

# Chapitre 4

## Conclusions et perspectives

### Sommaire

---

<b>4.1</b>	<b>Résumés des contributions</b>	<b>125</b>
4.1.1	Ré-identification sans coordination pour les CRDTs pour Séquence	125
4.1.2	Éditeur de texte collaboratif P2P chiffré de bout en bout	127
<b>4.2</b>	<b>Perspectives</b>	<b>129</b>
4.2.1	Définition de relations de priorité pour minimiser les traitements	129
4.2.2	Détection et fusion manuelle de versions distantes	130
4.2.3	Étude comparative des différents modèles de synchronisation pour CRDTs	133
4.2.4	Approfondissement du patron de conception de Pure Operation-Based CRDTs	135

---

Dans ce chapitre, nous revenons sur les contributions présentées dans cette thèse. Nous rappelons le contexte dans lequel elles s'inscrivent, récapitulons leurs spécificités et apports, et finalement présentons leurs limites que nous identifions. Puis, nous concluons ce manuscrit en présentant plusieurs pistes de recherche qui nous restent à explorer à l'issue de cette thèse. Les premières s'inscrivent dans la continuité directe de nos travaux sur un mécanisme de ré-identification pour CRDTs pour Séquence dans un système P2P à large échelle sujet au churn. Les dernières traduisent quant à elles notre volonté de recentrer nos travaux sur le domaine plus général des CRDTs.

### 4.1 Résumés des contributions

#### 4.1.1 Ré-identification sans coordination pour les CRDTs pour Séquence

Pour privilégier leur disponibilité, latence et tolérance aux pannes, les systèmes distribués peuvent adopter le paradigme de la réplication optimiste [13]. Ce paradigme consiste à relaxer la cohérence de données entre les noeuds du système pour leur permettre de consulter et modifier leur copie locale sans se coordonner. Leur copies peuvent alors temporairement diverger avant de converger de nouveau une fois les modifications de chacun

propagées. Cependant, cette approche nécessite l'emploi d'un mécanisme de résolution pour assurer la convergence même en cas de modifications concurrentes. Pour cela, l'approche des CRDTs [51, 16] propose d'utiliser des types de données dont les modifications sont nativement commutatives.

Depuis la spécification des CRDTs, la littérature a proposé plusieurs de ces mécanismes résolution de conflits automatiques pour le type de données Séquence [48, 49, 50, 46]. Cependant, ces approches souffrent toutes d'un surcoût croissant de manière monotone. Ce problème a été identifié par la communauté, et celle-ci a proposé pour y répondre des mécanismes permettant soit de réduire la croissance du surcoût [68, 69], soit d'effectuer une GC du surcoût [49, 1, 2]. Nous avons cependant déterminé que ces mécanismes ne sont pas adaptés aux systèmes P2P à large échelle souffrant de churn et utilisant des CRDTs pour Séquence à granularité variable.

Dans le cadre de cette thèse, nous avons donc souhaité proposer un nouveau mécanisme adapté à ce type de systèmes. Pour cela, nous avons suivi l'approche proposée par [1, 2] : l'utilisation d'un mécanisme pour ré-assigner de nouveaux identifiants aux éléments stockés dans la séquence. Nous avons donc proposé un nouveau mécanisme appartenant à cette approche pour le CRDT LogootSplit [4].

Notre proposition prend la forme d'un nouvel CRDT pour Séquence à granularité variable : RenamableLogootSplit. Ce nouveau CRDT associe à LogootSplit un nouveau type de modification, *ren*, permettant de produire une nouvelle séquence équivalente à son état précédent. Cette nouvelle modification tire profit de la granularité variable de la séquence pour produire un état de taille minimale : elle assigne à tous les éléments des identifiants de position issus d'un même intervalle. Ceci nous permet de minimiser les métadonnées que la séquence doit stocker de manière effective. De plus, le passage à une représentation interne minimale de la séquence nous permet d'améliorer le coût des modifications suivantes en termes de calculs.

Afin de gérer les opérations concurrentes aux opérations *ren*, nous définissons pour ces dernières un algorithme de transformation. Pour cela, nous définissons un mécanisme d'époques nous permettant d'identifier la concurrence entre opérations. De plus, nous introduisons une relation d'ordre strict total, *priority*, pour résoudre de manière déterministe le conflit provoqué par deux opérations *ren*, c.-à-d. pour déterminer quelle opération *ren* privilégier. Finalement, nous définissons deux algorithmes, **renameId** et **revertRenameId**, qui permettent de transformer les opérations concurrentes à une opération *ren* pour prendre en compte l'effet de cette dernière. Ainsi, notre algorithme permet de détecter et de transformer les opérations concurrentes aux opérations *ren*, sans nécessiter une coordination synchrone entre les noeuds. Le surcoût induit par ce mécanisme, notamment en termes de calculs, est toutefois contrebalancé par l'amélioration précisée précédemment, c.-à-d. la réduction de la taille de la séquence.

Finalement, le mécanisme que nous proposons est partiellement générique : il peut être adapté à d'autres CRDTs pour Séquence à granularité variable, e.g. un CRDT pour Séquence appartenant à l'approche à pierres tombales. Dans le cadre d'une telle démarche, nous pourrions réutiliser le système d'époques, la relation *priority* et l'algorithme de contrôle qui identifie les transformations à effectuer. Pour compléter une telle adaptation, nous devrions cependant concevoir de nouveaux algorithmes **renameId** et **revertRenameId** spécifiques et adaptés au CRDT choisi.

Le mécanisme de renommage que nous présentons souffre néanmoins de plusieurs limites. La première d'entre elles concerne ses performances. En effet, notre évaluation expérimentale a mis en lumière le coût important en l'état de la modification *ren* par rapport aux autres modifications en termes de calculs (cf. section 2.4.3, page 90). De plus, chaque opération *ren* comporte une représentation de l'ancien état qui doit être maintenue par les noeuds jusqu'à leur stabilité causale. Le surcoût en métadonnées introduit par un ensemble d'opérations *ren* concurrentes peut donc s'avérer important, voire pénalisant (cf. section 2.4.3, page 87). Pour répondre à ces problèmes, nous identifions trois axes d'amélioration :

- (i) La définition de stratégies de déclenchement du renommage efficaces. Le but de ces stratégies serait de déclencher le mécanisme de renommage de manière fréquente, de façon à garder son temps d'exécution acceptable, mais tout visant à minimiser la probabilité que les noeuds produisent des opérations *ren* concurrentes, de façon à minimiser le surcoût en métadonnées.
- (ii) La définition de relations *priority* efficaces. Nous développons ce point dans la sous-section 4.2.1.
- (iii) La proposition d'algorithmes de renommage efficaces. Cette amélioration peut prendre la forme de nouveaux algorithmes pour `renameId` et `revertRenameId` offrant une meilleure complexité en temps. Il peut aussi s'agir de la conception d'une nouvelle approche pour renommer l'état et gérer les modifications concurrentes, e.g. un mécanisme de renommage basé sur le journal des opérations (cf. sous-section 2.5.7, page 99).

Une seconde limite de `RenamableLogootSplit` que nous identifions concerne son mécanisme de GC des métadonnées introduites par le mécanisme de renommage. En effet, pour fonctionner, ce dernier repose sur la stabilité causale des opérations *ren*. Pour rappel, la stabilité causale représente le contexte causal commun à l'ensemble des noeuds du système. Pour le déterminer, chaque noeud doit récupérer le contexte causal de l'ensemble des noeuds du système. Ainsi, l'utilisation de la stabilité causale comme pré-requis pour la GC de métadonnées constitue une contrainte forte, voire prohibitive, dans les systèmes P2P à large échelle sujet au churn. En effet, un noeud du système déconnecté de manière définitive suffit pour empêcher la stabilité causale de progresser, son contexte causal étant alors indéterminé du point de vue des autres noeuds. Il s'agit toutefois d'une limite récurrente des mécanismes de GC distribués et asynchrones [49, 24, 88]. Nous présentons une piste de travail possible pour pallier ce problème dans la sous-section 4.2.2.

#### 4.1.2 Éditeur de texte collaboratif P2P chiffré de bout en bout

Les applications collaboratives permettent à des utilisateur-rices de réaliser collaborativement une tâche. Elles permettent à plusieurs utilisateur-rices de consulter la version actuelle du document, de la modifier et de partager leurs modifications avec les autres. Ceci permet de mettre en place une réflexion de groupe, ce qui améliore la qualité du résultat produit [75, 76].

