Réconciliation par consensus des mises à jour des répliques partielles d'un document structuré

Maurice TCHOUPÉ TCHENDJI *, Milliam M. ZEKENG NDADJI*

* Département de Maths-Informatique
Faculté des Sciences, Université de Dschang
BP 67, Dschang-Cameroun
ttchoupe@yahoo.fr
ndadjimaxime@yahoo.fr

RÉSUMÉ. Dans un workf ow d'édition coopérative asynchrone d'un document structuré, chacun des co-auteurs reçoit dans les différentes phases du processus d'édition une copie du document pour y insérer sa contribution. Pour des raisons de conf dentialité, cette copie peut n'être qu'une réplique partielle ne contenant que les parties du document (global) qui sont d'un intérêt avéré pour le co-auteur considéré. Remarquons que certaines parties peuvent être d'un intérêt avéré pour plus d'un co-auteur ; elles seront par conséquent accessibles en concurrence. Quand vient le moment de la synchronisation (à la f n d'une phase du processus d'édition par exemple), il faut fusionner toutes les contributions de tous les co-auteurs en un document unique. Du fait de l'asynchronisme de l'édition et de l'existence potentielle des parties offrant des accès concurrents, des conf its peuvent surgir et rendre les répliques partielles non fusionnables dans leur entièreté: elles sont incohérentes ou en conf its. Nous proposons dans ce papier une approche de fusion dite par consensus de telles répliques partielles à l'aide des automates d'arbre. Plus précisément, à partir des mises à jour des répliques partielles, nous construisons un automate d'arbre dit du consensus qui accepte exactement les documents du consensus. Ces documents sont les préf xes maximums ne contenant pas de conf it des répliques partielles fusionnées.

ABSTRACT. In an asynchronous cooperative editing workf ow of a structured document, each of the co-authors receive in the different phases of the editing process a copy of the document to insert its contribution. For conf dentiality reasons, this copy may be only a partial replica containing only parts of the (global) document which are of demonstrated interest for the considered co-author. Note that some parts may be a demonstrated interest over a co-author; they will therefore be accessible concurrently. When it's synchronization time (for eg. at the end of a phase of the process), we want to merge all contributions of all authors in a single document. Due to the asynchronism of edition and to the potential existence of the document parts offering concurrent access, conf icts may arise and make partial replicas unmergeable in their entirety: they are inconsistent or in conf ict. We propose in this paper a merging approach said by consensus of such partial replicas using tree automata. Specif cally, from the partial replicas update, we build a tree automaton that accepts exactly the consensus documents. These documents are the maximum pref xes containing no conf ict of partial replicas merged.

MOTS-CLÉS : Documents Structurés, Workf ow d'Edition Coopérative, Fusion des Répliques Partielles, Conf its, Consensus, Automates d'Arbre, Produit d'automates, Evaluation Paresseuse.

KEYWORDS : Structured Documents, Worf ow of Cooperative Edition, Merging Partial Replicas, Conf ict, Consensus, Tree Automata, Automata Product, Lazy Evaluation.

1. Introduction

Une proportion importante des documents manipulés et/ou échangés par les applications présente une structure régulière définie par un modèle grammatical (DTD-*Document Type Definition*-, schéma, ...) : on les appelle des *documents structurés*. La puissance toujours croissante des réseaux de communication en terme de débit et de sûreté ainsi que le souci d'efficacité a révolutionné la façon d'éditer de tels documents : au modèle classique d'un auteur éditant en local et de façon autonome son document, s'est adjoint l'édition coopérative (asynchrone) dans laquelle, plusieurs auteurs situés sur des sites géographiquement éloignés se coordonnent pour éditer de façon asynchrone un même document structuré (fig. 1). Dans de tels processus d'édition coopérative asynchrone, les phases d'éditions désynchronisées dans lesquelles chaque co-auteur édite sur son site sa copie du document, alternent avec les phases de synchronisation-redistributions dans lesquelles les différentes contributions (répliques locales) sont fusionnées en un unique document, qui est par la suite redistribué aux différents co-auteurs pour la poursuite de l'édition.

Pour des raisons de confidentialité, il est parfois souhaitable qu'un co-auteur n'ait accès qu'à certaines informations c-à-d, à des parties du document appartenant à des types donnés (des *sortes*) du modèle du document. Ainsi, la réplique t_i éditée par le co-auteur c_i depuis le site i n'est donc qu'une *réplique partielle* du document (global[1]) t ; elle est obtenue via une *opération de projection* qui supprime convenablement du document global t les parties qui ne sont pas accessibles au co-auteur considéré. Nous appelons *vue* d'un co-auteur, l'ensemble des *sortes* qui lui sont accessibles [1].

Quand les éditions locales asynchrones se font sur des répliques partielles, on peut supposer que chaque co-auteur possède sur son site un modèle local de document le guidant dans son édition à l'aide duquel on peut garantir que toute mise à jour d'une réplique partielle valide par rapport à ce modèle local est cohérente vis-à-vis du modèle global du document[2]. Ainsi, du fait de l'asynchronisme de l'édition, les seules incohérences qu'on puisse avoir quand arrive le moment de la synchronisation sont celles issues de l'édition concurrente du même noeud (du point de vue du document global) par plusieurs co-auteurs : on dit que les répliques partielles concernées sont en conflits. Ce papier propose une approche de détection et de résolution de tels conflits par *consensus* pendant la phase de synchronisation-redistribution, en se servant d'un automate d'arbre dit du consensus, pour représenter l'ensemble des documents qui sont les consensus d'éditions concurrentes réalisées sur les répliques partielles.

Un document structuré t est représentable intentionnellement par un arbre possèdant éventuellement des bourgeons[3] [1]. Intuitivement, synchroniser ou fusionner consensuellement les m-à-j $t_1, \ldots t_n$ de n répliques partielles d'un document t, consiste à trouver un document t_c conforme au modèle global, intégrant tous les noeuds des t_i non en conflits et dans lequel, tous les noeuds en conflits sont remplacés par des bourgeons. L'algorithme de fusion consensuelle présenté dans ce papier est une adaptation de celui de fusion présenté dans [1] qui ne gère pas les conflits. Techniquement, la démarche permettant d'obtenir les documents faisant partie du consensus est la suivante : (1) pour chaque m-à-j t_i^{maj} d'une réplique partielle t_i, nous associons un automate d'arbre $\mathcal{A}^{(i)}$ reconnaissant les arbres (conforme au modèle global) dont t_i^{maj} est une projection. (2) L'automate consensuel $\mathcal{A}_{(sc)}$ engendrant les documents du consensus est obtenu en effectuant

1. Nous désignons par *document global* ou tout simplement *document* quand il n'y a pas d'ambiguïté, le document comprenant toutes les parties.
2. Intuitivement, un modèle local de document sera dit cohérent vis-à-vis du modèle global si tout document partiel t_i qui lui est conforme est la réplique partielle d'au moins un document (global) t conforme au modèle global.
3. Un *bourgeon* est un noeud feuille d'un arbre indiquant qu'une édition doit être effectuée à ce niveau dans l'arbre. Editer un bourgeon revient à le remplacer par un sous arbre en se servant des productions de la grammaire du document.

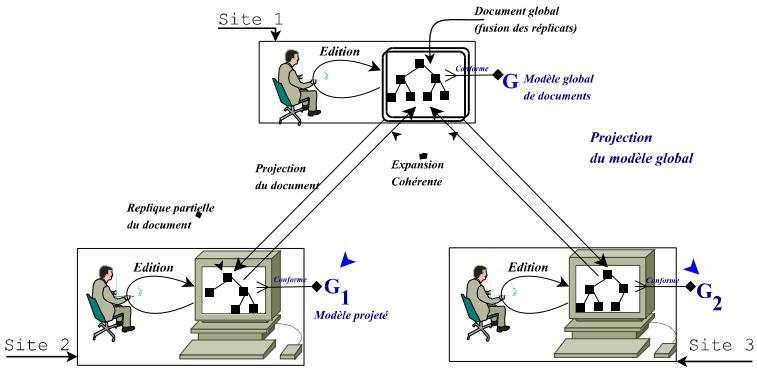

Figure 1. *L'édition coopérative désynchronisée ; site1 : édition+fusion du document (global) conformément au modèle (global) G de documents ; sites 2 et 3 : éditions des répliques partielles conformément aux modèles projetés de documents G_1 et G_2 obtenus à partir du modèle global G.*

un *produit synchrone* des automates $\mathcal{A}^{(i)}$ au moyen d'un opérateur commutatif et associatif noté $\otimes^{\Omega}$: $\mathcal{A}_{(sc)} = \otimes_i^{\Omega} \mathcal{A}^{(i)}$ que nous définissons. Il suffit alors de générer l'ensemble des arbres (ou ceux les plus représentatifs) acceptés par l'automate $\mathcal{A}_{(sc)}$, pour avoir les documents du consensus.

