# Une représentation des arborescences pour la recherche de sous-structures fréquentes

Federico del Razo Lopez, Anne Laurent, Maguelonne Teisseire

# Une représentation des arborescences pour la recherche de sous-structures fréquentes

Federico Del Razo Lopez, Anne Laurent, Maguelonne Teisseire

LIRMM - Université Montpellier II
161 rue Ada 34392 Montpellier cedex 5
{delrazo,laurent,teisseire}@lirmm.fr

**Résumé.** La recherche de structures fréquentes au sein de données arborescentes est une problématique actuellement très active qui trouve de nombreux intérêts dans le contexte de la fouille de données comme, par exemple, la construction automatique d'un schéma médiateur à partir de schémas XML. Dans ce contexte, de nombreuses propositions ont été réalisées mais les méthodes de représentation des arborescences sont très souvent trop coûteuses. Dans cet article, nous proposons donc une méthode originale de représentation de ces données. Les propriétés de cette représentation peuvent être avantageusement utilisées par les algorithmes de recherche de structures fréquentes (sous-arbres fréquents). La représentation proposée et les algorithmes associés ont été évalués sur des jeux de données synthétiques montrant ainsi l'interêt de l'approche proposée.

## 1 Introduction

L'explosion du volume de données disponible sur internet conduit aujourd'hui à réfléchir sur les moyens d'interroger les grosses masses d'information afin de retrouver les informations souhaitées. Les utilisateurs ne pouvant pas connaître les modèles sous-jacents des données qu'ils souhaitent accéder, il est donc nécessaire de leur fournir les outils automatiques de définition de schémas médiateurs. Un schéma médiateur fournit une interface permettant l'interrogation des sources de données par l'utilisateur au travers de requêtes. L'utilisateur pose alors ses requêtes de manière transparente à l'hétérogénéité et la répartition des données.

XML étant maintenant prépondérant sur internet, la recherche de moyens d'intégration de tels schémas est indispensable. Si les recherches permettant l'accès aux données quand un schéma d'interrogation est connu sont maintenant bien avancées (Xyleme, 2001), les recherches concernant la définition automatique d'un schéma médiateur restent incomplètes et sont donc non satisfaisantes (Tranier et al., 2004). Dans le but de proposer une approche permettant de répondre à cette dernière problématique, nous nous focalisons sur la recherche de sous-structures fréquentes au sein d'une base de données de schémas XML. Une sous-structure fréquente est un sous-arbre se trouvant dans *la plupart* des schémas XML considérés. Cette proportion est examinée au sens d'un *support* qui correspond à un nombre minimal d'arbres de la base dans lesquels doit se retrouver le sous-arbre pour être considéré comme *fréquent*. Une telle recherche

est complexe dans la mesure où il est nécessaire de traduire l'ensemble des schémas en une structure aisément manipulable. Cette transformation des données conduit parfois à doubler ou tripler la taille de la base initiale dès lors que l'on souhaite utiliser des propriétés spécifiques permettant d'améliorer le processus de fouille. Il n'existe pas de solution efficace à ce problème alliant une représentation compacte à des propriétés intéressantes. L'objet de cet article est la définition d'une nouvelle structure répondant à cet objectif puisque les propriétés de la représentation proposée peuvent permettre une génération des candidats et un élagage aussi performants que les approches de référence (Asai et al., 2002, Zaki, 2002, Termier et al., 2002).

La section 3 présente le cœur de notre proposition définissant une nouvelle méthode de représentation des données arborescentes ainsi qu'un aperçu de la méthode mise en œuvre pour la génération et l'élagage des candidats. La section 4 présente les résultats des premières expérimentations menées. Enfin, la section 5 conclut et présente les principales perspectives associées à nos travaux.

# 2 Travaux connexes

## 2.1 Définitions préliminaires

Les travaux les plus significatifs concernant l'extraction de sous-arbres fréquents se trouvent dans (Kuramochi et Karypis, 2001, Asai et al., 2002, Yan et Han, 2002, Zaki, 2002, Termier et al., 2002). Les définitions suivantes sont inspirées de ces travaux.

Un *arbre* est un graphe connexe sans cycle. Il est composé d'un ensemble de nœuds reliés par des arcs tels qu'il existe un nœud particulier nommé *racine* et tel que tous les autres nœuds hormis la racine sont composés d'un ensemble de sous-arbres. On parle d'*arbre ordonné* si l'ordre des sous-arbres importe, et d'*arbre non ordonné* sinon.

