CALCOLO DEL DIAMETRO NEI MODELLI DI STREAMING E SLIDING-WINDOW

ALESSIO MARINUCCI

RICCARDO FELICI

Sommario

Introduzione

Obiettivo dello studio

Analisi del problema del calcolo del diametro nei modelli di streaming e sliding-window.

Sviluppo di algoritmi approssimati ed esatti.

Adattamento ai flussi di dati multidimensionali.

Motivazioni

L'IMPORTANZA DEI DATASET MASSIVI

APPLICAZIONI PRATICHE

INNOVAZIONE ALGORITMICA

Definizione del Problema

- **Calcolo del Diametro**: massima distanza tra due punti in un insieme.
- **Modelli di dati utilizzati**: streaming e sliding-window.
- **Limiti da rispettare**: restrizioni di spazio, tempo e accuratezza.

Contesto e Lavori Correlati

Modelli Utilizzati

Modello di streaming: I dati arrivano sotto forma di flusso continuo e non possono essere memorizzati interamente. Lo spazio disponibile è limitato e gli algoritmi devono elaborare ogni dato in tempo reale.

Modello sliding-window: Si considerano solo i dati più recenti all'interno di una finestra temporale o di dimensione fissata. L'obiettivo è calcolare il diametro in base ai dati "attivi", ignorando quelli più vecchi.

Sfide Tecniche

Limiti di spazio

Limiti di tempo

Dati dinamici

Modello sublineare

Trade-off tra accuratezza e risorse

Alta dimensionalità dei dati

Lavori Precedenti

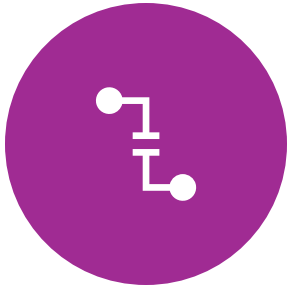
Istogrammi in Streaming

Quantili in Streaming

Norme in Streaming

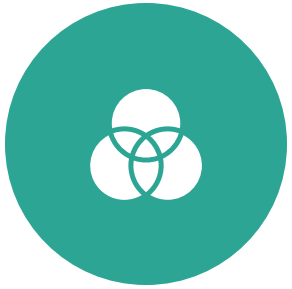
Tecniche Geometriche per Proprietà Avanzate

Limiti degli studi precedenti



Istogrammi in Streaming: Le approssimazioni possono non catturare accuratamente la distribuzione dei dati, specialmente con flussi ad alta varianza.

Quantili in Streaming: Le approssimazioni dei quantili potrebbero non essere abbastanza precise per applicazioni che richiedono stime esatte.



Norme in Streaming: Le approssimazioni delle norme, come quelle per le norme $L1$, $L2$ e $L\infty$, potrebbero ridurre la precisione nella stima di distanze massime, influenzando negativamente il calcolo del diametro.

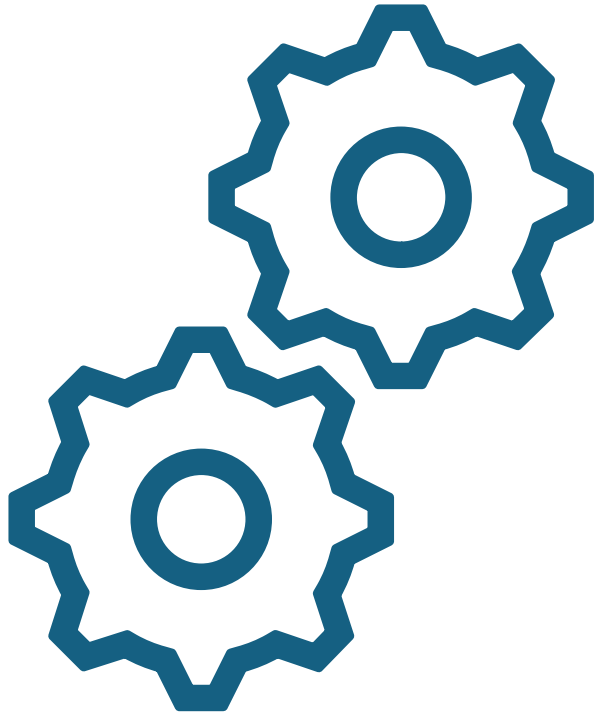
Tecniche Geometriche per Proprietà Avanzate: L'uso degli istogrammi radiali e dell'inviluppo convesso è una tecnica potente per stimare il diametro, ma la suddivisione in settori angolari o il mantenimento di soli punti rappresentativi può escludere punti significativi che influenzano la distanza massima.

Applicazioni pratiche

TELEFONIA WIRELESS

RETI DI SENSORI

Algoritmi e Metodologie



Algoritmo di Streaming

- Una prima soluzione dell'algoritmo per approssimare il diametro, con un errore limitato, può scorrere i punti del piano una sola volta.
- Per ogni punto proietta la posizione su una serie di linee con diverse inclinazioni.
- Mantiene solo i punti estremi per ogni linea.
- Quando proiettiamo due punti su linee consecutive con angoli leggermente diversi, la distanza tra i punti proiettati può cambiare leggermente.
- Questo cambiamento dipende dall'angolo tra le due linee consecutive. Se l'angolo è troppo grande, questo cambiamento potrebbe essere significativo, introducendo un errore maggiore di ε ·Diametro esatto.
- Questo significa che l'angolo tra linee consecutive deve essere al massimo ε.

Algoritmo di Streaming

1. Prendi il primo punto del flusso come centro e dividi il piano in settori in base a un angolo $\theta = \frac{\varepsilon}{2(1-\varepsilon)}$ dove ε è il margine d'errore. Sia *S* l'insieme dei settori.
2. Durante l'attraversamento del flusso, per ciascun settore, registra il punto più lontano in quel settore rispetto al centro. Mantieni anche la massima distanza, R_c, tra il centro e qualsiasi altro punto in *P*.
3. Sia | ab | la distanza tra i punti *a* e *b*. Definiamo $D_{max}^{ij} = \max |uv|$ la distanza massima tra i punti sugli archi dei settori *i* e *j*, e definiamo $D_{min}^{ij} = \min |uv|$ la distanza minima tra i punti sugli stessi archi. Restituisci $\max\{R_c, max_{i,j \in S}\ D_{min}^{ij}\}$ come il diametro dell'insieme di punti *P*.

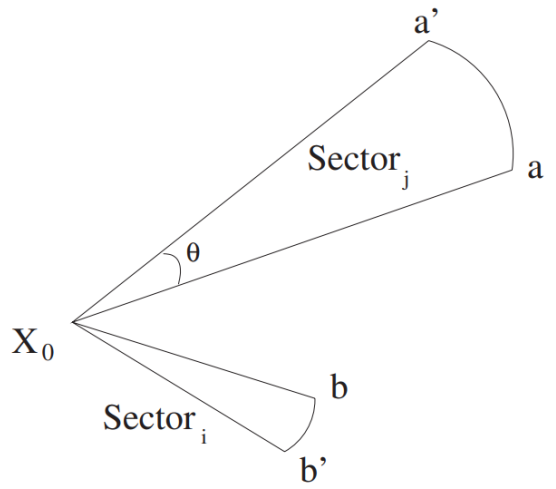
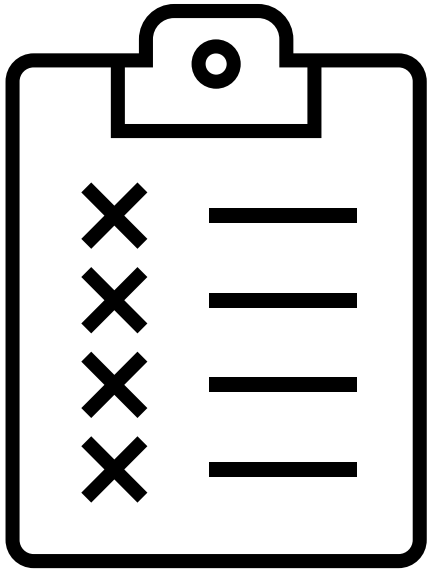


Fig. 1. Two examples of sectors.

