Actes de la conférence BDA 2018

Adina Magda Florea, Dan Vodislav

BDA 2018

Gestion de Données – Principes, Technologies et Applications

34e conférence
22-26 octobre 2018
Université Politehnica de Bucarest

Actes de la conférence BDA 2018
Conférence soutenue par Inria, l'Université Cergy-Pontoise, l'ETIS, l'Université Politehnica de Bucarest, Qwant et le GdR MaDICS.

Site de la conférence : `https://bda2018.ensea.fr/`
Actes en ligne : `https://hal.inria.fr/BDA2018`

Message des organisateurs

La 34ème édition de la conférence sur la « Gestion de Données – Principes, Technologies et Applications » (BDA 2018) a eu lieu à l'Université Politehnica de Bucarest, en Roumanie, du 22 au 26 octobre 2018. La conférence, rendez-vous incontournable de la communauté francophone en gestion de données, a été organisée en première en Roumanie, en collaboration avec la Faculté d'Automatique et Informatique de l'Université Politehnica Bucarest. Ces actes regroupent les versions courtes des articles présentés lors de cette édition de la conférence.

La recherche en gestion de données n'a jamais été aussi active, variée, ouverte sur d'autres champs de l'informatique et, au-delà, sur les grands défis socio-économiques. L'omniprésence des données massives change en profondeur la manière dont les différentes phases du processus d'acquisition et de valorisation des données sont mises en œuvre. Ces données sont volumineuses, hétérogènes, incomplètes, imprécises, produites dynamiquement et avec divers degrés de structuration. L'exploitation de ce type de données par des applications issues de domaines scientifiques et métier très variés pose de nombreux défis pour la communauté de recherche en informatique tant sur le plan fondamental qu'appliqué. La généralisation des techniques d'apprentissage automatique dans l'exploitation des données massives apporte de nouveaux défis liés à la fois à l'accès efficace aux données pour l'apprentissage et à l'inclusion de ces techniques dans les processus classiques de traitement de données. Les technologies développées pour la gestion de données doivent également s'adapter à un environnement matériel et logiciel en perpétuelle mutation, incluant des infrastructures de calcul et de stockage, des architectures et des méthodes de traitement nouvelles, massivement parallèles et distribuées. La mobilité, les objets connectés, la production continue de données ou de messages sur les réseaux sociaux, les architectures distribuées et la virtualisation sont omniprésents et sont en train de révolutionner la manière de fournir et d'utiliser les systèmes informatiques.

Poursuivant la tradition de rencontres annuelles de la communauté de gestion de données francophone, BDA 2018 a invité des participants académiques et industriels à soumettre leurs travaux récents pour rendre compte des défis et des avancées scientifiques et industrielles dans ce domaine extrêmement dynamique.

Le programme scientifique a comporté 4 conférences invitées, ainsi que 24 présentations d'articles longs, 4 présentations d'articles courts, 8 démonstrations et 13 présentations de doctorants. Une originalité de la conférence BDA est de proposer deux catégories pour les articles : articles originaux et articles déjà acceptés ou publiés dans une conférence internationale de renom. Cette deuxième catégorie a pour vocation d'attirer les meilleurs papiers de l'année de la communauté française et de donner à leurs auteurs l'opportunité de présenter leurs travaux devant la communauté de recherche nationale.

Nous tenons à remercier tous les auteurs pour la qualité de leurs contributions, Camelia Constantin et le comité de sélection des démonstrations, ainsi que tous les membres du comité de programme de BDA 2018. Tout particulièrement nos remerciements vont vers l'Université Politehnica de Bucarest pour l'excellent accueil réservé à la conférence et vers toute l'équipe d'organisation des journées BDA 2018.

Adina Magda Florea, Université Politechnica Bucarest, Co-présidente des journées
Dan Vodislav, Université de Cergy-Pontoise, Co-président des journées et président du comité de programme

Table des matières

1 Comités de BDA 2018

Co-présidents des Journées
Adina Magda Florea (Université Politechnica Bucarest) & Dan Vodislav (Université de Cergy-Pontoise)

Président du comité de Programme
Dan Vodislav (Université de Cergy-Pontoise)

Président du comité des démonstrations
Camelia Constantin (Sorbonne Université)

Comité de Programme
Bernd Amann (Sorbonne Université)
Sihem Amer-Yahia (CNRS, LIG)
Nicolas Anciaux (Inria Saclay – Île-de-France)
Gabriel Antoniu (Inria Rennes – Bretagne Atlantique)
Ladjel Bellatreche (ENSMA)
Angela Bonifati (Université Lyon 1)
Pierre Bourhis (CNRS, CRIStAL)
Radu Ciucanu (Université Clermont Auvergne)
Sarah Cohen-Boulakia (Université Paris-Sud)
Alin Deutsch (Université de Californie San Diego)
Cedric Du Mouza (CNAM)
Francois Goasdoue (Université Rennes 1)
Daniela Grigori (Université Paris-Dauphine, Université PSL)
David Gross-Amblard (Université Rennes 1)
Abdelkader Hameurlain (Université Paul Sabatier)
Dimitris Kotzinos (Université de Cergy Pontoise)
Frederique Laforest (Université de Saint-Étienne)
Philippe Lamarre (INSA Lyon)
Sofian Maabout (Université de Bordeaux)
Silviu Maniu (Université Paris-Sud)
Ioana Manolescu (Inria Saclay – Île-de-France)
Florent Masseglia (Inria Sophia Antipolis – Méditerranée)
Mariana Mocanu (Université Politehnica de Bucharest)
Pascal Molli (Université de Nantes)
Amedeo Napoli (CNRS, LORIA)
Esther Pacitti (Université de Montpellier 2)
Themis Palpanas (Université Paris-Descartes)
Florin Radulescu (Université Politehnica de Bucharest)
Pierre Senellart (École normale supérieure, Université PSL)
Nicolae Tapus (Université Politehnica de Bucharest)
Genoveva Vargas Solar (CNRS, LIG & LAFMIA)
Karine Zeitouni (Université de Versailles-Saint-Quentin)

Comité des démonstrations

Hubert Naacke (Sorbonne Université)
Nicolas Travers (CNAM)
Romuald Thion (Université Lyon 1)
Florin Pop (Université Politehnica de Bucharest)
Fatiha Saïs (Université Paris-Sud)
Zoltan Miklos (Université de Rennes 1)
Ciprian Dobre (Université Politehnica de Bucharest)
Olivier Teste (Université Toulouse Jean Jaurès)

2 Conférenciers invités

2.1 Communication Cost in Parallel Query Evaluation
Dan Suciu

Bio : Dan Suciu is a Professor in Computer Science at the University of Washington. He received his Ph.D. from the University of Pennsylvania in 1995, was a principal member of the technical staff at AT&T Labs and joined the University of Washington in 2000. Suciu is conducting research in data management, with an emphasis on topics related to Big Data and data sharing, such as probabilistic data, data pricing, parallel data processing, data security. He is a co-author of two books Data on the Web : from Relations to Semistructured Data and XML, 1999, and Probabilistic Databases, 2011. He is a Fellow of the ACM, holds twelve US patents, received the best paper award in SIGMOD 2000 and ICDT 2013, the ACM PODS Alberto Mendelzon Test of Time Award in 2010 and in 2012, the 10 Year Most Influential Paper Award in ICDE 2013, the VLDB Ten Year Best Paper Award in 2014, and is a recipient of the NSF Career Award and of an Alfred P. Sloan Fellowship. Suciu is an associate editor for the Journal of the ACM, VLDB Journal, ACM TWEB, and Information Systems and is a past associate editor for ACM TODS and ACM TOIS. Suciu's PhD students Gerome Miklau, Christopher Re and Paris Koutris received the ACM SIGMOD Best Dissertation Award in 2006, 2010, and 2016 respectively, and Nilesh Dalvi was a runner up in 2008.

Abstract : What is the minimum amount of communication required to compute a query in parallel, on a cluster of servers ? In the simplest case when we join two relations without skewed data, then we can get away by simply reshuffling the data once. But if the data is skewed, or if we need to compute multiple joins, then it turns out that the total communication cost is significantly larger than the input data. In this talk I will describe a class of algorithms for which we can prove formally that their communication cost is optimal. I will end by describing several open questions, including a concrete queries where the optimal communication cost is unknown.

