

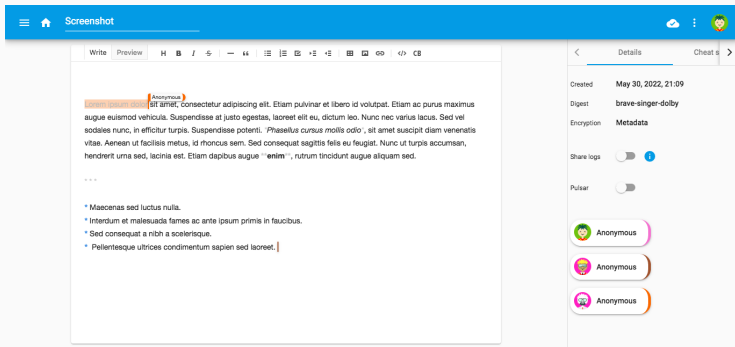
# Ré-identification sans coordination dans les types de données répliquées sans conflits

---

Matthieu Nicolas ([matthieu.nicolas@loria.fr](mailto:matthieu.nicolas@loria.fr))

20 décembre 2022

<i>Rapporteurs :</i>	Hanifa Boucheneb Davide Frey	Professeure, Polytechnique Montréal Chargé de recherche, HdR, Inria Rennes Bretagne-Atlantique
<i>Examineurs :</i>	Hala Skaf-Molli Stephan Merz	Maîtresse de conférences, HdR, Nantes Université, LS2N Directeur de Recherche, Inria Nancy - Grand Est
<i>Encadrants :</i>	Olivier Perrin Gérald Oster	Professeur des Universités, Université de Lorraine, LORIA Maître de conférences, Université de Lorraine, LORIA



- Application pair-à-pair
- Permet à groupes de rédiger collaborativement documents texte
- Garantit confidentialité & souveraineté de ses données

\*. Disponible à : <https://mutehost.loria.fr>

TODO : Illustrer une appli P2P

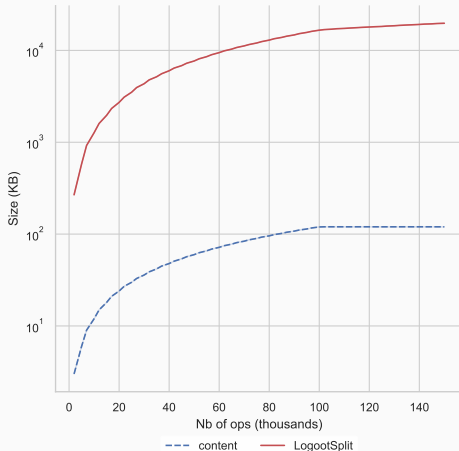
## Problématiques

En l'absence d'autorités centrales, comment

- authentifier les utilisateur-rices ?
- vérifier leurs droits d'accès ?
- résoudre les conflits de modifications ?

# Évaluation de MUTE

## Taille du texte comparée à taille de la séquence répliquée



- 1% contenu...
- ...99% métadonnées

Comment peut-on réduire le surcoût mémoire  
des mécanismes de résolution de conflits dans  
les applications pair-à-pair ?

# Conflict-free Replicated Data Types (CRDTs)<sup>[1]</sup>

- Nouvelles spécifications des types de données, e.g. *Ensemble* ou *Séquence*
- Conçues pour données répliquées

---

[1]. SHAPIRO et al., « Conflict-Free Replicated Data Types ».

# Conflict-free Replicated Data Types (CRDTs)<sup>[1]</sup>

- Nouvelles spécifications des types de données, e.g. *Ensemble* ou *Séquence*
- Conçues pour données répliquées

## Propriétés des CRDTs

- Permettent modifications **sans coordination**
- Garantissent la **convergence forte**

---

[1]. SHAPIRO et al., « Conflict-Free Replicated Data Types ».

# Conflict-free Replicated Data Types (CRDTs)<sup>[1]</sup>

- Nouvelles spécifications des types de données, e.g. *Ensemble* ou *Séquence*
- Conçues pour données répliquées

## Propriétés des CRDTs

- Permettent modifications **sans coordination**
- Garantissent la **convergence forte**

## Convergence forte

Ensemble des noeuds ayant intégrés le même ensemble de modifications obtient des états équivalents, sans nécessiter d'actions ou messages supplémentaires

---

[1]. SHAPIRO et al., « Conflict-Free Replicated Data Types ».



# LogootSplit<sup>[3]</sup>, un CRDT pour le type Séquence

- Assigne **identifiant de position** à chaque élément de la séquence

---

[2]. PREGUICA et al., « A Commutative Replicated Data Type for Cooperative Editing ».

[3]. ANDRÉ et al., « Supporting Adaptable Granularity of Changes for Massive-Scale Collaborative Editing ».