Dans ce qui suit, après la présentation (sec. 2) de quelques concepts et définitions relatifs à l'édition coopérative et aux automates d'arbre, nous exposons (sec. 3) le processus de construction de l'opérateur $\otimes_i^{\Omega}$. La section 4 est consacrée à la conclusion. En annexe, nous déroulons complètement l'exemple introduit dans la section 3 en mettant en exergue les concepts manipulés dans ce papier.

2. Edition coopérative structurée et notion de réplique partielle

2.1. Document structuré, édition et conformité

Il est usuel de représenter la structure abstraite d'un document structuré par un arbre et son modèle par une grammaire algébrique abstraite ; un document structuré valide est alors un arbre de dérivation pour cette grammaire. Une grammaire algébrique définit la structure de ses instances (les documents qui lui sont conformes) au moyen des productions. Une production, généralement notée $p : X_0 \to X_1 \ldots X_n$ est assimilable dans ce contexte à une règle de structuration présentant comment le symbole X_0 situé en partie gauche de la production se décompose en une séquence d'autre symboles $X_1 \ldots X_n$ situés dans sa partie droite. Plus formellement,

Définition 1 *Une **grammaire algébrique abstraite** est la donnée $\mathbb{G} = (\mathcal{S}, \mathcal{P}, A)$ d'un ensemble fini $\mathcal{S}$ de **symboles grammaticaux** ou **sortes** qui correspondent aux différentes **catégories syntaxiques** en jeu, d'un symbole grammatical $A \in \mathcal{S}$ particulier, appelé **axiome**, et d'un ensemble fini $\mathcal{P} \subseteq \mathcal{S} \times \mathcal{S}^*$ de **productions**. Une production $P = \left(X_{P(0)}, X_{P(1)} \cdots X_{P(n)}\right)$ est notée $P : X_{P(0)} \to X_{P(1)} \cdots X_{P(n)}$ et $|P|$ désigne la longueur de la partie droite de P.*

Pour certains traitements sur les arbres (documents) il est nécessaire de désigner précisément un noeud particulier. Plusieurs techniques d'indexation existent parmi lesquelles celle dite de numérotation dynamique par niveau [6] basée sur des identificateurs à longueur variable inspirés

de la classification décimale de *Dewey*. Suivant ce système d'indexation, on peut définir un arbre comme suit :

Définition 2 *Un* ***arbre*** *dont les noeuds sont étiquetés dans un alphabet $\mathcal{S}$ est une fonction t : $\mathbb{N}^* \to \mathbf{S}$ dont le domaine $Dom(t) \subseteq \mathbb{N}^*$ est un ensemble clos par préfixe tel que pour tout $u \in Dom(t)$ l'ensemble $\{i \in \mathbb{N} \mid u \cdot i \in Dom(t)\}$ est un intervalle d'entiers $[1, \cdots, n] \cap \mathbb{N}$ non vide ($\varepsilon \in Dom(t)(racine)$) ; l'entier n est l'**arité** du noeud d'adresse u. $t(w)$ est la valeur (étiquette) du noeud de t d'adresse w. Si $t_1, \cdots, t_n$ sont des arbres et $a \in \mathbf{S}$, on note $t = a[t_1, \ldots, t_n]$ l'arbre t de domaine $Dom(t) = \{\varepsilon\} \cup \{i \cdot u \mid 1 \leq i \leq n,\ u \in Dom(t_i)\}$ avec $t(\varepsilon) = a$ et $t(i \cdot u) = t_i(u)$.*

Soient t un document et $\mathbb{G} = (\mathcal{S}, \mathcal{P}, A)$ une grammaire. t est un arbre de dérivation pour $\mathbb{G}$ si sa racine est étiquetée par l'axiome A de $\mathbb{G}$, et si pour tout noeud interne n_0 étiqueté par le sorte X_0, et dont les fils $n_1, \ldots n_n$, sont étiquetés respectivement par les sortes $X_1, \ldots X_n$, il existe une production $P \in \mathcal{P}$ telle que, $P : X_0 \to X_1 \cdots X_n$ et $|P| = n$. On dit aussi dans ce cas que t appartient au langage engendré par $\mathbb{G}$ à partir du symbole A et on note $t \in \mathscr{L}(\mathbb{G},\ A)$ ou encore $t \therefore \mathbb{G}$.

Un document structuré en cours d'édition est représenté par un arbre contenant des *bourgeons* (ou *noeuds ouverts*) qui indiquent dans un arbre, les seuls lieux où des éditions (mises à jour) sont possibles[4]. Les bourgeons sont typés ; un *bourgeon de sorte X* est un noeud feuille étiqueté X_ω : il ne peut être édité (étendu en un sous-arbre) qu'en utilisant une *X-production* (production ayant X en membre gauche). Ainsi donc, un document structuré en cours d'édition et ayant la grammaire $\mathbb{G} = (\mathcal{S}, \mathcal{P}, A)$ pour modèle est un arbre de dérivation pour la grammaire étendue $\mathbb{G}_\Omega = (\mathcal{S}, \mathcal{P} \cup \mathcal{S}_\Omega, A)$ obtenue de $\mathbb{G}$ en ajoutant à l'ensemble $\mathcal{P}$ des productions et pour tout sorte $X \in \mathcal{S}$, une nouvelle ε-production $X_\Omega : X \to \varepsilon$; $\mathcal{S}_\Omega = \{X_\Omega : X \to \varepsilon,\ X \in \mathcal{S}\}$.

Pour décider de la conformité d'un document complètement édité ou en cours d'édition, on peut utiliser comme outil formel les automates d'arbre. Comme nous le verrons ci-dessous, il est facile de construire un automate d'arbre (un reconnaisseur/générateur) à partir d'une grammaire donnée. En effet, quand on regarde les productions d'une grammaire, on peut remarquer que chaque sorte est associé à un ensemble de productions. On peut donc de ce point de vue considérer une grammaire comme une application $gram : symb \to [(prod,\ [symb])]$qui associe à chaque sorte une liste de couples formés d'un nom de production et de la liste des sortes en partie droite de cette production. Une telle observation suggère qu'une grammaire peut être interprétée comme un automate d'arbre (descendant) utilisable pour la reconnaissance ou pour la génération de ses instances.

Définition 3 *Un* ***automate d'arbre*** *(descendant) défini sur Σ est la donnée $\mathcal{A} = (\Sigma, Q, R, q_0)$ d'un ensemble Σ de symboles ; ses éléments sont les étiquettes des noeuds des arbres à reconnaître, d' un ensemble Q d'états, d'un état particulier $q_0 \in Q$ appelé état initial, et d'un ensemble fini $R \subseteq Q \times \Sigma \times Q^*$ de transitions.*

– Un élément de R est noté $q \to (\sigma, [q_1, \cdots, q_n])$: intuitivement, il s'agit de la liste des états $[q_1, \cdots, q_n]$ accessibles à partir d'un état q donné en franchissant une transition étiquetée σ.