2.2 Learning Models over Relational Databases
Dan Olteanu

Bio : Dan Olteanu is Professor of Computer Science at the University of Oxford and Computer Scientist at RelationalAI. He received his BSc in Computer Science from Politehnica University of Bucharest in 2000 and his PhD from the University of Munich in 2005. He spends his time understanding hard computational challenges and designing simple and scalable solutions towards these challenges. He has published over 70 papers in the areas of database systems, AI, and theoretical computer science, contributing to XML query processing, incomplete information and probabilistic databases, factorised databases, scalable and incremental in-database optimisation, and the commercial systems LogicBlox and RelationalAI. He co-authored the book « Probabilistic Databases » (2011). He is the recipient of an ERC Consolidator grant (2016) and an Oxford Outstanding Teaching award (2009). He has served as member of over 60 programme committees, associate editor for PVLDB and IEEE TKDE, as track chair for IEEE ICDE'15, group leader for ACM SIGMOD'15, vice chair for ACM SIGMOD'17, and co-chair for AMW'18. He is currently serving as associate editor for ACM TODS. Eight of Olteanu's former students received the prize for the best thesis in their respective year and cohort at Oxford.

Abstract : In this talk I will overview recent results on learning classification and regression models over training datasets defined by feature extraction queries over relational databases. I will argue that the performance of the learning task can benefit tremendously from the sparsity in the relational data and the structure of the relational feature extraction queries. The mainstream approach to learning over relational data is to materialise the training dataset, export it out of the database system, and then import it into specialised statistical software packages. These three steps are very expensive and completely unnecessary.

I will instead argue for an in-database approach that avoids these steps by decomposing the learning task into aggregates and pushing them past the joins of the feature extraction queries. This approach comes with lower asymptotic complexity than the mainstream approach and several orders-of-magnitude speed-up over

state-of-the-art systems such as TensorFlow, R, and Scikit-learn whenever the latter do not exceed memory limitation or internal design limitations.

I will also highlight on-going work on linear algebra over databases and point out exciting directions for future work.

This work is based on long-standing collaboration with my PhD students Maximilian Schleich and Jakub Zavodny and more recent collaboration with Mahmoud Abo-Khamis and Hung Q. Ngo from RelationalAI and XuanLong Nguyen from University of Michigan.

2.3 Democracy Big Bang : What data management can(not) do for journalism
Ioana Manolescu

Bio : Ioana Manolescu is the lead of the CEDAR Inria team, focusing on rich data analytics at cloud scale. She is a member of the PVLDB Endowment Board of Trustees, and a co-president of the ACM SIGMOD Jim Gray PhD dissertation committee. Recently, she has been a general chair of the IEEE ICDE 2018 conference, an associate editor for PVLDB 2017 and 2018, and the program chair of SSDBBM 2016. She has co-authored more than 130 articles in international journals and conferences, and contributed to several books. Her main research interests include data models and algorithms for computational fact-checking, performance optimizations for semistructured data and the Semantic Web, and distributed architectures for complex large data.

Abstract : The tremendous power of Big Data has not been lost on journalists. As more and more human activity leaves electronic traces, or happens exclusively (or first) in an electronic manner, content management technologies such as databases, knowledge representation, information retrieval and natural language processing are increasingly called upon to help journalists automate and expedite their work.

In the last years, I have worked to understand the connections between existing (or missing!) algorithms and techniques, and real-world problems faced by actual journalists, doing their work, increasingly precariously due to the shift in advertising revenue, but so precious to the functioning of a modern society. I will discuss results obtained in this area together with colleagues, within the ANR ContentCheck project and the Inria-AIST joint team WebClaimExplain, then present a subjective list of open problems and a perspective on the future of data journalism and fact-checking as content management problems I believe our community should study.

2.4 Blockchain 2.0 : opportunities and risks
Patrick Valduriez

Bio : Patrick Valduriez is a senior scientist at Inria and LIRMM, University of Montpellier, France. He has also been a professor of computer science at University Pierre et Marie Curie (UPMC) in Paris and a researcher at Microelectronics and Computer Technology Corp. in Austin, Texas. He received his Ph. D. degree and Doctorat d'Etat in CS from UPMC in 1981 and 1985, respectively. He is the head of the Zenith team (between Inria and University of Montpellier, LIRMM) that focuses on data science, in particular data management in large-scale distributed and parallel systems and scientific data management. He has authored and co-authored many technical papers and several textbooks, among which "Principles of Distributed Database Systems". He currently serves as associate editor of several journals, including the VLDB Journal, Distributed and Parallel Databases, and Internet and Databases. He has served as PC chair of major conferences such as SIGMOD and VLDB. He was the general chair of SIGMOD04, EDBT08 and VLDB09. He obtained the best paper award at VLDB00. He was the recipient of the 1993 IBM scientific prize in Computer Science in France and the 2014 Innovation Award from Inria – French Academy of Science – Dassault Systems. He is an ACM Fellow.

Abstract : Popularized by bitcoin and other digital currencies, the blockchain has the potential to revolutionize our economic and social systems. Blockchain was invented for bitcoin to solve the double spending problem of previous digital currencies without the need of a trusted, central authority. The original

blockchain is a public, distributed ledger that can record and share transactions among a number of computers in a secure and permanent way. It is a complex distributed database infrastructure, combining several technologies such as P2P, data replication, consensus protocols and cryptography.

The term Blockchain 2.0 refers to new applications of the blockchain to go beyond transactions and enable exchange of assets without powerful intermediaries. Examples of applications are smart contracts, persistent digital ids, intellectual property rights, blogging, voting, reputation, etc. Blockchain 2.0 could dramatically cut down transaction costs, by automating operations and removing intermediaries. It could allow people to monetize their own information and creators of intellectual property to be properly compensated. The potential impact on society is also huge, as excluded people could join the global economy, e.g. by having digital bank accounts for free.

In this talk, I will introduce Blockchain 2.0 technologies and applications, and discuss the opportunities and risks. In developing countries, for instance, the lack of existing infrastructure and regulation may be a chance to embrace the blockchain revolution and leapfrog traditional solutions. But there are also risks, related to regulation, security, privacy, or integration with existing practice, which must be well understood and addressed.

Résumés des articles longs

Explanations and Transparency in Collaborative Workflows

Serge Abiteboul
Pierre Bourhis
Victor Vianu

ABSTRACT

We pursue an investigation of data-driven collaborative workflows. In the model, peers can access and update local data, causing side-effects on other peers' data. In this paper, we study means of explaining to a peer her local view of a global run, both at runtime and statically. We consider the notion of "scenario for a given peer" that is a subrun observationally equivalent to the original run for that peer. Because such a scenario can sometimes differ significantly from what happens in the actual run, thus providing a misleading explanation, we introduce and study a faithfulness requirement that ensures closer adherence to the global run. We show that there is a unique minimal faithful scenario, that explains what is happening in the global run by extracting only the portion relevant to the peer. With regard to static explanations, we consider the problem of synthesizing, for each peer, a "view program" whose runs generate exactly the peer's observations of the global runs. Assuming some conditions desirable in their own right, namely transparency and boundedness, we show that such a view program exists and can be synthesized. As an added benefit, the view program rules provide provenance information for the updates observed by the peer.

Maintenance Incrémentale du Skycube Négatif

Karim Alami
LaBRI, Université de Bordeaux
Talence, France
karim.alami@u-bordeaux.fr

Sofian Maabout
LaBRI, Université de Bordeaux
Talence, France
maabout@u-bordeaux.fr

ABSTRACT

Soient $T(Id, D_1, \ldots, D_d)$ une table $X \subseteq \{D_1, \ldots, D_d\}$ un sous-ensemble de dimensions, ou sous-espace, $Sky(T, X)$ dénote le skyline de T vis à vis de X, i.e., l'ensemble des enregistrements de T qui ne sont pas dominés respectivement à X. Pour T, il y a $2^d - 1$ requêtes skylines possibles en fonction du X choisi. Pour optimiser toutes ces requêtes, une manière de procéder consiste à les pré-calculer toutes et les stocker. Ce calcul est coûteux en termes de temps et d'espace mémoire. Un travail antérieur a proposé le skycube négatif (NSC) qui est une structure qui stocke pour chaque enregistrement et d'une manière compressée, l'ensemble des sous-espaces où il est dominé. L'efficacité de cette structures en termes de temps de calcul, d'espace de stockage et d'optimisation des requêtes a déjà été établie. Dans le présent article, nous montrons comment mettre à jour le NSC sans avoir à le recalculer entièrement suite à une insertion/suppression d'un (ensemble de) tuple(s).