Errore Approssimativo



L'errore approssimativo si riferisce alla **differenza tra il valore esatto** e la **stima** calcolata tramite algoritmi approssimati.

Gli algoritmi di streaming e sliding-window introducono un errore perché **non conservano tutti i dati**. L'errore dipende dal livello di **approssimazione** scelto.

TRADE-OFF

Un errore maggiore permette di risparmiare memoria e tempo di calcolo, ma riduce la precisione del risultato. Aumentare la memoria (e quindi la capacità di elaborazione) può ridurre l'errore, ma aumenta il costo computazionale.

CLAIM 1

DEFINIZIONE

- La distanza tra due punti in un settore *i* e un settore *j* non è maggiore di $\max\{R_c, D_{max}^{ij}\}$
  - $|uv| \leq \max\{R_c, D_{max}^{ij}\}$

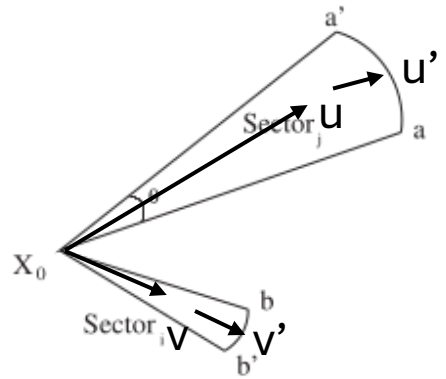


Fig. 1. Two examples of sectors.

DIMOSTRAZIONE

- Sia u un punto nel settore i e sia v un punto nel settore j. Estendiamo x_0u fino a raggiungere l'arco aa'. Indichiamo il punto di intersezione con u'. Allo stesso modo, estendiamo x_0v fino a raggiungere l'arco bb'. Indichiamo il punto di intersezione con v'
- Limitiamo $|uu'|$ e $|vv'|$
- Distanza $|uu'| \leq R_c$ e $|vv'| \leq R_c$
- Distanza $|u'v'| \leq D_{max}^{ij}$ (distanza tra aa' e bb')
- Utilizzando la disuguaglianza triangolare
  - $|uv| \leq |uu'| + |u'v'| + |v'v|$
  - $|uv| \leq R_c + D_{max}^{ij} + R_c$
  - $|uv| \leq \max\{R_c, D_{max}^{ij}\}$ ■

CLAIM 2

DEFINIZIONE

- $D_{max}^{ij} \leq D_{min}^{ij} + length(aa') + length(bb') \leq D_{min}^{ij} + 2R_c \cdot \theta.$

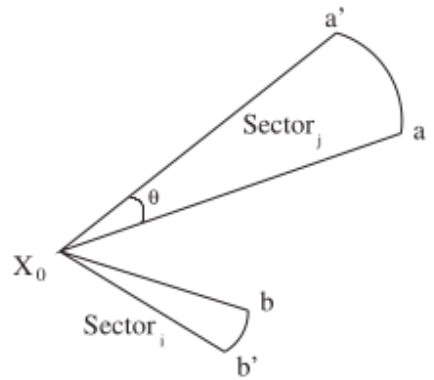
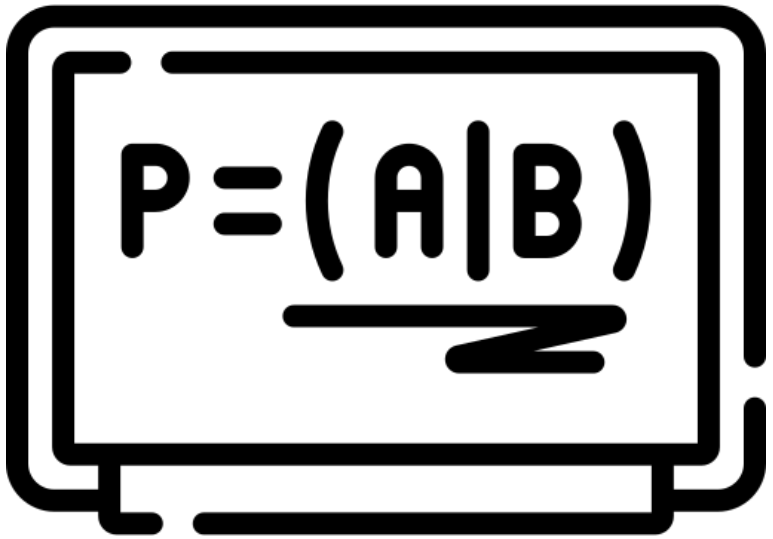


Fig. 1. Two examples of sectors.

DIMOSTRAZIONE

- Sia $| uv | = D_{max}^{ij}$ e $| u'v' | = D_{min}^{ij}$. Poiché u, u' appartengono all'arco aa' e v, v' a bb', esiste un percorso da u a v, ovvero $u \sim u' \sim v' \sim v$.
  $D_{max}^{ij} \leq |uu'| + D_{min}^{ij} + |vv'| \leq D_{min}^{ij} + 2R_c \cdot \theta$ ■

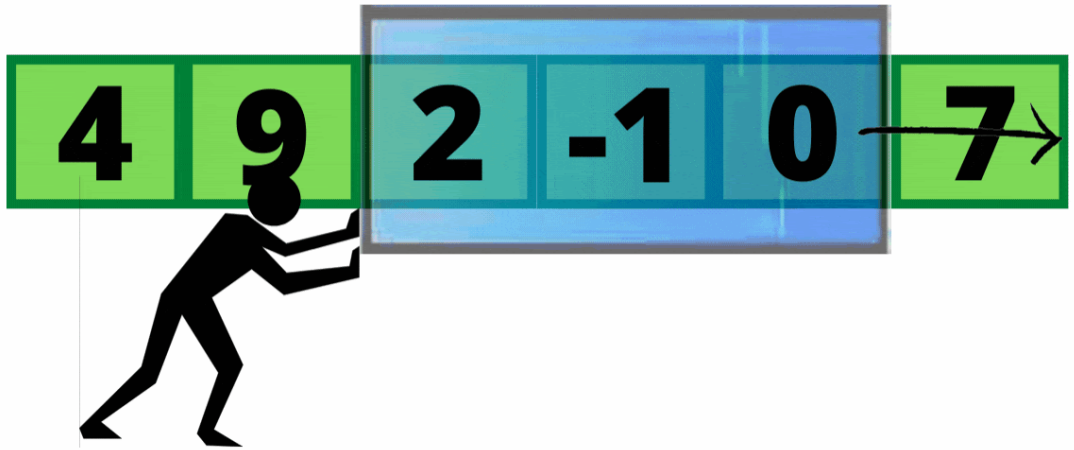
- Vogliamo calcolare il diametro approssimato ($diam$) con un errore ε
  - $\max\{R_c, D_{min}^{ij}\} \leq diam \leq diam_{true} \leq \max\{R_c, D_{max}^{ij}\}$
  1. **Caso 1**: $R_c \geq D_{min}^{ij}$
     - $R_c \geq (1-\varepsilon) D_{max}^{ij}$
  2. **Caso 2**: $R_c \leq D_{min}^{ij}$
     - $D_{min}^{ij} \geq (1-\varepsilon) D_{max}^{ij}$
  - $\theta \leq \frac{\varepsilon}{2(1-\varepsilon)}$



Teorema 1

Esiste un algoritmo che approssima il diametro bidimensionale nel modello di streaming con un'approssimazione ε, utilizzando spazio per $O(\frac{1}{\varepsilon})$ punti. Per elaborare ciascun punto, richiede $O(\log(\frac{1}{\varepsilon}))$.