# LogootSplit<sup>[3]</sup>, un CRDT pour le type Séquence

- Assigne **identifiant de position** à chaque élément de la séquence

## Propriétés des identifiants de position<sup>[2]</sup>

1. Unique
2. Immuable
3. Ordonnable par une relation d'ordre strict total  $<_{id}$
4. Appartenant à un espace dense

---

[2]. PREGUICA et al., « A Commutative Replicated Data Type for Cooperative Editing ».

[3]. ANDRÉ et al., « Supporting Adaptable Granularity of Changes for Massive-Scale Collaborative Editing ».

# LogootSplit<sup>[3]</sup>, un CRDT pour le type Séquence

- Assigne **identifiant de position** à chaque élément de la séquence

## Propriétés des identifiants de position<sup>[2]</sup>

1. Unique
2. Immuable
3. Ordonnable par une relation d'ordre strict total  $<_{id}$
4. Appartenant à un espace dense

- **Ordonne les éléments** entre eux **en utilisant** leurs **identifiants**

---

[2]. PREGUICA et al., « A Commutative Replicated Data Type for Cooperative Editing ».

[3]. ANDRÉ et al., « Supporting Adaptable Granularity of Changes for Massive-Scale Collaborative Editing ».

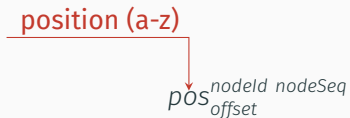
## Identifiant

- Composé d'un ou plusieurs tuples suivants

$$pos_{offset}^{nodeId \ nodeSeq}$$

## Identifiant

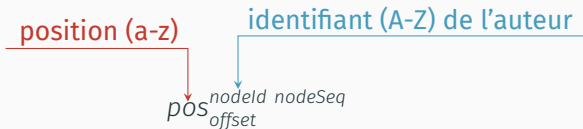
- Composé d'un ou plusieurs tuples suivants



# Identifiant LogootSplit

## Identifiant

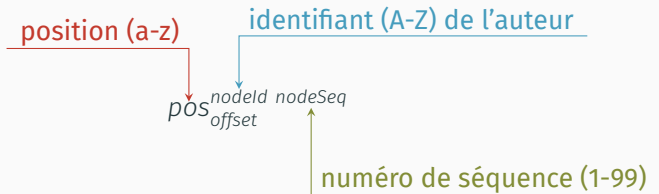
- Composé d'un ou plusieurs tuples suivants



# Identifiant LogootSplit

## Identifiant

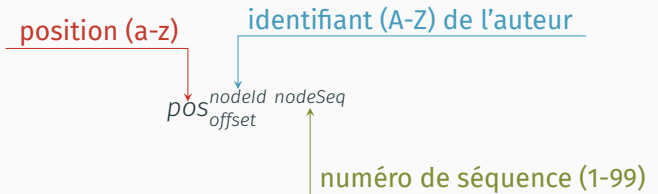
- Composé d'un ou plusieurs tuples suivants



# Identifiant LogootSplit

## Identifiant

- Composé d'un ou plusieurs tuples suivants



## Exemples

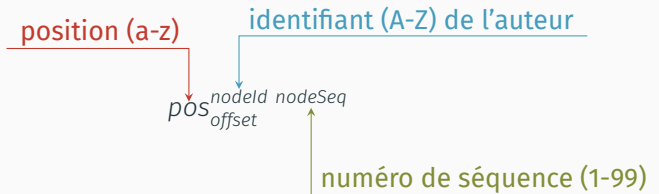
$d_0^{F5}$



# Identifiant LogootSplit

## Identifiant

- Composé d'un ou plusieurs tuples suivants



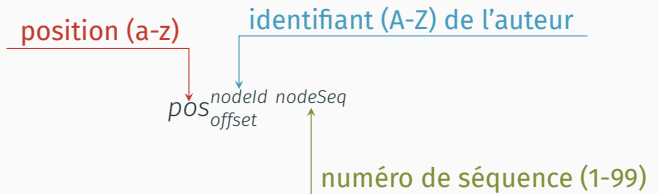
## Exemples

$$d_0^{F5} <_{id} m_0^{C1}$$

# Identifiant LogootSplit

## Identifiant

- Composé d'un ou plusieurs tuples suivants



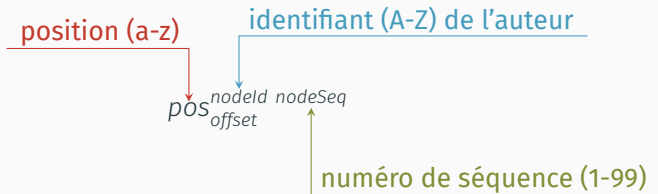
## Exemples

$$d_0^{F5} <_{id} m_0^{C1} <_{id} m_0^{C1} f_0^{E1}$$

# Identifiant LogootSplit

## Identifiant

- Composé d'un ou plusieurs tuples suivants



## Exemples

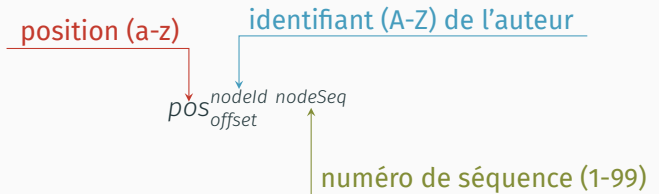
$$d_0^{F5} <_{id} m_0^{C1} <_{id} m_0^{C1} f_0^{E1}$$