Cependant, les applications collaboratives sont historiquement des applications centralisées, e.g. Google Docs [77]. Ce type d'architecture induit des défauts d'un point de vue

technique, e.g. faible capacité de passage à l'échelle et faible tolérance aux pannes, mais aussi d'un point de vue utilisateur, e.g. perte de la souveraineté des données et absence de garantie de pérennité.

Les travaux de l'équipe Coast s'inscrivent dans une mouvance souhaitant résoudre ces problèmes et qui a conduit à la définition d'un nouveau paradigme d'applications : les Local-First Softwares (LFS) [85]. Le but de ce paradigme est la conception d'applications collaboratives, P2P, pérennes et rendant la souveraineté de leurs données aux utilisateur-rices.

Dans le cadre de cette démarche, l'équipe Coast développe depuis plusieurs années l'application Multi User Text Editor (MUTE), un éditeur de texte web collaboratif P2P temps réel chiffré de bout en bout. Cette application sert à la fois de plateforme de démonstration et de recherche pour les travaux de l'équipe, mais aussi de Proof of Concept (PoC) pour les LFS.

Dans le cadre de cette thèse, nous avons implémenté dans MUTE nos travaux de recherche portant sur le nouvel CRDT pour le type Séquence : RenamableLogootSplit. MUTE a aussi servi à l'équipe pour présenter ses travaux concernant l'authentification des utilisateur-rices dans un système P2P [7]. Finalement, MUTE nous a permis de nous d'étudier et/ou de présenter les travaux de recherche existants concernant :

- (i) Les protocoles distribués d'appartenance au groupe [8].
- (ii) Les mécanismes d'anti-entropie [31].
- (iii) Les protocoles d'établissement de clés de chiffrement de groupe [11].
- (iv) Les protocoles d'établissement de topologies réseaux efficaces [10].
- (v) Les mécanismes de conscience de groupe.

MUTE offre donc, à notre connaissance, le tour d'horizon le plus complet des travaux de recherche permettant la conception d'applications LFS. Cependant, cela ne dispense pas MUTE de souffrir de plusieurs limites.

Tout d'abord, l'environnement web implique un certain nombre de contraintes, notamment au niveau des technologies et protocoles disponibles. Notamment, le protocole WebRTC repose sur l'utilisation de serveurs de signalisation, c.-à-d. de points de rendez-vous des pairs, et de serveurs de relai, c.-à-d. d'intermédiaires pour communiquer entre pairs lorsque les configurations de leur réseaux respectifs interdisent l'établissement d'une connexion directe. Ainsi, les applications P2P web doivent soit déployer et maintenir leur propre infrastructure de serveurs, soit reposer sur une infrastructure existante, e.g. celle proposée par OpenRelay [89]. Afin de minimiser l'effort requis aux applications P2P et la confiance exigée à leurs utilisateur-rices, nous devons supporter la mise en place d'une telle infrastructure transparente et pérenne.

Une autre limite de ce système que nous identifions concerne l'utilisabilité des systèmes P2P de manière générale. L'expérience vécue suivante constitue à notre avis un exemple éloquent des limites actuelles de l'application MUTE dans ce domaine. Après avoir rédigé une version initiale d'un document, nous avons envoyé le lien du document à notre collaborateur pour relecture et validation. Lorsque notre collaborateur a souhaité accéder au document, celui-ci s'est retrouvé devant une page blanche : comme nous nous étions déconnecté du système entretemps, c.-à-d. plus aucun pair n'était disponible pour

effectuer une synchronisation. Notre collaborateur était donc dans l'incapacité de récupérer l'état et d'effectuer sa tâche. Afin de pallier ce problème, une solution possible est de faire reposer MUTE sur un réseau P2P global, e.g. le réseau de InterPlanetary File System (IPFS) [90], et d'utiliser les pairs de ce dernier, potentiellement des pairs étrangers à l'application, comme pairs de stockage pour permettre une synchronisation future. Cette solution limite ainsi le risque qu'un pair ne puisse récupérer l'état du document faute de pairs disponibles. Cependant, elle nécessite de mettre en place un mécanisme de réplication de données additionnel. Ce mécanisme de réplication sur des pairs additionnels doit cependant garantir qu'il n'introduit pas de vulnérabilités, e.g. la possibilité pour les pairs de stockage sélectionnés de reconstruire et consulter le document.

## 4.2 Perspectives

### 4.2.1 Définition de relations de priorité pour minimiser les traitements

Dans la sous-section 2.3.2, nous avons spécifié la relation *priority* (cf. Définition 41, page 68). Pour rappel, cette relation doit établir un ordre strict total sur les époques de notre mécanisme de renommage.

Cette relation nous permet ainsi de résoudre le conflit provoqué par la génération de modifications *ren* concurrentes en les ordonnant. Grâce à cette relation d'ordre, les noeuds peuvent déterminer vers quelle époque de l'ensemble des époques connues progresser. Cette relation permet ainsi aux noeuds de converger à une époque commune à terme.

La convergence à terme à une époque commune présente plusieurs avantages :

- (i) Réduire la distance entre les époques courantes des noeuds, et ainsi minimiser le surcoût en calculs par opération du mécanisme de renommage. En effet, il n'est pas nécessaire de transformer une opérations livrée avant de l'intégrer si celle-ci provient de la même époque que le noeud courant.
- (ii) Définir un nouveau PPAC entre les époques courantes des noeuds. Cela permet aux noeuds d'appliquer le mécanisme de GC pour supprimer les époques devenues obsolètes et leur anciens états associés, pour ainsi minimiser le surcoût en métadonnées du mécanisme de renommage.

Il existe plusieurs manières pour définir la relation *priority* tout en satisfaisant les propriétés indiquées. Dans le cadre de ce manuscrit, nous avons utilisé l'ordre lexicographique sur les chemins des époques dans l'*arbre des époques* pour définir *priority*. Cette approche se démarque par :

- (i) Sa simplicité.
- (ii) Son surcoût limité, c.-à-d. cette approche n'introduit pas de métadonnées supplémentaires à stocker et diffuser, et l'algorithme de comparaison utilisé est simple.
- (iii) Sa propriété arrangeante sur les déplacements des noeuds dans l'arbre des époques. De manière plus précise, cette définition de *priority* impose aux noeuds de se déplacer que vers l'enfant le plus à droite de l'arbre des époques. Ceci empêche les

noeuds de faire un aller-retour entre deux époques données. Cette propriété permet de passer outre une contrainte concernant le couple de fonctions `renameId` et `revertRenameId` : leur réciprocity.

Cette définition présente cependant plusieurs limites. La limite que nous identifions est sa décorrélation avec le coût et le bénéfice de progresser vers l'époque cible désignée. En effet, l'époque cible est désignée de manière arbitraire par rapport à sa position dans l'arbre des époques. Il est ainsi possible que progresser vers cette époque détériore l'état de la séquence, c.-à-d. augmente la taille des identifiants et augmente le nombre de blocs. De plus, la transition de l'ensemble des noeuds depuis leur époque courante respective vers cette nouvelle époque cible induit un coût en calculs, potentiellement important (cf. Figure 2.17, page 96).

Pour pallier ce problème, il est nécessaire de proposer une définition de *priority* prenant l'aspect efficacité en compte. L'approche considérée consisterait à inclure dans les opérations *ren* une ou plusieurs métriques qui représente le travail accumulé sur la branche courante de l'arbre des époques, e.g. le nombre d'opérations intégrées, les noeuds actuellement sur cette branche... L'ordre strict total entre les époques serait ainsi construit à partir de la comparaison entre les valeurs de ces métriques de leur opération *ren* respective.

Il conviendra d'adjoindre à cette nouvelle définition de *priority* un nouveau couple de fonctions `renameId` et `revertRenameId` respectant la contrainte de réciprocity de ces fonctions, ou de mettre en place une autre implémentation du mécanisme de renommage ne nécessitant pas cette contrainte, telle qu'une implémentation basée sur le journal des opérations (cf. sous-section 2.5.7, page 99).

Il conviendra aussi d'étudier la possibilité de combiner l'utilisation de plusieurs relations *priority* pour minimiser le surcoût global du mécanisme de renommage, e.g. en fonction de la distance entre deux époques.

Finalement, il sera nécessaire de valider l'approche proposée par une évaluation comparative par rapport à l'approche actuelle. Cette évaluation pourrait consister à monitorer le coût du système pour observer si l'approche proposée permet de réduire les calculs de manière globale. Plusieurs configurations de paramètres pourraient aussi être utilisées pour déterminer l'impact respectif de chaque paramètre sur les résultats.