1 INTRODUCTION

Depuis son introduction à la communauté gestion de données dans [2], la requête skyline a suscité un large intérêt et différents aspects ont été traités dans la littérature, e.g., [1, 4–7]. Les requêtes skyline *multi-dimensionnelles* en représentent un cas particulier. Ici, on considère que les utilisateurs ont la possibilité de choisir la combinaison d'attributs sur lesquels l'évaluation de la requête va se baser. /

Exemple 1. *Soit la relation* Hotel *dont l'instance est illustrée dans Fig. 1. Les hôtels non dominés respectivement à $X = PDE$ sont h_1, h_2 et h_3. h_4 est dominé car, e.g., h_2 est meilleur que lui selon les trois critères P, D et E. Un utilisateur qui a suffisament de moyens, peut s'affranchir du critère P et ne considérer que D et E pour chercher les meilleurs hôtels alors qu'inversement, un étudiant qui a peu de moyens ne considèrera que le critère P. Dans ce dernier cas, h_3 est le meilleur alors que selon DS, h_2 et h_3 sont les meilleurs. La figure 2 montre les différents résultats en fonction des critères choisis : Noter que dans l'exemple ci-dessus, on cherche à minimiser le prix et la distance alors que l'on veut maximiser le nombre d'étoiles.*

Id	*(P)rix*	*(D)istance APlage*	*(E)oiles*
h_1	*100*	*200*	*4*
h_2	*120*	*150*	*2*
h_3	*80*	*220*	*4*
h_4	*250*	*200*	*1*

Figure 1: Hotel **relation**

Sous-espaces	*Skyline*	*Sous-espace*	*Skyline*
PDE	h_1, h_2, h_3	*PD*	h_2, h_3
PE	h_1, h_2, h_3	*DE*	h_1, h_2
P	h_3	*D*	h_2
E	h_1, h_3		

Figure 2: Le Skycube de la table Hotel

2 LE SKYCUBE NÉGATIF (NSC)

La structure NSC présentée dans [3] se base sur la remarque suivante : Pour un enregistrement t_1, rien qu'en le comparant à un autre tuple t_2, on déduit un *sous-ensemble* de sous-espaces où t_1 est dominé. En reprenant l'exemple 1 et en comparant h_4 à h_1, on obtient la *paire* $\langle PE|D\rangle$ signifiant que h_1 est meilleur que h_4 en P et E alors qu'ils sont égaux en D. De cette paire, on déduit que que h_4 ne fait partie d'aucun skyline relatif à un sous espace X tel que (i) $X \subseteq (PE \cup D)$ et (ii) $X \cap PE \neq \emptyset$.

En comparant donc chaque enregistrement t à tous les autres, on obtient un ensemble de paires, noté $Paires(t)$. Ce dernier encode tous les espaces où t est dominé. Ainsi, vérifier si $t \in Sky(T, X)$ revient à vérifier s'il n'existe pas $p = \langle Y|Z\rangle \in Paires(t)$ telle que $X \subseteq YZ$ et $X \cap Y \neq \emptyset$.

L'ensemble $Paires(t)$ peut être compressé en éliminant les paires superflues. Par exemple, soit $Paires(t) = \{p_1 : \langle AB|C\rangle, p_2 : \langle C|B\rangle, p_3 : \langle C|D\rangle\}$. On peut observer que p_2 peut être supprimée de $Paires(t)$ sans perte d'information. En effet, p_2 couvre $\{CB, C\}$. CB est couvert par p_1 alors que C est couvert par p_3.

Obtenir un sous-ensemble Q de $Paires(t)$ qui lui soit équivalent et de taille minimal a été montré NP-difficile [3] et un algorithme glouton avec garantie de l'approximation y a été proposé. Remarquer que la minimisation de $Paires(t)$ réduit à la fois l'espace de stockage de NSC et le temps de l'évaluation des requêtes.

3 MISE À JOUR DU NSC

Quand un enregistrement est ajouté ou supprimé, il faut mettre à jour le NSC correspondant. Nous commençons d'abord par rappeler la propriété suivante : pour obtenir $Paires(t)$ il suffit de comparer t non pas à *tous* les t' mais seulment à ceux qui sont dans le *topmost* skyline, i.e., le skyline respectivement à *toutes* les dimensions. Ainsi, on suppose que toute $p \in Paires(t)$ est obtenu en comparant t à un élément de *topmost*.

Nous présentons notre solution pour la prise en compte de la suppression et on abordera ensuite l'insertion.

3.1 Suppression

En se basant sur la propriété ci-dessus, on peut d'ores et déjà enoncer le fait suivant :

— Si l'on supprime t et $t \notin topmost$ alors pour chaque t', $Paires(t')$ n'a pas besoin d'être modifiée.

Dans le cas où $t \in topmost$, deux cas de figure peuvent se présenter pour t' selon que la paire obtenue en comparant t' à t, notons la $comp(t', t)$, soit dans $Paires(t')$ ou pas :

— Si $comp(t', t) \notin Paires(t')$ alors la suppression de t n'a aucun effet sur $Paires(t')$.

— Si par contre $comp(t', t) \in Paires(t')$ alors il faut mettre à jours cette dernière.

Dans le dernier cas, il se peut qu'il existe un $t'' \in topmost$ tel que $comp(t', t'') = comp(t', t)$. Dans ce cas aussi, $Paires(t')$ n'a pas à être modifiée. Pour identifier facilement cette situation, on ajoute à chaque paire un compteur indiquant le nombre de tuples du topmost qui ont permis de l'obtenir. Ainsi, lors de la suppression de t, on décrémente le compteur de $comp(t', t)$ et ce n'est que si ce dernier passe à 0 que la mise à jour de $Paires(t')$ devient nécessaire.

3.2 Insertion

Toujours en se basant sur la propriété du *Topmost*, on peut dire que :

— En insérant t, si $t \notin Topmost$, alors $Paires(t')$ n'a pas à être modifié quel que soit t'.

Si t est dans le nouveau $Topmost$ alors non seulement il faut ajouter $comp(t', t)$ à $Paires(t)$ mais en plus t' peut dominer des éléments t'' de l'ancien $Topmost$ et donc plus besoin de garder $comp(t', t'')$ dans $Paires(t'')$. Noter néanmoins que le fait de garder ces paires ne remet pas en cause la correction de NSC c'est juste dans un soucis de réduction de l'espace de stockage autant que possible. Il est important de noter ici que la suppression d'une paire obtenue suite à une comparison avec un t'' qui n'est plus dans le $Topmost$ n'engendre pas la ré-introduction de nouvelles paires qui auraient été supprimées lors de la compression sémantique effectuée par l'algorithme glouton.

4 EVALUATION EXPÉRIMENTALE

Dans cette section, nous montrons quelques uns des résultats expérimentaux que nous obtenons avec notre méthode. Pour ce faire, et faute de place, nous ne présentons que les résultats obtenus avec des données synthétiques (anti-corrélées). Comme solution de base pour la mise à jour, nous avons sélectionné celle qui consiste à reconstruire le NSC à chaque mise à jour. Aussi, nous distinguons deux méthodes de mise à jour : (i) séquentielle, qui itère une même procédure sur un ensemble de tuples insérés/supprimés et (ii) batch, qui fait d'abord un taitement de l'ensemble des insertions/suppressions puis met à jour NSC .

On considère une table T avec un nombre de dimensions (attributs) $d = 14$ dont le NSC correspondant est déjà construit et on mesure le temps nécessaire pour prendre en compte l'insertion (resp. suppression) d'un ensemble de tuples Δ^+ (resp. $\Delta-$. On fait varier à la fois la taille de T ainsi que celle de Δ^+/Δ^-.

Ce que l'on peut observer à partir des Figures 3 et 4 et que la mise à jour en Batch est globalement 100 fois plus rapide que la reconstruction. On note aussi que quand le nombre de tuples insérés/supprimés est petit (une dizaine), la méthode séquentielle est la plus performante.

Nous avons effectué les mêmes expériences avec d'autres types de données (corrélées et indépendantes) et on arrive aux mêmes conclusions. Aussi, le facteur d a son importance car plus d est grand et plus le speed-up de notre approche par rapport à la reconstruction est grand.