$$i_0^{B1} <_{id} ? <_{id} i_1^{B1}$$

# Identifiant LogootSplit

## Identifiant

- Composé d'un ou plusieurs tuples suivants



## Exemples

$$d_0^{F5} <_{id} m_0^{C1} <_{id} m_0^{C1} f_0^{E1}$$

$$i_0^{B1} <_{id} i_0^{B1} f_0^{A1} <_{id} i_1^{B1}$$

- Coûteux de stocker les identifiants de chaque élément

B	A	N	J	O
$m_0^{C1}$	$m_1^{C1}$	$m_2^{C1}$	$m_3^{C1}$	$m_4^{C1}$

# Bloc LogootSplit

- Coûteux de stocker les identifiants de chaque élément

B	A	N	J	O
$m_0^{C1}$	$m_1^{C1}$	$m_2^{C1}$	$m_3^{C1}$	$m_4^{C1}$

- Aggrège en un **bloc** éléments ayant **identifiants contigus**

## Identifiants contigus

Deux identifiants sont contigus si et seulement si les deux identifiants sont identiques à l'exception de leur dernier offset et que leur derniers offsets sont consécutifs.

# Bloc LogootSplit

- Coûteux de stocker les identifiants de chaque élément

B	A	N	J	O
$m_0^{C1}$	$m_1^{C1}$	$m_2^{C1}$	$m_3^{C1}$	$m_4^{C1}$

- Aggrège en un **bloc** éléments ayant **identifiants contigus**

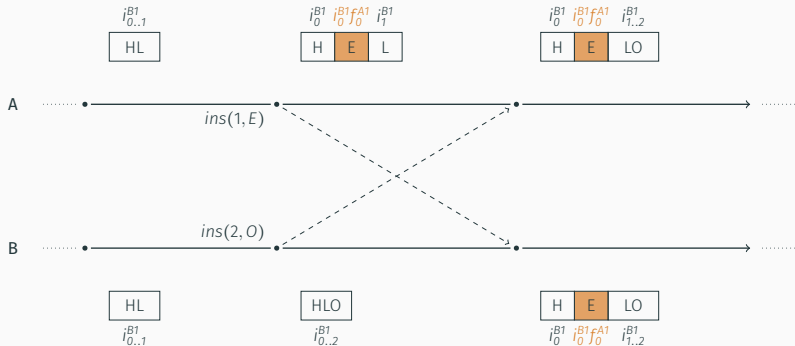
## Identifiants contigus

Deux identifiants sont contigus si et seulement si les deux identifiants sont identiques à l'exception de leur dernier offset et que leur derniers offsets sont consécutifs.

- Note l'intervalle d'identifiants d'un bloc :  $pos_{begin..end}^{nodeId nodeSeq}$

BANJO
$m_{0..4}^{C1}$

# Exemple insertions concurrentes

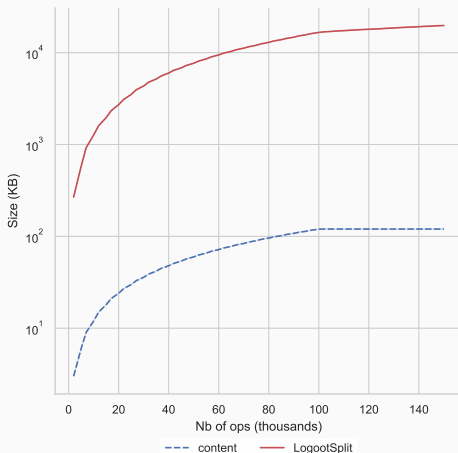




# Limites de LogootSplit

- Croissance non-bornée de la taille des identifiants
- Fragmentation en blocs courts

Taille du contenu comparé à la taille de la séquence LogootSplit



## L'approche core-nebula<sup>[4]</sup>

- Ré-assigne des identifiants courts aux éléments, c.-à-d. les *renomme*
- Transforme les opérations *insert* et *remove* concurrentes...
- ...mais ne supportent pas opérations *rename* concurrentes

Inadaptée aux applications pair-à-pair

---

[4]. ZAWIRSKI et al., « Asynchronous rebalancing of a replicated tree ».

Proposition : Mécanisme de renommage  
supportant les renommages concurrents