– Si $q \to \left(\sigma_1, [q_1^1, \cdots, q_{n_1}^1]\right), \cdots, q \to \left(\sigma_k, [q_1^k, \cdots, q_{n_k}^k]\right)$ désigne l'ensemble des transitions associées à l'état q, on note next $q = [\left(\sigma_1, [q_1^1, \cdots, q_{n_1}^1]\right), \cdots, \left(\sigma_k, [q_1^k, \cdots, q_{n_k}^k]\right)]$ la liste formée des couples $\left(\sigma_i, [q_1^i, \cdots, q_{n_i}^i]\right)$. Dans le cas où q est un état terminal (c-à-d. aucune transition n'est franchissable depuis l'état q), next $q = [\,]$.

4. Nous nous intéressons dans ce papier qu'à l'*édition positive* basée sur une réplication optimiste [5] partielle des documents édités ; en effet, les documents édités ne font que croitre : il n'y a pas d'effacement possible dès qu'une synchronisation a été effectuée.

On peut interpréter une grammaire $\mathbb{G} = (\mathcal{S}, \mathcal{P}, A)$ comme un automate d'arbre (descendant) [2]$\mathcal{A} = (\Sigma, Q, R, q_0)$ en considérant que : (1) $\Sigma = \mathcal{S}$ est le type des étiquettes des noeuds de l'arbre à reconnaitre. (2) $Q = \mathcal{S}$ est le type des états (tous considérés comme finals) et, (3) $q \rightarrow (\sigma, [q_1, \cdots, q_n])$ est une transition de l'automate lorsque la paire $(\sigma, [q_1, \cdots, q_n])$ apparait dans la liste $(gram\ \ q)$[5]. On notera $\mathcal{A}_{\mathbb{G}}$ l'automate d'arbre dérivé à partir de $\mathbb{G}$.

Pour reconnaître un arbre à l'aide d'un automate d'arbre à partir d'un état initial, il suffit de : (1) Associer l'état initial à la racine de l'arbre. (2) Si un noeud étiqueté A est associé à l'état q, alors A doit être égal à q. Si ce noeud possède n successeurs non encore associés à des états, et que la transition $q \rightarrow (A, [q_1, \cdots, q_n])$ est une transition de l'automate, alors on associe les états de q_1 jusqu'à q_n à chacun de ces n successeurs. (3) L'arbre est reconnu si on a ainsi réussi à associer un état à chacun des noeuds de l'arbre. On note $\mathcal{L}(\mathcal{A}, q)$ (langage d'arbre) l'ensemble des arbres acceptés par l'automate $\mathcal{A}$ à partir de l'état initial q.

Comme pour les automates sur les mots, on peut définir un produit synchrone sur les automates d'arbre pour obtenir l'automate reconnaissant l'intersection, l'union, ...des langages réguliers d'arbre [2]. Nous introduisons ci-dessous la définition du produit synchrone de k automates d'arbre dont une adaptation sera utilisée dans la section suivante pour la dérivation de l'automate du consensus.

Définition 4 ***Produit synchrone de*** k ***automates :*** *Soient* $\mathcal{A}_1 = (\Sigma, Q^{(1)}, R^{(1)}, q_0^{(1)}), \ldots, \mathcal{A}_k = (\Sigma, Q^{(k)}, R^{(k)}, q_0^{(k)})$ *k automates d'arbre. Le produit synchrone de ces k automates* $\mathcal{A}_1 \otimes \cdots \otimes \mathcal{A}_k$ *noté* $\otimes_i \mathcal{A}^{(i)}$*, est l'automate* $\mathcal{A}_{(sc)} = (\Sigma, Q, R, q_0)$ *défini comme suit :* ***(a)*** *Ses états sont les vecteurs d'états :* $Q = Q^{(1)} \times \cdots \times Q^{(k)}$ *;* ***(b)*** *Son état initial est formé par le vecteur des états initiaux des différents automates :* $q_0 = (q_0^{(1)}, \cdots, q_0^{(k)})$ *;* ***(c)*** *Ses transitions sont données par :* $(q^{(1)}, \ldots, q^{(k)}) \xrightarrow{a} \left((q_1^{(1)}, \ldots, q_1^{(k)}), \ldots, (q_n^{(1)}, \ldots, q_n^{(k)})\right) \Leftrightarrow \left(q^{(i)} \xrightarrow{a} (q_1^{(i)}, \ldots, q_n^{(i)}) \quad \forall i,\ 1 \leq i \leq k\right)$

2.2. Notions de vue, de projection, de projection inverse et de fusion

2.2.1. Vue, projection associée et fusion

L'arbre de dérivation donnant la représentation (globale) d'un document structuré édité de façon coopérative, rend visible l'ensemble des symboles grammaticaux de la grammaire ayant participé à sa construction. Comme déjà mentionné à la section 1, pour des raisons de confidentialité (degré d'accréditation), un co-auteur manipulant un tel document n'aura pas accès nécessairement à l'ensemble de tous ces symboles grammaticaux ; seul un sous-ensemble d'entre eux peut être jugé pertinent pour lui : c'est sa *vue*. Une vue $\mathcal{V}$ est donc un sous-ensemble de symboles grammaticaux ($\mathcal{V} \subseteq \mathcal{S}$).

Une réplique partielle de t suivant la vue $\mathcal{V}$, est une copie partielle de t obtenue en supprimant dans t tous les noeuds étiquétés par des symboles n'appartenant pas à $\mathcal{V}$. La figure 2 présente un document t (au centre) ainsi que deux répliques partielles $t_{\mathcal{V}_1}$ (à gauche) et $t_{\mathcal{V}_2}$ (à droite) obtenues respectivement par projections à partir des vues $\mathcal{V}_1 = \{A, B\}$ et $\mathcal{V}_2 = \{A, C\}$. Pratiquement, une réplique partielle est obtenue via une opération de *projection* notée π. On note donc $\pi_{\mathcal{V}}(t) = t_{\mathcal{V}}$ le fait que $t_{\mathcal{V}}$ est une réplique partielle obtenue par projection de t suivant la vue $\mathcal{V}$.

Notons $t_{\mathcal{V}_i} \leq t_{\mathcal{V}_i}^{maj}$ le fait que le document $t_{\mathcal{V}_i}^{maj}$ soit une m-à-j du document $t_{\mathcal{V}_i}$, c-à-d que $t_{\mathcal{V}_i}^{maj}$ est obtenu de $t_{\mathcal{V}_i}$ en remplaçant certains de ses bourgeons par des arbres. Dans un processus d'édition coopérative asynchrone, il existe des points de synchronisations[6] au cours desquels

5. Rappel : $gram$ est l'application obtenue par abstraction de $\mathbb{G}$ et a pour type : $gram : symb \rightarrow [(prod,\ [symb])]$.
6. Un point de synchronisation peut être déf ni statiquement ou déclenché par un co-auteur dès que certaines propriétés sont satisfaites.

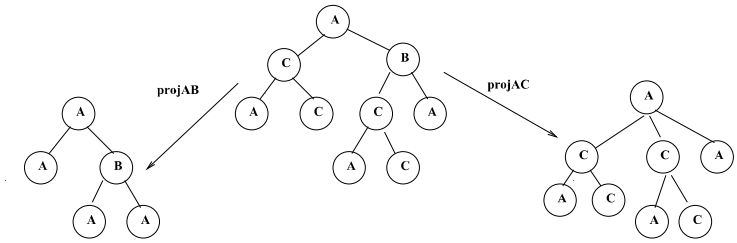

Figure 2. *Exemple de projections effectuées sur un document.*

on essaye de fusionner toutes les contributions $t^{maj}_{\mathcal{V}_i}$ des différents co-auteurs pour obtenir un document global unique t_f[7]. Un algorithme de fusion n'intégrant pas la gestion des conflits et s'appuyant sur une solution au problème de la *projection inverse* est donné dans [1].

2.2.2. Réplique partielle et projection inverse (expansion)

La *projection inverse* d'une réplique partielle mise à jour $t^{maj}_{\mathcal{V}_i}$ relativement à une grammaire $\mathbb{G} = (\mathcal{S}, \mathcal{P}, A)$ donnée, c'est l'ensemble $T^{maj}_{i\mathcal{S}}$ des documents conformes à $\mathbb{G}$ qui admettent $t^{maj}_{\mathcal{V}_i}$ comme réplique partielle suivant la vue $\mathcal{V}_i$: $T^{maj}_{i\mathcal{S}} = \left\{ t^{maj}_{i\mathcal{S}} \therefore \mathbb{G} \mid \pi_{\mathcal{V}_i}(t^{maj}_{i\mathcal{S}}) = t^{maj}_{\mathcal{V}_i} \right\}$.

