FLAIR

Storing Unbounded Data Streams on Mobile Devices to Unlock User Privacy at the Edge

Olivier Ruas *et al.*

Project members

Olivier Ruas
Pathway

Rémy Raes
Inria

Adrien Luxey
Inria

Romain Rouvoy
Inria

01

Introduction

- Geolocation data is sensitive
- Privacy VS utility
- Location privacy protection mechanisms (LPPMs)

  V. Primault et al. “Time Distortion Anonymization for the Publication of Mobility Data with High Utility”. *IEEE Trustcom/BigDataSE/ISPA*. 2015
- In-situ privacy

- Geolocation data is sensitive
- Privacy VS utility
- Location privacy protection mechanisms (LPPMs)

  V. Primault et al. "Time Distortion Anonymization for the Publication of Mobility Data with High Utility". *IEEE Trustcom/BigDataSE/ISPA*. 2015
- In-situ privacy

- Geolocation data is sensitive
- Privacy VS utility
- Location privacy protection mechanisms (LPPMs)

  V. Primault et al. "Time Distortion Anonymization for the Publication of Mobility Data with High Utility". *IEEE Trustcom/BigDataSE/ISPA*. 2015
- In-situ privacy

- Geolocation data is sensitive
- Privacy VS utility
- Location privacy protection mechanisms (LPPMs)

  V. Primault et al. "Time Distortion Anonymization for the Publication of Mobility Data with High Utility". *IEEE Trustcom/BigDataSE/ISPA*. 2015
- In-situ privacy

- Mobile devices
- Limited resources
  - CPU
  - RAM
  - Storage
  - Battery
  - Network

- Mobile devices
- Limited resources
  - CPU
  - RAM
  - Storage
  - Battery
  - Network

1. Store geo-data on mobile phones → FLAIR
2. Protect location privacy → PROMESSE
3. Evaluate privacy → POI-attack

1. Store geo-data on mobile phones → FLAIR
2. Protect location privacy → PROMESSE
3. Evaluate privacy → POI-attack

1. Store geo-data on mobile phones → FLAIR
2. Protect location privacy → PROMESSE
3. Evaluate privacy → POI-attack

1. Store geo-data on mobile phones → FLAIR
2. Protect location privacy → PROMESSE
3. Evaluate privacy → POI-attack

1. Store geo-data on mobile phones → FLAIR
2. Protect location privacy → PROMESSE
3. Evaluate privacy → POI-attack

1. Store geo-data on mobile phones → FLAIR
2. Protect location privacy → PROMESSE
3. Evaluate privacy → POI-attack

02

FLAIR modeling

Storing time-series

- On mobile: raw points (SQLite)
- Elsewhere: modeling (SWAB, Greycat...)

  E. Keogh et al. "An online algorithm for segmenting time series". *ICDM*. 2001

FLAIR : Fast piecewise LineAr InteRpolation

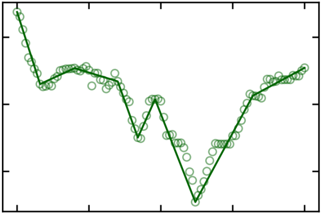
- Such memory gain
- Much fast
- Wow

Storing time-series

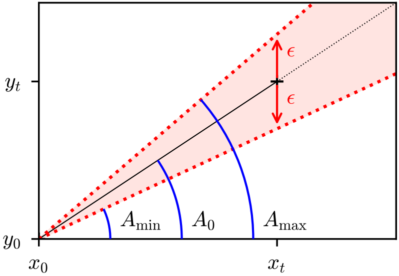
- On mobile: raw points (SQLite)
- Elsewhere: modeling (SWAB, Greycat...)

  E. Keogh et al. "An online algorithm for segmenting time series". *ICDM*. 2001

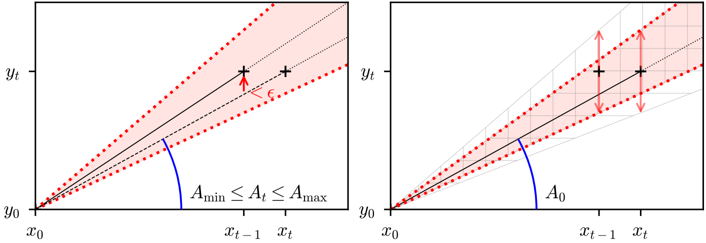
FLAIR : Fast piecewise LineAr InteRpolation