Modello Sliding-Window



Rounding-Subroutine

Crea una rappresentazione efficiente in termini di spazio per un insieme di punti, chiamati **cluster**. Una volta costruito il cluster, il diametro dell'insieme originale può essere approssimato calcolando il diametro del cluster.

1. **Scelta del centro**: si seleziona un punto c come centro.
2. Dato un punto b si arrotonda verso un punto a se
$$|ac| \leq |bc| \leq (1 + \hat{\varepsilon})|ac|$$
3. Si divide lo spazio in intervalli $[c, t_0), [t_0, t_1), [t_1, t_2), \dots, [t_{k-1}, t_k]$ dove $|ct_i| = (1 + \hat{\varepsilon})^i d$. Con d distanza minima tra c e un altro punto. I punti nell'intervallo$[t_i, t_{i+1}]$ vengono arrotondati alla posizione t_i.
4. **Eliminazione dei duplicati**: se più punti sono arrotondati alla stessa posizione si conserva solo il punto più recente.

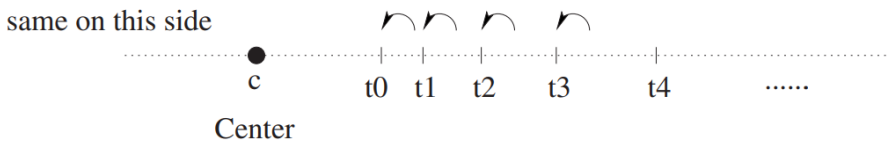


Fig. 2. Rounding points in each interval.

Cluster

- Un **cluster** è composto dal centro c e dai punti rimasti dopo l'arrotondamento.
- Diciamo che un punto a nel cluster rappresenta un punto b nell'insieme originale se b è stato arrotondato alla posizione di a dalla subroutine, indipendentemente dal fatto che b sia stato eliminato. Chiamiamo il punto a il **Punto Rappresentativo** del punto b originale.
- Il **volume** di un cluster è definito come il numero di punti nell'insieme originale che sono rappresentati dal cluster. La dimensione di un cluster è il numero di punti che compongono il cluster:

$$k \leq 2 \log_{(1+\hat{\varepsilon})} \frac{D}{d} = 2 \frac{\log {}^{D}/_{d}}{\log e \ln(1+\hat{\varepsilon})} \leq \frac{4}{\hat{\varepsilon} \log e} \log \frac{D}{d}$$

CLAIM 3

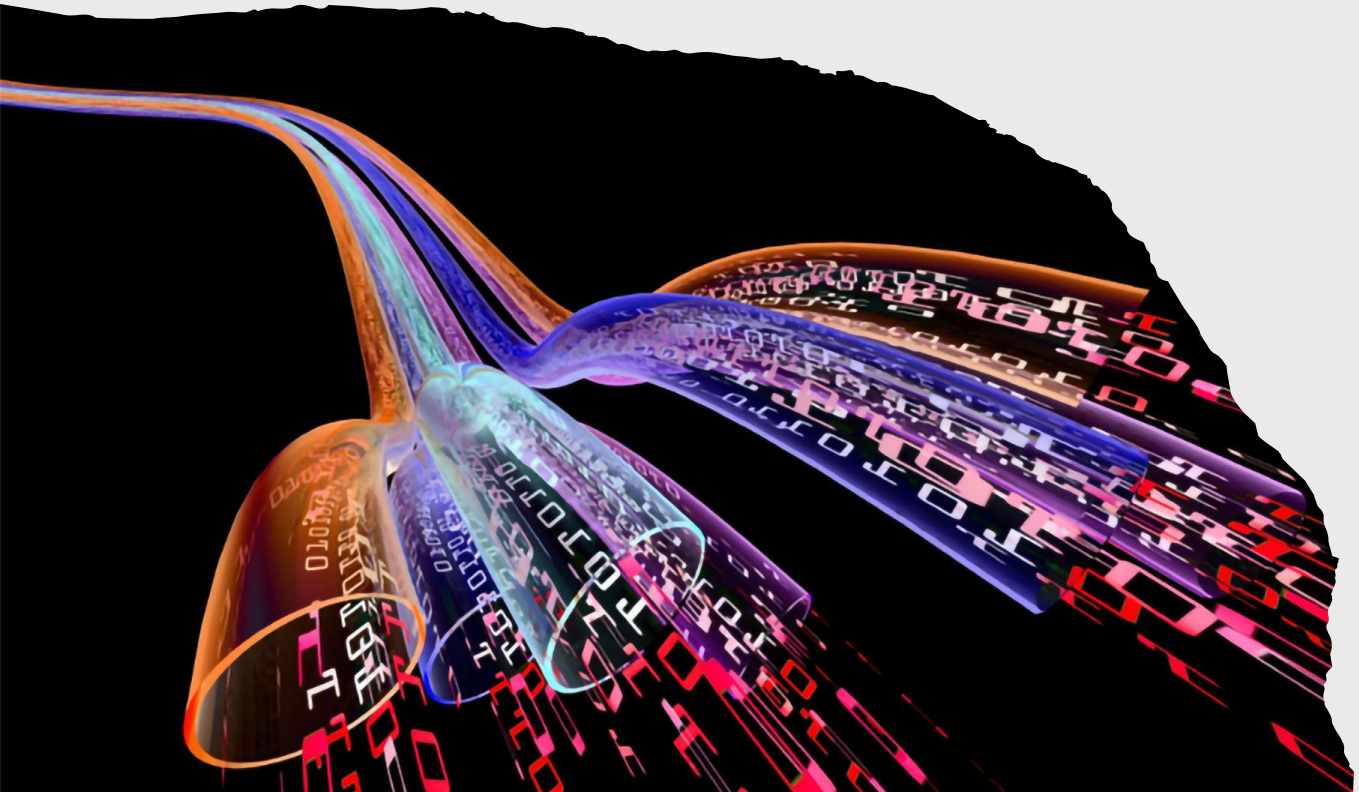
DEFINIZIONE

Quando la subroutine viene invocata su un insieme di punti:

1. Viene costruito un cluster che può essere utilizzato per approssimare il diametro dell'insieme
2. Se c'è uno spostamento tra un punto dell'insieme e il suo rappresentante nel cluster, il punto sarà sempre arrotondato **verso il centro** del cluster.

L'obiettivo è usare i cluster per rappresentare i punti in modo da ridurre i calcoli per approssimare il diametro, minimizzando gli errori introdotti dal processo di approssimazione

Algoritmo per Sliding-Window



1. ‘Clusterizzazione’ dei Punti

- Ogni cluster rappresenta un intervallo continuo nella finestra.
- Il punto più recente è scelto come centro.

2. Arrotondamento

- Applicare arrotondamento per rappresentare i punti con errore massimo controllato.

3. Struttura Multi-Livello

- Organizzare i cluster su livelli 1,2,…,$\lceil \log n \rceil$.
- Max 2 cluster per livello.

4. Combinazione di Cluster

- Se ci sono più di 2 cluster in un livello, si combinano i più vecchi in un livello superiore.

5. Coprire la Finestra

- Tutti i punti nella finestra sono rappresentati da $O(\log n)$ cluster.