Une solution à ce problème utilisant les automates d'arbre est donnée dans [1]. Dans celle ci, on se sert des productions de la grammaire $\mathbb{G}$, pour associer à une vue $\mathcal{V}_i \subseteq \mathcal{S}$ un automate d'arbre $\mathcal{A}^{(i)}$ tel que les arbres qu'il reconnait à partir d'un état initial construit à partir de $t^{maj}_{\mathcal{V}_i}$ sont exactement ceux qui ont cette réplique partielle comme projection suivant la vue $\mathcal{V}_i$: $T^{maj}_{i\mathcal{S}} = \mathcal{L}(\mathcal{A}^{(i)}, q_{t_{\mathcal{V}_i}})$. Le lecteur interessé peut consulter [1] pour une description plus détaillée du processus d'association d'un automate d'arbre à une vue et l'annexe pour une illustration.

3. Réconciliation par consensus

3.1. Problématique et principe de la solution de la réconciliation par consensus

Dans un processus d'édition coopérative asynchrone de plusieurs répliques partielles d'un document, quand on atteint un point de synchronisation, on peut se retrouver avec des répliques non fusionnables dans leur entièreté car, contenant des m-à-j non compatibles[8] : il faut les réconcilier. On peut le faire en remettant en cause (annulation) certaines actions d'édition locales afin de lever les conflits et aboutir à une version globale cohérente dite de consensus.

Les études portant sur la réconciliation des versions d'un document reposent sur des heuristiques [4] dans la mesure où il n'y a pas de solution générale à ce problème. Dans notre cas, étant donné que toutes les actions d'édition sont réversibles[9] et qu'il est facile de localiser les conflits lors de la tentative de fusion des répliques partielles (voir section 3.2), nous disposons d'une méthode canonique pour éliminer les conflits : nous remplaçons lors de la fusion tout noeud (du document global) dont les répliques sont en conflits par un bourgeon. On élague donc au niveau

7. Il peut arriver que l'édition doit être poursuivie après la fusion (c'est le cas s'il existe encore des bourgeons dans le document fusionné) : on doit redistribuer à chacun des n co-auteurs une réplique (partielle) $t_{\mathcal{V}_i}$ de t_f telle que $t_{\mathcal{V}_i} = \pi_{\mathcal{V}_i}(t_f)$ pour la poursuite du processus d'édition.

8. C'est notamment le cas s'il existe au moins un noeud du document global accessible par plus d'un co-auteur et édité par au moins deux d'entre eux en utilisant des productions différentes.

9. Rappel : Les actions d'éditions effectuées sur une réplique partielle peuvent être annulées tant qu'elles n'ont pas encore été intégrées dans le document gobal.

des noeuds où un conflit apparaît, en remplaçant le sous arbre correspondant par un bourgeon du type approprié, indiquant que cette partie du document n'est pas encore éditée : les documents obtenus sont appelés les consensus. Ce sont les préfixes maximaux sans conflits de la fusion des documents issus des différentes expansions des diverses répliques partielles m-à-j.

Le problème de la fusion consensuelle de n répliques patielles dont le modèle global est donné par une grammaire $\mathbb{G} = (\mathcal{S}, \mathcal{P}, A)$ peut donc s'énoncer comme suit :

Problème de la fusion consensuelle : Etant donné n vues $(\mathcal{V}_i)_{1\leq i\leq n}$ et n répliques partielles $(t^{maj}_{\mathcal{V}_i})_{1\leq i\leq n}$, fusionner consensuellement la famille $(t^{maj}_{\mathcal{V}_i})_{1\leq i\leq n}$ consiste à rechercher les plus grands documents $t^{maj}_{\mathcal{S}} \therefore \mathbb{G}$ satisfaisant : $\forall i \in 1,\ldots,n \ \left(\exists t_i \therefore \mathbb{G},\ t_i \leq t^{maj}_{\mathcal{S}},\ \pi_{\mathcal{V}_i}(t_i) \leq t^{maj}_{\mathcal{V}_i}\right)$

La solution que nous proposons à ce problème découle d'une instrumentalisation de celle proposée pour l'expansion (section 2.2.2). En effet, nous nous servons d'un opérateur associatif et commutatif noté $\otimes^{\Omega}$ pour synchroniser les automates d'arbre $\mathcal{A}^{(i)}$ construits pour réaliser les différentes expansions afin de générer l'automate d'arbre de la fusion consensuelle. En notant $\mathcal{A}_{(sc)}$ cet automate, les documents du consensus sont les arbres du langage engendré par l'automate $\mathcal{A}_{(sc)}$ à partir d'un état initial construit à partir du tuple formé des états initiaux des automates $(\mathcal{A}^{(i)})$: $\mathscr{L}(\mathcal{A}_{(sc)},\ (q_{t^{maj}_{\mathcal{V}_i}})) = consensus\{\mathscr{L}(\mathcal{A}^{(i)},\ q_{t^{maj}_{\mathcal{V}_i}})\}$. $\mathcal{A}_{(sc)}$ est obtenu en procédant de la façon suivante : (1) Pour chaque vue $\mathcal{V}_i$, construire l'automate $\mathcal{A}^{(i)}$ qui réalisera l'expansion de la réplique partielle $t^{maj}_{\mathcal{V}_i}$ comme indiqué précédemment (sec. 2.2.2) : $\mathscr{L}(\mathcal{A}^{(i)},\ q_{t^{maj}_{\mathcal{V}_i}}) = T^{maj}_{i\mathcal{S}}$. (2) Au moyen de l'opérateur $\otimes^{\Omega}$, calculer l'automate générant le langage du consensus $\mathcal{A}_{(sc)} = \otimes^{\Omega}_i \mathcal{A}^{(i)}$.

3.2. Calcul du consensus

Avant de présenter l'algorithme du calcul du consensus, précisons en utilisant les notions introduites à la section 2.1 que deux documents t_1 et t_2 sont en conflits (et on note $t_1 \curlyvee t_2$) s'il existe une adresse $w \in Dom(t_1) \cap Dom(t_2)$ telle que l'étiquette du noeud n_1 situé à l'adresse w dans t_1 est différente de celle du noeud n_2 située à la même adresse dans t_2 ; n_1 et n_2 ne sont pas des bourgeons. c-à-d. $(t_1 \curlyvee t_2) \Leftrightarrow (\exists w \in Dom(t_1) \cap Dom(t_2),\ t_1(w) \neq X_\omega,\ t_2(w) \neq X_\omega,\ t_1(w) \neq t_2(w)))$ [10]

3.2.1. Consensus entre plusieurs (deux) documents

Soient $t_1, t_2 : \mathbb{N}^* \to \mathbf{A}$ deux arbres (documents) en conflits de domaines respectifs $Dom(t_1)$ et $Dom(t_2)$. L'arbre consensuel $t_c : \mathbb{N}^* \to \mathbf{A}$ issu de t_1 et t_2 est tel que : (1) Le domaine de t_c est l'union des domaines des deux arbres auquel on soustrait les éléments appartenant aux domaines des sous-arbres issus des noeuds en conflits (on élague au niveau des noeuds en conflits).