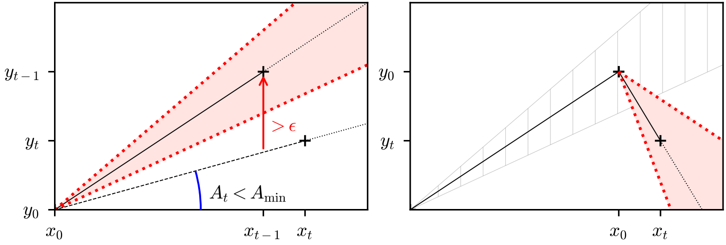
- Such memory gain
- Much fast
- Wow



- Parameter ϵ: tolerated error
- Persisted model: (x_0, y_0, A_0)
- In-memory: $(A_{\min}, A_{\max})$ cone



- New point fits current model
- Model is updated



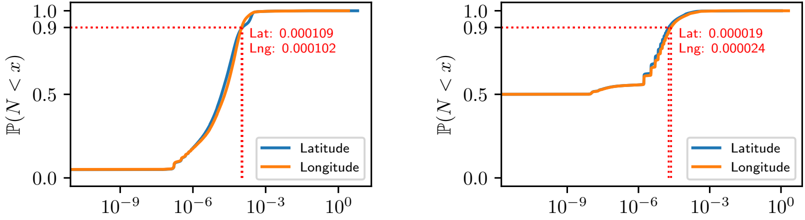
- New point does not fit current model
- Model is saved, and a new one is created

- Tolerated ϵ requires data knowledge

CDF of latitude and longitude variations of successive locations in CABSPOTTING and PRIVAMOV.

- We used $\epsilon = 10^{-3}$

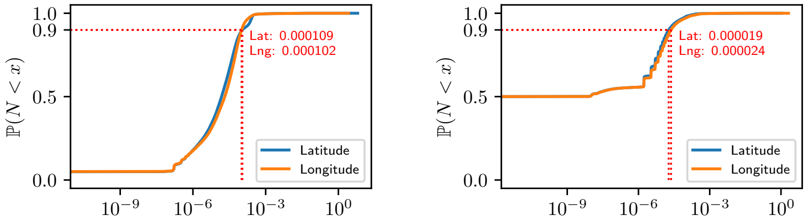
- Tolerated ϵ requires data knowledge



CDF of latitude and longitude variations of successive locations in CABSPOTTING and PRIVAMOV.

- We used $\epsilon = 10^{-3}$

- Tolerated ϵ requires data knowledge



CDF of latitude and longitude variations of successive locations in CABSPOTTING and PRIVAMOV.

- We used $\epsilon = 10^{-3}$

PRIVAMOV GPS

- 5.0 GB in SQLite
- 25 MB in FLAIR $\epsilon = 10^{-3}$

$\rightarrow$ **99.95% gain**

CABSPOTTING (536 taxis)

Per-taxi gain CDF with different ϵ

Throughput on random values

- Sequential writes: 3505 times faster
- Random reads: 2343 times faster

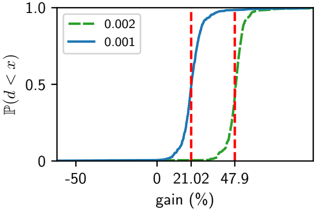
... than competitors (SWAB & Greycat)

PrivaMov GPS

- 5.0 GB in SQLite
- 25 MB in FLAIR $\epsilon = 10^{-3}$

$\rightarrow$ **99.95% gain**

CabSpotting (536 taxis)



Per-taxi gain CDF with different ϵ

Throughput on random values

- Sequential writes: 3505 times faster
- Random reads: 2343 times faster

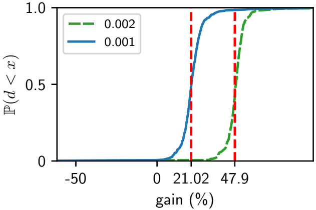
... than competitors (SWAB & Greycat)

PrivaMov GPS

- 5.0 GB in SQLite
- 25 MB in FLAIR $\epsilon = 10^{-3}$

$\rightarrow$ **99.95% gain**

CabSpotting (536 taxis)