Struttura del cluster

- I punti originali sono le foglie di un albero.
- Due punti consecutivi formano un cluster di livello 1.
- Due cluster consecutivi di livello 1 possono essere uniti per formare un cluster di livello 2.
- Il processo continua fino al livello più alto.
- Ad ogni livello si mantengono al massimo due cluster. Si rappresenta l'intera finestra con $O(\log n)$ cluster.
- Quando la finestra scorre in avanti
  - Nuovi punti vengono aggiunti
  - Nuovi cluster vengono formati
  - Se ci sono troppi cluster in un livello vengono uniti

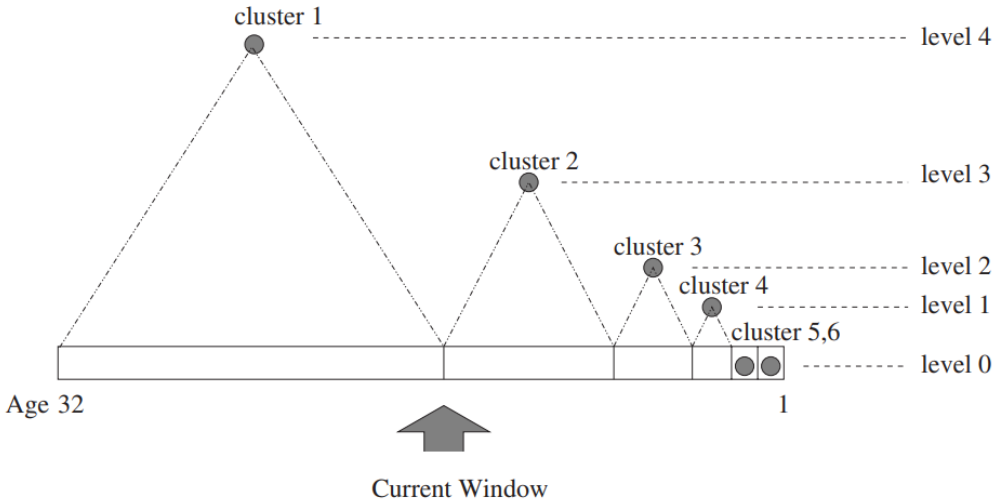
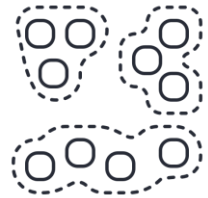


Fig. 3. Clusters built for the first window.

Processo di Fusione dei Cluster



Per unire due cluster c_1 e c_2, centrati rispettivamente in Ctr_1 e Ctr_2 (con Ctr_2 più recente di Ctr_1), i passi sono i seguenti:

1. **Centro del nuovo cluster**
   Usare Ctr_2 come centro del nuovo cluster c_3 , dato che è il punto più recente.
2. **Eliminazione di punti superflui**
   Rimuovere i punti di c_1 che si trovano tra Ctr_1 e Ctr_2, poiché questi punti non sono interessanti per il calcolo del diametro.
3. **Eliminazione di punti ridondanti**
   Per ogni punto p in c_1, se la distanza $| p - Ctr_2 |$ soddisfa la condizione: $|pCtr_2| < |pCtr_1| \leq (1 + \hat{\varepsilon})|Ctr_1Ctr_2|$ allora p viene eliminato per evitare ridondanze.
4. **Creazione del nuovo cluster**
   Definire P_{merge} come l'unione dei punti rimanenti in c_1 e di tutti i punti in c_2.
   Applicare una subroutine di arrotondamento a P_{merge}, usando Ctr_2 come centro.
5. **Ricalcolo della distanza minima**
   Il valore di d (distanza minima da Ctr_2 a un altro punto in P_{merge}) può cambiare rispetto al cluster originale c_2. La subroutine potrebbe arrotondare anche i punti di c_2 per garantire una rappresentazione compatta e un errore controllato.

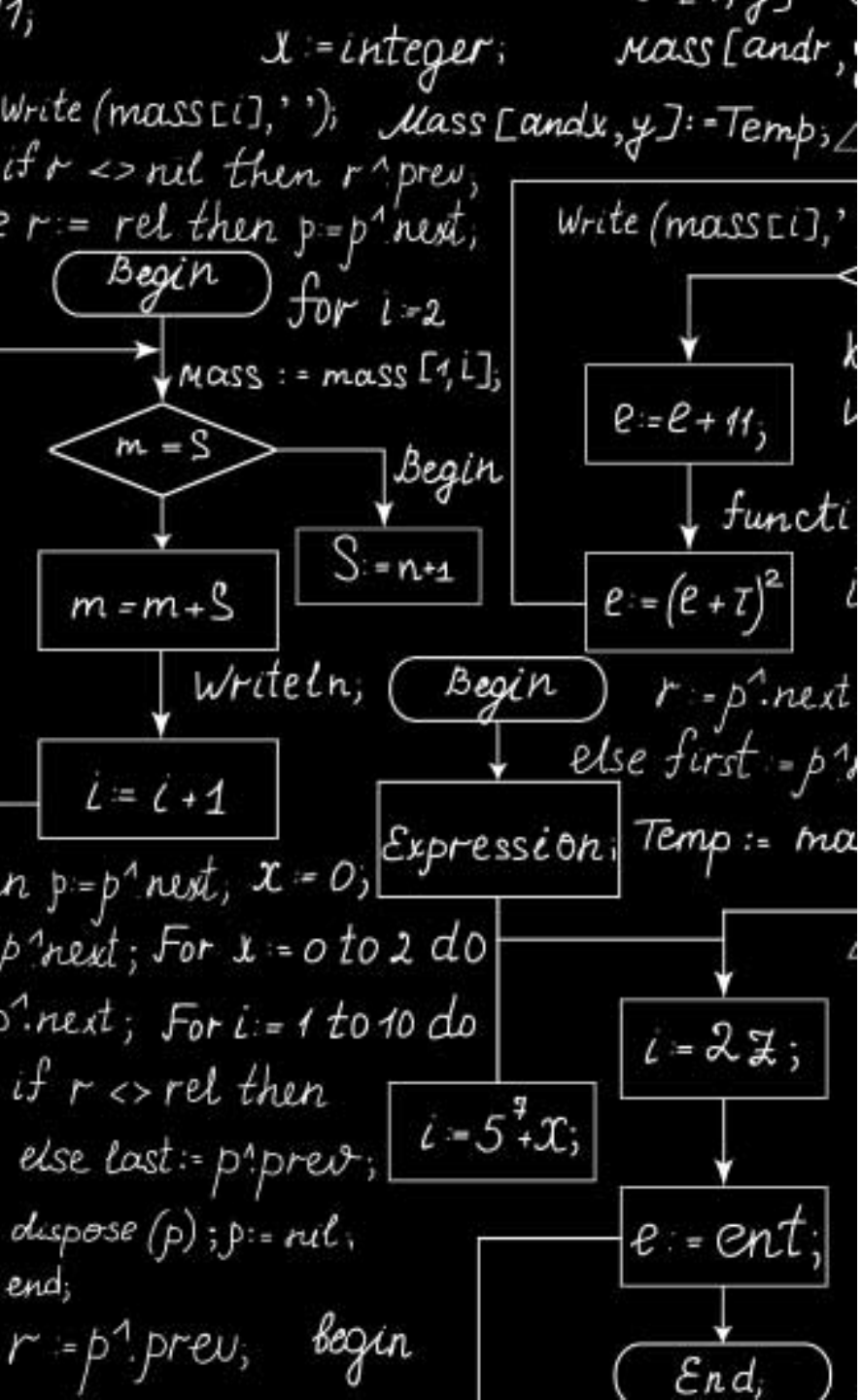
LEMMA 1

DEFINIZIONE

- Nella subroutine di arrotondamento o nel processo di fusione, un punto viene scartato se: non realizzerà alcun diametro o è rappresentato (da un punto rappresentativo) nel cluster risultante dalla subroutine di arrotondamento o dal processo di fusione.