(2) Quand un noeud n_1 de t_1 est en conflit avec un noeud n_2 de t_2, ils apparaissent dans l'arbre consensuel t_c sous la forme d'un (unique) bourgeon. Ainsi,

$$\forall w \in Dom(t_c),\ t_c(w) = \begin{cases} t_1(w) & si \quad t_1(w) = t_2(w) \\ t_1(w) & si \quad t_2(w) = X_\omega \\ t_2(w) & si \quad t_1(w) = X_\omega \\ t_1(w) & si \quad w \notin Dom(t_2)\ et\ \exists u, v \in \mathbb{N}^*\ tq\ w = u.v,\ t_2(u) = X_\omega \\ t_2(w) & si \quad w \notin Dom(t_1)\ et\ \exists u, v \in \mathbb{N}^*\ tq\ w = u.v,\ t_1(u) = X_\omega \\ X_\omega & si \quad t_1(w) \neq X_\omega\ et\ t_2(w) \neq X_\omega\ et\ t_1(w) \neq t_2(w) \end{cases}$$

3.2.2. Construction de l'automate du consensus

La prise en compte des documents avec des bourgeons nécessite le réaménagement de certains modèles utilisés. Par exemple, dans ce qui suit, nous manipulerons les *automates d'arbre avec états de sortie* en lieu et place des automates d'arbre introduits dans la définition 3. Intuiti-

10. Rappel : on note X_ω l'étiquette d'un bourgeon : $(t(w) = X_\omega) \Leftrightarrow$ le noeud situé à l'adresse w est un bourgeon.

vement, un état q d'un automate est qualifié d'*état de sortie* s'il ne lui est associé qu'une unique transition $q \rightarrow (X_\omega, [])$ permermetant de reconnaitre un arbre réduit à un bourgeon de type $X \in \Sigma$. Ainsi, un automate d'arbre à états de sortie $\mathcal{A}$ est un quintuplet $(\Sigma, Q, R, q_0, exit)$ où Σ, Q, R, q_0 désignent les mêmes objets que ceux introduits dans la définition 3, et *exit* est un prédicat défini sur les états ($exit : Q \rightarrow Bool$). Tout état q de Q pour lequel *exit* q est *Vrai* est un état de sortie.

Dans la section 3.2.1 ci-dessus, nous avons dit que, "*quand deux noeuds sont en conflits, ils apparaissent dans l'arbre consensuel sous la forme d'un (unique) noeud ouvert*". Du point de vue de la synchronisation d'automates, la notion de "noeuds en conflits" se traduit par la notion "d'états en conflits" (que nous précisons ci-dessous) et l'extrait précédent se traduit par "*quand deux états sont en conflits, ils apparaissent dans l'automate du consensus sous la forme d'un (unique) état de sortie*". Ainsi, si on considère deux familles de transitions $q_o^1 -> [(a_1^1, qs_1) \dots (a_{n_1}^1, qs_{n_1})]$ et $q_o^1 -> [(a_1^2, qs'_1) \dots (a_{n_2}^2, qs'_{n_2})]$ associées aux états q_0^1 et q_0^2 de deux automates d'arbres $auto_1$ et $auto_2$, on dira que les états q_0^1 et q_0^2 (qui ne sont pas des états de sortie) sont en conflits (et on note $q_0^1 \curlyvee q_0^2$) s'il n'existe pas de transition partant de chacun d'eux et portant la même étiquette, c-à-d. il n'existe pas d'étiquette de transition a^3 telle que (a^3, qs) appartient à $[(a_1^1, qs_1) \dots (a_{n_1}^1, qs_{n_1})]$ et (a^3, qs') appartient à $[(a_1^2, qs'_1) \dots (a_{n_2}^2, qs'_{n_2})]$. Il est alors évident que deux automates donnés admettent un automate consensuel si leurs états initials ne sont pas en conflit.

L'automate synchronisé consensuel $\mathcal{A}_{(sc)} = \otimes_i^\Omega \mathcal{A}^{(i)}$ dont le processus de construction est décrit par l'algorithme 1 est un automate à états de sortie. Il est obtenu en considérant lors du calcul du produit synchrone des automates $\mathcal{A}^{(i)}$ que : (1) quand un automate $\mathcal{A}^{(j)}$ quelconque a atteint un état de sortie[11], il ne contribue plus au comportement mais, ne s'oppose pas à la synchronisation des autres automates : on dit qu'il est *endormi* (algo. 1 ligne 9). (2) un état $q = (q^1, \cdots, q^k)$ est un état de sortie si : (a) tous les états composites q^i sont endormis (algo. 1 ligne 17) ou (b) s'il existe deux états quelconques q^i et q^j, $i \neq j$, composants de l'état q qui sont en conflits (algo. 1 ligne 18) $\left(exit\ (q^{(1)}, \dots, q^{(k)})\right) \Leftrightarrow \left(\left(exit\ q^{(i)}, \forall i \in \{1 \dots k\}\right) \text{ ou } \left(\exists i, j,\ i \neq j,\ q^{(i)} \curlyvee q^{(j)}\right)\right)$.

Ainsi, l'automate synchronisé consensuel $\mathcal{A}_{(sc)} = \otimes_i^\Omega \mathcal{A}^{(i)}$ est construit comme suit (algo. 1) :

– Ses états sont les vecteurs d'états : $Q = Q^{(1)} \times \cdots \times Q^{(k)}$;

– Son état initial est formé par le vecteur des états initiaux des différents automates : $q_0 = (q_0^{(1)}, \cdots, q_0^{(k)})$ (algo. 1 ligne 2) ;

– Ses transitions sont données par :

- Si $(exit\ q)$ alors $q \rightarrow (X_\omega, [])$ est une transition de $\mathcal{A}_{(sc)}$ (algo. 1 ligne 18),
- sinon $(q^{(1)}, \dots, q^{(k)}) \xrightarrow{a} \left((q_1^{(1)}, \dots, q_1^{(k)}), \dots, (q_n^{(1)}, \dots, q_n^{(k)})\right) \Leftrightarrow \forall i,\ 1 \leq i \leq k$ ou bien
  - $*$ $exit\ q^{(i)}$ et $\left(q_j^{(i)} = q^{(i)}, \forall j,\ 1 \leq j \leq n\right)$ /* $q^{(i)}$ *est endormi* */ , sinon
  - $*$ $q^{(i)} \xrightarrow{a} \left(q_1^{(i)}, \dots, q_n^{(i)}\right)$

$\otimes_i^\Omega$ est donc une relaxation de la synchronisation d'automates que nous avions introduite dans la définition 4 et $\mathcal{L}(\mathcal{A}_{(sc)},\ (q_{t_{\mathcal{V}_i}^{maj}})) = consensus(\mathcal{L}(\mathcal{A}^{(i)},\ q_{t_{\mathcal{V}_i}^{maj}}))$. Il ne reste plus qu'à appliquer un générateur[13] à $\mathcal{A}_{(sc)}$ comme dans [1] pour obtenir les documents les plus simples qu'il accepte, c-à-d. ceux du consensus.

11. Le noeud correspondant dans le document de la projection inverse est un bourgeon et traduit le fait que l'auteur correspondant ne l'a pas édité. Dans le cas où il serait partagé avec un autre co-auteur l'ayant édité dans sa réplique, c'est l'édition réalisée par ce dernier qui sera retenue lors de la fusion.

13. Il est facile d'écrire un générateur (voir dans [1]) qui à partir d'un automate d'arbre énumère ses arbres acceptés les plus simples, c-à-d., dans aucune branche, un état n'est utilisé plus d'une fois pour la génération des noeuds de la branche.

entrée: - k automates $\mathcal{A}_i = (\Sigma, Q_i, R_i, q_0^{(i)}, exit_i)$ $1 \leq i \leq k$, associés aux k répliques ;
- La grammaire globale $\mathbb{G} = (\mathcal{S}, \mathcal{P}, A)$;

sortie : L'Automate $\mathcal{A}_{(sc)} = (\Sigma, Q, R, q_0, exit) = \otimes_i^{\Omega} \mathcal{A}^{(i)}$ qui accepte les documents du consensus