Per-taxi gain CDF with different ϵ

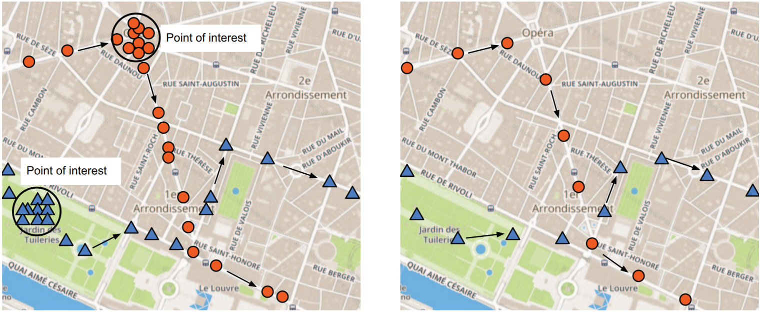
Throughput on random values

- Sequential writes: 3505 times faster
- Random reads: 2343 times faster

... than competitors (SWAB & Greycat)

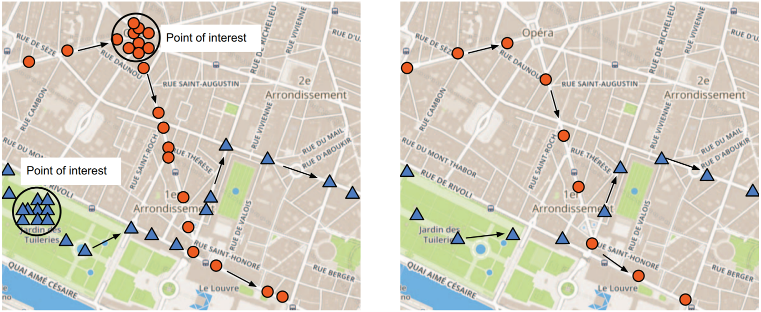
03

Protecting users' privacy at the edge



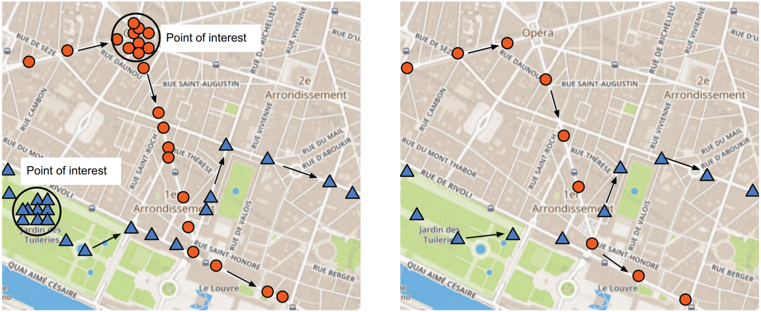
V. Primault et al. “Time Distortion Anonymization for the Publication of Mobility Data with High Utility”. *IEEE Trustcom/BigDataSE/ISPA*. 2015

- How can we evaluate privacy provided by this LPPM?
- We attack *in-situ*!



V. Primault et al. "Time Distortion Anonymization for the Publication of Mobility Data with High Utility". *IEEE Trustcom/BigDataSE/ISPA*. 2015

- How can we evaluate privacy provided by this LPPM?
- We attack *in-situ*!



V. Primault et al. “Time Distortion Anonymization for the Publication of Mobility Data with High Utility”. *IEEE Trustcom/BigDataSE/ISPA*. 2015

- How can we evaluate privacy provided by this LPPM?
- We attack *in-situ*!

- Classical POI inference algorithm is slow
- New implementation: **Divide & Stay (D&S)**

Platform	POI-attack	D&S	Speed-up
Desktop	59 min 20 s	32 s	×111
iOS	1 h 00 min 01 s	22 s	×164
Android	1 h 58 min 04 s	59 s	×120

Computation times on PRIVAMOV user #1

- Classical POI inference algorithm is slow
- New implementation: **Divide & Stay (D&S)**

Platform	POI-attack	D&S	Speed-up
Desktop	59 min 20 s	32 s	×111
iOS	1 h 00 min 01 s	22 s	×164
Android	1 h 58 min 04 s	59 s	×120

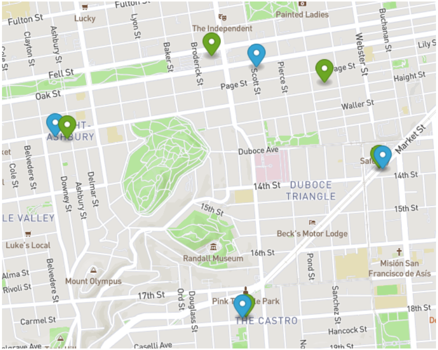
Computation times on PRIVAMOV user #1

- Classical POI inference algorithm is slow
- New implementation: **Divide & Stay (D&S)**

