

Nettoyage des données XML : combien ça coûte ?

Laure Berti-Équille

IRISA, Campus Universitaire de Beaulieu, 35042 Rennes cedex
berti@irisa.fr
<http://www.irisa.fr>

Résumé. L'objectif de cet article est de présenter un travail en cours qui consiste à proposer, implanter et valider expérimentalement un modèle pour estimer le coût d'un processus de nettoyage de documents XML. Notre approche de calcul de coût est basée sur une méthode par calibration selon une analyse probabiliste. Pour cela, nous proposons de calculer des probabilités de pollution et au préalable de détection des différents types de pollutions. Pour valider notre modèle, nous avons choisi de polluer artificiellement une collection de données XML avec l'ensemble des types d'erreurs possibles (erreurs typographiques, ajout de doublons, de valeurs manquantes, tronquées, censurées, etc.) et d'estimer, grâce au modèle proposé, le nombre et le coût des opérations nécessaires au nettoyage des données afin de proposer des stratégies de réparation ciblées et économes. Les expérimentations en cours ne sont pas rapportées dans cet article.

1 Introduction

Le nettoyage automatique des données se décompose classiquement en trois étapes : 1) examiner les données afin de détecter les incohérences, les données manquantes, les erreurs, les doublons, etc. 2) choisir les transformations pour résoudre les problèmes, 3) et enfin, appliquer les transformations choisies au jeu de données. La plupart des outils utilisés pour le nettoyage des données par *Extraction-Transformation-Loading (ETL)* permettent l'extraction d'expressions régulières et structures (*patterns*) à partir des données, ainsi que leur transformation et formatage par l'application de différentes fonctions (sélection, fusion, *clustering*, etc.) (Vassiliadis 2003) dont généralement, on ignore *a priori* le coût. Bien qu'il existe de nombreux travaux (Dasu 2003), (Winkler 2003), (Rahm 2000) outils et prototypes (Telcordia (Caruso 2000), AJAX (Galhardas 2001), Potter's Wheel (Raman 2001), ArktoS (Vassiliadis 2000), IntelliClean (Low 2000), Tailor (Elfeky 2002)) développés pour « nettoyer » les données relationnelles, très peu de travaux à l'exception des récents travaux de Weis et Naumann (Weis 2004), ont jusqu'ici été menés pour le nettoyage de données XML et, à notre connaissance, aucun n'a abordé l'estimation du coût d'un nettoyage de données *a fortiori* pour des données XML. C'est dans ce cadre qu'a débuté notre travail dont l'objectif est de proposer, d'implanter et valider expérimentalement un modèle de coût global permettant d'estimer combien peut coûter un processus de nettoyage sur un document XML artificiellement pollué pour les besoins de nos expériences.

La suite de l'article s'organise de la façon suivante : la section 2 propose notre démarche illustrée par un exemple simple qui énumère les différents types de pollution possibles dans un document XML. La section 3 présente plus formellement notre modèle de coût avec ses

définitions préliminaires et ses paramètres. Enfin, la section 4 conclut l'article et présente brièvement nos perspectives de travail.

2 Problématique

La Figure 1 illustre sur un exemple simple de document XML (*clean.xml*) quelques-uns des types d'erreurs pouvant être introduites artificiellement (*dirty.xml*) pour valider notre modèle.