DIMOSTRAZIONE

- **Caso dell'arrotondamento**: un punto b viene scartato solo se esiste già un altro punto rappresentativo a per b nel cluster, con a più recente di b.
- **Caso della fusione**:
  - Punti di c_1 che si trovano tra Ctr_1 e Ctr_2. Non possono formare un diametro, poiché ogni distanza che potrebbero realizzare è inferiore alla distanza realizzata da uno dei due centri
  - $S = \{p \in c_1 \mid |pCtr_2| < |pCtr_1| \leq (1 + \varepsilon)|Ctr_1Ctr_2|\}$
    I punti in S sono rappresentati da Ctr_2 nel cluster risultante. ■



Algoritmo Sliding-Window Diameter

Fase di Aggiornamento: Quando Arriva un Nuovo Punto

- **Verifica dei Punti di Confine del Cluster Più Vecchio**:
  - Se uno dei punti di confine è scaduto (cioè non è più nella finestra), viene rimosso e il cluster viene aggiornato di conseguenza.
- **Creazione di un Nuovo Cluster per il nuovo punto**:
  - Il punto appena arrivato forma un cluster di dimensione 1.
- **Fusione dei Cluster**:
  - A partire dal cluster più recente, si vanno a fondere i cluster, seguendo le regole predefinite.
  - Dopo ogni fusione, i punti di confine dei nuovi cluster risultanti dalla fusione vengono aggiornati.
- **Aggiornamento dei Punti di Confine della Finestra**:
  - Se necessario, vengono aggiornati i punti di confine che rappresentano l'intera finestra dopo il processo di fusione dei cluster.

Risposta alla Query: Report del Diametro della Finestra

- La query comporta la restituzione della distanza tra i punti di confine della finestra, che viene definita come **diametro della finestra**.
- Il **diametro della finestra** è la distanza tra i punti estremi (la coordinata più grande e la più piccola) dell'intera finestra.

CLAIM 4

DEFINIZIONE

Se un punto viene arrotondato più volte durante la sua durata di vita, tutti i dislocamenti dovuti all'arrotondamento avvengono nella stessa direzione. Ovvero per tutte le posizioni p_i e i corrispondenti centri Ctr_i, $|p_0Ctr_i| \geq |p_iCtr_i|$.

Se un punto realizza $diam_p$, allora la distanza tra il punto originale p_0 e qualsiasi centro di cluster non supera il diametro $diam_p$.
Per $i = 1,2, \ldots, t$ $|p_0Ctr_i| \leq diam_p$

Fig. 4. A point may be moved in each rounding but all the displacements are in the same direction.

DIMOSTRAZIONE

1. Supponiamo che la prima volta che il punto p viene arrotondato venga spostato a destra.
2. Se successivamente p viene arrotondato a sinistra per la prima volta al passo i, allora secondo il CLAIM 3, il centro Ctr_{i-1} si trova a destra di p, mentre Ctr_i a sinistra.
3. Inoltre, p appartiene al cluster con centro Ctr_{i-1} prima della fusione.
4. Secondo le regole del processo, p sarebbe stato scartato, poiché si trova tra i due centri (Ctr_{i-1} e Ctr_i) e appartiene al cluster più vecchio.
5. Inoltre come mostrato nella prova del LEMMA 1, un punto del genere non realizza il diametro e non sarebbe più rappresentato da alcun punto di riferimento nei cluster della finestra, quindi viene scartato. ■

LEMMA 2

DEFINIZIONE

L'errore totale di arrotondamento di un punto p, prima che scada o non venga più rappresentato da alcun punto di riferimento nella finestra è al massimo $\hat{\varepsilon} \log n \cdot diam_p$.

DIMOSTRAZIONE: CASO 1: ARROTONDAMENTO

1. Quando si arrotonda il punto, si mantiene la condizione che $|p_i Ctr_{i+1}| \leq (1 + \hat{\varepsilon})|p_{i+1} Ctr_{i+1}|$.
2. Quindi, l'errore di arrotondamento $Err_{i+1} = |p_i p_{i+1}|$ è limitato da $\hat{\varepsilon}|p_{i+1} Ctr_{i+1}|$.
3. Poiché secondo il CLAIM 4, $|p_0 Ctr_{i+1}|$ è una stima della distanza tra il punto originale e il centro del cluster, possiamo affermare che
$$Err_{i+1} \leq \hat{\varepsilon}|p_0 Ctr_{i+1}|$$
vincolando l'errore di arrotondamento.

LEMMA 2

DIMOSTRAZIONE: CASO 2: FUSIONE DI CLUSTER

1. Nel processo di fusione, se p_i soddisfa una certa condizione verrà scartato.
2. Secondo la prova del LEMMA 1, in questo caso, il nuovo punto rappresentante di p sarà Ctr_{i+1} e quindi p_{i+1} sarà Ctr_{i+1}. L'errore di arrotondamento in questa fase sarà:

   $$Err_{i+1} = |p_i p_{i+1}| \leq \hat{\varepsilon}|Ctr_i Ctr_{i+1}| \leq |p_i Ctr_i|$$

   Ancora una volta usando il CLAIM 4 l'errore di arrotondamento è limitato da

   $$Err_{i+1} \leq \hat{\varepsilon}|p_0 Ctr_i|$$

   Un punto può partecipare a un massimo di $\log n$ fusioni, quindi l'errore totale dovuto agli spostamenti è:

   $$\sum_i Err_i = \hat{\varepsilon} \log n \cdot diam_p$$ ■

   Per limitare l'errore a $\frac{1}{2}\varepsilon$ si impone che $\hat{\varepsilon} \leq \frac{\varepsilon}{2 \log n}$ con n numero di elementi in finestra.

LEMMA 2

NUMERO DI PUNTI IN UN CLUSTER

Dopo il processo di arrotondamento il numero di punti in un cluster è al massimo $O(\frac{1}{\varepsilon}\log n \log\left(\frac{D}{d}\right))$.

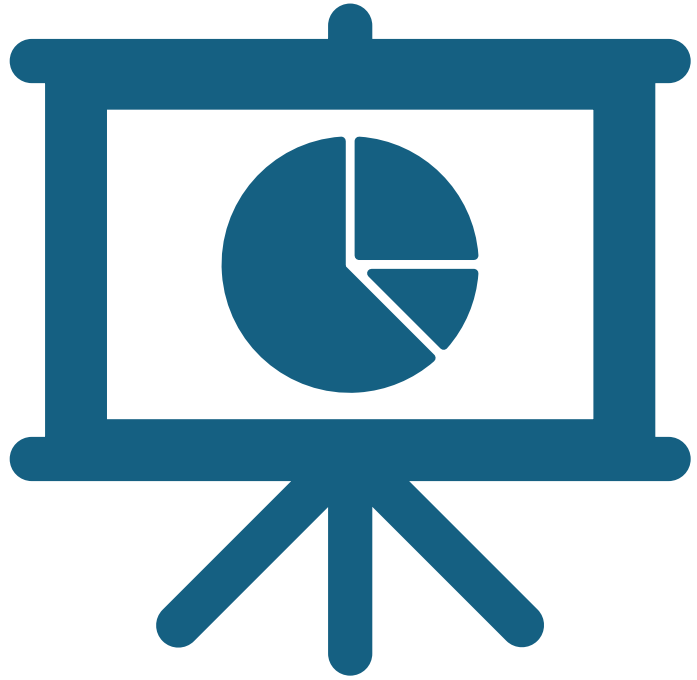
- **Bounding of Distance d:**
  - Ogni cluster ha un centro, e la distanza d tra il centro del cluster e qualsiasi altro punto originale rappresentato nel cluster è limitata da $\hat{\varepsilon}$ volte la distanza minima tra il centro del cluster e un altro punto originale.
  - Questo significa che, se il centro del cluster è troppo lontano rispetto alla distanza minima tra i punti originali, il processo di arrotondamento potrebbe non essere accurato, ma questo errore viene controllato.
- **Definizione di R:**
  - Denotiamo con R il massimo rapporto tra il diametro e la distanza minima non nulla tra due punti originali in una finestra, su tutte le finestre. R rappresenta quanto può variare il diametro rispetto alla distanza minima tra due punti in una finestra.