Platform	**POI-attack**	**D&S**	**Speed-up**
Desktop	59 min 20 s	32 s	×111
iOS	1 h 00 min 01 s	22 s	×164
Android	1 h 58 min 04 s	59 s	×120

Computation times on PRIVAMOV user #1

- Ensure POI inference results are the same between:
  - > classic dataset + *POI-attack*
  - > FLAIR-modeled dataset + *D&S*



- POIs inference with and without *FLAIR*
- *In-situ* LPPM usage

	without PROMESSE		with PROMESSE	
Algorithm	**Raw POIs**	**FLAIR**	**Raw POIs**	**FLAIR**
POI-attack	30	31	0	0
D&S	30	30	0	0
POI-attack ∩ D&S	21	20	-	-

Inferred POIs on CABSPOTTING user #0

- That's how we store big amounts of data and protect them!

- POIs inference with and without *FLAIR*
- *In-situ* LPPM usage

	without PROMESSE		with PROMESSE	
Algorithm	**Raw POIs**	**FLAIR**	**Raw POIs**	**FLAIR**
POI-attack	30	31	0	0
D&S	30	30	0	0
POI-attack ∩ D&S	21	20	-	-

Inferred POIs on CABSPOTTING user #0

- That's how we store big amounts of data and protect them!

- POIs inference with and without *FLAIR*
- *In-situ* LPPM usage

	without PROMESSE		with PROMESSE	
Algorithm	**Raw POIs**	**FLAIR**	**Raw POIs**	**FLAIR**
POI-attack	30	31	0	0
D&S	30	30	0	0
POI-attack ∩ D&S	21	20	-	-

Inferred POIs on CABSPOTTING user #0

- That's how we store big amounts of data and protect them!

FLAIR

Storing Unbounded Data Streams on Mobile Devices to Unlock User Privacy at the Edge

Olivier Ruas *et al.*

FLAIR insertions

Algorithm 1 FLAIR insertion using parameter $\epsilon \in \mathbb{R}^{+*}$

Before: $M; (x_0, x_{t-1}) \in \mathbb{R}^{2+}; (y_0, y_{t-1}, A_0, A_{\min}, A_{\max}) \in \mathbb{R}^5$

function INSERT($x_t \in \mathbb{R}^+, y_t \in \mathbb{R}$)
- $(x_t^\Delta, y_t^\Delta) \leftarrow (x_t - x_0, y_t - y_0)$ ▹ Compute A_t
- $A_t \leftarrow y_t^\Delta / x_t^\Delta$
- **if** $A_{\min} \leq A_t \leq A_{\max}$ **then**
    - $A_0 \leftarrow A_t$ ▹ Update model
    - $A_{\min} \leftarrow \max\left(A_{\min}, \frac{y_t^\Delta - \epsilon}{x_t^\Delta}\right)$
    - $A_{\max} \leftarrow \min\left(A_{\max}, \frac{y_t^\Delta + \epsilon}{x_t^\Delta}\right)$
- **else**
    - $\mathcal{M}.\text{insert}(x_0, y_0, A_0)$ ▹ Persist model
    - $(x_0, y_0) \leftarrow (x_{t-1}, y_{t-1})$ ▹ Build new model
    - $(x_t^\Delta, y_t^\Delta) \leftarrow (x_t - x_0, y_t - y_0)$
    - $A_0 \leftarrow y_t^\Delta / x_t^\Delta$
    - $A_{\min} \leftarrow \left(y_t^\Delta - \epsilon\right) / x_t^\Delta$
    - $A_{\max} \leftarrow \left(y_t^\Delta + \epsilon\right) / x_t^\Delta$
- **end if**
- $(x_{t-1}, y_{t-1}) \leftarrow (x_t, y_t)$ ▹ Update penultimate

end function

Algorithm 2 FLAIR approximate read

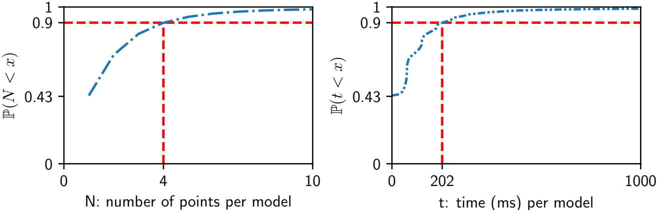
Before: Current model (x_0, y_0, A_0);
Memory $\mathcal{M}$ containing previous models

```
function READ(x ∈ ℝ⁺)
    if x_0 ≤ x then
        return A_0 × (x − x_0) + y_0
    end if
    Select i s.t. (x_i, y_i, A_i) ∈ M ∧ x_i ≤ x < x_{i+1}
    return A_i × (x − x_i) + y_i
end function
```

