần tử hiệu quả.

Cơ sở lý thuyết

Cơ sở lý thuyết

Các thuật toán xác suất phổ biến nhất để ước lượng số lượng được sử dụng trong thực tế là họ các thuật toán LogLog bao gồm thuật toán *LogLog*, được đề xuất bởi Marianne Durand và Philippe Flajolet vào năm 2003 [2], và các kế thừa của nó *HyperLogLog* và *HyperLogLog++*.
Các thuật toán này sử dụng một phương pháp tương tự như thuật toán Đếm Xác Suất trong việc ước lượng số lượng n bằng cách quan sát số lượng lớn nhất của các số không dấu hàng đầu trong biểu diễn nhị phân của các giá trị. Tất cả chúng đều yêu cầu một bộ nhớ phụ trợ và thực hiện một lần duyệt qua dữ liệu để tạo ra một ước lượng về số lượng.
Mỗi phần tử trong tập dữ liệu được tiền xử lý bằng cách áp dụng một hàm băm h chuyển đổi các phần tử thành số nguyên phân bố đều đặn đủ trên một phạm vi scala $\{0, 1, ..., 2^M - 1\}$ hoặc, tương đương, trên tập hợp các chuỗi nhị phân có độ dài M:

$$h(x) = j = \sum_{k=0}^{M-1} j_k \cdot 2^k := (i_0 i_1 ... i_{M-1})_2 \, , i_k \in \{0, 1\}. \quad (5.1)$$

Cơ sở lý thuyết

Đầu tiên, chúng ta chia tập dữ liệu ban đầu hoặc dòng dữ liệu đầu vào thành một số tập con, mỗi tập con này được lập chỉ mục bởi một trong m bộ đếm đơn giản. Sau đó, theo phương pháp trung bình ngẫu nhiên, vì có một hàm băm đơn giản, chúng ta chọn bộ đếm cho phần tử cụ thể x bằng cách sử dụng một phần của giá trị băm của nó $h(x)$, trong khi phần còn lại được sử dụng để cập nhật bộ đếm tương ứng.
Tất cả các thuật toán được thảo luận ở đây dựa trên việc quan sát các mẫu 0^k1 xuất hiện ở đầu của các giá trị cho bộ đếm cụ thể, và gán mỗi mẫu với chỉ số của nó, gọi là hạng (rank). Hạng tương đương với vị trí bit 1 đầu tiên từ trái qua phải trong biểu diễn nhị phân của giá trị băm của phần tử được lập chỉ mục và có thể được tính bằng công thức:

$$rank(i) = \begin{cases} \min_{i_k \neq 0}, \text{for } i > 0, \\ M \text{for } i = 0 \end{cases}$$

Cơ sở lý thuyết

Mỗi bộ đếm đơn giản xây dựng quan sát về số lượng của riêng mình dựa trên các hạng đã nhìn thấy, ước lượng cuối cùng của số lượng được tạo ra từ quan sát bằng một hàm đánh giá.
Liên quan đến lưu trữ, thuật toán *LogLog* đề xuất một giải pháp tiết kiệm lưu trữ cùng với một hàm đánh giá và phương pháp hiệu chỉnh sai lệch tốt hơn.

LogLog algorithm

Ý tưởng cơ bản của thuật toán *LogLog* bắt đầu bằng việc tính toán hạng cho mỗi phần tử đầu vào dựa trên một hàm băm đơn giản h. Vì chúng ta có thể mong đợi rằng khoảng $\frac{n}{2^k}$ phần tử có thể có $rank(\cdot) = k$, trong đó n là tổng số phần tử được lập chỉ mục vào một bộ đếm, hạng quan sát tối đa có thể cung cấp một dấu hiệu tốt về giá trị của $log_2 n$:

$$R = \max_{x \in D} (rank(x)) \approx log_2 n.$$

Tuy nhiên, ước lượng như vậy có sai số khoảng ± 1.87 lần nhị phân, điều này không thực tế. Để giảm sai số, thuật toán *LogLog* sử dụng một kỹ thuật phân nhóm dựa trên việc trung bình ngẫu nhiên và chia tập dữ liệu thành $m = 2^p$ tập con $S_0, S_1, ..., S_{m-1}$, trong đó tham số độ chính xác p xác định số bit được sử dụng trong điều hướng.
Do đó, đối với mỗi phần tử x từ tập dữ liệu, p bit đầu tiên của giá trị băm $h(x)$ M-bit có thể được lấy để tìm ra chỉ số j của tập con thích hợp.