$$\log(\frac{D}{d}) \leq \log R + \log\frac{1}{\varepsilon} = \mathrm{O}(\log R + \log\log n + \log\frac{1}{\varepsilon})$$

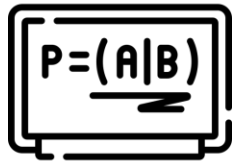
Il numero di punti in un cluster può quindi essere vincolato da

$$O\left(\frac{1}{\varepsilon}\log n\left(\log R + \log\log n + \log\frac{1}{\varepsilon}\right)\right)$$

Risultati e Teoremi



TEOREMA 2



DEFINIZIONE:

Esistenza di un algoritmo ε-approssimato per calcolare il diametro in una dimensione su una finestra mobile di dimensione n:

- Utilizza $O(\frac{1}{\varepsilon}\log^3 n\,(\log R + \log\log n + \log(\frac{1}{\varepsilon})))$ bit di spazio
- Risponde ad una query sul diametro in tempo $O(1)$
- Ad ogni scorrimento della finestra, processa un nuovo punto con un tempo massimo di

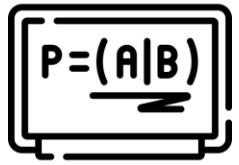
$$O\left(\frac{1}{\varepsilon}\log^2 n\left(\log R + \log\log n + \log\left(\frac{1}{\varepsilon}\right)\right)\right)$$

- Con una modifica, è possibile processare i punti in tempo ammortizzato $O(\log n)$, usando

$$O\left(\frac{1}{\varepsilon}\log^2 n\left(\log n + \log\log R + \log\left(\frac{1}{\varepsilon}\right)\right)\right)\left(\log R + \log\log n + \log\left(\frac{1}{\varepsilon}\right)\right)$$

bit di spazio.

TEOREMA 2

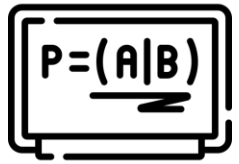


DIMOSTRAZIONE:

1. Dal LEMMA 1, ogni punto che può contribuire al diametro in una finestra ha un punto rappresentativo in uno dei cluster mantenuti dall'algoritmo

2. Per imporre l'errore limitato si impone $\hat{\varepsilon} \leq \frac{\varepsilon}{2 \log n}$
   - Dal LEMMA 2, il massimo spostamento tra la posizione originale di un punto p e il suo punto rappresentativo è $\hat{\varepsilon} \log n \cdot diam_p$.
     Dove $diam_p$ p il diametro realizzato da p.
   - Dato che $\hat{\varepsilon} \log n \leq \frac{\varepsilon}{2}$, l'errore massimo introdotto da un punto rappresentativo è $\frac{\varepsilon}{2} \cdot diam_p$

TEOREMA 2



DIMOSTRAZIONE:

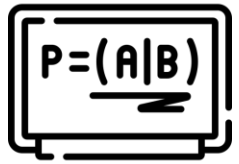
3. Poiché il diametro calcolato dall'algoritmo è determinato da due punti rappresentativi, l'errore totale è dato dalla somma degli errori di ciascun punto rappresentativo

$$Errore\ Totale \leq \frac{\varepsilon}{2} \cdot diam_p + \frac{\varepsilon}{2} \cdot diam_q = \varepsilon \cdot diam_{TRUE}$$

   Quindi il diametro riportato dall'algoritmo è compreso tra $(1 - \varepsilon)$ volte il diametro vero e il valore esatto del diametro vero.

4. Supponiamo che il diametro riportato dall'algoritmo sia maggiore del diametro vero. Ciò implicherebbe che la distanza tra due punti rappresentativi $|R(p)R(p')|$ è maggiore della distanza tra i punti originali $|pp'|$.
   Tuttavia ciò contraddice il CLAIM 4, che garantisce che lo spostamento di un punto verso un centro durante il processo di arrotondamento non può aumentare la distanza tra i punti rappresentativi rispetto ai punti originali. Il processo di clustering approssima i punti in modo da minimizzare le distanze interne ai cluster o tra i centri. Questo implica che la distanza massima tra i rappresentativi (i centri) è sempre **inferiore o uguale** alla distanza massima tra i punti originali.

TEOREMA 2



DIMOSTRAZIONE: INFORMAZIONI MANTENUTE PER CIASCUN CLUSTER

1. Posizione del centro e punto più vicino al centro, per calcolare velocemente i limiti del cluster
2. Età di tutti i punti, in base alla finestra corrente. Sono necessari $\boldsymbol{O(\log n)}$ bit per punto
3. Posizioni relative dei punti rispetto al centro. Codificate tramite un vettore di bit o altre rappresentazioni compatte.
4. **Requisiti di spazio**:
   - Numero massimo di punti in un cluster:

$$O(\frac{1}{\varepsilon}\log n(\log R + \log\log n + \log\frac{1}{\varepsilon}))$$

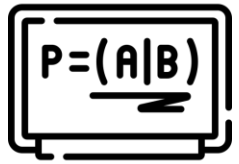
   - Spazio totale per mantenere le informazioni relative a tutti i punti di un cluster:

$$O(\frac{1}{\varepsilon}\log^2 n(\log R + \log\log n + \log\frac{1}{\varepsilon}))$$

   - Spazio complessivo, considerando $O(\log n)$ cluster:

$$O(\frac{1}{\varepsilon}\log^3 n(\log R + \log\log n + \log\frac{1}{\varepsilon}))$$