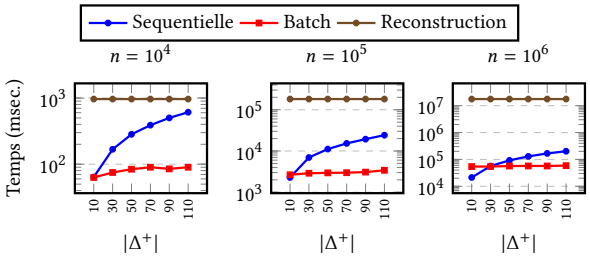

Figure 3: Insertion de Δ^+ en variant $n = |T|$

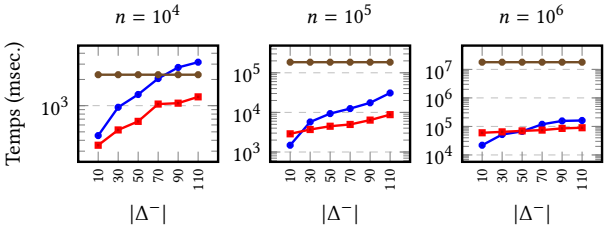

Figure 4: Suppression de Δ^- en variant $n = |T|$

5 CONCLUSION

Nous avons présenté dans cet article une solution pour la prise en compte efficace des mises à jours d'une relation afin de les répercuter sur la structure d'indexation NSC permettant de répondre efficacement aux requêtes skyline. Les expériences montrent que notre solution est largement plus rapide que celle sui consiste à reconstruire le NSC ce qui lui confère le caractère incrémental et qui rend NSC réellement utilisable.

RÉFÉRENCES

[1] K. S. Bøgh, S. Chester, and I. Assent. Skyalign : a portable, work-efficient skyline algorithm for multicore and GPU architectures. *VLDB Journal*, 25(6) :817–841, 2016.

[2] S. Börzsönyi, D. Kossmann, and K. Stocker. The skyline operator. In *Proc. of ICDE conf.*, pages 421–430, 2001.

[3] N. Hanusse, P. Kamnang-Wanko, and S. Maabout. Computing and summarizing the negative skycube. In *Proc. of CIKM Conference*, pages 1733–1742, 2016.

[4] K. Hose and A. Vlachou. A survey of skyline processing in highly distributed environments. *VLDB Journal*, 21(3) :359–384, 2012.

[5] J. Liu, L. Xiong, J. Pei, J. Luo, and H. Zhang. Finding pareto optimal groups : Group-based skyline. *PVLDB*, 8(13) :2086–2097, 2015.

[6] Y. Park, J. Min, and K. Shim. Efficient processing of skyline queries using mapreduce. *IEEE Trans. Knowl. Data Eng.*, 29(5) :1031–1044, 2017.

[7] J. Pei, B. Jiang, X. Lin, and Y. Yuan. Probabilistic skylines on uncertain data. In *Proceedings of the 33rd International Conference on Very Large Data Bases, University of Vienna, Austria, September 23-27, 2007*, pages 15–26, 2007.

Skyline Multi-dimensionnelle sur des Données en Flux

Karim Alami
LaBRI, Université de Bordeaux
Talence, France
karim.alami@u-bordeaux.fr

Sofian Maabout
LaBRI, Université de Bordeaux
Talence, France
maabout@u-bordeaux.fr

1 RÉSUMÉ

Dans le présent papier, nous proposons une structure d'indexation pour optimiser les requêtes skyline multi-dimensionnelles dans un contexte de données en flux. Quelques travaux ont traité le problème de la maintenance du skyline dans ce contexte, toutefois aucun n'a considéré le cas de skyline par rapport à un sous-ensemble de dimensions. Soient $D = \{D_1, \ldots, D_d\}$ un ensemble de dimensions, $T(id, D)$ un flux de données et ω la taille de la fenêtre de temps glissante, i.e., la durée de vie d'un enregistrement, notre structure répond à des requêtes de la forme $sky(T, X, k)$ skyline de T vis à vis du sous-espace X choisi et $k \leq \omega$ permet de restreindre la requête aux k flux les plus récents.

2 INTRODUCTION

L'opérateur skyline [1] est pertinent pour récupérer les meilleurs éléments d'un ensemble d'éléments, i.e., ceux qui ne sont pas dominés sur tous les critères.

Dans cet article, on traite le problème suivant : Etand donnés $D = \{D_1, \ldots, D_d\}$ un ensemble de dimensions et $T(Id, D_1, \ldots, D_d)$ une table continuellement étendue par un ensemble de données δ. On assume que tous les éléments ont une même *durée de vie* ω, qui dénote aussi la taille de la fenêtre de temps glissante. Maintenant, étant donné un sous-espace $X \subseteq \{D_1, \ldots, D_d\}$, on veut calculer le skyline par rapport à X en ne prenant en compte que les k flux les plus récents tel que $k \leq \omega$.

L'exemple suivant illustre le contexte que l'on étudie.

Example 1. *On considère l'ensemble de données dans le tableau 1, la dernière colonne représentant l'heure d'arrivée. On peut voir T comme une séquence de 5 transactions δ_i, dont chacune a inséré un ensemble d'éléments à l'instant i.*

Soit $\omega = 3$, notre système permet de répondre à une requête qui inclut les k flux les plus récents tel que $k \in \{1, 2, 3\}$.

Notez que pour chaque dimension, on assume que les valeurs les plus petites sont préférées.

Prenons le sous-espace AB et $k = 3$ alors $Sky(T, AB, 3) = \{r_5, r_6\}$. Aussi pour le sous-espace BC et $k = 2$ $Sky(T, BC, 3) = \{r_7, r_8\}$

A notre connaissance [4] a été le premier papier à traiter la problématique de skyline avec un flux de données. Cependant, aucun travail n'a considéré la requête par rapport à un sous-ensemble de dimension. D'un autre côté, les travaux qui traitent du problème de requêtes skyline multidimensionnelles, ne les traitent que dans un environnement où les données sont statiques. La structure qu'on propose dans le présent papier est basée sur la structure *NSC* présentée dans le papier [2].

Id	*A*	*B*	*C*	*heure arr.*
r_1	2	2	2	0
r_2	4	2	3	0
r_3	5	1	3	1
r_4	1	1	3	1
r_5	1	0	4	2
r_6	0	1	5	2
r_7	2	0	3	3
r_8	2	1	1	3
r_9	6	6	6	4

Table 1: T

3 STRUCTURE DE DONNÉES POUR LA REQUÊTE SKYLINE SUR DES DONNÉES EN FLUX

La solution qu'on propose se base sur la structure d'indexation présentée dans le papier [2] et qui consiste à : pour chaque enregistrement r, stocker les sous-espaces où r est dominé en forme de *paires de sous-espaces* en le comparant avec tous les autres enregistrements, e.g. en comparant r_2 à r_1 (voir table 1), on garde l'information $\langle AC|B\rangle$ tel que AC est le sous-espace où r_1 est strictement meilleur que r_2 et B est l'espace où ils sont égaux. En comparant un enregistrement r à tous les autres enregistrements de l'ensemble des données, on obtient un ensemble de paires $Paires(r)$. De cet ensemble, pour vérfier si r appartient au skyline par rapport à un sous-espace, il suffit de calculer la couverture de chaque paire dans $Paires(r)$ de la façon suivante : soit X, Y des sous-espaces de D, $couvert(\langle X|Y\rangle) = \{Z \in D | Z \subseteq XY \wedge X \cap Y \neq \emptyset\}$. Il est intéressant de noter que les paires d'un enregistrement sont organisées en liste d'ensembles de paires tel que les paires calculées avec un ensemble d'enregistrements arrivés au même moment sont rassemblées dans le même ensemble.

3.1 Traitement d'un Enregistrement à son Insertion

Dans cette section, nous expliquons comment nous calculons les paires d'un enregistrement récemment ajouté.

L'algorithme 1 décrit le calcul des paires d'un enregistrement à son arrivée. Pour un enregistrement r récemment ajouté, nous calculons une paire p avec chaque enregistrement r' actif, i.e. la différence entre l'heure d'arrivée $arr(r')$ et l'instant actuel t_c est plus petite que ω, puis la paire est placée dans $Paires(r, arr(r'))$ (ligne 3-5).

Algorithm 1: computePairs

Input: record r, T
Output: Paires(r)

```
Paires(r) ← ∅
begin
    foreach j ∈ [max(0, t_c − ω + 1), t_c] do
        foreach r' ∈ T ∧ (arr(r') = j) do
            Paires(r, j)∪ ← compare(r, r')
return Paires(r)
```

Compression de Paires(r). L'ensemble des paires $Paires(r)$ peut être réduit sans perte d'informations.

Example 2. *La table 2 représente les paires de r*5.