## 4.2.2 Détection et fusion manuelle de versions distantes

À l'issue de cette thèse, nous constatons plusieurs limites des mécanismes de résolution de conflits automatiques dans les systèmes P2P à large échelle sujet au churn. La première d'entre elles est l'utilisation d'un contexte causal. Le contexte causal est utilisé par les mécanismes de résolution de conflits pour :

- (i) Satisfaire le modèle de cohérence causale, c.-à-d. assurer que si nous avons deux modifications  $m_1$  et  $m_2$  telles que  $m_1 \rightarrow m_2$ , alors l'effet de  $m_2$  supplantera celui de  $m_1$ . Ceci permet d'éviter des anomalies de comportement de la part de la structure de données du point de vue des utilisateur-rices, par exemple la résurgence d'un élément supprimé au préalable.
- (ii) Permettre de préserver l'intention d'une modification malgré l'intégration préalable de modifications concurrentes.

Le contexte causal est utilisé de manière différente en fonction du mécanisme de résolution de conflit. Dans l'approche OT, le contexte causal est utilisé par l'algorithme de contrôle pour déterminer les modifications concurrentes à une modification lors de son intégration, afin de prendre en compte leurs effets. Dans l'approche CRDT, le contexte causal est utilisé par la structure de données répliquée à la génération de la modification pour en faire une modification nativement commutative avec les modifications concurrentes, c.-à-d. pour en faire un élément du sup-demi-treillis représentant la structure de données répliquée.

Le contexte causal peut être représenté de différentes manières. Par exemple, le contexte causal peut prendre la forme d'un vecteur de version [28, 29] ou d'un Directed Acyclic Graph (DAG) des modifications [30]. Cependant, de manière intrinsèque, le contexte causal ne fait que de croître au fur et à mesure que des modifications sont effectuées ou que des noeuds rejoignent le système, incrémentant son surcoût en métadonnées, calculs et bande-passante.

La stabilité causale permet cependant de réduire le surcoût lié au contexte causal. En effet, la stabilité causale permet d'établir le contexte commun à l'ensemble des noeuds, c.-à-d. l'ensemble des modifications que l'ensemble des noeuds ont intégré. Ces modifications font alors partie de l'histoire commune et n'ont plus besoin d'être considérées par les mécanismes de résolution de conflits automatiques. La stabilité causale permet donc de déterminer et de tronquer la partie commune du contexte causal pour éviter que ce dernier ne pénalise les performances du système à terme.

La stabilité causale est cependant une contrainte forte dans les systèmes P2P dynamiques à large échelle sujet au churn. Il ne suffit en effet que d'un noeud déconnecté pour empêcher la stabilité causale de progresser. Pour répondre à ce problème, nous avons dès lors tout un spectre d'approches possibles, proposant chacune un compromis entre le surcoût du contexte causal et la probabilité de rejeter des modifications. Les extrémités de ce spectre d'approches sont les suivantes :

- (i) Considérer tout noeud déconnecté comme déconnecté de manière définitive, et donc les ignorer dans le calcul de la stabilité causale. Cette première approche permet à la stabilité causale de progresser, et ainsi aux noeuds connectés de travailler dans des conditions optimales. Mais elle implique cependant que les modifications potentielles des noeuds déconnectés soient perdues, c.-à-d. de ne plus pouvoir les intégrer en l'absence d'un lien entre leur contexte causal de génération et le contexte causal actuel de chaque autre noeud. Il s'agit là de la stratégie la plus agressive en terme de GC du contexte causal.
- (ii) Assurer en toutes circonstances la capacité d'intégration des modifications des noeuds, même ceux déconnectés. Cette seconde approche permet de garantir que les modifications potentielles des noeuds déconnectés pourront être intégrées automatiquement, dans l'éventualité où ces derniers se reconnectent à terme. Mais elle implique de bloquer potentiellement de manière définitive la stabilité causale et donc le mécanisme de GC du contexte causal. Il s'agit là de la stratégie la plus timide en terme de GC du contexte causal.

La seconde limite que nous constatons est la limite des mécanismes actuels de résolution de conflits automatiques pour préserver l'intention des utilisateur-rices. Par exemple,

les mécanismes de résolution de conflits automatiques pour le type Séquence présentés dans ce manuscrit (cf. section 1.3, page 23) définissent l'intention de la manière suivante : *l'intégration de la modification par les noeuds distants doit reproduire l'effet de la modification sur la copie d'origine*. Cette définition assure que chaque modification est porteuse d'une intention, mais limite voire ignore toute la dimension sémantique de la dite intention. Nous conjecturons que l'absence de dimension sémantique réduit les cas d'utilisation de ces mécanismes.

Considérons par exemple une édition collaborative d'un même texte par un ensemble de noeuds. Lors de la présence d'une faute de frappe dans le texte, e.g. le mot "HLLO", plusieurs utilisateur-rices peuvent la corriger en concurrence, c.-à-d. insérer l'élément "E" entre "H" et "L". Les mécanismes de résolution de conflits automatiques permettent aux noeuds d'obtenir des résultats qui convergent mais à notre sens insatisfaisant, e.g. "HEEEEEELLO". Nous considérons ce type de résultats comme des anomalies, au même titre que l'entrelacement [63]. Dans le cadre de collaborations temps réel à échelle limitée, nous conjecturons cependant qu'une granularité fine des modifications permet de pallier ce problème. En effet, les utilisateur-rices peuvent observer une anomalie produite par le mécanisme de résolution de conflits automatique, déduire l'intention initiale des modifications concernées et la restaurer par le biais d'actions supplémentaires de compensation.

Cependant, dans le cadre de collaborations asynchrones ou à large échelle, nous conjecturons que ces anomalies de résolution de conflits s'accumulent. Cette accumulation peut atteindre un seuil rendant laborieuse la déduction et le rétablissement de l'intention initiale des modifications. Le travail imposé aux utilisateur-rices pour résoudre ces anomalies par le biais d'actions de compensation peut alors entraver la collaboration. Pour reprendre l'exemple de l'édition collaborative de texte, nous pouvons constater de tels cas suite à de la duplication de contenu et/ou l'entrelacement de mots, phrases voire paragraphes nuisant à la clarté et correction du texte. Il convient alors de s'interroger sur le bien-fondé de l'utilisation de mécanismes de résolutions de conflits automatiques pour intégrer un ensemble de modifications dans l'ensemble des situations.

Ainsi, pour répondre aux limites des mécanismes de résolution conflits automatiques dans les systèmes P2P à large échelle, c.-à-d. l'augmentation de leur surcoût et la pertinence de leur résultat, nous souhaitons proposer une approche combinant un ou des mécanismes de résolution de conflits automatiques avec un ou des mécanismes de résolution de conflits manuels. L'idée derrière cette approche est de faire varier le mécanisme de résolution de conflits utilisé pour intégrer des modifications. Le choix du mécanisme de résolution de conflits utilisé peut se faire à partir de la valeur d'une distance calculée entre la version courante de la donnée répliquée et celle de la génération de la modification à intégrer, ou d'une évaluation de la qualité du résultat de l'intégration de la modification. Par exemple :

- (i) Si la distance calculée se trouve dans un intervalle de valeurs pour lequel nous disposons d'un mécanisme de résolution de conflits automatique satisfaisant, utiliser ce dernier. Ainsi, nous pouvons envisager de reposer sur plusieurs mécanismes de résolution de conflits automatiques, de plus en plus complexes et pertinents mais coûteux, sans dégrader les performances du système dans le cas de base.
- (ii) Si la distance calculée dépasse la distance seuil, c.-à-d. que nous ne disposons plus



à ce stade de mécanismes de résolution de conflits automatiques satisfaisants, faire intervenir les utilisateur-rices par le biais d'un mécanisme de résolution de conflits manuel. L'utilisation d'un mécanisme manuel n'exclut cependant pas tout pré-travail de notre part pour réduire la charge de travail des utilisateur-rices dans le processus de fusion.

Dans un premier temps, cette approche pourrait se focaliser sur un type d'application spécifique, e.g. l'édition collaborative de texte.

Cette approche nous permettrait de répondre aux limites soulevées précédemment. En effet, elle permettrait de limiter la génération d'anomalies par le mécanisme de résolution de conflits automatique en faisant intervenir les utilisateur-rices. Puis, puisque nous déléguons aux utilisateur-rices l'intégration des modifications à partir d'une distance seuil, nous pouvons dès lors reconsidérer les métadonnées conservées par les noeuds pour les mécanismes de résolution de conflits automatiques. Notamment, nous pouvons identifier les noeuds se trouvant au-delà de cette distance seuil d'après leur dernier contexte causal connu et ne plus les prendre en compte pour le calcul de la stabilité causale. Cette approche permettrait donc de réduire le surcoût lié au contexte causal et limiter la perte de modifications, tout en prenant en considération l'ajout de travail aux utilisateur-rices.