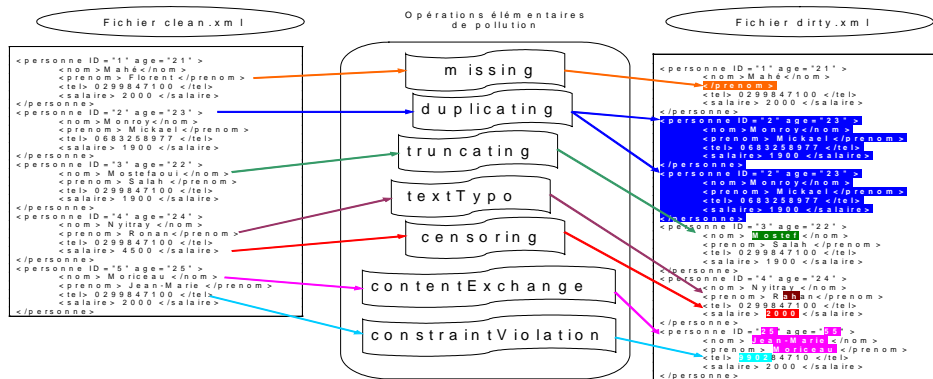


FIG. 1 - Exemple de pollution d'un document XML

A partir d'une hypothèse assez intuitive qui est que plus la durée de vie d'un document XML est longue et ses mises à jour nombreuses, plus le risque d'introduire des erreurs à chaque mise à jour est grand, nous avons défini un taux de pollution qui prend en compte la probabilité qu'un arbre XML soit (ou non) pollué et préalablement détecté (ou non) comme tel. Ce taux de pollution, défini formellement dans la section suivante, peut être utilisé à des fins de diagnostic par un classifieur pour estimer la probabilité que le document soit correct ou bien pollué. Dans le cas d'un document détecté pollué, le modèle de coût que nous proposons permet d'estimer le coût global d'un scénario de nettoyage selon le type de pollution. Afin d'évaluer expérimentalement notre approche, nous avons pollué artificiellement une collection de documents XML. Pour chaque nœud d'un arbre XML, nous calculons la distance de similarité existante entre ce nœud et sa version artificiellement polluée et l'identifions au taux de pollution du nœud considéré sous l'hypothèse qu'il n'y a pas d'erreur dans la classification (le nœud étant détecté pollué à juste titre). Ceci nous permet d'avoir une estimation globale du coût d'un scénario de nettoyage qui est par la suite raffiné selon le type de pollution et le coût des opérateurs de nettoyage nécessaires.

3 Modèle de coût

3.1 Définitions préliminaires

Définition 1. [Document XML]

Un document XML est défini comme un arbre $T = (N, E)$. L'ensemble des nœuds $N = NE \cup NA \cup NV$, où NE est l'ensemble des éléments, NA l'ensemble des attributs et NV est l'ensemble des valeurs d'éléments et d'attributs de type texte ou numérique. E est l'ensemble

des arcs reliant les noeuds, en particulier, l'arc $(u, v) \in E$ si et seulement si v est une valeur ou un fils de u . Un élément u est un fils de l'élément v si $(v, u) \in E$. Un élément u est le parent d'un noeud v si $(u, v) \in E$. Un noeud u est un ancêtre d'un noeud v s'il existe une série d'arcs qui relient u à v .

Définition 2. [Processus de nettoyage de données XML]

Un processus de nettoyage de données XML, noté C , est un quadruplet $\{M, B, O, S\}$, tel que M est le modèle de données XML (basé sur une DTD ou un schéma XSD), B est l'unité de base d'information (c'est-à-dire le contenu d'un noeud élément ou d'un noeud attribut), O est un ensemble d'opérateurs de nettoyage qui agit sur les instances du modèles de données. Chaque opérateur accepte en entrée une à plusieurs collections d'unités de base B et produit une collection d'unités de base en sorti. S est un ensemble possible de scénarii de nettoyage, c'est-à-dire un ensemble de listes d'opérateurs à utiliser pour le nettoyage.

Définition 3. [Document XML pollué]

Soit N un ensemble fini de noeuds dont le type est donné par le modèle de données M , soit P un ensemble fini d'estampilles de pollution. Un document XML pollué est un arbre $T_p = (Np, Ep)$, $p \in P$, où (i) l'ensemble des noeuds $Np \subseteq N$; (ii) l'ensemble des arcs $Ep \subseteq Np \times Np \times Np$ défini un arbre racine valide par rapport à M (c'est-à-dire satisfaisant le schéma imposé par le modèle de données entre les différents types de noeuds), ayant le quadruplet $(parent, left, flag^*(n, p), n)$ qui spécifie que le noeud n a le noeud "*parent*" pour parent et le noeud "*left*" pour noeud frère à gauche. Le prédicat $flag^*(n, p)$ est vrai si le noeud n contient directement ou indirectement une (ou plusieurs) fonction(s) de pollution notée p .

Définition 4. [Fonction de pollution]

Une fonction de pollution notée $p(T, nodeNumber, [minHeight, maxHeight, N, parameters])$ a pour paramètres l'arbre T , un nombre de noeuds à polluer (qui peut être un entier ou exprimé en pourcentage), et de façon optionnelle les profondeurs minimale ou maximale localisant la région où est appliquée la pollution dans l'arbre T ou encore l'ensemble des noms des noeuds ciblés (étiquettes) et un ensemble de paramètres plus spécifiques selon le type de pollution.