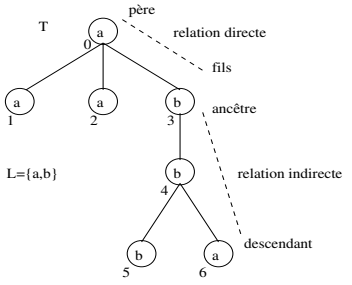


FIG. 1 – *Arbre*

Soit un alphabet d'étiquettes $\mathcal{L} = \{a, b, c, ...\}$, on considère un *arbre étiqueté et ordonné* noté $T = \{r, N, B, L, F\}$ où : $r$ est la racine de l'arbre, $N$ est l'ensemble des nœuds, $B$ est l'ensemble des arêtes tel que $B \subseteq V^2$, $(L : N \rightarrow \mathcal{L})$ est une fonction qui associe une étiquette aux nœuds dans $N$ et $F$ est une relation d'ordre de droite à

gauche entre frères.

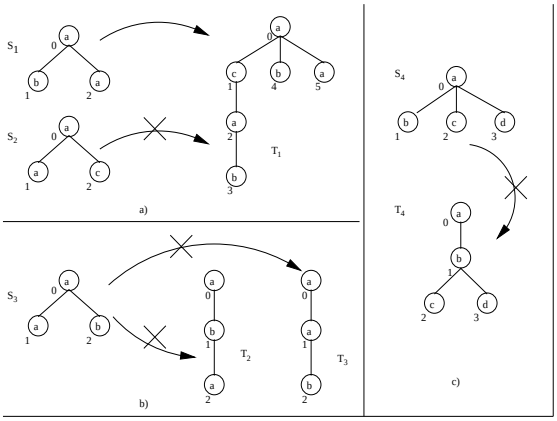


FIG. 2 – *Inclusion et non inclusion de S dans T*

Plusieurs nœuds pourront avoir la même étiquette, ce qui introduit un problème de polysémie (une même étiquette renvoie à plusieurs nœuds différents). La *taille* de $T$, notée $|T|$, est le nombre de nœuds dans $T$. l'*ordre* de $T$ correspond au nombre de branches dans $T$.

Chaque nœud $n$ est associé à un numéro unique $i$, correspondant à sa position par un parcours en profondeur d'abord. $n_i$ fait alors référence au $i$-ème nœud en utilisant un schéma de numération ($i = 0...|T| - 1$).

Pour chaque couple $(x, y) \in N \times N$, il peut exister une relation *directe* entre $x$ et $y$ si $x$ est le père de $y$ ou bien une relation *indirecte* si $x$ est un ancêtre de $y$ (il faut alors suivre plusieurs arcs successifs pour aller de $x$ à $y$). La figure 1 montre une relation directe (père-fils) entre les nœuds 0 et 3 de $T$, et une relation indirecte (ancêtre-descendant) entre les nœuds 3 et 6.

On note $S \preceq T$ le fait que le sous-arbre $S$ est *inclus* dans l' arbre étiqueté ordonné $T$ (voir figure 2). Dans le cadre d'une relation indirecte *ancêtre-descendant*, $S \preceq T$ si et seulement si les suivantes conditions sont satisfaites :

1. $N_S \subset N_T$
2. Pour toutes les branches $b_S = (x, y) \in B_S$, $x$ est ancêtre de $y$ dans $T$
3. Pour toutes les branches $b_T = (x, y) \in B_T$, $x$ est ancêtre de $y$ dans $S$
4. Pour toutes les branches $b_{1S} = (x, y_1) \in B_S$ et $b_{2S} = (x, y_2) \in B_S$ telles que $y_1 \prec_F y_2$, $y_1 \prec_F y_2$ dans $T$

## 2.2 Les représentations existantes

L'algorithme *TREEMINER* (Zaki, 2002) propose une méthode d'extraction de sous-arbres fréquents. De même que dans notre approche, ces travaux reposent sur une

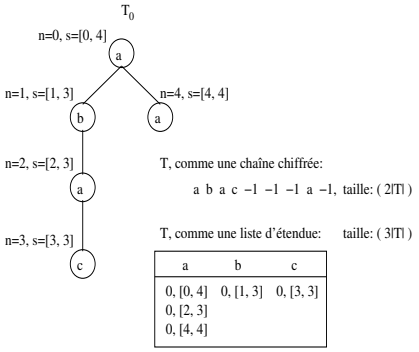


FIG. 3 – *Représentation proposée dans (Zaki, 2002)*

représentation originale des arbres facilitant la gestion des candidats. Néanmoins, la représentation verticale adoptée aboutit à stocker trois fois la taille d'un arbre comme l'indique la figure 3.