Pour mener à bien ce travail, il conviendra tout d'abord de définir la notion de distance entre versions de la donnée répliquée. Nous envisageons de baser cette dernière sur les deux aspects temporel et spatial, c.-à-d. en utilisant la distance entre contextes causaux et la distance entre contenus. Dans le cadre de l'édition collaborative, nous pourrions pour cela nous baser sur les travaux existants pour évaluer la distance entre deux textes.

Il conviendra ensuite de déterminer comment établir la valeur seuil à partir de laquelle la distance entre versions est jugée trop importante. Les approches d'évaluation de la qualité du résultat [91] pourront être utilisées pour déterminer un couple  $\langle$  méthode de calcul de la distance, valeur de distance  $\rangle$  spécifiant les cas pour lesquels les méthodes de résolution de conflits automatiques ne produisent plus un résultat satisfaisant. Le couple obtenu pourra ensuite être confirmé par le biais d'expériences utilisateurs inspirées de [66, 67].

Finalement, il conviendra de proposer un mécanisme de résolution de conflits adapté pour gérer les éventuelles fusions d'une même modification de façon concurrente par un mécanisme automatique et par un mécanisme manuel, ou à défaut un mécanisme de conscience de groupe invitant les utilisateur-rices à effectuer des actions de compensation.

### 4.2.3 Étude comparative des différents modèles de synchronisation pour CRDTs

Comme évoqué dans l'état de l'art (cf. section 1.2.2, page 20), un nouveau modèle de synchronisation pour CRDT fut proposé récemment [25]. Ce dernier propose une synchronisation des noeuds par le biais de différences d'états.

Pour rappel, ce nouveau modèle de synchronisation se base sur le modèle de synchronisation par états. Il partage les mêmes pré-requis, à savoir la nécessité d'une fonction **merge** associative, commutative et idempotente. Cette dernière doit permettre de la fusion toute paire d'états possible en calculant leur borne supérieure, c.-à-d. leur LUB [17].

La spécificité de ce nouveau modèle de synchronisation est de calculer pour chaque modification la différence d'état correspondante. Cette différence correspond à un élément irréductible du sup-demi-treillis du CRDT [37], c.-à-d. un état particulier de ce dernier. Cet élément irréductible peut donc être diffusé et intégré par les autres noeuds, toujours à l'aide de la fonction `merge`.

Ce modèle de synchronisation permet alors d'adopter une variété de stratégies de synchronisation, e.g. diffusion des différences de manière atomique, fusion de plusieurs différences puis diffusion du résultat..., et donc de répondre à une grande variété de cas d'utilisation.

Dans notre comparaison des modèles de synchronisation (cf. section 1.2.2, page 21), nous avons justifié que les CRDTs synchronisés par différences d'états peuvent être utilisés dans les mêmes contextes que les CRDTs synchronisés par états et que les CRDTs synchronisés par opérations. Cette conclusion nous mène à reconsidérer l'intérêt des autres modèles de synchronisation de nos jours.

Par exemple, un CRDT synchronisé par différences d'états correspond à un CRDT synchronisé par états dont nous avons identifié les éléments irréductibles. La différence entre ces deux modèles de synchronisation semble reposer seulement sur la possibilité d'utiliser ces éléments irréductibles pour propager les modifications, en place et lieu des états complets. Nous conjecturons donc que le modèle de synchronisation par états est rendu obsolète par celui par différences d'états. Il serait intéressant de confirmer cette supposition.

En revanche, l'utilisation du modèle de synchronisation par opérations conduit généralement à une spécification différente du CRDT, les opérations permettant d'encoder plus librement les modifications. Notamment, l'utilisation d'opérations peut mener à des algorithmes d'intégration des modifications différents que ceux de la fonction `merge`. Il convient de comparer ces algorithmes pour déterminer si le modèle de synchronisation par opérations peut présenter un intérêt en termes de surcoût.

Au-delà de ce premier aspect, il convient d'explorer d'autres pistes pouvant induire des avantages et inconvénients pour chacun de ces modèles de synchronisation. À l'issue de cette thèse, nous identifions les pistes suivantes :

- (i) La composition de CRDTs, c.-à-d. la capacité de combiner et de mettre en relation plusieurs CRDTs au sein d'un même système, afin d'offrir des fonctionnalités plus complexes. Par exemple, une composition de CRDTs peut se traduire par l'ajout de dépendances entre les modifications des différents CRDTs composés. Le modèle de synchronisation par opérations nous apparaît plus adapté pour cette utilisation, de par le découplage qu'il induit entre les CRDTs et la couche de livraison de messages.
- (ii) L'utilisation de CRDTs au sein de systèmes non-sûrs, c.-à-d. pouvant compter un ou plusieurs adversaires byzantins [92]. Dans de tels systèmes, les adversaires byzantins peuvent générer des modifications différentes mais qui sont perçues comme identiques par les mécanismes de résolution de conflits. Cette attaque, nommée *équivoque*, peut provoquer la divergence définitive des copies. [88] propose une solution adaptée aux systèmes P2P à large échelle. Celle-ci se base notamment sur l'utilisation de journaux infalsifiables. Il convient alors d'étudier si l'utilisation de ces structures ne limite pas le potentiel du modèle de synchronisation par différences

d'états, e.g. en interdisant la diffusion des modifications par états complets.

Un premier objectif de notre travail serait de proposer des directives sur le modèle de synchronisation à privilégier en fonction du contexte d'utilisation du CRDT.

Ce travail permettrait aussi d'étudier la combinaison des modèles de synchronisation par opérations et par différences d'états au sein d'un même CRDT. Le but serait notamment d'identifier les paramètres conduisant à privilégier un modèle de synchronisation par rapport à l'autre, de façon à permettre aux noeuds de basculer dynamiquement entre les deux.

#### 4.2.4 Approfondissement du patron de conception de Pure Operation-Based CRDTs

BAQUERO et al. [24] proposent un framework pour concevoir des CRDTs synchronisés par opérations : Pure Operation-Based CRDTs. Ce framework a plusieurs objectifs :

- (i) Proposer une approche partiellement générique pour définir un CRDT synchronisé par opérations.
- (ii) Factoriser les métadonnées utilisées par le CRDT pour le mécanisme de résolution de conflits, notamment pour identifier les éléments, et celles utilisées par la couche livraison, notamment pour identifier les opérations.
- (iii) Inclure des mécanismes de GC de ces métadonnées pour réduire la taille de l'état.

Pour cela, les auteurs se limitent aux CRDTs purs synchronisés par opérations, c.-à-d. les CRDTs dont les modifications enrichies de leurs arguments et d'une estampille fournie par la couche de livraison des messages sont commutatives. Pour ces CRDTs, les auteurs proposent un framework générique permettant leur spécification sous la forme d'un journal partiellement ordonné des opérations, nommé PO-Log. Les auteurs associent le PO-Log à une couche de livraison Reliable Causal Broadcast (RCB) des opérations.

Les auteurs définissent ensuite le concept de stabilité causale. Ce concept leur permet de retirer les métadonnées de causalité des opérations du PO-Log lorsque celles-ci sont déterminées comme étant causalement stables.

Finalement, les auteurs définissent un ensemble de relations, spécifiques à chaque CRDT, qui permettent d'exprimer la *redondance causale*. La redondance causale permet de spécifier quand retirer une opération du PO-Log, car rendue obsolète par une autre opération.

Cette approche souffre toutefois de plusieurs limites. Tout d'abord, elle repose sur l'utilisation d'une couche de livraison RCB. Cette couche satisfait le modèle de livraison causale. Mais pour rappel, ce modèle induit l'ajout de données de causalité précises à chaque opération, sous la forme d'un vecteur de version ou d'une barrière causale. Nous jugeons ce modèle trop coûteux pour un système P2P dynamique à large échelle sujet au churn.

En plus du coût induit en termes de métadonnées et de bande-passante, le modèle de livraison causale peut aussi introduire un délai superflu dans la livraison des opérations. En effet, ce modèle impose que tous les messages précédant un nouveau message d'après la relation *happens-before* soient eux-mêmes livrés avant de livrer ce dernier. Il en résulte que

des opérations peuvent être mises en attente par la couche livraison, e.g. suite à la perte d'une de leurs dépendances d'après la relation *happens-before*, alors que leurs dépendances réelles ont déjà été livrées et que les opérations sont de fait intégrables en l'état. Plusieurs travaux [93, 35] ont noté ce problème. Pour y répondre et ainsi améliorer la réactivité du framework Pure Operation-Based, ils proposent d'exposer les opérations mises en attente par la couche livraison au CRDT. Bien que fonctionnelle, cette approche induit toujours le coût d'une couche de livraison respectant le modèle de livraison causale et nous fait considérer la raison de ce coût, le modèle de livraison n'étant dès lors plus respecté.