Définition 5. [Scénario de nettoyage]

Un scénario de nettoyage Sp consiste à appliquer un ensemble d'opérations à un ensemble de noeud pollués selon un type de pollution p afin d'obtenir un ensemble de noeuds majoritairement non pollués.

Notre modèle de coût se base sur le calcul d'une distance de similarité entre les noeuds du document XML sain et ceux de son correspondant artificiellement pollué. Cette distance est basée sur la distance *q-gram* entre les chaînes de caractères et elle est définie comme la norme L_1 de $G_q(x)[v] - G_q(y)[v]$ telle $D_q(x, y) = \sum_{v \in \Sigma^q} |G_q(x)[v] - G_q(y)[v]|$, avec le *q-gram* $v = a_1 a_2 \dots a_q \in \Sigma^q$ étant une chaîne de caractères dans l'alphabet fini Σ de longueur q et x, y deux chaînes de caractères quelconques dans l'alphabet Σ . $G_q(x)[v]$ (resp. $G_q(y)[v]$) représente le nombre d'occurrences du *q-gram* v dans x (resp. y). La distance *q-gram* consiste donc à mesurer le nombre de caractères non communs entre les deux chaînes de caractères en prenant une fenêtre d'observation de longueur q . Par exemple, prenons $q=2$,

Nettoyage de données XML

et $x = "clean"$ et $y = "clue"$, les 2-grams de x et y sont les suivants : (cl, le, ea, an) et (cl, lu, ue) , on obtient pour x , $G_2(x)[cl] = 1$, $G_2(x)[le] = 1$, $G_2(x)[ea] = 1$, $G_2(x)[an] = 1$ et $G_2(x)[v] = 0$ pour les autres 2-grams de x . En listant les 2-grams commençant par cl, le, ea, an, lu, ue dans cet ordre. On obtient un profil pour x tel que $(1, 1, 1, 1, 0, 0, \dots)$ et pour y tel que $(1, 0, 0, 0, 1, 1, 0, \dots)$ et la distance $D_2(x, y) = 5$.

Définition 6 [Distance entre contenu de deux nœuds XML] La distance entre le contenu de deux nœuds n_1 et n_2 est définie telle que :

$$contentDistance(n_1, n_2) = \begin{cases} \min\left(\frac{infoSize(n_1) + infoSize(n_2)}{2}, qdist(text(n_1), text(n_2))\right) & \text{pour } n_1 \text{ et } n_2 \text{ des nœuds de type texte} \\ \begin{cases} sametag(n_1, n_2) + \sum_C \min(qdist(val(n_1, a), val(n_2, a)), attrInfo(a)) + c_a D & \text{pour } n_1 \text{ et } n_2 \text{ des nœuds de type élément} \\ 1 & \text{pour les autres cas} \end{cases} \end{cases}$$

Les fonctions *infoSize* et *attrInfo* renvoient respectivement la longueur arrondie du contenu et des valeurs d'attribut d'un nœud et sont définies de la façon suivante :

$$infoSize(n) = \begin{cases} \max(|text(n)| - c_t, 1), \text{ avec } n, \text{ un nœud de type texte} \\ c_e + \sum_{attr(n,a)} (c_a + \max(|val(n, a)| - c_v, 1)), \text{ avec } n, \text{ un nœud de type élément} \end{cases}$$

$$attrInfo(a) = \min(|val(n_1, a)| - c_v, 1) + \min(|val(n_2, a)| - c_v, 1)$$

c_t et c_v sont des constantes seuils permettant de diminuer la contribution de la similarité des textes courts et des valeurs d'attributs par rapport aux autres. Par exemple, une valeur d'attribut de plus courte longueur que c_v sera traitée comme ayant une longueur de 1. c_a et c_e sont respectivement le contenu d'information correspondant au nom de l'attribut, ou au nom de l'élément. C est l'ensemble de tous les attributs de n_1 et n_2 . D est le nombre d'attributs de n_1 et n_2 non présents à la fois dans les deux nœuds. *sametag*(..) est une fonction qui retourne c_e dans le cas où les noms d'éléments sont égaux et 0 sinon. *val*(n, a) retourne la valeur de l'attribut a du nœud n , et *text*(n) retourne le contenu textuel du nœud n .