L'approche proposée dans (Asai et al., 2002) est dédiée aux arbres ordonnés et adopte une structure permettant des performances très intéressantes. Mais cette représentation, illustrée figure 4, conduit également à tripler la taille de la base afin de stocker les informations nécessaires.

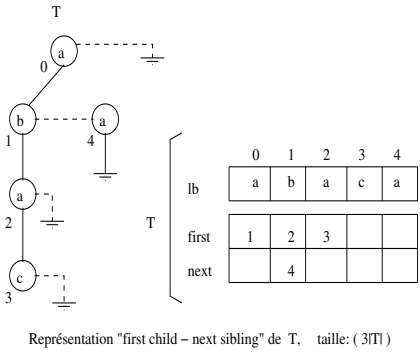


FIG. 4 – *Représentation proposée dans (Asai et al., 2002)*

Des proposition récentes se basent sur une représentation efficace des arbres mais néanmoins, elles n'utilisent pas des propriétés aussi intéressantes que les travaux précédents afin d'améliorer les traitement des structures candidates. Nous pouvons citer (Wang et al., 2004) dont la structure est illustrée figure 5 et (Chi et al., 2004), (Chi et al., 2003) proposant une représentation des arbres illustrée figure 6.

L'objectif est alors de proposer à la fois une représentation peu coûteuse de la base

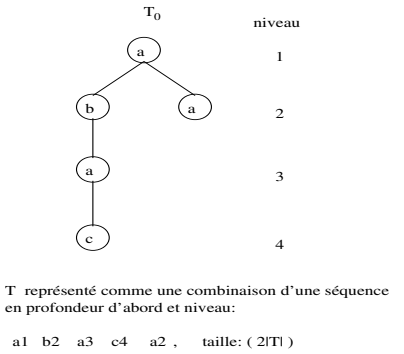


FIG. 5 – *Représentation proposée dans (Wang et al., 2004)*

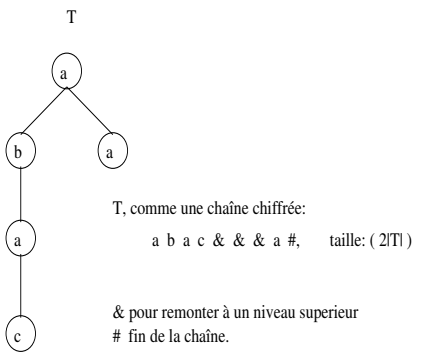


FIG. 6 – *Représentation proposée dans (Chi et al., 2004,Chi et al., 2003)*

et des propriétés intéressantes pour améliorer le processus de fouille de données. C'est dans ce contexte que se situe notre proposition décrite au paragraphe suivant.

# 3 Proposition

Dans cet article, nous proposons une nouvelle approche permettant l'extraction efficace de sous-arbres fréquents au sein d'une base de données arborescentes. Notre proposition s'appuie sur une méthode de représentation des arbres originale qui permet de générer de manière efficace les sous-arbres candidats puis d'élaguer les sous-arbres non fréquents (après calcul du support).

## 3.1 Représentation des arbres

Pour la représentation d'un arbre $T$, nous profitons de la propiété suivante : chaque nœud dans l'arbre possède un seul parent. Nous proposons d'utiliser deux vecteurs pour représenter un arbre comme indiqué dans (Weiss, 1998). Le premier, nommé $st$, conserve la position du père de chaque nœud. Les nœuds de l'arbre sont numérotés en profondeur d'abord, la racine de $T$ correspondant à l'index 0 et ayant une valeur $st[0] = -1$ pour indiquer que la racine n'a pas de père. Les valeurs $st[i], i = 0, 1, ..., k-1$ correspondent alors aux positions du père des nœuds $i$, comme illustré sur la figure 7.

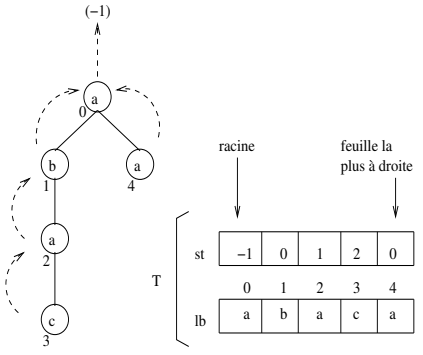


FIG. 7 – *Représentation d'un arbre*

Cette représentation permet de retrouver en temps constant le père d'un nœud. De plus, elle permet la localisation directe de la *feuille la plus à droite* par rapport à l'index $k$. En parcourant l'arbre, on peut bâtir toutes les relations directes *père-fils* entre nœuds.