$R \leftarrow \emptyset$ et $N_i \leftarrow \emptyset, \forall i, 1 \leq i \leq k$; `/* N_i est un ensemble de transitions */`
Au départ, $q_0 = (q_0^{(1)}, \cdots, q_0^{(k)})$ est l'unique état de Q et il est non marqué;
si $\forall i, exit_i\ q_0^{(i)}$ *et* $q_0^{(i)} \rightarrow (X_\omega, [\,]) \in R_i$ **alors**
  Ajouter à R la transition $q_0 \rightarrow (X_\omega, [\,])$; positionner $(exit_i\ q_0)$ à $TRUE$; Retourner $\mathcal{A}_{(sc)}$;
tantque *il existe un état non marqué* $q = (q^{(1)}, \cdots, q^{(k)})$ *dans* Q **faire**
  marquer q;
  si *not* $(exit\ q)$ **alors** `/* q n'est pas un état de sortie */`
    pourTout $P : X_0 \rightarrow X_1 \ldots X_n \in \mathcal{P}$ *telle que*
    $\left[q^{(i)} \rightarrow \left(P,\ \left[q_1^{(i)}, \ldots, q_n^{(i)}\right]\right) \in R_i \text{ ou } \left(q^{(i)} \rightarrow (X_{0_\omega}, [\,]) \in R_i \cup N_i\right)\right]$, $\forall i, 1 \leq i \leq k$ **faire**
      Ajouter à R la transition $q \rightarrow \left(P, \left[(q_1^{(1)}, \ldots, q_1^{(k)}), \ldots, (q_n^{(1)}, \ldots, q_n^{(k)})\right]\right)$ dans laquelle
      si $exit_i\ q^{(i)}$ **alors** `/* q^(i) est endormi */`
        $q_l^{(i)} = q^{(i)}$, $\forall l,\ 1 \leq l \leq n$
        pourTout $P' : X_i \rightarrow X_1' \ldots X_{m'}' \in \mathcal{P}$ **faire** `/* Forward état endormi */`
          Ajouter à N_i la transition $q^{(i)} \rightarrow \left(X_{l_\omega}', [\,]\right), \forall l,\ 1 \leq l \leq m'$;
      pourTout $q_j = (q_j^{(1)}, \ldots, q_j^{(k)})$, $q_j \notin Q$, $1 \leq j \leq n$ **faire**
        Ajouter q_j à Q[12] ; `/* q_j est un nouvel état */`;
        si $\left(\forall i,\ exit_i\ q_j^{(i)}\right)$ *ou* `/* q_j est un état de sortie */`
        $\left(\nexists P'' : X_j \rightarrow X_1'' \ldots X_{m''}'' \in \mathcal{P},\ \left(q^{(i)} \rightarrow \left(P'',\ \left[q_1^{(i)}, \ldots, q_{m''}^{(i)}\right]\right) \in R_i \text{ ou } \left(q^{(i)} \rightarrow (X_{j_\omega}, [\,]) \in R_i \cup N_i\right)\right)\right)$, $\forall i$
        `/* les états composites de q_j sont en conflits */` **alors**
        Ajouter à R la transition $q_j \rightarrow (X_{j_\omega}, [\,])$ et positionner $(exit\ q_j)$ à $TRUE$;
Retourner $\mathcal{A}_{(sc)}$;

Algorithm 1: Algorithme de construction de l'automate du consensus

3.3. Illustration

La figure 3 est une illustration d'un processus d'édition coopérative asynchrone engendrant des répliques partielles (fig. 3c et fig. 3e) en conflits [14] par rapport à la grammaire constituée des productions suivantes : $P_1 : A \rightarrow C\,B$ $\quad P_2 : A \rightarrow \varepsilon$ $\quad P_3 : B \rightarrow C\,A$
$P_4 : B \rightarrow B\,B$ $\quad P_5 : C \rightarrow A\,C$ $\quad P_6 : C \rightarrow C\,C$ $\quad P_7 : C \rightarrow \varepsilon$

Initialement dans ce processus, deux répliques partielles (fig. 3b et fig. 3d) sont obtenues par projections du document global (fig. 3a). Suite à leur m-à-j (fig. 3c et fig. 3e) un point de synchronisation est atteint et par application de la démarche décrite dans la section 3.1 c-à-d., association des automates d'arbres $\mathcal{A}^{(1)}$ et $\mathcal{A}^{(2)}$ respectivement aux répliques partielles *tv1* et *tv2*, leur synchronisation consensuelle en l'automate $\mathcal{A}_{(sc)} = \mathcal{A}^{(1)} \otimes^{\Omega} \mathcal{A}^{(2)}$ puis, calcul du langage accepté par cet automate, et enfin, génération des documents du consensus (fig. 3h). Rappelons que le présent exemple est explicitement repris en annexe (sec. A) et la figure (fig. 4) de cet annexe donne l'ensemble des documents les plus simples du consensus.

14. En réalisant les expansions respectives de chacune des répliques, on obtient les documents des f gures respectives f g. 3f et f g. 3g sur lesquels on peut observer aisément un conf it mis en exergue par des zones grisées.

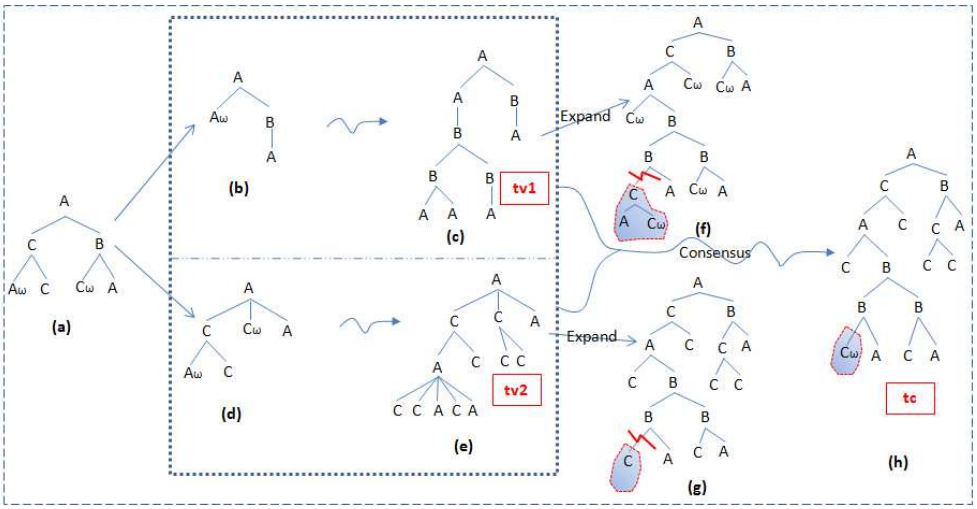

Figure 3. *Exemple de workflow d'une édition avec conflit et consensus correspondant.*

4. Conclusion

Nous avons présenté dans ce papier une approche de réconciliation dite par *consensus*, des répliques partielles d'un document soumis à un processus d'édition coopérative asynchrone. L'approche proposée s'appuie sur une relaxation du produit synchrone d'automates pour construire un automate pouvant générer les documents du consensus.

Les algorithmes présentés dans ce papier ont été implémentés en haskell [3] et expérimentés sur bien des exemples (dont celui explicité en annexe (sec. A)) avec des résultats probants. Nous nous investissons actuellement à la production d'un prototype d'expérimentation via une interface graphique des algorithmes proposés dans un environnement véritablement distribué.

5. Bibliographie

[1] E. Badouel and M. Tchoupé, *«Merging hierarchically structured documents in workflow systems »*, Proceedings of the Ninth Workshop on Coalgebraic Methods in Computer Science (CMCS 2008), Budapest. Electronic Notes in Theoretical Computer Science, 203(5), pp. 3-24, 2008.

[2] H. Comon, M. Dauchet, R. Gilleron, D. Lugiez, S. Tison and M. Tommasi, *«Tree automata techniques and applications »*, Draft, Available at http ://www. grappa. univ-lille3. fr/tata/, 2005.

[3] *Haskell*, A Purely Functional Language. http ://www.haskell.org.

[4] T. Mens, *«A State-of-the-Art Survey on Software Merging »*, Journal of IEEE Transactions on Software Engineering, 28(5), pp. 449-462, 2002.

[5] Y. Saito and M. Shapiro, *«Optimistic replication »*, In ACM Computing Surveys, Vol. V. No. N. 3, pp. 1-44, 2005.

[6] B. Timo and R. Erhard, *«Supporting Efficient Streaming and Insertion of XML Data in RDBMS »*, In Proc. Int. Workshop Data Integration over the Web (DIWeb), pp. 70-81, 2004.