Ensuite, ce framework impose que la modification **prepare** ne puisse pas inspecter l'état courant du noeud. Cette contrainte est compatible avec les CRDTs pour les types de données simples qui sont considérés dans [24], e.g. le Compteur ou l'Ensemble. Elle empêche cependant l'expression de CRDTs pour des types de données plus complexes, e.g. la Séquence ou le Graphe. Nous jugeons dommageable qu'un framework pour la conception de CRDTs limite de la sorte son champ d'application.

Finalement, les auteurs ne considèrent que des types de données avec des modifications à granularité fixe. Ainsi, ils définissent la notion de redondance causale en se limitant à ce type de modifications. Par exemple, ils définissent que la suppression d'un élément d'un ensemble rend obsolète les ajouts précédents de cet élément. Cependant, dans le cadre d'autres types de données, e.g. la Séquence, une modification peut concerner un ensemble d'éléments de taille variable. Une opération peut donc être rendue obsolète non pas par une opération, mais par un ensemble d'opérations. Par exemple, les suppressions d'éléments formant une sous-chaîne rendent obsolète l'insertion de cette sous-chaîne. Ainsi, la notion de redondance causale est incomplète et souffre de l'absence d'une notion d'obsolescence partielle d'une opération.

Pour répondre aux différents problèmes soulevés, nous souhaitons proposer une extension du framework Pure Operation-Based CRDTs. Nos objectifs sont les suivants :

- (i) Proposer un framework mettant en lumière la présence et le rôle de deux modèles de livraison :
  - (i) Le modèle de livraison minimal requis par le CRDT pour assurer la convergence forte à terme [16].
  - (ii) Le modèle de livraison utilisé par le système, qui doit être égal ou plus contraint que modèle de livraison minimal du CRDT. Ce second modèle de livraison est une stratégie permettant au système de respecter un modèle de cohérence donné et régissant les règles de compaction de l'état. Il peut être amené à évoluer en fonction de l'état du système et de ses besoins. Par exemple, un système peut par défaut utiliser le modèle de livraison causale pour assurer le modèle de cohérence causal. Puis, lorsque le nombre de noeuds atteint un seuil donné, le système peut passer au modèle de livraison FIFO pour assurer le modèle de cohérence PRAM afin de réduire les coûts en bande-passante.
- (ii) Étendre la notion de redondance causale pour prendre en compte la redondance partielle des opérations. De plus, nous souhaitons rendre cette notion accessible à la couche de livraison, pour détecter au plus tôt les opérations désormais obsolètes et prévenir leur diffusion.

- (iii) Identifier et classifier les mécanismes de résolution de conflits, pour déterminer lesquels sont indépendants de l'état courant pour la génération des opérations et lesquels nécessitent d'inspecter l'état courant dans **prepare**.



# Bibliographie

- [1] Mihai LETIA, Nuno PREGUIÇA et Marc SHAPIRO. « Consistency without concurrency control in large, dynamic systems ». In : *LADIS 2009 - 3rd ACM SIGOPS International Workshop on Large Scale Distributed Systems and Middleware*. T. 44. Operating Systems Review 2. Big Sky, MT, United States : Assoc. for Computing Machinery, oct. 2009, p. 29–34. DOI : 10.1145/1773912.1773921. URL : <https://hal.inria.fr/hal-01248270>.
- [2] Marek ZAWIRSKI, Marc SHAPIRO et Nuno PREGUIÇA. « Asynchronous rebalancing of a replicated tree ». In : *Conférence Française en Systèmes d'Exploitation (CFSE)*. Saint-Malo, France, mai 2011, p. 12. URL : <https://hal.inria.fr/hal-01248197>.
- [3] Matthieu NICOLAS, Victorien ELVINGER, Gérald OSTER, Claudia-Lavinia IGNAT et François CHAROY. « MUTE : A Peer-to-Peer Web-based Real-time Collaborative Editor ». In : *ECSCW 2017 - 15th European Conference on Computer-Supported Cooperative Work*. T. 1. Proceedings of 15th European Conference on Computer-Supported Cooperative Work - Panels, Posters and Demos 3. Sheffield, United Kingdom : EUSSET, août 2017, p. 1–4. DOI : 10.18420/ecscw2017\\_p5. URL : <https://hal.inria.fr/hal-01655438>.
- [4] Luc ANDRÉ, Stéphane MARTIN, Gérald OSTER et Claudia-Lavinia IGNAT. « Supporting Adaptable Granularity of Changes for Massive-Scale Collaborative Editing ». In : *International Conference on Collaborative Computing : Networking, Applications and Worksharing - CollaborateCom 2013*. Austin, TX, USA : IEEE Computer Society, oct. 2013, p. 50–59. DOI : 10.4108/icst.collaboratecom.2013.254123.
- [5] Victorien ELVINGER. « Réplication sécurisée dans les infrastructures pair-à-pair de collaboration ». Theses. Université de Lorraine, juin 2021. URL : <https://hal.univ-lorraine.fr/tel-03284806>.
- [6] Hoang-Long NGUYEN, Claudia-Lavinia IGNAT et Olivier PERRIN. « Trusternity : Auditing Transparent Log Server with Blockchain ». In : *Companion of the The Web Conference 2018*. Lyon, France, avr. 2018. DOI : 10.1145/3184558.3186938. URL : <https://hal.inria.fr/hal-01883589>.
- [7] Hoang-Long NGUYEN, Jean-Philippe EISENBARTH, Claudia-Lavinia IGNAT et Olivier PERRIN. « Blockchain-Based Auditing of Transparent Log Servers ». In : *32th IFIP Annual Conference on Data and Applications Security and Privacy (DBSec)*. Sous la dir. de Florian KERSCHBAUM et Stefano PARABOSCHI. T. LNCS-10980.

- Data and Applications Security and Privacy XXXII. Part 1 : Administration. Bergamo, Italy : Springer International Publishing, juil. 2018, p. 21–37. DOI : 10.1007/978-3-319-95729-6\\_2. URL : <https://hal.archives-ouvertes.fr/hal-01917636>.
- [8] Abhinandan DAS, Indranil GUPTA et Ashish MOTIVALA. « SWIM : scalable weakly-consistent infection-style process group membership protocol ». In : *Proceedings International Conference on Dependable Systems and Networks*. 2002, p. 303–312. DOI : 10.1109/DSN.2002.1028914.
- [9] Armon DADGAR, James PHILLIPS et Jon CURREY. « Lifeguard : Local health awareness for more accurate failure detection ». In : *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*. IEEE. 2018, p. 22–25.
- [10] Brice NÉDELEC, Julian TANKE, Davide FREY, Pascal MOLLI et Achour MOSTÉFAOUI. « An adaptive peer-sampling protocol for building networks of browsers ». In : *World Wide Web* 21.3 (2018), p. 629–661.
- [11] Mike BURMESTER et Yvo DESMEDT. « A secure and efficient conference key distribution system ». In : *Advances in Cryptology — EUROCRYPT’94*. Sous la dir. d’Alfredo DE SANTIS. Berlin, Heidelberg : Springer Berlin Heidelberg, 1995, p. 275–286. ISBN : 978-3-540-44717-7.
- [12] Rachid GUERRAOUI, Matej PAVLOVIC et Dragos-Adrian SEREDINSCHI. « Trade-offs in replicated systems ». In : *IEEE Data Engineering Bulletin* 39.ARTICLE (2016), p. 14–26.
- [13] Yasushi SAITO et Marc SHAPIRO. « Optimistic Replication ». In : *ACM Comput. Surv.* 37.1 (mar. 2005), p. 42–81. ISSN : 0360-0300. DOI : 10.1145/1057977.1057980. URL : <https://doi.org/10.1145/1057977.1057980>.
- [14] Douglas B TERRY, Marvin M THEIMER, Karin PETERSEN, Alan J DEMERS, Mike J SPREITZER et Carl H HAUSER. « Managing Update Conflicts in Bayou, a Weakly Connected Replicated Storage System ». In : *SIGOPS Oper. Syst. Rev.* 29.5 (déc. 1995), p. 172–182. ISSN : 0163-5980. DOI : 10.1145/224057.224070. URL : <https://doi.org/10.1145/224057.224070>.
- [15] Leslie LAMPORT. « Time, Clocks, and the Ordering of Events in a Distributed System ». In : *Commun. ACM* 21.7 (juil. 1978), p. 558–565. ISSN : 0001-0782. DOI : 10.1145/359545.359563. URL : <https://doi.org/10.1145/359545.359563>.
- [16] Marc SHAPIRO, Nuno M. PREGUIÇA, Carlos BAQUERO et Marek ZAWIRSKI. « Conflict-Free Replicated Data Types ». In : *Proceedings of the 13th International Symposium on Stabilization, Safety, and Security of Distributed Systems*. SSS 2011. 2011, p. 386–400. DOI : 10.1007/978-3-642-24550-3\_29.
- [17] Nuno M. PREGUIÇA, Carlos BAQUERO et Marc SHAPIRO. « Conflict-free Replicated Data Types (CRDTs) ». In : *CoRR* abs/1805.06358 (2018). arXiv : 1805.06358. URL : <http://arxiv.org/abs/1805.06358>.