Le deuxième vecteur, nommé $lb$, est utilisé pour enregistrer les étiquettes de l'arbre avec $lb[i], i = 0, 2, ..., k-1$ représentant l'étiquette de chaque nœud $n_i \in T$.

La structure adoptée permet une représentation des arbres peu coûteuse puisqu'elle se réduit à $2|T|$. De plus elle possède des propriétés intéressantes, évoquées au paragraphe suivant, pouvant être utilisées lors de la recherche de sous structures fréquentes.

## 3.2 Génération et élagage des candidats

Les candidats de taille 1 sont obtenus en parcourant tous les nœuds des arbres de la base de données. Chaque nœud voit son support incrémenté lors de ce parcours et seuls sont conservés les nœuds dont le support final est supérieur au support minimal défini par l'utilisateur. La base de données est alors transformée pour ne conserver que les nœuds fréquents, comme illustré par la figure 8.

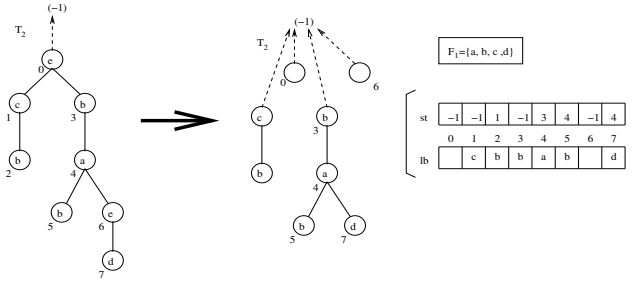


FIG. 8 – *Transformation de la base de données après génération de* $F_1$

Les candidats de taille 2 sont générés en combinant deux à deux tous les fréquents de taille 1.

La base de données est alors mise à jour, en modifiant les arbres de la racine, des sommets et des feuilles afin de ne conserver que les liens entre les nœuds fréquents. Les figures 9, 10 et 11 illustrent ce processus, en considérant $\sigma = 7$ et $F_2 = \{a - d, a - b\}$.

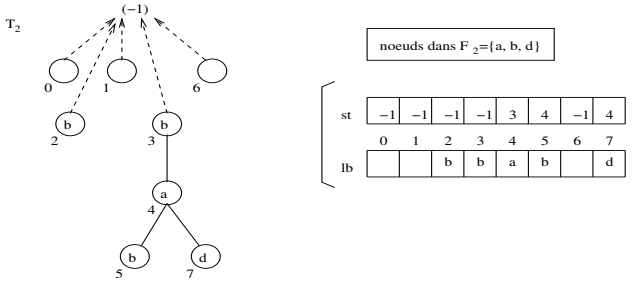


FIG. 9 – *Transformation de la base - racine*

La génération des candidats de taille $k \geq 3$ s'effectue de la même manière que dans les approches classiques de type Apriori (Agrawal et Srikant, 2002), par combinaison

des fréquents de taille $k-1$.

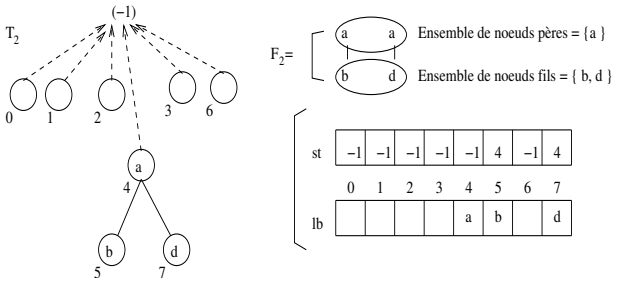


FIG. 10 – *Transformation de la base - sommets*

L'originalité de notre approche réside dans l'utilisation de notre représentation pour l'élagage des candidats non fréquents. Le calcul du support de chaque candidat consiste à compter le nombre d'arbres de la base qui contiennent ce sous-arbre candidat. Pour ce faire, pour chaque arbre de la base, il s'agit de chercher les *points d'ancrage* sur lesquels la racine du sous-arbre à tester peut s'instancier. Pour chaque point d'ancrage trouvé, on cherche alors à instancier l'ensemble des nœuds de l'arbre candidat au sein de l'arbre courant testé. On note que nous cherchons une instanciation *exacte* du candidat au sein des arbres de la base. Si tous les nœuds du candidat ont été trouvés, l'arbre supporte le candidat. Le support de la structure candidate est alors incrémenté. Ce qui n'est pas le cas si tous les nœuds de l'arbre ont été parcourus sans trouver l'ensemble des nœuds du candidat.