$$j = (i_0 i_1 ... i_{p-1})_2 ,$$

LogLog algorithm

và phần còn lại (M-p) bit được lập chỉ mục vào bộ đếm tương ứng COUNTER[j] để tính hạng và quan sát R_j theo công thức (3.9).
Dưới sự phân phối công bằng, mỗi tập con nhận $\frac{n}{m}$ phần tử, do đó quan sát R_j từ các bộ đếm $\{COUNTER[j]\}_{j=0}^{m-1}$ có thể cung cấp một dấu hiệu về giá trị của $\log_2 \frac{n}{m}$, và bằng cách sử dụng trung bình số học của chúng với một số sự hiệu chỉnh, chúng ta có thể giảm thiểu phương sai của một quan sát duy nhất:

$$n = \alpha_m \cdot m \cdot 2^{\frac{1}{m}\sum_{j=0}^{m-1} R_j},$$

với $\alpha_m = \left(\Gamma\left(-\frac{1}{m}\right) \cdot \frac{1-2^{\frac{1}{m}}}{\log_2}\right)^m$, $\Gamma(\cdot)$ là gamma function.
Tuy nhiên, đối với hầu hết các trường hợp thực tế, $m \geq 64$ là đủ để chỉ sử dụng $\alpha_m \approx 0.39701$.

LogLog algorithm

Algorithm 1: Estimatin cardinality with *LogLog*

Input: Dataset D

Input: Array of m *LogLog* counters with hash function h

Output: Cardinality estimation

$COUNTER[j] \leftarrow 0,\ j = 0...m-1$

for $x \in D$ **do**

$i \leftarrow \mathrm{h(x)} := (i_0 i_1 ... i_{M-1})_2,\ i_k \in \{0,1\}$

$j \leftarrow (i_0 i_1 ... i_{M-1})_2$

$r \leftarrow \mathrm{rank}((i_p i_{p+1} ... i_{M-1})_2)$

$COUNTER[j] \leftarrow \max(\mathrm{COUNTER[j],r})$

end

$$R \leftarrow \frac{1}{m} \sum_{k=0}^{m-1} \mathrm{COUNTER[j]}$$

return $\alpha_m \cdot m \cdot 2^R$

LogLog algorithm

Sai số tiêu chuẩn δ của thuật toán *LogLog* có mối quan hệ nghịch với số lượng bộ đếm sử dụng m và có thể được xấp xỉ gần như là:

$$\delta \approx \frac{1.3}{\sqrt{m}}$$

> *Do đó, với* $m = 256$, *sai số tiêu chuẩn là khoảng 8% và với* $m = 1024$, *nó giảm xuống còn khoảng 4%.*

Yêu cầu lưu trữ của thuật toán *LogLog* có thể được ước tính là

$O(log_2 log_2 n)$ bit nếu cần đến đếm đến n. Cụ thể hơn, tổng không gian được yêu cầu bởi thuật toán để đếm đến n là $m \cdot log_2 log_2 \frac{n}{m}(1 + O(1))$.

LogLog algorithm

So sánh với thuật toán Đếm Xác suất trong đó mỗi bộ đếm yêu cầu 16 hoặc 32 bit, thuật toán *LogLog* yêu cầu bộ đếm nhỏ hơn nhiều $\{COUNTER[j]\}_{j=0}^{m-1}$, thường là 5 bit mỗi bộ đếm. Tuy nhiên, trong khi thuật toán *LogLog* cung cấp hiệu quả lưu trữ tốt hơn so với thuật toán Đếm Xác suất, nó đôi chút ít chính xác hơn.
Giả sử chúng ta cần đếm định lượng cho đến 2^{30}, tức là khoảng 1 tỷ, với độ chính xác khoảng 4%. Như đã đề cập, cho sai số tiêu chuẩn như vậy, cần $m = 1024$ ngăn, mỗi ngăn sẽ nhận xấp xỉ $\frac{n}{m} = 2^{20}$ phần tử.
$log_2\left(log_2 2^{20}\right) \approx 4.32$, do đó, chỉ cần phân bổ khoảng 5 bit cho mỗi ngăn (tức là một giá trị nhỏ hơn 32). Do đó, để ước lượng định lượng lên đến khoảng 10^9 với sai số tiêu chuẩn là 4%, thuật toán yêu cầu 1024 ngăn với 5 bit mỗi ngăn, tức là tổng cộng 640 byte.