$Paires(r_5, 0)$	$Paires(r_5, 1)$	$Paires(r_5, 2)$
$\langle C\|\emptyset\rangle$	$\langle C\|\emptyset\rangle$	$\langle A\|\emptyset\rangle$
$\langle C\|\emptyset\rangle$	$\langle C\|A\rangle$	

Table 2: Paires de r_5

Observez que $couvert(\langle C|\emptyset\rangle) \subset couvert(\langle C|A\rangle)$ *dans* $Paires(r_5, 1)$. *Cela signifie que nous pouvons ignorer* $\langle C|\emptyset\rangle$ *à partir de* $Paires(r_5, 1)$ *et que nous ne perdons pas les informations de couverture de* $Paires(r_5, 1)$.

Aussi si $k = 1$, *nous calculons les ensembles couverts par les paires dans* $Paires(r_5, 2)$. *Si* $k = 2$, *nous calculons la couverture des paires dans* $(r_5, 1)$ *et* $Paires(r_5, 2)$. *De même, si* $k = 3$, *nous calculons la couverture des paires dans les trois ensembles de* $Pair(r_5)$. *Observez que* $couvert(Paires(r_5, 0)) \subset cover(Paires(r_5, 1))$ *par conséquent, le fait de supprimer tout le contenu de* $Paires(r_5, 0)$ *ne modifie pas le résultat d'une requête où* $k = 3$.

Ainsi les paires conservées sont $\langle C|A\rangle$ *dans* $Paires(r_5, 1)$ *et* $\langle A|\emptyset\rangle$ *dans* $Paires(r_5, 2)$

Nous formulons cette étape dans le problème suivant :

Problème 1. Etant donnée une liste d'ensemble de paires $Paires(r)$, $\forall j \in [max(0, arr(r) - \omega + 1), arr(r)]$, trouver $s(j) \subseteq Paires(r, j)$ t.q. $cover(s(j) \cup Paires(r, j+1)) = cover(Paires(r, j) \cup Paires(r, j+1))$ et $s(j)$ est de taille minimum.

Theorem 1. *Problème 1 est NP-difficile.*

3.2 Mise à Jour des Paires d'un Enregistrement

Dans cette section, nous expliquons la procéudre de mise à jour des paires d'un enregistrement r lorsque de nouveaux enregistrements sont ajoutés et que certains existants expirent.

Mise à jour des paires par rapport aux enregistrements expirés. A l'insertion de nouveaux enregistrements, la transaction la plus ancienne arrivée à $t_c - \omega$ est supprimée. Pour tous les enregistrements toujours actifs, les paires calculées avec les enregistrements expirés sont également supprimées. Ces paires sont situées dans $Paires(r, t_c - \omega)$.

Mise à jour des paires par rapport aux enregistrements récemment ajoutés. Pout tout enregistrement r toujours actif, on le compare avec les enregistrements récemment ajoutés et on stocke les paires dans l'ensemble des paires les plus récentes. S'en suit la procédure de compression des nouvelles paires.

4 RÉSULTATS EXPÉRIMENTAUX

Dans cette section, nous allons montrer quelques résultats expérimentaux qui démontrent les performances de notre solution à répondre aux requêtes skyline multi-dimensionnelles dans un contexte de données en flux. Nous comparons notre solution avec l'algorithme de l'état de l'art *BSkyTree*[3]. Faute de place, nous ne présentons que les résultats obtenus avec des données synthétiques indépendantes. Dans la figure 1 on évalue le nombre de requêtes qui peuvent être traitées entre deux transactions successives par *NSCt* ou *BSkyTree*. On considère un flux de données indépendantes avec 12 dimensions, une taille maximale de 10000 enregistrements et des transactions qui insèrent 1000 enregistrements à chaque cycle. On voit que globalement, *NSCt* évalue beaucoup plus de requêtes que *BSkyTree*($\approx \times 100$), même en prenant en compte le temps de mise à jour de *NSCt*. Aussi, l'écart se creuse en augmentant l'intervalle de temps entre deux transactions, ce qui montre que plus l'écart est grand, plus c'est intéressant d'utiliser *NSCt* dans un contexte de données en flux.

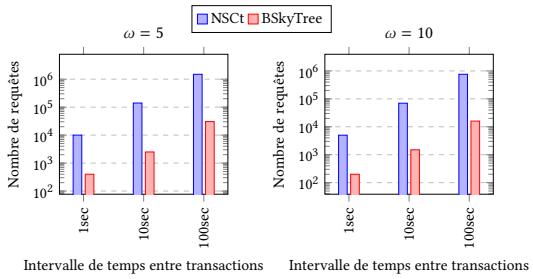

Figure 1: Nombre de requêtes évaluées avec des données indépendantes, $n = 10^4$, $d = 12$ et $|\delta| = 10^3$

5 CONCLUSION

Nous avons proposé une structure de données ainsi qu'une procédure efficace pour sa maintenance dans le but d'accélérer le temps d'évaluation d'une requête skyline multi-dimensionnelle sur des données en flux. À l'avenir, nous prévoyons d'étudier une version distribuée de notre solution.

RÉFÉRENCES

[1] S. Börzsönyi, D. Kossmann, and K. Stocker. The skyline operator. In *Proc. of ICDE conf.*, pages 421–430, 2001.
[2] N. Hanusse, P. Kamnang-Wanko, and S. Maabout. Computing and summarizing the negative skycube. In *Proc. of CIKM Conference*, pages 1733–1742, 2016.
[3] J. Lee and S. won Hwang. BSkyTree : scalable skyline computation using a balanced pivot selection. In *Proc. of EDBT conf.*, 2010.
[4] Y. Tao and D. Papadias. Maintaining sliding window skylines on data streams. *IEEE Transactions on Knowledge and Data Engineering*, 18(3) :377–391, 2006.

Liens entre largeur et structure en compilation de connaissances

Antoine Amarilli
Mikael Monet
Pierre Senellart

ABSTRACT

De nombreuses tâches d'évaluation de requêtes peuvent être vues comme de la compilation de connaissances : le résultat de la requête est compilé comme un circuit de lignage qui permet de calculer la réponse. Pour de telles tâches, il est important de pouvoir borner certains paramètres de largeur du circuit en question, par exemple la largeur d'arbre ou la largeur linéaire, et ainsi de convertir le circuit vers des classes structurées, par exemple les NNF déterministes structurées (d-SDNNF) ou les OBDD. Dans cet article, nous montrons comment établir des liens entre la largeur des circuits et la taille de leurs représentations structurées, en prouvant des bornes inférieures et des bornes supérieures. Notre borne supérieure montre comment nous pouvons convertir des circuits de largeur d'arbre bornée en d-SDNNF, en temps linéaire en la taille du circuit. Contrairement aux résultats existants, notre borne est constructive et sa dépendance en la largeur d'arbre n'est que simplement exponentielle. Nous montrons également une borne inférieure qui porte sur les formules monotones en DNF ou CNF, lorsque l'on fait l'hypothèse d'une borne constante sur l'arité (la taille des clauses) et le degré (le nombre d'occurrences de chaque variable) : nous établissons que toute d-SDNNF (respectivement, SDNNF) pour une telle DNF (respectivement, CNF) doit être de taille exponentielle en sa largeur d'arbre. Nous montrons le même résultat pour la largeur linéaire pour une compilation vers les OBDD. Ces bornes inférieures, contrairement à la plupart des résultats existants, s'appliquent à *toute* formule de la classe concernée, et pas seulement à une famille ad hoc de telles formules. Ainsi, pour notre langage de DNF et de CNF, la largeur linéaire et la largeur d'arbre caractérisent respectivement l'efficacité de la compilation vers les OBDD et vers les (d-)SDNNF : la compilation est simplement exponentielle en le paramètre de largeur. Nous concluons par une application de nos bornes inférieures à une tâche d'évaluation de requêtes.

Énumération sur les arbres avec support des réétiquetages

Antoine Amarilli
Pierre Bourhis
Stefan Mengel

ABSTRACT

Nous étudions l'évaluation sur les arbres de requêtes MSO avec des variables libres, suivant le paradigme des algorithmes d'énumération. Des travaux antérieurs ont montré comment énumérer les réponses à de telles requêtes avec un prétraitement en temps linéaire et avec un délai linéaire en la taille de chaque réponse, c'est-à-dire avec un délai constant lorsque les variables libres sont du premier ordre. Nous étendons ce résultat afin de pouvoir également appliquer des opérations de *réétiquetage* qui permettent de mettre à jour l'étiquette des nœuds de l'arbre d'entrée. Notre résultat principal montre que nous pouvons énumérer les résultats de requêtes MSO sur des arbres, avec un prétraitement en temps linéaire et avec un délai linéaire en la taille de chaque réponse, mais également appliquer de telles opérations de réétiquetage en temps logarithmique. Pour démontrer ce résultat, nous réutilisons la structure d'énumération à base de circuits que nous avions développée dans nos travaux précédents, et nous développons des techniques pour maintenir son index à jour sous ces opérations de réétiquetage. Nous montrons également que l'énumération sous réétiquetages peut en particulier être appliquée à certains langages de requêtes inspirés par des applications pratiques : nous considérons en particulier les requêtes aggrégatives, les requêtes avec opérateur de groupe, et les requêtes paramétrées. Cet article a déjà été publié (catégorie "P") dans les actes de la conférence ICDT'18 [**?**].