3.2 Paramètres du modèle de coût pour l'estimation globale

Suite à ces définitions, nous avons établi plusieurs paramètres permettant de modéliser, dans un premier temps, le coût global d'un nettoyage. Soient $P_p(n)$ représentant la probabilité qu'un nœud soit détecté pollué, $P_c(n)$ représentant la probabilité qu'un nœud soit détecté correct, $P_{cdp}(n)$ représentant la probabilité qu'un nœud soit détecté pollué alors qu'il est correct, $P_{pdc}(n)$ représentant la probabilité qu'un nœud soit détecté correct alors qu'il est pollué. On formule alors le taux de pollution de type p pour le nœud n tel que :

$$Pollution(p, n) = E\left[\frac{\min(Cost(U), Cost(M))}{Cost(M)}\right] P_{pdc} \int_{n_{clean}} P_p(n) dn \quad (1a)$$

$$+ E\left[\frac{\min(Cost(U) + Cost(S_p), Cost(M))}{Cost(M)}\right] P_{pdc} \left[1 - \int_{n_{clean}} P_p(n) dn\right] \quad (1b)$$

$$+ E\left[\frac{\min(Cost(U), Cost(M))}{Cost(M)}\right] P_{cdp} \left[1 - \int_{n_p} P_c(n) dn\right] \quad (1c)$$

$$+ E\left[\frac{\min(Cost(U) + Cost(D), Cost(M))}{Cost(M)}\right] P_{cdp} \int_{n_p} P_c(n) dn \quad (1d)$$

Le terme (1a) définit les nœuds pollués qui sont malencontreusement détectés corrects et nous nous intéressons à la probabilité qu'un nœud soit classifié pollué (P_p) lorsque les

nœuds ont toutes les caractéristiques de nœuds corrects (*Nclean*), d'où l'intégration sur cette surface. Le terme est pondéré par la probabilité qu'un nœud est détecté correct alors qu'il est en fait pollué (*Ppdc*) et par le temps de son traitement (représentant par un écart-type sur les temps de mise à jour et de maintenance). Généralement les temps de mise à jour (*U*), de détection (*D*) et de nettoyage de la pollution *p* (*Sp*) sont plus courts que la durée de vie totale du fichier et donc que le temps de sa maintenance (*M*). Le terme (1b) prend en compte les nœuds restant classifiés pollués, pondéré par la probabilité que le nœud soit détecté correct alors qu'il est pollué et par le temps de traitement (écart-type sur les temps de mise à jour, de nettoyage et de maintenance). Les termes (1c) et (1d) sont similaires. Le terme (1d) définit les nœuds corrects qui sont malencontreusement détectés pollués, pondéré par la probabilité qu'un nœud est détecté pollué alors qu'il est en fait correct (*Pcdp*) et par le temps de son traitement (écart-type sur les temps de mise à jour et de maintenance). Le terme (1d) prend en compte les nœuds restants classifiés corrects pondéré par la probabilité que le nœud soit détecté pollué alors qu'il est correct et par le temps de traitement incluant la mise à jour, la détection et la maintenance. Le taux de pollution peut être réécrit de la façon suivante :

$$Pollution(p, n) = A \cdot P_{pdc} \int_{N_{clean}} P_p(n) dn + B \cdot P_{cdp} \int_{Sp} P_c(n) dn$$

avec $A = E \left[\frac{\min(Cost(U), Cost(M))}{Cost(M)} \right] - E \left[\frac{\min(Cost(U) + Cost(S_p), Cost(M))}{Cost(M)} \right]$

$$+ E \left[\frac{\min(Cost(U) + Cost(S_p), Cost(M))}{Cost(M)} \right] P_{pdc}$$

$$+ E \left[\frac{\min(Cost(U), Cost(M))}{Cost(M)} \right] P_{cdp}$$

$B = E \left[\frac{\min(Cost(U) + Cost(D), Cost(M))}{Cost(M)} \right] - E \left[\frac{\min(Cost(U), Cost(M))}{Cost(M)} \right]$