[7] A. VAN DEURSEN, P. KLINT, AND J. VISSER, *«Domain-specific languages : An annotated bibliography »*, ACM SIGPLAN Notices, 35(6), pp. 36-36, June 2000.

A. Annexe

Nous illustrons l'algorithme de fusion consensuelle avec la grammaire de la section 3.3. Nous associons les automates $\mathcal{A}^{(1)}$ et $\mathcal{A}^{(2)}$ respectivement aux répliques partielles m-à-j $tv1$ et $tv2$ (fig. 3c et fig. 3e), puis nous construisons l'automate du consensus $\mathcal{A}_{(sc)} = \mathcal{A}^{(1)} \otimes^{\Omega} \mathcal{A}^{(2)}$ par application de la démarche décrite dans la section 3.2.2 et enfin, présentons les documents les plus simples du consensus (fig. 4).

Linéarisation d'un document structuré

Afin de simplifier la présentation, nous représentons dans ce qui suit les arbres par leur linéarisation sous la forme d'un mot de Dyck. Pour ce faire, nous associons une paire (différentes) de parenthèses à chaque symbole grammaticale et la linéarisation d'un arbre est obtenue en effectuant un parcours en profondeur d'abord de l'arbre résultant.

Les schémas de transition pour la vue $\{A,B\}$

Une liste d'arbres (forêts) est représentée par la concaténation de leurs linéarisations. Nous utilisons la parenthèse ouvrante '(' et fermante ')' pour représenter les symboles de Dyck associés au symbole visible A et le crochet ouvrant '[' et fermant ']' pour représenter ceux associés au symbole visible B. Chaque transition des automates associés aux répliques partielles suivant la vue $\{A,B\}$ est conforme à l'un des schémas de règles suivants :

$$\begin{array}{llll||llll}
\langle A,w_1\rangle & \longrightarrow & (P_1,[\langle C,u\rangle,\langle B,v\rangle]) & \text{si } w_1=u[v] & \langle A,w_2\rangle & \longrightarrow & (P_2,[\,]) & \text{si } w_2=\varepsilon \\
\langle B,w_3\rangle & \longrightarrow & (P_3,[\langle C,u\rangle,\langle A,v\rangle]) & \text{si } w_3=u(v) & \langle B,w_4\rangle & \longrightarrow & (P_4,[\langle B,u\rangle,\langle B,v\rangle]) & \text{si } w_4=[u][v] \\
\langle C,w_5\rangle & \longrightarrow & (P_5,[\langle A,u\rangle,\langle C,v\rangle]) & \text{si } w_5=(u)v & \langle C,w_6\rangle & \longrightarrow & (P_6,[\langle C,u\rangle,\langle C,v\rangle]) & \text{si } w_6=uv\neq\varepsilon \\
\langle C,w_7\rangle & \longrightarrow & (C_\omega,[\,]) & \text{si } w_7=\varepsilon & & & &
\end{array}$$

Ces schémas de règles sont obtenus à partir des productions de la grammaire [1] et les couples $\langle X,w_i\rangle$ sont des états dans lesquels X est un symbole grammatical et w_i une forêt codée dans le langage de Dyck. Le premier schéma par exemple, stipule que les arbres de syntaxe abstraite (AST) générés à partir de l'état $\langle A,w_1\rangle$ sont ceux obtenus en utilisant la production P_1 pour créer un arbre de la forme $P_1[t_1,t_2]$; t_1 et t_2 étant générés respectivement à partir des états $\langle C,u\rangle$ et $\langle B,v\rangle$ tel que $w_1=u[v]$. L'état $\langle C,w_7\rangle$ avec $w_7=\varepsilon$ étant un état de sortie [1], la règle $\langle C,w_7\rangle \longrightarrow (C_\omega,[\,])$ liée à la production P_7 stipule que l'AST généré à partir de l'état $\langle C,w_7\rangle$ est réduit à un bourgeon de type C (C est le symbole situé en partie gauche de P_7).

Construction de l'automate $\mathcal{A}^{(1)}$ associé à $tv1$

Ayant associé les symboles de Dyck '(' et ')' (resp. '[' et ']') au symbole grammatical A (resp. B), la linéarisation de la réplique partielle $tv1$ (fig. 3.c) donne $(([[()()][()]])[()])$. A étant l'axiome de la grammaire, l'état initial de l'automate $\mathcal{A}^{(1)}$ est $q_0^1=\langle A,(([[()()][()]])[()])\rangle$. En ne considérant que les états accessibles à partir de q_0^1 et en appliquant les schémas de règles présentés précédemment, nous obtenons l'automate d'arbre suivant pour la réplique $tv1$ (fig. 3.c) :

$$\begin{array}{llll}
q_0^1 & \longrightarrow & (P_1,[q_1^1,q_2^1]) & \text{avec } q_1^1=\langle C,([[()()][()]])\rangle \text{ et } q_2^1=\langle B,()\rangle \\
q_1^1 & \longrightarrow & (P_5,[q_3^1,q_4^1]) & \text{avec } q_3^1=\langle A,[[()()][()]]\rangle \text{ et } q_4^1=\langle C,\varepsilon\rangle \\
q_1^1 & \longrightarrow & (P_6,[q_4^1,q_1^1]) \mid (P_6,[q_1^1,q_4^1]) & \\
q_2^1 & \longrightarrow & (P_3,[q_4^1,q_5^1]) & \text{avec } q_5^1=\langle A,\varepsilon\rangle \\
q_3^1 & \longrightarrow & (P_1,[q_4^1,q_6^1]) & \text{avec } q_6^1=\langle B,[()()][()]\rangle \\
q_4^1 & \longrightarrow & (C_\omega,[\,]), & q_5^1 \longrightarrow (P_2,[\,]) \\
q_6^1 & \longrightarrow & (P_4,[q_7^1,q_2^1]) & \text{avec } q_7^1=\langle B,()()\rangle \\
q_7^1 & \longrightarrow & (P_3,[q_8^1,q_5^1]) & \text{avec } q_8^1=\langle C,()\rangle \\
q_8^1 & \longrightarrow & (P_5,[q_5^1,q_4^1]), & q_8^1 \longrightarrow (P_6,[q_8^1,q_4^1]) \mid (P_6,[q_4^1,q_8^1])
\end{array}$$

L'état $q_4^1=\langle C,\varepsilon\rangle$ est le seul état de sortie de $\mathcal{A}^{(1)}$. Il est aisé de vérifier que le document de la figure 3f issu de la projection inverse de $tv1$ appartient au langage accepté par l'automate $\mathcal{A}^{(1)}$.

Construction de l'automate $\mathcal{A}^{(2)}$ associé à $tv2$

Comme précédemment, en associant au symbole grammatical C (resp. A) les symboles de Dyck

'[' et ']' (resp. '(' et ')'), on obtient les schémas des transitions pour les automates associés aux répliques partielles suivant la vue $\{A,C\}$.

La linéarisation de la réplique partielle $tv2$ (fig. 3e) donne $([([\,][\,]()[\,]())[\,]][[\,][\,]]())$. L'automate $\mathcal{A}^{(2)}$ associé à cette réplique a pour état initial $q_0^2 = \langle A, [([\,][\,]()[\,]())[\,]][[\,][\,]]()\rangle$ et pour transitions :