Phát biểu bài toán

```
The theme provides sensible defaults to
\emph{emphasize} text, \alert{accent} parts
or show \textbf{bold} results.
```

becomes

The theme provides sensible defaults to *emphasize* text, accent parts or show **bold** results.

Font feature test

- Regular
- *Italic*
- Small Caps
- **Bold**
- ***Bold Italic***
- **Bold Small Caps**
- `Monospace`
- *`Monospace Italic`*
- `Monospace Bold`
- *`Monospace Bold Italic`*

Lists

Items

- Milk
- Eggs
- Potatoes

Enumerations

1. First,
2. Second and
3. Last.

Descriptions

PowerPoint Meeh.

Beamer Yeeeha.

Animation

- This is important

Animation

- This is important
- Now this

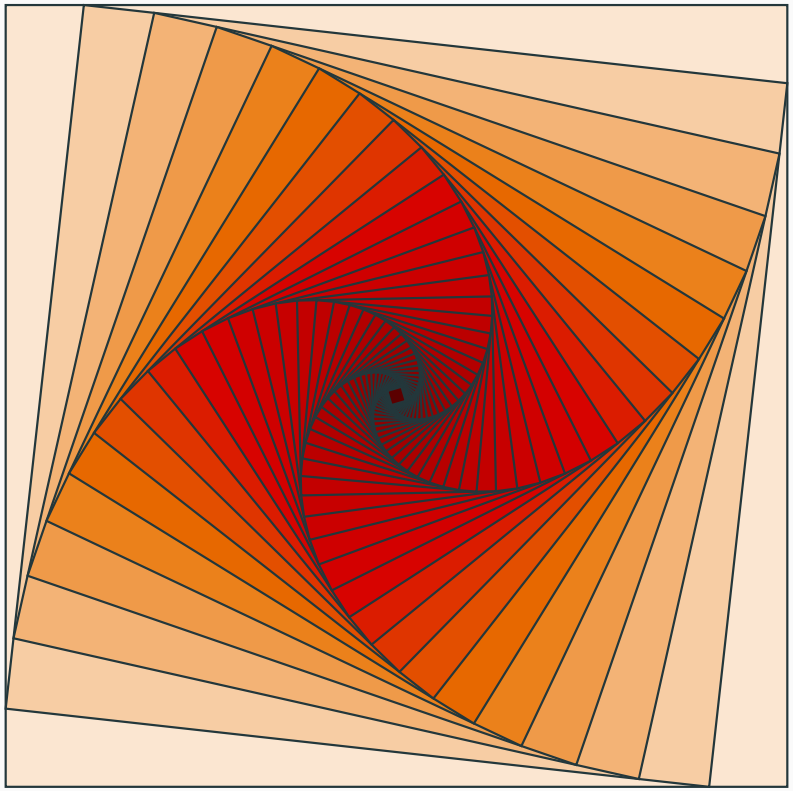
Animation

- This is important
- Now this
- And now this

Animation

- This is really important
- Now this
- And now this

Figures



Hình 1: Rotated square from texample.net.

Tables

Bảng 1: Largest cities in the world (source: Wikipedia)

City	Population
Mexico City	20,116,842
Shanghai	19,210,000
Peking	15,796,450
Istanbul	14,160,467

Blocks

Three different block environments are pre-defined and may be styled with an optional background color.

Default
Block content.

Alert
Block content.

Example
Block content.

Default

Block content.

Alert

Block content.