TEOREMA 2



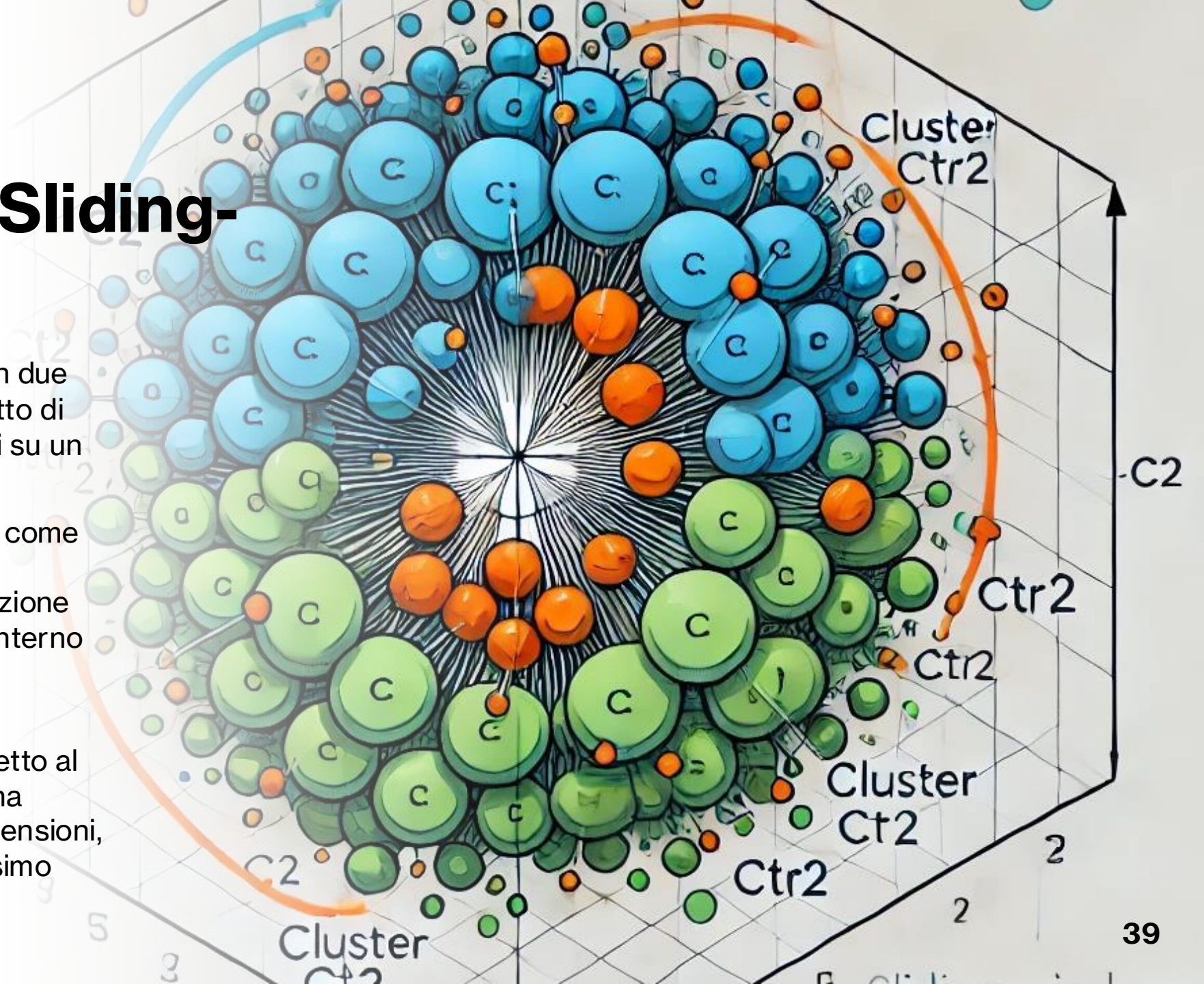
DIMOSTRAZIONE: TEMPO DI AGGIORNAMENTO DEI CLUSTER

1. **Gestione delle fusioni:**
   - In condizioni peggiori si possono fondere $O(\log n)$ cluster
   - Ogni cluster contiene $O(\frac{1}{\varepsilon}\log n(\log R + \log\log n + \log\frac{1}{\varepsilon}))$ punti
   - Tempo massimo per la fusione: $O(\frac{1}{\varepsilon}\log^2 n(\log R + \log\log n + \log\frac{1}{\varepsilon}))$
2. **Uso di vettori di bit per ottimizzare le posizioni relative**:
   - Se i vettori di bit sono sparsi si rischia di sprecare tempo
   - Alternativa: rappresentare direttamente le posizioni relative di ciascun punto
   - Spazio richiesto per ogni punto: $O(\log n + \log\log R + \log\frac{1}{\varepsilon})$
   - Con questa modifica, la gestione del cluster durante la fusione non introduce overhead.
3. **Costo ammortizzato**:
   - Un punto partecipa a massimo $O(\log n)$ fusioni durante la sua vita
   - Costo ammortizzato per aggiornamento: $O(\log n)$

■

Estensione all’algoritmo Sliding-Window 2D

- L'algoritmo sliding-window in due dimensioni estende il concetto di finestra scorrevole per punti su un piano bidimensionale.
- Ogni punto è rappresentato come una coppia (x,y) e il focus è mantenere una rappresentazione compatta dei punti attivi all’interno della finestra corrente, approssimando il diametro.
- I punti sono arrotondati rispetto al centro del cluster usando una distanza euclidea in due dimensioni, mantenendo un errore massimo controllato (ε)

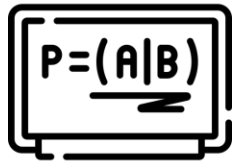


TEOREMA 3

Esiste un algoritmo di approssimazione ε per mantenere il diametro in due dimensioni in una finestra scorrevole di dimensione n utilizzando $O((\frac{1}{\varepsilon^{3/2}})log^3 n(\log R + \log\log n + \log(\frac{1}{\varepsilon}))$ bit di spazio, dove R è il massimo, su tutte le finestre, del rapporto tra il diametro e la distanza minima non nulla tra due punti in quella finestra.

Lower Bounds

TEOREMA 4



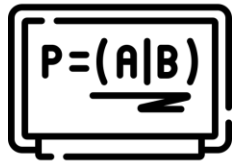
DEFINIZIONE:

Qualsiasi algoritmo di streaming che calcola il diametro esatto di n punti, anche se ciascun punto può essere codificato utilizzando al massimo O(log n) bit, richiede Ω(n) bit di spazio.

DIMOSTRAZIONE:

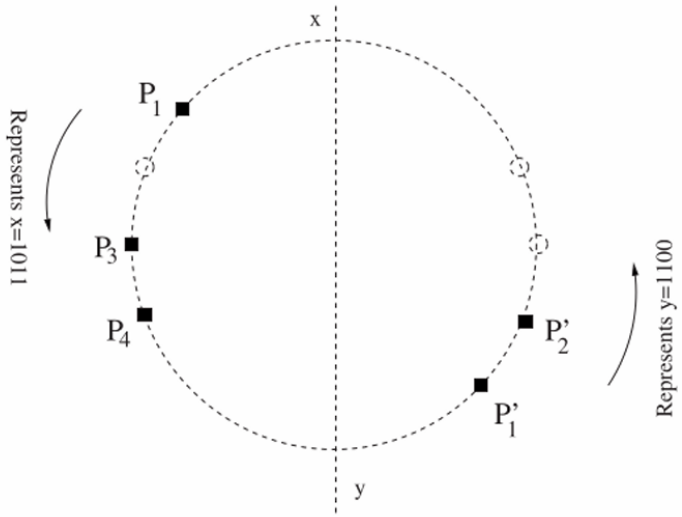
- Riduciamo il problema del calcolo del diametro al problema di disgiunzione tra insiemi.
- Il problema di disgiunzione ha una complessità lineare che dipende dalle dimensioni degli insiemi iniziali.
- Consideriamo due punti su un cerchio. Ogni punto p_i ha un punto p_i' che è il punto opposto ($|p_i p_i'|$ è $diametro$).
- La distanza tra p_i e ogni altro punto è minore di $|p_i p_i'|$. Per ogni elemento i, ci sono i due punti sul cerchio (p_i e p_i').
- La presenza di questi punti sul cerchio è legata alla presenza dell'elemento i nei sottoinsiemi disgiunti x e y. Se i appartiene a x, appare solo p_i . Se i appartiene a y, appare solo $p_{i'}$.