Rewriting-Based Query Answering for Semantic Data Integration Systems

Maxime Buron
François Goasdoué
Ioana Manolescu
Marie-Laure Mugnier

ABSTRACT

We consider the integration of heterogeneous data under a global RDF graph data model, based on the knowledge expressed in an ontology describing the concepts relevant for an application, and a set of entailment rules which characterize the logical relationships between these concepts.We consider both standard RDF Schema rules and user-specified ones. We propose a data integration architecture to compute certain query answers in this setting. Existing approaches to query answering in the presence of knowledge (expressed here in the ontology and the entailment rules) involve either the materialization of inferences in the data or the reformulation of the query. Both approaches have well-known drawbacks. We introduce a new approach to query answering, based on an reduction to view-based query answering.This approach avoids both materialization in the data and query reformulation. We define restrictions of our general architecture under which our method is correct, and formally prove its correctness.

Extracting Linked Data from statistic spreadsheets

Tien-Duc Cao
Ioana Manolescu
Xavier Tannier

ABSTRACT

Fact-checking journalists typically check the accuracy of a claim against some trusted data source. Statistic databases such as those compiled by state agencies or by reputed international organizations are often used as trusted data sources, as they contain valuable, high-quality information. However, their usability is limited when they are shared in a format such as HTML or spreadsheets: this makes it hard to find the most relevant dataset for checking a specific claim, or to quickly extract from a dataset the best answer to a given query. We provide a conceptual model for the open data comprised in statistics published by INSEE, the national French economic and societal statistics institute. Then, we describe a novel method for extracting RDF Linked Open Data, to populate an instance of this model. We used our method to produce RDF data out of 20k+ Excel spreadsheets, and our validation indicates a 91% rate of successful extraction. We also present a novel algorithm enabling the exploitation of such statistic tables, by (*i*) identifying the statistic datasets most relevant for a given fact-checking query, and (*ii*) extracting from each dataset the best specific (precise) query answer it may contain. We have implemented our approach and experimented on the complete corpus of statistics obtained from INSEE, the French national statistic institute. Our experiments and comparisons demonstrate the effectiveness of our proposed method.

Discovering Complex Temporal Dependencies between heterogenous sensor data streams

Amine El Ouassouli
Lionel Robinault
Marian Scuturici

ABSTRACT

In addition to sensor heterogeneity, the monitoring applications must handle different temporal data models (e.g time series, event sequences). In this paper, we address the problem of discovering directly actionable high level knowledge from such data. We model temporal information through interval-based streams describing environment states. We propose an approach to discover efficiently Complex Temporal Dependencies (CTD) between state streams, called CTD-Miner. A CTD is modeled similarly to a conjunctive normal form and maintains temporal information (time delays) between states. CTD-Miner is robust to temporal variability of data and uses chi-2 independence tests to determine the most appropriate time lags between states. This testis also used to perform pruning of trivial dependencies according to information gathered during the search space exploration.Finally, we validate our approach via synthetic data and a case study in a real-world smart environment context using video cameras.

Reducing Filter Bubbles With a Community Aware Model

Quentin Grossetti
Camelia Constantin
Cedric Du Mouza
Nicolas Travers

ABSTRACT

After facing tremendous growth, main social network platforms took an important place in modern societies. In order to increase user engagement, they rely heavily on recommender systems and fears arise that personalizing content might trap users into a filter bubble exacerbating divisions within societies. After a thorough study of the detected communities in a large Twitter dataset and their role in information propagation, we present a metric to compute the strength of the filter bubble. Our results show that filter bubble effects are in fact limited for a majority of users. We find that, counterintuitively, in most cases recommender systems tend to open users perspectives. However, for some specific users, the bubble effect is noticeable and we propose a model relying on communities to provide a list of recommendations closer to the user's usage of the platform.

Une Algèbre de Motifs pour l'Évaluation et l'Analyse de la Complétude des Données et l'Éxactitude des Requêtes

Fatma Zohra Hannou
Bernd Amann
Mohamed-Amine Baazizi
firstname.lastname@lip6.fr
Sorbonne Université, CNRS, Laboratoire d'Informatique de Paris 6, F-75005 Paris, France

ACM Reference format:
Fatma Zohra Hannou, Bernd Amann, and Mohamed-Amine Baazizi. 2019. Une Algèbre de Motifs pour l'Évaluation et l'Analyse de la Complétude des Données et l'Éxactitude des Requêtes . In *Proceedings of ACM Conference, Washington, DC, USA, July 2017 (Conference'17),* 2 pages.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Introduction

Les données manquantes sont un problème majeur dans les environnements informatiques distribués, ouverts et peu fiables. Dans les entrepôts de données, les informations sont souvent incomplètes à cause de défaillances matérielles ou logicielles, d'incompatibilités de données, d'autorisations d'accès aux données sources, etc. Dans toutes ces situations, l'interrogation et l'agrégation de données incomplètes peuvent conduire à des réponses manquantes et incorrectes. Nous proposons une nouvelle approche où la complétude des données est évaluée par rapport à des données de référence et résumée à l'aide de motifs. Nous montrons qu'il est possible de générer des ensembles de motifs automatiquement à partir de données de référence et d'utiliser ces ensembles pour raisonner sur la qualité des réponses aux requêtes sur des données incomplètes.

Travaux connexes

Le travail précurseur de [Mot89] suggère un modèle fondé sur des méta-tables décrivant des contraintes de complétude et d'exactitude sur les données. Les méta-tables sont similaires à nos tables de motif, où les nuplets servent à définir les données disponibles, valides et non valides. La complétude d'une requête est vérifiée en montrant l'existence d'une réécriture de la requête avec des vues complètes. Une autre idée de ces travaux est la définition d'une algèbre manipulant des méta-tables pour produire des ensembles corrects (mais pas forcément complets) de méta-nuplets satisfaits par une requête en entrée. Plus récemment, [RKNS15] présente une approche consistant à associer des motifs de complétude à des tables de données et une algèbre permettant d'interroger ces motifs afin de produire des informations sur l'exhaustivité des réponses. Nous adoptons une approche similaire avec une sémantique différente de *complétude relative* [FG10] (voir plus loin). [LNRN14] analyse différents types d'anomalies de résultats partiels inférées par l'observation d'anomalies d'accès physique aux données. Les motifs de complétude font la distinction entre les anomalies de cardinalité (incomplètes, fantômes, indéterminées) et de validité (crédibles et non crédibles) à différents niveaux de granularité (entrée, opérateur, colonne, partition) et permettent d'étudier comment ces anomalies se propagent dans un plan de requête. Nous poursuivons un objectif similaire concernant la propagation de complétude en utilisant des opérateurs à la granularité de la partition (jusqu'à des nuplets individuels). Notre travail se situe dans la lignée des *modèles de complétude relative* qui utilisent des données de référence (maître) pour évaluer la complétude des données [FG10]. [SKL$^+$17] introduit *m-tables* (inspiré de *c-tables* [IL88]) pour représenter les informations de complétude et une algèbre pour annoter les réponses à une requête avec des informations de certitude. D'autres travaux existants traitent de l'explication des réponses manquantes [HHT09, HH10] ou des requêtes *why-not* [TC10, BHT15]. Ces travaux supposent que les données sont complètes et se concentrent sur l'analyse et la correction des requêtes erronnées.

Motifs de complétude

Prenons l'exemple où Pierre souhaite étudier la fiabilité d'un réseau de capteurs de température et les éventuels problèmes de qualité des données liés aux défaillances des capteurs. Le réseau de capteurs observe les températures dans un certain nombre de pièces (stockées dans une table **LOC**) et produit une table **Energy** de mesures quotidiennes. Pierre veut étudier la fiabilité de ce réseau et détecter toutes les valeurs manquantes sur une période de temps **CAL**. Comparer les données collectées **Energy** aux ensembles de données de référence **LOC** et **CAL** est réalisable pour de petites tables, mais devient rapidement impossible dans un contexte réel où la table **Energy** peut contenir des milliers, voire des millions de mesures. Pour faciliter l'analyse, Pierre utilise une représentation compacte basée sur un motif de partitions incomplètes, comme indiqué dans le tableau $\mathbf{P_E}$ (Tableau 1).