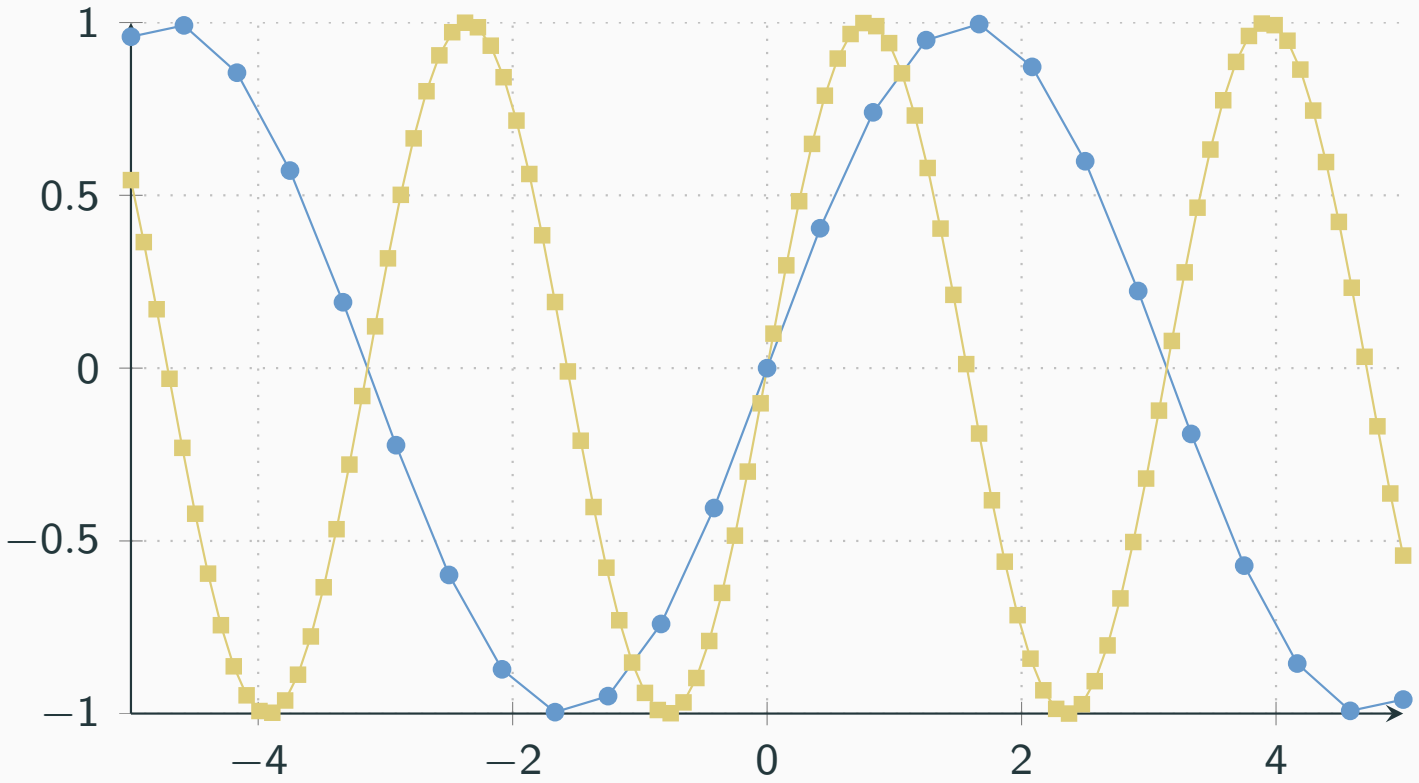
Example

Block content.