TEOREMA 4

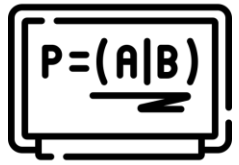


DIMOSTRAZIONE

- Se quindi l'elemento i appare sia in x che in y, i due insiemi non sono disgiunti e quindi esistono p_i e p_i' sul cerchio, che mi consentono di calcolare il diametro.
- In conclusione, determinare se due insiemi x e y sono disgiunti, equivale a vedere se ci sono coppie di elementi antipodali (p_i e p_i') sul cerchio, il che dipende dal fatto che l'elemento i si trovi sia in x che y.
- Se il problema della disgiunzione ha complessità Ω(n), anche il problema del diametro avrà una complessità lineare. In altre parole: se potessimo risolvere il problema del diametro con meno di Ω(n) bit, possiamo risolvere anche l'altro ma è una contraddizione. ■



TEOREMA 5



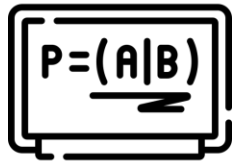
DEFINIZIONE:

Per mantenere, in una finestra scorrevole di dimensione n, il diametro esatto di un insieme di punti su una linea, anche se ciascun punto dell'insieme può essere codificato utilizzando al massimo $O(\log n)$ bit, richiede $\Omega(n)$ bit di spazio

DIMOSTRAZIONE:

- Immaginiamo di avere una sequenza di punti lungo una linea (lunga $2n-2$). Gli ultimi $n-1$ punti sono fissi, cioè le loro posizioni non cambiano.
- La finestra considera solo un certo numero di punti alla volta (n).
- Calcolare il diametro significa trovare i due punti più lontani nella finestra.
- I primi $n-2$ punti possono assumere $n-1$ valori diversi.
- Calcoliamo la distanza tra il primo punto e l'ultimo punto. E' difficile perché cambiano i valori e cambiano i diametri, per ogni finestra.

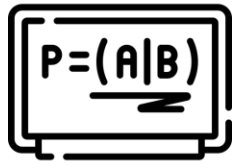
TEOREMA 5



DIMOSTRAZIONE

- Ci possono essere $2^{n/2}$ combinazioni di valori, ovvero servono almeno $\log(2^{n/2}) = \frac{n}{2}$ bit di spazio.
- Per calcolare il diametro esatto in una finestra serve tanta memoria quanto la dimensione della finestra (n). ■

TEOREMA 6

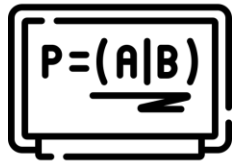


DEFINIZIONE:

Sia R il massimo, su tutte le finestre, del rapporto tra il diametro e la distanza minima non nulla tra due punti qualsiasi in quella finestra. Per mantenere **approssimativamente** il diametro dei punti su una linea in una finestra scorrevole di dimensione n, richiede:

- Se $\log R \leq (\frac{3 \log e}{2})\varepsilon \cdot n^{1-\delta}$: $\boldsymbol{\Omega}(\frac{\mathbf{1}}{\boldsymbol{\varepsilon}} \boldsymbol{log\ R\ log\ n})$ bit di spazio. Per qualche costante $\delta < 1$
- Se $\log R \geq (\frac{3 \log e}{2})\varepsilon \cdot n$: $\boldsymbol{\Omega}(\boldsymbol{n})$ bit di spazio

TEOREMA 6



DIMOSTRAZIONE

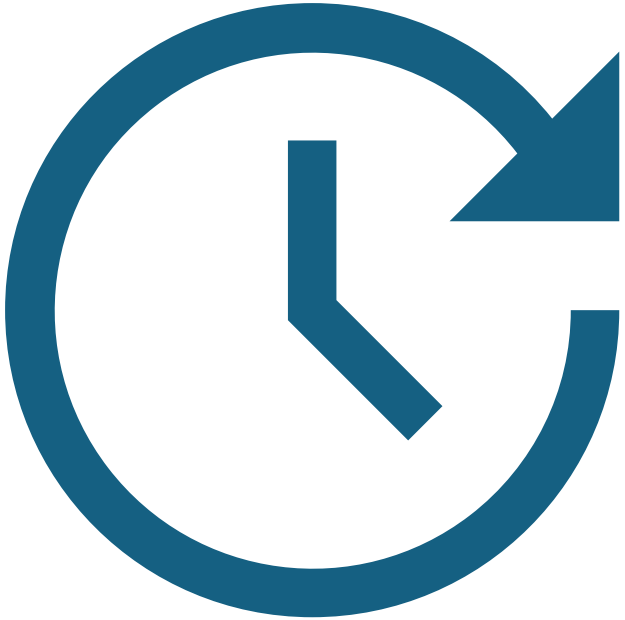
Partiamo dalla costruzione della sequenza dei punti con valori scelti come potenze di $(1+\varepsilon)$:

$(1+\varepsilon)^{3k}$, con k un intero compreso tra 1 e m, con $m = \frac{1}{3\log(1+\varepsilon)} \cdot \log R$.

Il teorema quantifica lo spazio minimo necessario per mantenere un'approssimazione del diametro ε.

- **Caso 1** - $\log R \leq (\frac{3\log e}{2})\varepsilon \cdot n^{1-\delta}$: lo spazio richiesto è $\boldsymbol{\Omega(\frac{1}{\varepsilon} log\, R\, log\, n)}$ per distinguere bene punti molto vicini o punti molto lontani. Il numero di possibili configurazioni dei punti nella finestra è limitato, ma aumenta comunque con ε, $\log R$, e $\log n$.
- **Caso 2** - $\log R \geq (\frac{3\log e}{2})\varepsilon \cdot n$: lo spazio richiesto è $\boldsymbol{\Omega(n)}$ poiché ci sono punti molto vicini e molto lontani nella stessa finestra. Per mantenere un'approssimazione accurata, è necessario memorizzare molte informazioni sui punti. ■

Conclusioni e Sviluppi Futuri



Sintesi dei Risultati

Sono stati sviluppati algoritmi per il calcolo del **diametro** nei modelli di streaming e sliding-window:

1. **Streaming**: Algoritmi approssimati che utilizzano spazio $O(\frac{1}{\varepsilon})$ bit e garantiscono un margine di errore ε.
2. **Sliding-Window**: Tecniche innovative per aggiornare dinamicamente i calcoli man mano che i dati escono dalla finestra.

Punti di Forza

Approcci scalabili e sub-lineari che bilanciano risorse computazionali (spazio e tempo) e accuratezza.

Applicabilità in contesti pratici come reti di sensori, monitoraggio geografico, e analisi di dati in tempo reale.

Limitazioni

Gli algoritmi funzionano bene per basse dimensioni (d=2), ma diventano meno efficienti in spazi ad alta dimensionalità

Esiste ancora un divario tra i limiti superiori e inferiori nella complessità spaziale per alcuni problemi.

Sviluppi Futuri

1. **Algoritmi per Alta Dimensionalità**:

- Estendere le tecniche di approssimazione al calcolo del diametro e di altre misure geometriche in spazi ad alta dimensionalità (d>2).
- Utilizzo di tecniche di **riduzione dimensionale**, come proiezioni casuali, per mitigare la "maledizione della dimensionalità".

2. Estensione a Problemi Geometrici Complessi:

- Studio di altre misure geometriche in streaming, come:
  - **Volume e perimetro**.
  - **Riconoscimento di pattern** basati sulla forma dei dati.
- Integrazione con algoritmi per clustering e classificazione.

Sviluppi Futuri

3. Modelli più generali:

- Finestre Elastiche.
- Modelli probabilistici

4. Ottimizzazioni Computazionali:

- Riduzione dei costi computazionali per scenari distribuiti e multi-stream.
- Parallelizzazione degli algoritmi per applicazioni ad alta velocità.

GRAZIE PER L’ATTENZIONE!