Table 1: Tables de gabarit $\mathbf{P_E}$ et $\overline{\mathbf{P_E}}$

$\mathbf{P_E}$	fl	ro	we	da
p_0	*	*	w1	*
p_1	f2	*	*	*
p_2	f1	r1	*	Mon
...	...	...	...	...

$\overline{\mathbf{P_E}}$	fl	ro	we	da
$\overline{p}_0$	*	r2	w2	*
$\overline{p}_2$	f1	r1	*	Tue
...	...	...	...	...

En examinant le motif p_0, Pierre peut comprendre que toutes les mesures sont disponibles pour la première semaine **w1** (la partition identifiée par p_0 est terminée). Le même argument est valable pour p_1, indiquant la «complétude» du deuxième étage **f2**. La table de motifs *couvre toutes* les partitions complètes de la table de données et, plus formellement, nous appelons $T = (\mathbf{Energy}, \mathbf{LOC} \times \mathbf{CAL})$ une *table contrainte* et $\mathbf{P_E}$ a *couverture stricte* de T. $\mathbf{P_E}$ est également un couverture *minimal* permet à Pierre non seulement de raisonner sur toutes les données disponibles, mais également de déduire

des informations sur les mesures manquantes. Par exemple, le motif $[f1, r1, *, Mon]$ n'indique pas seulement la complétude pour la semaine $w1$, mais également l'absence de mesures pour d'autres étages, pièces et jours. L'existence d'une table de référence permet également de générer le *complément* $\overline{\mathbf{P_E}}$ de toutes les partitions *vides* dans la table **Energy**. Ce tableau montre qu'aucune mesure n'est disponible pour la chambre $r2$ aux deux étages et pour les deux jours de la semaine $w2$.

Algèbre de motif

Une *table de motif* est une table relationnelle qui peut être interrogée à l'aide de l'algèbre relationnelle standard $\Omega = \{\sigma, \pi, \bowtie, \cup, \cap, -\}$ (et SQL). Nous étendons l'algèbre relationnelle avec deux opérateurs *unfold* et *fold* qui *spécialisent* (déplient) et *généralisent* (replient) les motifs d'une table de motif P avec la garantie supplémentaire que le tableau résultant est une couverture stricte de P:

- L'opérateur *unfold* $\triangleleft_A(P, R)$ génère pour une table de motif P et la table de référence R une table de motifs *équivalente* $P' \equiv_R P$ où toutes les valeurs des attributs $a_i \in A$ sont des valeurs constantes. Notez également qu'un dépliage (unfold) complet sur tous les attributs génère D.
- L'opérateur fold $\triangleright_{a_i}$ est l'opérateur inverse de $\triangleleft_{a_i}$ et *généralise* (plie) tous les sous-ensembles S qui peuvent être remplacé par un seul motif $p_{a_i:*}$ avec une valeur générique pour l'attribut a_i.

L'algèbre relationnelle étendue avec les opérateurs $fold$ et $unfold$ peut être utilisée pour définir un certain nombre d'opérateurs de niveau supérieur dans les tables de motifs:

- Génération de tables de motifs: Le compactage (fold) complet $\triangleright(P, R)$ de P génère une *couverture stricte* (et minimale) de T.
- Génération des tables de patterns pour les résultats de requête: Soit $T = (D, R)$ une table contrainte et $Q(D)$ une requête relationnelle standard sur la table de données D faisant uniquement référence aux attributs de référence de D. Il est possible d'obtenir la couverture stricte (minimale) $P^*(T')$ de la table contrainte $T' = (Q(D), Q(R))$ en compactant le résultat de $Q(T)$ par rapport au résultat de la référence requête $Q(R)$: $P^*(T') = \triangleright(Q(D), Q(R))$ (voir les lignes pointillées rouges dans la figure 1). Une autre méthode consiste à réécrire $Q(D)$ dans une nouvelle *requête de motifs* $\widehat{Q}(P, R)$ sur une couverture stricte (pas nécessairement minimale) $P(T)$ de la table contrainte T telle que $\widehat{Q}$ prend en entrée le couple(P, R) et produit le nouveau couple $(P^*(T'), Q(R))$ (voir la ligne bleue continue dans la figure 1). Nous montrons

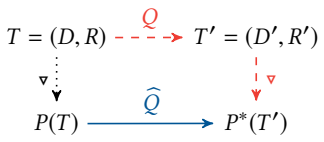

Figure 1: Requêtes de données et de signatures

dans la version étendue de l'article que cette deuxième solution (par réécriture) est généralement plus efficace que la première solution (par pliage du résultat).

- Projection sûre: Un autre problème lié à la complétude concerne les requêtes analytiques susceptibles d'agréger des informations sur des partitions incomplètes [Sho97]. Considérons, par exemple, la requête SQL avec la table de référence $R(fl, ro, we, da)$:

```
select fl, we, avg (khw) from Energy
group by fl, nous
```

Comme indiqué précédemment, le motif de complétude du résultat peut être calculé en compactant le résultat de $\pi_{fl,we}(T) = (\pi_{fl,we}(D), \pi_{f,w}(R))$. Il est également facile de voir que des partitions incomplètes dans D conduisent à des résultats d'agrégation incorrects (par exemple, tous les résultats pour floor $f2$ et pour la semaine $w1$ sont corrects, alors que le résultat pour floor $f1$ est incorrect) . Des couvertures strictes permettent de filtrer des partitions complètes en fonction des attributs supprimés ro et da avant l'application de la projection: L'opérateur de *projection sûre* $\widehat{\pi}^*$ compacte d'abord tous les motifs sur les attributs projetés et filtre toutes les dimensions incomplètes avant la projection. Cela garantit que le résultat ne contient que des motifs correspondant à des partitions complètes avec les attributs supprimés:

$$\widehat{\pi}^*_{\neg A_\pi}(P, R) = (\pi_{\neg A_\pi}(\sigma_{\theta_\pi}(\triangleright_{A_\pi}(P, R))), \pi_{\neg A_\pi}(R)) \quad (1)$$

où A_π indique les attributs supprimés par une projection et $\theta_\pi = \bigwedge_{a_i \in A_\pi}(a_i = *)$ filtre tous les motifs incomplets pour les attributs A_π.

Conclusion

La version étendue de l'article [HAB19] fournit plus de détails sur la mise en œuvre et l'optimisation de cette algèbre de motif, y compris deux algorithmes efficaces pour l'opérateur fold $\triangleright$ (l'opérateur unfold peut être implémenté en SQL). Des résultats expérimentaux sur les performances de la solution sont également présentés.

REFERENCES

[BHT15] Nicole Bidoit, Melanie Herschel, and Aikaterini Tzompanaki. Efficient computation of polynomial explanations of why-not questions. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 713–722. ACM, 2015.

[FG10] Wenfei Fan and Floris Geerts. Relative Information Completeness. *ACM Trans. Database Syst.*, 35(4):27:1–27:44, October 2010.

[HAB19] Fatma-Zohra Hannou, Bernd Amann, and Mohamed-Amine Baazizi. Explaining Query Answer Completeness and Correctness with Minimal Pattern Covers. January 2019. https://hal.archives-ouvertes.fr/hal-01982575.

[HH10] Melanie Herschel and Mauricio A Hernández. Explaining missing answers to spjua queries. *Proceedings of the VLDB Endowment*, 3(1-2):185–196, 2010.

[HHT09] Melanie Herschel, Mauricio A Hernández, and Wang-Chiew Tan. Artemis: A system for analyzing missing answers. *Proceedings of the VLDB Endowment*, 2(2):1550–1553, 2009.

[IL88] Tomasz Imieliński and Witold Lipski. Incomplete information in relational databases. In *Readings in Artificial Intelligence and Databases*, pages 342–360. Elsevier, 1988.

[LNRN14] Willis Lang, Rimma V. Nehme, Eric Robinson, and Jeffrey F. Naughton. Partial results in database systems. In *International Conference on Management of Data, SIGMOD*, pages 1275–1286, Snowbird, USA, June 2014.

[Mot89] Amihai Motro. Integrity = Validity + Completeness. *ACM Trans. Database Syst.*, 14(4):480–502, December 1989.