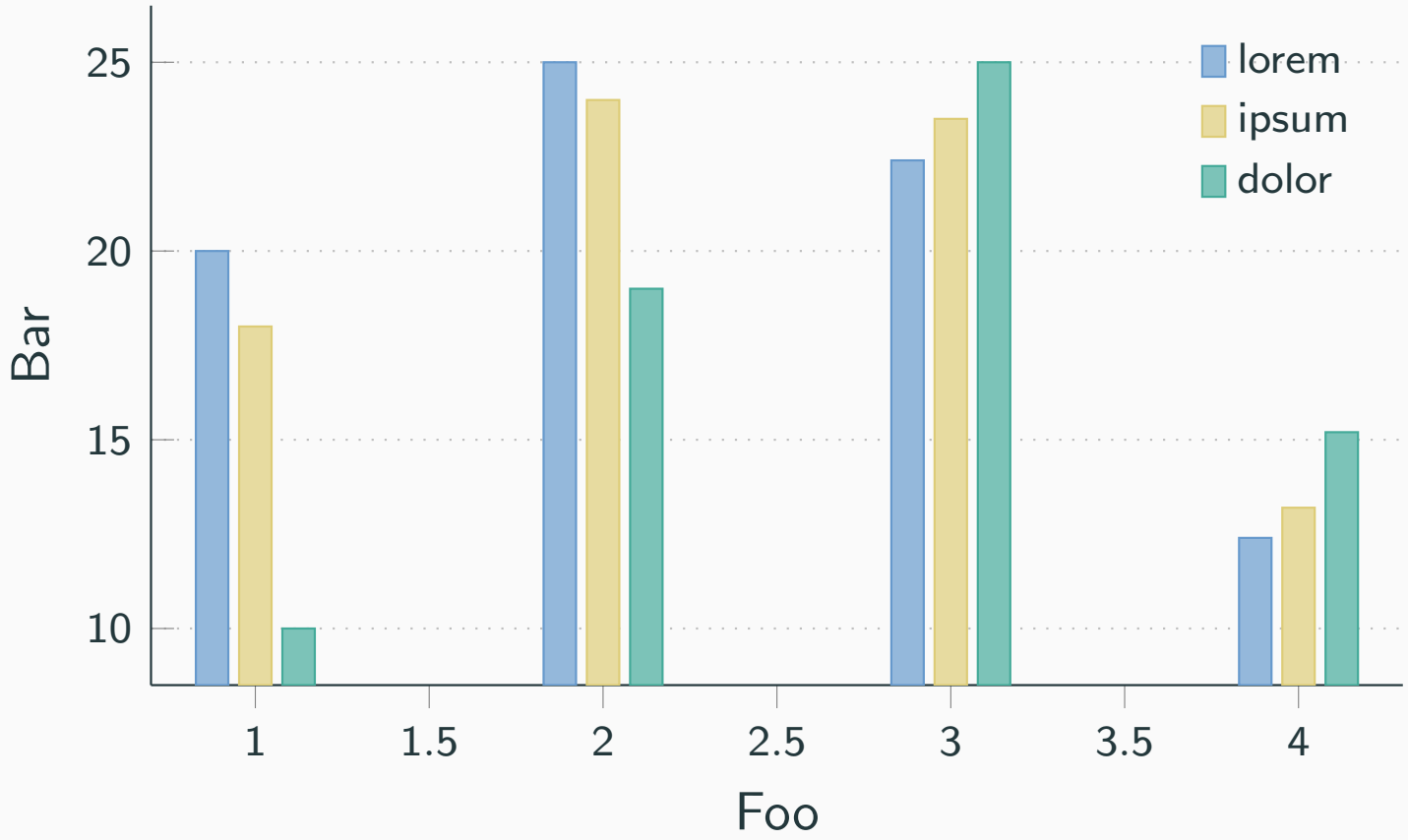
Math

$$e = \lim_{n \to \infty} \left(1 + \frac{1}{n}\right)^n$$

Line plots



Bar charts



Quotes

Veni, Vidi, Vici

Frame footer

metropolis defines a custom beamer template to add a text to the footer. It can be set via

```
\setbeamertemplate{frame footer}{My custom footer}
```

References

Some references to showcase [allowframebreaks] [2, 4, 1, 3, 5]

Kiến thức nền tảng

Kiến thức nền tảng

Get the source of this theme and the demo presentation from

`github.com/matze/mtheme`

Phương pháp thực hiện

Bài toán 1

Bài toán 1: Phát triển thuật toán để ước lượng số lượng phần tử (cardinality estimation) trong một khoảng thời gian trên một dòng dữ liệu (data stream):
Trong phương pháp này, chúng tôi sẽ trình bày quá trình phát triển thuật toán sử dụng HyperLogLog để ước lượng số lượng phần tử trên một dòng dữ liệu. Bằng cách sử dụng cấu trúc dữ liệu HyperLogLog thep khung thời gian, ví dụ như mỗi giờ hoặc mỗi phút và các kỹ thuật tối ưu, chúng tôi xây dựng một thuật toán hiệu quả và chính xác để đếm số lượng phần tử duy nhất trong dữ liệu dòng.