Traditionnellement, dans la théorie de la classification statistique, les variables *A* et *B* représentent les coûts d'une mauvaise classification. Dans notre formule, ces quantités représentent des pourcentages de temps pendant lequel un nœud est classifié correct ou pollué. Ainsi, le taux de pollution peut être défini comme le pourcentage de temps d'une mauvaise classification avec des temps de traitement additionnels pour résoudre cette mauvaise classification. Pour minimiser le taux de pollution *Pollution(p, n)*, le mécanisme de détection doit classier le nœud *n* soit correct soit pollué selon la règle de décision suivante :

$$n \text{ est pollué par une pollution de type } p \text{ si } \frac{P_p(n) \cdot P_{pdc}(n)}{P_c(n) \cdot P_{cdp}(n)} \geq \frac{B}{A}$$

$$n \text{ est correct si } \frac{P_p(n) \cdot P_{pdc}(n)}{P_c(n) \cdot P_{cdp}(n)} < \frac{B}{A}$$

Dans nos expériences, connaissant *Cost(U)*, *Cost(M)* et la distance entre un nœud *n* et sa version artificiellement polluée *np* est identifiée à la probabilité de pollution du nœud, et nous évaluons *A* et *B* et déterminons *Cost(Sp)* et *Cost(D)* de façon à ce que soit vérifiée l'inégalité suivante :

$$contentDistance(n, np) \geq \frac{E \left[\frac{\min(Cost(U) + Cost(D), Cost(M))}{Cost(M)} \right] - E \left[\frac{\min(Cost(U), Cost(M))}{Cost(M)} \right]}{E \left[\frac{\min(Cost(U), Cost(M))}{Cost(M)} \right] - E \left[\frac{\min(Cost(U) + Cost(S_p), Cost(M))}{Cost(M)} \right]}$$

4 Conclusion

Cet article présente succinctement un travail en cours d'expérimentation qui consiste à proposer, implanter et valider un modèle permettant d'estimer globalement le coût probable d'un processus de nettoyage de documents XML. Pour cela, nous proposons de calculer des probabilités de pollution et de détection sur les différents types d'erreurs possibles sur les données (erreurs typographiques, ajout de doublons, de valeurs manquantes, etc.) qui préservent la validité structurelle des documents XML par rapport à leur modèle de données (DTD ou XSD). Selon les résultats des expériences de pollution artificielle en cours qui

permettent de corroborer notre modèle global, nous envisageons de raffiner notre approche au niveau du coût de chaque type d'opérateur de nettoyage, notamment en utilisant les historiques des scénarii de nettoyage sur des documents XML.

Références

- Caruso F., Cochinwala M., Ganapathy U., Lalk G. et Missier P. (2000), Telcordia's Database Reconciliation and Data Quality Analysis Tool, Conf. VLDB, 2000.
- Dasu T. et Johnson T. (2003), Exploratory Data Mining and Data Cleaning, Wiley, 2003.
- Elfeky M.G., Verykios V.S., et Elmagarmid A.K. (2002), Tailor: A Record Linkage Toolbox, Intl. ICDE Conf., 2002.
- Galhardas H., Florescu D., Shasha D., Simon E. et Saita C. (2001), Declarative Data Cleaning: Language, model and algorithms, Intl. Conf. VLDB, 2001.
- Low W.L., Lee M.L. et Ling T.W. (2001), A Knowledge-Based Approach for Duplicate Elimination in Data Cleaning, Information System, Vol. 26 (8), 2001.
- Rahm E. et Do H., (2000)Data Cleaning: Problems and Current Approaches. IEEE Data Eng. Bull., 23(4): 3-13, 2000.
- Raman V. et Hellerstein J. M. (2001), Potter's Wheel: An Interactive Data Cleaning System, Intl. Conf. VLDB, 2001.
- Vassiliadis P., Vagena Z., Skiadopoulos S. et Karayannidis N. (2003), ARKTOS: A Tool For Data Cleaning and Transformation in Data Warehouse Environments. IEEE Data Eng. Bull., 23(4): 42-47, 2000.
- Vassiliadis P., Simitsis A., Georgantas P. et Terrovitis M. (2003), A Framework for the Design of ETL Scenarios, Conf. on Advanced Information Systems Engineering (CAiSE '03), Klagenfurt, Austria, 2003.
- Weis M. et Naumann F. (2004), Detecting Duplicate Objects in XML Documents, ACM Workshop on Information Quality in Information Systems, 2004.
- Winkler W. (2003), Data Cleaning Methods, Intl. Conf. KDD, 2003.