- 
- [18] Nuno M. PREGUIÇA. « Conflict-free Replicated Data Types : An Overview ». In : *CoRR* abs/1806.10254 (2018). arXiv : 1806.10254. URL : <http://arxiv.org/abs/1806.10254>.
  - [19] B. A. DAVEY et H. A. PRIESTLEY. *Introduction to Lattices and Order*. 2<sup>e</sup> éd. Cambridge University Press, 2002. DOI : 10.1017/CB09780511809088.
  - [20] Paul R. JOHNSON et Robert THOMAS. *RFC0677 : Maintenance of duplicate databases*. RFC Editor, 1975.
  - [21] Weihai YU et Sigbjørn ROSTAD. « A Low-Cost Set CRDT Based on Causal Lengths ». In : *Proceedings of the 7th Workshop on Principles and Practice of Consistency for Distributed Data*. New York, NY, USA : Association for Computing Machinery, 2020. ISBN : 9781450375245. URL : <https://doi.org/10.1145/3380787.3393678>.
  - [22] Marc SHAPIRO, Nuno PREGUIÇA, Carlos BAQUERO et Marek ZAWIRSKI. *A comprehensive study of Convergent and Commutative Replicated Data Types*. Research Report RR-7506. Inria – Centre Paris-Rocquencourt ; INRIA, jan. 2011, p. 50. URL : <https://hal.inria.fr/inria-00555588>.
  - [23] Carlos BAQUERO, Paulo Sérgio ALMEIDA et Ali SHOKER. « Making Operation-Based CRDTs Operation-Based ». In : *Proceedings of the First Workshop on Principles and Practice of Eventual Consistency*. PaPEC '14. Amsterdam, The Netherlands : Association for Computing Machinery, 2014. ISBN : 9781450327169. DOI : 10.1145/2596631.2596632. URL : <https://doi.org/10.1145/2596631.2596632>.
  - [24] Carlos BAQUERO, Paulo Sergio ALMEIDA et Ali SHOKER. *Pure Operation-Based Replicated Data Types*. 2017. arXiv : 1710.04469 [cs.DC].
  - [25] Paulo Sérgio ALMEIDA, Ali SHOKER et Carlos BAQUERO. « Efficient State-Based CRDTs by Delta-Mutation ». In : *Networked Systems*. Sous la dir. d’Ahmed BOUAJJANI et Hugues FAUCONNIER. Cham : Springer International Publishing, 2015, p. 62–76. ISBN : 978-3-319-26850-7.
  - [26] Paulo Sérgio ALMEIDA, Ali SHOKER et Carlos BAQUERO. « Delta state replicated data types ». In : *Journal of Parallel and Distributed Computing* 111 (jan. 2018), p. 162–173. ISSN : 0743-7315. DOI : 10.1016/j.jpdc.2017.08.003. URL : <http://dx.doi.org/10.1016/j.jpdc.2017.08.003>.
  - [27] Prince MAHAJAN, Lorenzo ALVISI, Mike DAHLIN et al. « Consistency, availability, and convergence ». In : *University of Texas at Austin Tech Report* 11 (2011), p. 158.
  - [28] Friedemann MATTERN et al. *Virtual time and global states of distributed systems*. Univ., Department of Computer Science, 1988.
  - [29] Colin FIDGE. « Logical Time in Distributed Computing Systems ». In : *Computer* 24.8 (août 1991), p. 28–33. ISSN : 0018-9162. DOI : 10.1109/2.84874. URL : <https://doi.org/10.1109/2.84874>.

- [30] Ravi PRAKASH, Michel RAYNAL et Mukesh SINGHAL. « An Adaptive Causal Ordering Algorithm Suited to Mobile Computing Environments ». In : *Journal of Parallel and Distributed Computing* 41.2 (1997), p. 190–204. ISSN : 0743-7315. DOI : <https://doi.org/10.1006/jpdc.1996.1300>. URL : <https://www.sciencedirect.com/science/article/pii/S0743731596913003>.
- [31] D. S. PARKER, G. J. POPEK, G. RUDISIN, A. STOUGHTON, B. J. WALKER, E. WALTON, J. M. CHOW, D. EDWARDS, S. KISER et C. KLINE. « Detection of Mutual Inconsistency in Distributed Systems ». In : *IEEE Trans. Softw. Eng.* 9.3 (mai 1983), p. 240–247. ISSN : 0098-5589. DOI : 10.1109/TSE.1983.236733. URL : <https://doi.org/10.1109/TSE.1983.236733>.
- [32] Giuseppe DECANDIA, Deniz HASTORUN, Madan JAMPANI, Gunavardhan KAKULAPATI, Avinash LAKSHMAN, Alex PILCHIN, Swaminathan SIVASUBRAMANIAN, Peter VOSSHALL et Werner VOGELS. « Dynamo : Amazon’s highly available key-value store ». In : *ACM SIGOPS operating systems review* 41.6 (2007), p. 205–220.
- [33] Nico KRUBER, Maik LANGE et Florian SCHINTKE. « Approximate Hash-Based Set Reconciliation for Distributed Replica Repair ». In : *2015 IEEE 34th Symposium on Reliable Distributed Systems (SRDS)*. 2015, p. 166–175. DOI : 10.1109/SRDS.2015.30.
- [34] Ricardo Jorge Tomé GONÇALVES, Paulo Sérgio ALMEIDA, Carlos BAQUERO et Vitor FONTE. « DottedDB : Anti-Entropy without Merkle Trees, Deletes without Tombstones ». In : *2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*. 2017, p. 194–203. DOI : 10.1109/SRDS.2017.28.
- [35] Jim BAUWENS et Elisa Gonzalez BOIX. « Improving the Reactivity of Pure Operation-Based CRDTs ». In : *Proceedings of the 8th Workshop on Principles and Practice of Consistency for Distributed Data*. PaPoC ’21. Online, United Kingdom : Association for Computing Machinery, 2021. ISBN : 9781450383387. DOI : 10.1145/3447865.3457968. URL : <https://doi.org/10.1145/3447865.3457968>.
- [36] Carlos BAQUERO, Paulo Sérgio ALMEIDA et Ali SHOKER. « Making Operation-Based CRDTs Operation-Based ». In : *Distributed Applications and Interoperable Systems*. Sous la dir. de Kostas MAGOUTIS et Peter PIETZUCH. Berlin, Heidelberg : Springer Berlin Heidelberg, 2014, p. 126–140.
- [37] Vitor ENES, Paulo Sérgio ALMEIDA, Carlos BAQUERO et João LEITÃO. « Efficient Synchronization of State-Based CRDTs ». In : *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. 2019, p. 148–159. DOI : 10.1109/ICDE.2019.00022.
- [38] Clarence A. ELLIS et Simon J. GIBBS. « Concurrency Control in Groupware Systems ». In : *Proceedings of the 1989 ACM SIGMOD International Conference on Management of Data*. SIGMOD ’89. Portland, Oregon, USA : Association for Computing Machinery, 1989, p. 399–407. ISBN : 0897913175. DOI : 10.1145/67544.66963. URL : <https://doi.org/10.1145/67544.66963>.