- Bước 1: Xác định khoảng thời gian Đầu tiên, chúng tôi sẽ xác định khoảng thời gian mà chúng tôi muốn đếm số lượng phần tử. Ví dụ, mỗi giờ hoặc mỗi phút.
- Bước 2: Lưu trữ HyperLogLog Tiếp theo, chúng tôi sẽ lưu trữ cấu trúc HyperLogLog cho mỗi khoảng thời gian. Cấu trúc dữ liệu sẽ bao gồm cặp $\langle T_1, HLL_1 \rangle$, trong đó T_1 là thời điểm đại diện cho khung thời gian cụ thể.
- Bước 3: Sử dụng kết quả Cuối cùng, khi cần, chúng tôi có thể truy

Bài toán 2

Bài toán 2: Mở rộng thuật toán để ước lượng số lượng phần tử trong một khoảng thời gian trên nhiều dòng dữ liệu: Trong phương pháp này, chúng tôi sẽ sẽ mở rộng thuật toán 1 để ước lượng trên nhiều dòng dữ liệu. Ví dụ khi chúng ta cần biết có bao nhiêu người dùng đã đăng nhập vào hệ thống vào ngày hôm qua, do dữ liệu người dùng được lưu ở trên nhiều hệ thống như web, application và cũng như trên các bộ phận khác nhau của doanh nghiệp. Khi đó chúng ta sẽ có nhiều nguồn dữ liệu khác nhau và cần một thuật toán để kết hợp các nguồn dữ liệu này để tổng hợp cho ra ước lượng số lượng cuối cùng.

- Bước 1: Tổng hợp dữ liệu Đầu tiên, chúng ta đã lưu trữ dữ liệu trên một dòng dữ liệu như thuật toán ở trên.
- Bước 2: Tổng hợp HyperLogLog Tiếp theo, chúng tôi sẽ sẽ tiến hành tổng hợp các dữ liệu từ nhiều nơi khác nhau $\langle T_1, HLL_1 \rangle, \langle T_2, HLL_2 \rangle, ..., \langle T_N, HLL_N \rangle$, trong đó $T_1, T_2, ..., T_N$ là các khoảng thời gian giống nhau nên $T_1 = T_2 = T_N$ và đặt chung là T, và HLL_1 là dữ liệu HyperLogLog trong khoảng thời gian, ví dụ từ 12:00 ngày hôm qua cho đến 12:00 ngày hôm nay.

KẾ HOẠCH TRIỂN KHAI

Kế hoạch triển khai

#	Tuần	Nội dung công việc
1	1 - 2	Tìm hiểu thêm các bài báo liên quan mới nhất và bổ sung cơ sở lý thuyết về các kỹ thuật ước lượng số lượng trên dòng dữ
2	3 - 4	Thu thập dữ liệu, chuẩn hoá và tiền xử lý. Hiện thực bài toán 1 ước lượng số lượng phần tử trên dòng dữ liệu
3	5 - 6	Hiện thực bài toán mở rộng để ước lượng số lượng phần tử tron trên nhiều dòng dữ liệu. Đánh giá hiệu suất và độ chính xác.
4	7 - 8	Phân tích và so sánh kết quả, đánh giá ưu nhược điểm của từng Đề suất phương pháp hay tối ưu hiệu suất và độ chính xác.
5	9 - 10	Ứng dụng kết quả nghiên cứu. Đề suất hướng phát triển và nghiên cứu tiếp theo.
6	11 - 12	Đề xuất và đánh giá các giải pháp
7	1 - 14	Tổng hợp kết quả và viết báo cáo

Thời gian dự kiến làm luận văn là từ ngày **05/09/2024** *đến* **15/12/2024**

Nội dung dự kiến của luận văn

Chương 1: Giới thiệu. Chương này nhấn mạnh về tầm quan trọng của việc phát triển kỹ thuật đếm số phần tử trên dòng dữ liệu trong ngữ cảnh dữ liệu lớn. Qua đó, ta sẽ hiểu rõ hơn về phạm vi và mục tiêu của nghiên cứu, cũng như cách tiếp cận vấn đề.