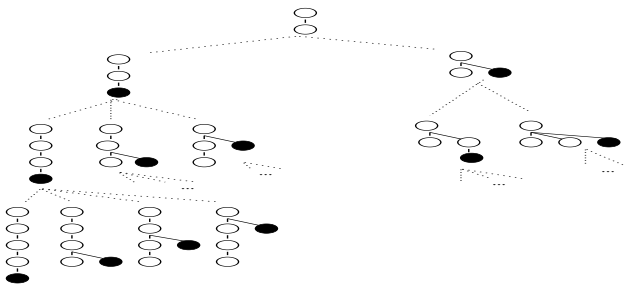

FIG. 11 – *Transformation de la base - feuilles*

FIG. 12 – *Résulats expérimentaux : temps de calcul*

# 4 Expérimentations

Des expérimentations sont menées sur les données issues du générateur de schémas XML développé par Alexandre Termier (Termier et al., 2002). Les résultats, illustrés figure 12, soulignent l'intérêt de notre approche, concernant les temps de calcul.

# 5 Conclusion et perspectives

Dans cet article, nous proposons une approche originale de représentation de données arborescentes dont les propriétés permettent une extraction efficace de sous-arbres fréquents. Les premières expérimentations réalisées sur des données synthétiques laissent envisager des résultats très prometteurs par rapport aux approches de référence. Notre objectif est donc de poursuivre dans cette voie et d'optimiser les différents algorithmes associés.
Ces travaux seront également utilisés dans le cadre de la médiation de données, les sous-arbres fréquents extraits servant de support à la construction automatique d'un schéma médiateur. Une telle solution peut également être adoptée dans le cadre de la fouille de données en ligne (data streams) pour le traitement à la volée de données XML. Cette perspective permettra de traiter les gros volumes de données transitant sur internet de manière efficace et rapide.

# Références

Agrawal, R. and Srikant, R. (2002). Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th VLDB Conference*, Santiago, Chile.

Asai, T., Abe, K., Kawasoe, S., Arimura, H., and Sakamoto, H. (2002). Efficient substructure discovery from large semi-structure data. In *2nd Annual SIAM Symposium on Data Mining, SDM2002*, Arlington, VA, USA. Springer-Verlag.

Chi, Y., Yang, Y., and Muntz, R. R. (2003). Indexing and mining free trees. In *International Conference on Data Mining 2003 (ICDM2003)*.

Chi, Y., Yang, Y., and Muntz, R. R. (2004). Cmtreeminer : Mining both closed and maximal frequent subtrees. In *The Eighth Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'04)*.

Kuramochi, M. and Karypis, G. (2001). Frequent subgraph discovery. In *IEEE International Conference on Data Mining (ICDM)*.

Termier, A., Rousset, M.-C., and Sebag, M. (2002). Treefinder, a first step towards XML data mining. In *IEEE Conference on Data Mining (ICDM)*, pages 450–457.

Tranier, J., Baraer, R., Bellahsene, Z., and Teisseire, M. (July, 7th - 9th 2004). Where's Charlie : Family based heuristics for peer-to-peer schema integration. In *Proceedings of the 8th International Database Engineering and Applications Symposium (IDEAS '04)*, Coimbra, Portugal.

Wang, C., Yuan, Q., Zhou, H., Wang, W., and Shi, B. (May 2004). Chopper : An efficient algorithm for tree mining. *Journal of Computer Science and Technology*, 19 :309–319.

Weiss, M. A. (1998). *Data Structures And Algorithm Analysis In C*.

Xyleme, L. (2001). A dynamic warehouse for xml data of the web. In *IEEE Data Engineering Bulletin*.

Yan, X. and Han, J. (2002). gspan : Graph-based substructure pattern mining. In *IEEE Conference on Data Mining (ICDM)*.

Zaki, M. (2002). Efficiently mining frequent trees in a forest. In *ACM-SIGKDD'02*.

## Summary

Mining frequent subtrees from tree databases is currently a very active research topic. This research has many interests, including for instance mediator schema building from XML schemas. In this framework, many works have been proposed. However, the way the data are represented is often very memory-consuming. In this paper, we propose a new method to represent such data. We show that this representation has many properties, which can be used in order to enhance subtree mining algorithms. Experiments are led to assess our proposition (data representation and its associated algorithms).