- 
- [39] Chengzheng SUN et Clarence ELLIS. « Operational transformation in real-time group editors : issues, algorithms, and achievements ». In : *Proceedings of the 1998 ACM conference on Computer supported cooperative work*. 1998, p. 59–68.
- [40] Matthias RESSEL, Doris NITSCHÉ-RUHLAND et Rul GUNZENHÄUSER. « An integrating, transformation-oriented approach to concurrency control and undo in group editors ». In : *Proceedings of the 1996 ACM conference on Computer supported cooperative work*. 1996, p. 288–297.
- [41] Chengzheng SUN, Yun YANG, Yanchun ZHANG et David CHEN. « A consistency model and supporting schemes for real-time cooperative editing systems ». In : *Australian Computer Science Communications* 18 (1996), p. 582–591.
- [42] David SUN et Chengzheng SUN. « Context-Based Operational Transformation in Distributed Collaborative Editing Systems ». In : *Parallel and Distributed Systems, IEEE Transactions on* 20 (nov. 2009), p. 1454–1470. DOI : 10.1109/TPDS.2008.240.
- [43] Chengzheng SUN, Xiaohua JIA, Yanchun ZHANG, Yun YANG et David CHEN. « Achieving convergence, causality preservation, and intention preservation in real-time cooperative editing systems ». In : *ACM Transactions on Computer-Human Interaction (TOCHI)* 5.1 (1998), p. 63–108.
- [44] Gérald OSTER, Pascal MOLLI, Pascal URSO et Abdessamad IMINE. « Tombstone Transformation Functions for Ensuring Consistency in Collaborative Editing Systems ». In : *2006 International Conference on Collaborative Computing : Networking, Applications and Worksharing*. 2006, p. 1–10. DOI : 10.1109/COLCOM.2006.361867.
- [45] Chengzheng SUN, Xiaohua JIA, Yanchun ZHANG, Yun YANG et David CHEN. « Achieving Convergence, Causality Preservation, and Intention Preservation in Real-Time Cooperative Editing Systems ». In : *ACM Trans. Comput.-Hum. Interact.* 5.1 (mar. 1998), p. 63–108. ISSN : 1073-0516. DOI : 10.1145/274444.274447. URL : <https://doi.org/10.1145/274444.274447>.
- [46] Stéphane WEISS, Pascal URSO et Pascal MOLLI. « Logoot : A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks ». In : *Proceedings of the 29th International Conference on Distributed Computing Systems - ICDCS 2009*. Montreal, QC, Canada : IEEE Computer Society, juin 2009, p. 404–412. DOI : 10.1109/ICDCS.2009.75. URL : <http://doi.ieeeecomputersociety.org/10.1109/ICDCS.2009.75>.
- [47] Bernadette CHARRON-BOST. « Concerning the size of logical clocks in distributed systems ». In : *Information Processing Letters* 39.1 (1991), p. 11–16.
- [48] Gérald OSTER, Pascal URSO, Pascal MOLLI et Abdessamad IMINE. « Data Consistency for P2P Collaborative Editing ». In : *ACM Conference on Computer-Supported Cooperative Work - CSCW 2006*. Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work. Banff, Alberta, Canada : ACM Press, nov. 2006, p. 259–268. URL : <https://hal.inria.fr/inria-00108523>.

- [49] Hyun-Gul ROH, Myeongjae JEON, Jin-Soo KIM et Joonwon LEE. « Replicated abstract data types : Building blocks for collaborative applications ». In : *Journal of Parallel and Distributed Computing* 71.3 (2011), p. 354–368. ISSN : 0743-7315. DOI : <https://doi.org/10.1016/j.jpdc.2010.12.006>. URL : <http://www.sciencedirect.com/science/article/pii/S0743731510002716>.
- [50] Nuno PREGUICA, Joan Manuel MARQUES, Marc SHAPIRO et Mihai LETIA. « A Commutative Replicated Data Type for Cooperative Editing ». In : *2009 29th IEEE International Conference on Distributed Computing Systems*. Juin 2009, p. 395–403. DOI : 10.1109/ICDCS.2009.20.
- [51] Marc SHAPIRO et Nuno PREGUIÇA. *Designing a commutative replicated data type*. Research Report RR-6320. INRIA, 2007. URL : <https://hal.inria.fr/inria-00177693>.
- [52] Paulo Sérgio ALMEIDA, Carlos BAQUERO, Ricardo GONÇALVES, Nuno PREGUIÇA et Victor FONTE. « Scalable and Accurate Causality Tracking for Eventually Consistent Stores ». In : *Distributed Applications and Interoperable Systems*. Sous la dir. de Kostas MAGOUTIS et Peter PIETZUCH. Berlin, Heidelberg : Springer Berlin Heidelberg, 2014, p. 67–81. ISBN : 978-3-662-43352-2.
- [53] Charbel RAHHAL, Stéphane WEISS, Hala SKAF-MOLLI, Pascal URSO et Pascal MOLLI. *Undo in Peer-to-peer Semantic Wikis*. Research Report RR-6870. INRIA, 2009, p. 18. URL : <https://hal.inria.fr/inria-00366317>.
- [54] Mehdi AHMED-NACER, Claudia-Lavinia IGNAT, Gérald OSTER, Hyun-Gul ROH et Pascal URSO. « Evaluating CRDTs for Real-time Document Editing ». In : *11th ACM Symposium on Document Engineering*. Sous la dir. d'ACM. Mountain View, California, United States, sept. 2011, p. 103–112. DOI : 10.1145/2034691.2034717. URL : <https://hal.inria.fr/inria-00629503>.
- [55] Stéphane WEISS, Pascal URSO et Pascal MOLLI. « Wooki : a P2P Wiki-based Collaborative Writing Tool ». In : t. 4831. Déc. 2007. ISBN : 978-3-540-76992-7. DOI : 10.1007/978-3-540-76993-4\_42.
- [56] Ben SHNEIDERMAN. « Response Time and Display Rate in Human Performance with Computers ». In : *ACM Comput. Surv.* 16.3 (sept. 1984), p. 265–285. ISSN : 0360-0300. DOI : 10.1145/2514.2517. URL : <https://doi.org/10.1145/2514.2517>.
- [57] Caroline JAY, Mashhuda GLENCROSS et Roger HUBBOLD. « Modeling the Effects of Delayed Haptic and Visual Feedback in a Collaborative Virtual Environment ». In : *ACM Trans. Comput.-Hum. Interact.* 14.2 (août 2007), 8-es. ISSN : 1073-0516. DOI : 10.1145/1275511.1275514. URL : <https://doi.org/10.1145/1275511.1275514>.
- [58] Hagit ATTIYA, Sebastian BURCKHARDT, Alexey GOTSMAN, Adam MORRISON, Hongseok YANG et Marek ZAWIRSKI. « Specification and Complexity of Collaborative Text Editing ». In : *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*. PODC '16. Chicago, Illinois, USA : Association for Computing Machinery, 2016, p. 259–268. ISBN : 9781450339643. DOI : 10.1145/2933057.2933090. URL : <https://doi.org/10.1145/2933057.2933090>.

- 
- [59] Hagit ATTIYA, Sebastian BURCKHARDT, Alexey GOTSMAN, Adam MORRISON, Hongseok YANG et Marek ZAWIRSKI. « Specification and space complexity of collaborative text editing ». In : *Theoretical Computer Science* 855 (2021), p. 141–160. ISSN : 0304-3975. DOI : <https://doi.org/10.1016/j.tcs.2020.11.046>. URL : <http://www.sciencedirect.com/science/article/pii/S0304397520306952>.
- [60] AUTOMERGE. *Automerge : data structures for building collaborative applications in Javascript*. URL : <https://github.com/automerge/automerge>.
- [61] Loïck BRIOT, Pascal URSO et Marc SHAPIRO. « High Responsiveness for Group Editing CRDTs ». In : *ACM International Conference on Supporting Group Work*. Sanibel Island, FL, United States, nov. 2016. DOI : 10.1145/2957276.2957300. URL : <https://hal.inria.fr/hal-01343941>.
- [62] Weihai YU. « A String-Wise CRDT for Group Editing ». In : *Proceedings of the 17th ACM International Conference on Supporting Group Work*. GROUP '12. Sanibel Island, Florida, USA : Association for Computing Machinery, 2012, p. 141–144. ISBN : 9781450314862. DOI : 10.1145/2389176.2389198. URL : <https://doi.org/10.1145/2389176.2389198>.
- [63] Martin KLEPPMANN, Victor B. F. GOMES, Dominic P. MULLIGAN et Alastair R. BERESFORD. « Interleaving Anomalies in Collaborative Text Editors ». In : *Proceedings of the 6th Workshop on Principles and Practice of Consistency for Distributed Data*. PaPoC '19. Dresden, Germany : Association for Computing Machinery, 2019. ISBN : 9781450362764. DOI : 10.1145/3301419.3323972. URL : <https://doi.org/10.1145/3301419.3323972>.
- [64] Matthew WEIDNER. *There Are No Doubly Non-Interleaving List CRDTs*. URL : [https://mattweidner.com/assets/pdf/List\\_CRDT\\_Non\\_Interleaving.pdf](https://mattweidner.com/assets/pdf/List_CRDT_Non_Interleaving.pdf).
- [65] Stéphane WEISS, Pascal URSO et Pascal MOLLI. « Logoot-Undo : Distributed Collaborative Editing System on P2P Networks ». In : *IEEE Transactions on Parallel and Distributed Systems* 21.8 (août 2010), p. 1162–1174. DOI : 10.1109/TPDS.2009.173. URL : <https://hal.archives-ouvertes.fr/hal-00450416>.
- [66] Claudia-Lavinia IGNAT, Gérald OSTER, Meagan NEWMAN, Valerie SHALIN et François CHAROY. « Studying the Effect of Delay on Group Performance in Collaborative Editing ». In : *Proceedings of 11th International Conference on Cooperative Design, Visualization, and Engineering, CDVE 2014, Springer 2014 Lecture Notes in Computer Science*. Proceedings of 11th International Conference on Cooperative Design, Visualization, and Engineering, CDVE 2014. Seattle, WA, United States, sept. 2014, p. 191–198. DOI : 10.1007/978-3-319-10831-5\_29. URL : <https://hal.archives-ouvertes.fr/hal-01088815>.
- [67] Claudia-Lavinia IGNAT, Gérald OSTER, Olivia FOX, François CHAROY et Valerie SHALIN. « How Do User Groups Cope with Delay in Real-Time Collaborative Note Taking ». In : *European Conference on Computer Supported Cooperative Work 2015*. Sous la dir. de Nina BOULUS-RODJE, Gunnar ELLINGSEN, Tone BRATTETEIG, Margunn AANESTAD et Pernille BJORN. Proceedings of the 14th European Conference on Computer Supported Cooperative Work. Oslo, Norway : Springer International