Chương 2: Các công trình nghiên cứu liên quan. Trong chương này, sẽ được trình bày các công trình nghiên cứu liên quan, khám phá các phương pháp giải quyết vấn đề cũng như những phạm vi và giới hạn của các nghiên cứu đó. Qua đó, ta có thể đánh giá tính khả thi của đề tài.

Chương 3: Kiến thức nền tảng. Trong chương này giới thiệu về dòng dữ liệu, và các tính chất của nó. Các phương pháp truy vấn và xử lý dữ liệu trên dòng dữ liệu. Giới thiệu về HyperLogLog và nguyên lý hoạt động của nó. Các kiến thức cơ bản về ước lượng số lượng phần tử trong dữ liệu lớn, và các khái niệm về đánh giá hiệu suất và độ chính xác trong đếm số phần tử trên dòng dữ liệu.

Chương 4: Hiện thực và thử nghiệm. Trong chương này sẽ trình bày chi tiết cách thức hiện thực của từng thuật toán.

Chương 5: Kết quả và đánh giá. Trong chương này sẽ nêu ra các kết

Kết luận

Kết luận

Việc giám sát và quản lý số lượng người dùng đóng vai trò quan trọng trong việc tối ưu hóa hiệu quả hoạt động, nâng cao trải nghiệm người dùng, hỗ trợ ra quyết định kinh doanh sáng suốt và đảm bảo an ninh mạng cho doanh nghiệp.

Phân bổ tài nguyên hợp lý: Đảm bảo hệ thống hoạt động ổn định, tránh quá tải, lãng phí tài nguyên, tối ưu hóa chi phí vận hành.

Nâng cao trải nghiệm người dùng: Giảm thiểu lỗi hệ thống, lag, giật, loading lâu, mang đến trải nghiệm mượt mà, thu hút và giữ chân khách hàng.

Phát hiện và khắc phục sự cố kịp thời: Nhận diện sớm các dấu hiệu bất thường, sự cố hệ thống, từ đó có biện pháp khắc phục nhanh chóng, hạn chế ảnh hưởng đến hoạt động kinh doanh.

Hiểu rõ hành vi người dùng: Phân tích dữ liệu truy cập, hành vi click chuột, sở thích, nhu cầu của người dùng để cá nhân hóa trải nghiệm, đề xuất sản phẩm/dịch vụ phù hợp, nâng cao hiệu quả marketing và dự báo xu hướng thị trường.

Hỗ trợ ra quyết định kinh doanh: Đánh giá hiệu quả chiến dịch

Questions?

Backup slides

Sometimes, it is useful to add slides at the end of your presentation to refer to during audience questions.

The best way to do this is to include the `appendixnumberbeamer` package in your preamble and call `\appendix` before your backup slides.

metropolis will automatically turn off slide numbering and progress bars for slides in the appendix.

Tài liệu tham khảo i

Y. Chabchoub and G. Heébrail.
Sliding hyperloglog: Estimating cardinality in a data stream over a sliding window.
In *2010 IEEE International Conference on Data Mining Workshops*, pages 1297–1303. IEEE, 2010.

M. Durand and P. Flajolet.
Loglog counting of large cardinalities.
In *Algorithms-ESA 2003: 11th Annual European Symposium, Budapest, Hungary, September 16-19, 2003. Proceedings 11*, pages 605–617. Springer, 2003.

O. Ertl.
Exaloglog: Space-efficient and practical approximate distinct counting up to the exa-scale.
arXiv preprint arXiv:2402.13726, 2024.

Tài liệu tham khảo ii

P. Flajolet, É. Fusy, O. Gandouet, and F. Meunier.
Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm.
Discrete mathematics & theoretical computer science, (Proceedings), 2007.

S. Heule, M. Nunkesser, and A. Hall.
Hyperloglog in practice: Algorithmic engineering of a state of the art cardinality estimation algorithm.
In *Proceedings of the 16th International Conference on Extending Database Technology*, pages 683–692, 2013.