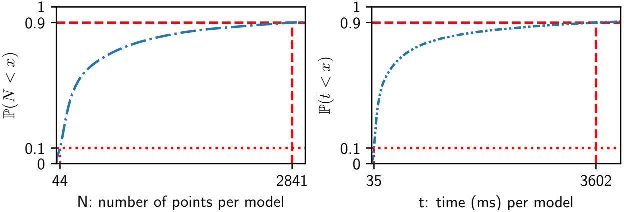
Algorithm 3 Divide & Stay (D&S)

Input: $T \in (\mathbb{R} \times \mathbb{G})^n; S \in \mathbb{N}^+; s \in [\![0; n-1]\!];$
$e \in [\![0; n-1]\!], (t_{min}, D_{max}) \in \mathbb{R}^{2+}$
Output: $STAYS \in (\mathbb{R} \times \mathbb{G})^n$

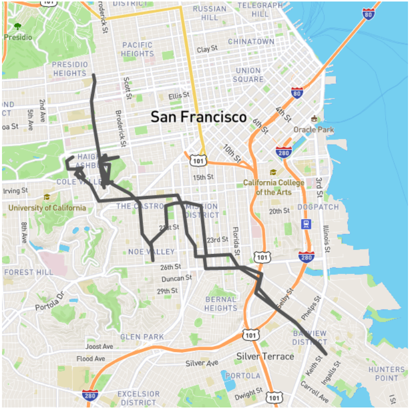
$STAYS \leftarrow \emptyset$
if $T.size() \leq S$ **then**
  return getStays(T.subtrace(s, e), m, D)
end if
$i = \lfloor (e+s)/2 \rfloor$
$t1 = T[i].t - T[s].t$
$d1 = \text{geo.dist}(T[s].g, T[i].g)$
if $\neg(d1 > D_{max} \wedge t1 \leq t_{min})$ **then**
  $STAYS{+} = D\&S(T, S, s, i, t_{min}, D_{max})$
end if
$t2 = T[e].t - T[i].t$
$d2 = geo.dist(T[i].g, T[e].g)$
if $\neg(d2 > D_{max} \wedge t2 \leq t_{min})$ **then**
  $STAYS{+} = D\&S(T, S, i, e, t_{min}, D_{max})$
end if
return $STAYS$

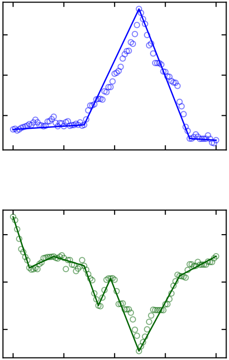


Cabspotting modeling



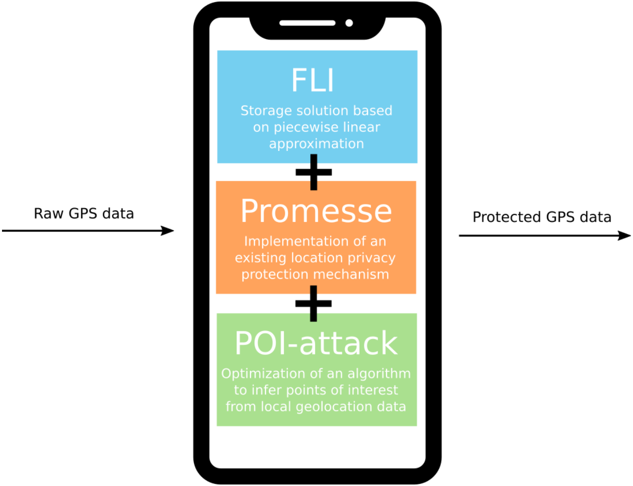
Privamov modeling





Modeled latitude and longitude.

A system?


Raw GPS data
FLI
Storage solution based on piecewise linear approximation
+
Promesse
Implementation of an existing location privacy protection mechanism
+
POI-attack
Optimization of an algorithm to infer points of interest from local geolocation data
Protected GPS data