- Publishing, sept. 2015, p. 223–242. DOI : 10.1007/978-3-319-20499-4\_12. URL : <https://hal.inria.fr/hal-01238831>.
- [68] Brice NÉDELEC, Pascal MOLLI, Achour MOSTÉFAOUI et Emmanuel DESMONTILS. « LSEQ : an adaptive structure for sequences in distributed collaborative editing ». In : *Proceedings of the 2013 ACM Symposium on Document Engineering*. DocEng 2013. Sept. 2013, p. 37–46. DOI : 10.1145/2494266.2494278.
- [69] Brice NÉDELEC, Pascal MOLLI et Achour MOSTÉFAOUI. « A scalable sequence encoding for collaborative editing ». In : *Concurrency and Computation : Practice and Experience* (), e4108. DOI : 10.1002/cpe.4108. eprint : <https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpe.4108>. URL : <https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.4108>.
- [70] Daniel ABADI. « Consistency Tradeoffs in Modern Distributed Database System Design : CAP is Only Part of the Story ». In : *Computer* 45.2 (2012), p. 37–42. DOI : 10.1109/MC.2012.33.
- [71] Matthieu NICOLAS. « Efficient renaming in CRDTs ». In : *Middleware 2018 - 19th ACM/IFIP International Middleware Conference (Doctoral Symposium)*. Rennes, France, déc. 2018. URL : <https://hal.inria.fr/hal-01932552>.
- [72] Matthieu NICOLAS, Gérald OSTER et Olivier PERRIN. « Efficient Renaming in Sequence CRDTs ». In : *7th Workshop on Principles and Practice of Consistency for Distributed Data (PaPoC'20)*. Heraklion, Greece, avr. 2020. URL : <https://hal.inria.fr/hal-02526724>.
- [73] Haifeng SHEN et Chengzheng SUN. « A log compression algorithm for operation-based version control systems ». In : *Proceedings 26th Annual International Computer Software and Applications*. 2002, p. 867–872. DOI : 10.1109/CMPSAC.2002.1045115.
- [74] Claudia-Lavinia IGNAT. « Maintaining consistency in collaboration over hierarchical documents ». Thèse de doct. ETH Zurich, 2006.
- [75] Sylvie NOËL et Jean-Marc ROBERT. « Empirical study on collaborative writing : What do co-authors do, use, and like ? ». In : *Computer Supported Cooperative Work (CSCW)* 13.1 (2004), p. 63–89.
- [76] Jim GILES. « Special Report Internet encyclopaedias go head to head ». In : *nature* 438.15 (2005), p. 900–901.
- [77] GOOGLE. *Google Docs*. URL : <https://docs.google.com/>.
- [78] ETHERPAD. *Etherpad*. URL : <https://etherpad.org/>.
- [79] Quang-Vinh DANG et Claudia-Lavinia IGNAT. « Performance of real-time collaborative editors at large scale : User perspective ». In : *Internet of People Workshop, 2016 IFIP Networking Conference*. Proceedings of 2016 IFIP Networking Conference, Networking 2016 and Workshops. Vienna, Austria, mai 2016, p. 548–553. DOI : 10.1109/IFIPNetworking.2016.7497258. URL : <https://hal.inria.fr/hal-01351229>.

- 
- [80] Barton GELLMAN et Laura POITRAS. *U.S., British intelligence mining data from nine U.S. Internet companies in broad secret program*. URL : [https://www.washingtonpost.com/investigations/us-intelligence-mining-data-from-nine-us-internet-companies-in-broad-secret-program/2013/06/06/3a0c0da8-cebf-11e2-8845-d970ccb04497\\_story.html](https://www.washingtonpost.com/investigations/us-intelligence-mining-data-from-nine-us-internet-companies-in-broad-secret-program/2013/06/06/3a0c0da8-cebf-11e2-8845-d970ccb04497_story.html).
- [81] Glen GREENWALD et Ewen MACASKILL. *NSA Prism program taps in to user data of Apple, Google and others*. URL : <https://www.theguardian.com/world/2013/jun/06/us-tech-giants-nsa-data>.
- [82] Brice NÉDELEC, Pascal MOLLI et Achour MOSTEFAOUI. « CRATE : Writing Stories Together with our Browsers ». In : *25th International World Wide Web Conference. WWW 2016*. ACM, avr. 2016, p. 231–234. DOI : 10.1145/2872518.2890539.
- [83] Jim PICK. *PeerPad*. URL : <https://peerpad.net/>.
- [84] Jim PICK. *Graf, Nikolaus*. URL : <https://www.serenity.re/en/notes>.
- [85] Martin KLEPPMANN, Adam WIGGINS, Peter van HARDENBERG et Mark MCGRANAGHAN. « Local-First Software : You Own Your Data, in Spite of the Cloud ». In : *Proceedings of the 2019 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software*. Onward! 2019. Athens, Greece : Association for Computing Machinery, 2019, p. 154–178. ISBN : 9781450369954. DOI : 10.1145/3359591.3359737. URL : <https://doi.org/10.1145/3359591.3359737>.
- [86] Peter van HARDENBERG et Martin KLEPPMANN. « PushPin : Towards Production-Quality Peer-to-Peer Collaboration ». In : *7th Workshop on Principles and Practice of Consistency for Distributed Data. PaPoC 2020*. ACM, avr. 2020. DOI : 10.1145/3380787.3393683.
- [87] Madhavan MUKUND, Gautham SHENOY et SP SURESH. « Optimized or-sets without ordering constraints ». In : *International Conference on Distributed Computing and Networking*. Springer. 2014, p. 227–241.
- [88] Victorien ELVINGER, Gérard OSTER et Francois CHAROY. « Prunable Authenticated Log and Authenticable Snapshot in Distributed Collaborative Systems ». In : *2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*. 2018, p. 156–165. DOI : 10.1109/CIC.2018.00031.
- [89] OPENRELAY. *OpenRelay*. URL : <https://openrelay.xyz/>.
- [90] Protocol LABS. *IPFS*. URL : <https://ipfs.io/>.
- [91] Quang Vinh DANG et Claudia-Lavinia IGNAT. « Quality Assessment of Wikipedia Articles : A Deep Learning Approach by Quang Vinh Dang and Claudia-Lavinia Ignat with Martin Vesely as Coordinator ». In : *SIGWEB Newsl.* Autumn (nov. 2016). ISSN : 1931-1745. DOI : 10.1145/2996442.2996447. URL : <https://doi.org/10.1145/2996442.2996447>.

- [92] Leslie LAMPORT, Robert SHOSTAK et Marshall PEASE. « The Byzantine Generals Problem ». In : *Concurrency : The Works of Leslie Lamport*. New York, NY, USA : Association for Computing Machinery, 2019, p. 203–226. ISBN : 9781450372701. URL : <https://doi.org/10.1145/3335772.3335936>.
- [93] Jim BAUWENS et Elisa Gonzalez BOIX. « Flec : A Versatile Programming Framework for Eventually Consistent Systems ». In : *Proceedings of the 7th Workshop on Principles and Practice of Consistency for Distributed Data*. PaPoC '20. Heraklion, Greece : Association for Computing Machinery, 2020. ISBN : 9781450375245. DOI : 10.1145/3380787.3393685. URL : <https://doi.org/10.1145/3380787.3393685>.