$q_0^2 \longrightarrow (P_1,[q_1^2,q_2^2])$ avec $q_1^2 = \langle C,([\,][\,]()[\,]())[\,]\rangle$ et $q_2^2 = \langle B,[[\,][\,]]()\rangle$
$q_1^2 \longrightarrow (P_5,[q_3^2,q_4^2])$ avec $q_3^2 = \langle A,[\,][\,]()[\,]()\rangle$ et $q_4^2 = \langle C,\varepsilon\rangle$
$q_2^2 \longrightarrow (P_3,[q_5^2,q_6^2])$ avec $q_5^2 = \langle C,[\,][\,]\rangle$ et $q_6^2 = \langle A,\varepsilon\rangle$
$q_3^2 \longrightarrow (P_1,[q_4^2,q_7^2])$ avec $q_7^2 = \langle B,[\,]()[\,]()\rangle$
$q_4^2 \longrightarrow (P_7,[\,])$ $q_5^2 \longrightarrow (P_6,[q_4^2,q_4^2])$ $q_6^2 \longrightarrow (P_2,[\,])$
$q_7^2 \longrightarrow (P_4,[q_8^2,q_7^2])$ | $(P_4,[q_9^2,q_{10}^2])$ | $(P_4,[q_{11}^2,q_{11}^2])$ | $(P_4,[q_{12}^2,q_{13}^2])$ | $(P_4,[q_7^2,q_8^2])$ avec $q_8^2 = \langle B,\varepsilon\rangle$, $q_9^2 = \langle B,[\,]\rangle$, $q_{10}^2 = \langle B,()[\,]()\rangle$, $q_{11}^2 = \langle B,[\,]()\rangle$, $q_{12}^2 = \langle B,[\,]()[\,]\rangle$ et $q_{13}^2 = \langle B,()\rangle$
$q_8^2 \longrightarrow (B_\omega,[\,])$ $q_9^2 \longrightarrow (P_4,[q_8^2,q_9^2])$ | $(P_4,[q_9^2,q_8^2])$
$q_{10}^2 \longrightarrow (P_4,[q_8^2,q_{10}^2])$ | $(P_4,[q_{13}^2,q_{11}^2])$ | $(P_4,[q_{14}^2,q_{13}^2])$ | $(P_4,[q_{10}^2,q_8^2])$ avec $q_{14}^2 = \langle B,()[\,]\rangle$
$q_{11}^2 \longrightarrow (P_3,[q_4^2,q_6^2])$
$q_{12}^2 \longrightarrow (P_4,[q_8^2,q_{12}^2])$ | $(P_4,[q_9^2,q_{14}^2])$ | $(P_4,[q_{11}^2,q_9^2])$ | $(P_4,[q_{12}^2,q_8^2])$
$q_{13}^2 \longrightarrow (P_4,[q_8^2,q_{13}^2])$ | $(P_4,[q_{13}^2,q_8^2])$
$q_{14}^2 \longrightarrow (P_4,[q_8^2,q_{14}^2])$ | $(P_4,[q_{13}^2,q_9^2])$ | $(P_4,[q_{14}^2,q_8^2])$

L'état $q_8^2 = \langle B,\varepsilon\rangle$ est le seul état de sortie de l'automate $\mathcal{A}^{(2)}$.

Construction de l'automate du consensus $\mathcal{A}_{(sc)}$

Par application du produit synchrone de plusieurs automates d'arbres décrit dans la section 3.2.2 aux automates $\mathcal{A}^{(1)}$ et $\mathcal{A}^{(2)}$, l'automate du consensus $\mathcal{A}_{(sc)} = \mathcal{A}^{(1)} \otimes^{\Omega} \mathcal{A}^{(2)}$ a pour état initial $q_0 = (q_0^1,q_0^2)$. $\mathcal{A}^{(1)}$ possède une transition de q_0^1 vers $[q_1^1,q_2^1]$ étiquetée P_1. De même, $\mathcal{A}^{(2)}$ possède une transition de q_0^2 vers $[q_1^2,q_2^2]$ étiquetée P_1. On a donc dans $\mathcal{A}_{(sc)}$ une transition etiquetée P_1 permettant d'accéder aux états $[q_1 = (q_1^1,q_1^2), q_2 = (q_2^1,q_2^2)]$ à partir de l'état initial $q_0 = (q_0^1,q_0^2)$. Suivant ce principe, on construit l'automate consensuel suivant :

$q_0 = (q_0^1,q_0^2)$
$q_0 \longrightarrow (P_1,[q_1,q_2])$ avec $q_1 = (q_1^1,q_1^2)$ et $q_2 = (q_2^1,q_2^2)$
$q_1 \longrightarrow (P_5,[q_3,q_4])$ avec $q_3 = (q_3^1,q_3^2)$ et $q_4 = (q_4^1,q_4^2)$
$q_2 \longrightarrow (P_3,[q_5,q_6])$ avec $q_5 = (q_4^1,q_5^2)$ et $q_6 = (q_5^1,q_6^2)$
$q_3 \longrightarrow (P_1,[q_4,q_7])$ avec $q_7 = (q_6^1,q_7^2)$
$q_4 \longrightarrow (P_7,[\,])$, $q_5 \longrightarrow (P_6,[q_4,q_4])$, $q_6 \longrightarrow (P_2,[\,])$
$q_7 \longrightarrow (P_4,[q_8,q_9])$ | $(P_4,[q_{10},q_{11}])$ | $(P_4,[q_{12},q_{13}])$ | $(P_4,[q_{14},q_{15}])$ | $(P_4,[q_{16},q_{17}])$ avec $q_8 = (q_7^1,q_8^2)$, $q_9 = (q_2^1,q_7^2)$, $q_{10} = (q_7^1,q_9^2)$, $q_{11} = (q_2^1,q_{10}^2)$, $q_{12} = (q_7^1,q_{11}^2)$, $q_{13} = (q_2^1,q_{11}^2)$, $q_{14} = (q_7^1,q_{12}^2)$, $q_{15} = (q_2^1,q_{13}^2)$, $q_{16} = (q_7^1,q_7^2)$ et $q_{17} = (q_2^1,q_8^2)$
$q_8 \longrightarrow (P_3,[q_{18},q_{19}])$ avec $q_{18} = (q_8^1,q_8^2)$ et $q_{19} = (q_5^1,q_8^2)$
$q_{12} \longrightarrow (P_3,[q_{20},q_6])$ avec $q_{20} = (q_8^1,q_4^2)$
$q_{13} \longrightarrow (P_3,[q_4,q_6])$
$q_{17} \longrightarrow (P_3,[q_{21},q_{19}])$ avec $q_{21} = (q_4^1,q_8^2)$
$q_{18} \longrightarrow (P_5,[q_{19},q_{21}])$ | $(P_6,[q_{18},q_{21}])$ | $(P_6,[q_{21},q_{18}])$

$q_{19} \longrightarrow (P_2, [\,]),$ $q_9 \longrightarrow (B_\omega, [\,])$ $q_{10} \longrightarrow (B_\omega, [\,])$ $q_{11} \longrightarrow (B_\omega, [\,])$ $q_{14} \longrightarrow (B_\omega, [\,])$ $q_{15} \longrightarrow (B_\omega, [\,])$ $q_{16} \longrightarrow (B_\omega, [\,])$ $q_{20} \longrightarrow (C_\omega, [\,])$ $q_{21} \longrightarrow (C_\omega, [\,])$

Les états $\{q_9, q_{10}, q_{11}, q_{14}, q_{15}, q_{16}, q_{20}, q_{21}\}$ sont les états de sortie de l'automate $\mathcal{A}_{(sc)}$. Ce sont des états dont les états composites sont soit en conflits (par exemple $q_9 = (q_2^1, q_7^2)$ et $q_2^1 \not\vee q_7^2$), soit sont tous des états de sortie (par exemple $q_{21} = (q_4^1, q_8^2)$).

L'utilisation de la fonction de génération des AST (avec bourgeons) les plus simples d'un langage d'arbre à partir de sont automate [1] sur l'automate $\mathcal{A}_{(sc)}$, produit *quatre* AST dont les arbres de dérivations (les consensus) sont schématisés sur la figure 4.

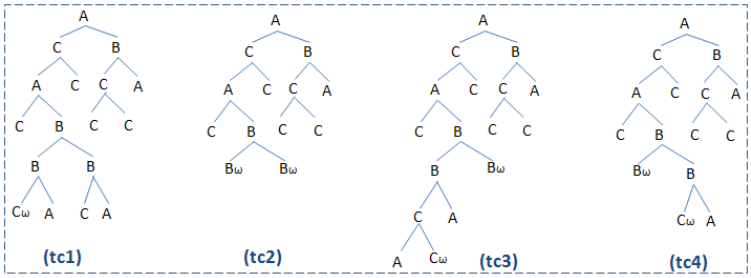

Figure 4. *Arbres consensuels générés à partir de l'automate* $\mathcal{A}_{(sc)}$