[RKNS15] Simon Razniewski, Flip Korn, Werner Nutt, and Divesh Srivastava. Identifying the extent of completeness of query answers over partially complete databases. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 561–576, Melbourne, Victoria, Australia, May 31 - June 4 2015.

[Sho97] Arie Shoshani. OLAP and statistical databases: Similarities and differences. In *Proceedings of the sixteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pages 185–196. ACM, 1997.

[SKL+17] Bruhathi Sundarmurthy, Paraschos Koutris, Willis Lang, Jeffrey F. Naughton, and Val Tannen. m-tables: Representing missing data. In *20th International Conference on Database Theory, ICDT*, pages 21:1–21:20, Venice, Italy, 2017.

[TC10] Quoc Trung Tran and Chee-Yong Chan. How to conquer why-not questions. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 15–26. ACM, 2010.

An experimental survey on big data frameworks (Highlight paper)

Wissem Inoubli
inoubliwissem@gmail.com
University of Tunis El Manar, Faculty of sciences of Tunis, LIPAH, 1060, Tunis, Tunisia

Sabeur Aridhi
sabeur.aridhi@loria.fr
University of Lorraine, LORIA, Campus Scientifique BP 239, Vandoeuvre-lès-Nancy, France

Haithem Mezni
haithem.mezni@fsjegj.rnu.tn
University of Jendouba, Avenue de l 'Union du Maghreb Arabe, Jendouba 8189, Tunisia

Mondher Maddouri
maddourimondher@yahoo.fr
College Of Business, University of Jeddah, P.O.Box 80327, Jeddah 21589, Saudi Arabia

Engelbert Mephu Nguifo
mephu@isima.fr
University Clermont Auvergne, CNRS, LIMOS, F-63000 Clermont-Ferrand, France

ABSTRACT

Recently, increasingly large amounts of data are generated from a variety of sources. Existing data processing technologies are not suitable to cope with the huge amounts of generated data. Yet, many research works focus on Big Data, a buzzword referring to the processing of massive volumes of (unstructured) data. Recently proposed frameworks for Big Data applications help to store, analyze and process the data. In this paper, we discuss the challenges of Big Data and we survey existing Big Data frameworks. We also present an experimental evaluation and a comparative study of the most popular Big Data frameworks with several representative batch.

CCS CONCEPTS

• **Information systems** → *Database management system engines.*

KEYWORDS

Big data, MapReduce, Hadoop, HDFS, Spark, Flink, Storm, Samza, Batch/stream processing

1 INTRODUCTION

In recent decades, increasingly large amounts of data are generated from a variety of sources. The size of generated data per day on the Internet has already exceeded two exabytes [1]. Within one minute, 72 h of videos are uploaded to Youtube, around 30.000 new posts are created on the Tumblr blog platform, more than 100.000 Tweets are shared on Twitter and more than 200.000 pictures are posted on Facebook [1]. Big Data problems lead to several research questions such as (1) how to design scalable environments, (2) how to provide fault tolerance and (3) how to design efficient solutions. Most existing tools for storage, processing and analysis of data are inadequate for massive volumes of heterogeneous data. Consequently, there is an urgent need for more advanced and adequate Big Data solutions. Many definitions of Big Data have been proposed throughout the literature. Most of them agreed that Big Data problems share four main characteristics, referred to as the four V's (Volume, Variety, Veracity and Velocity) [3]. In this paper, we first give an overview of most popular and widely used Big Data frameworks which are designed to cope with the above mentioned Big Data problems. We identify some key features which characterize Big Data frameworks. Then, we present an experimental study on Big Data processing systems with several representative batch stream and iterative workloads.

2 CATEGORIZATION OF BIG DATA FRAMEWORKS

We present in this section a categorization of the presented frameworks according to data format, processing mode, used data sources, programming model, supported programming languages and cluster manager (for more details, please see [2]). As explained in [2], Hadoop, Flink and Storm use the key-value format to represent their data. This is motivated by the fact that the key-value format allows access to heterogeneous data. For Spark, both RDD and key-value models are used to allow fast data access. We have also classified the studied big data frameworks into two categories: (1) batch mode and (2) stream mode. Additional, Hadoop processes the data in batch mode, whereas the other frameworks allow the stream processing mode. In terms of physical architecture, we notice tha all the studied frameworks are deployed in a cluster architecture, and each framework uses a specified cluster manager. We note that most of the studied frameworks use YARN as cluster manager.

3 EXPERIMETAL EVALUATION OF BIG DATA FRAMEWORKS

We have performed an extensive set of experiments to highlight the strengths and weaknesses of popular Big Data frameworks. We provide more informatiosn about our experiments and the obtained results in [2] and in the following link: https://members.loria.fr/SAridhi/files/software/bigdata/.

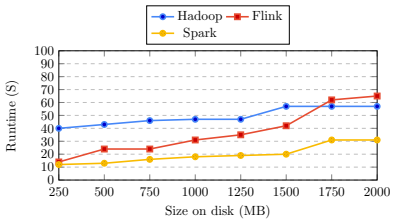

Figure 1: Impact of the size of the data on the average processing time

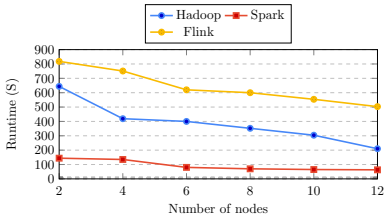

Figure 2: Impact of the number of machines on the average processing time (WordCount workload)

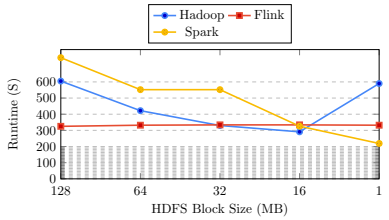

Figure 3: Impact of HDFS block size on the runtime (Kmeans workload with 10 million examples and 10 iterations)

3.1 Experimental results: Batch mode

This experiment aims to evaluate the impact of the size of the data on the processing time. The experiments are conducted using the WordCount workload and a set of text files with a size on disk varying from one GB to 100 GB. As shown in Fig. 1, Spark is the fastest framework in the case of small datasets, Flink is the next and Hadoop is the lowest, and Spark has kept its order in the case of big datasets and Hadoop showed good results compared to Flink (see [2]). In the next experiment, we tried to evaluate the scalability and the processing time of the considered frameworks based on the number of machines in the cluster. Fig. 2 shows that both Hadoop and Flink take higher time regardless of the cluster size, compared to Spark. In the next experiment, we try to show the impact of data partitioning on the studied frameworks. In our experimental setup, we used HDFS for storage. We varied the block size in our HDFS system and we run Kmeans with 10 iterations with all the used frameworks. Fig. 3 presents the impact of the HDFS block size on the processing time. As shown in Fig. 3, the curves are inflated proportionally to the size of the HDFS block size for both Hadoop and Spark while Flink does not imply any variation in the processing time.

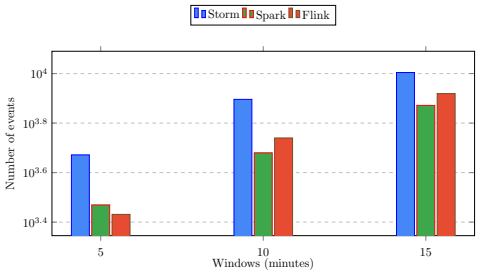

Figure 4: Impact of the window time on the number of processed events (100 KB per message)

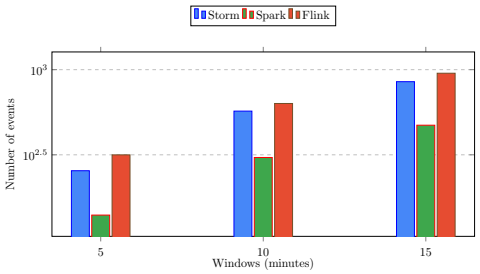

Figure 5: Impact of the window time on the number of processed events (500 KB per message)

3.2 Experimental results: Stream mode

The goal here is to compare the performance of the studied frameworks according to the number of processed messages within a period of time. In the first experiment, we send a tweet of 100 KB (in average) per message. The obtained results in Fig 4 show that Flink, Samza and Storm have better processing rates compared to Spark.

In the second experiment, we changed the sizes of the processed messages. We used 5 tweets per message (around 500 KB per message). The results presented in Fig. 5 show that Samza and Flink are very efficient compared to Spark, especially for large messages.

4 CONCLUSION

In this work, we surveyed popular frameworks for large-scale data processing. After a brief description of the main paradigms related to Big Data problems, we presented an overview of the Big Data frameworks Hadoop, Spark, Storm and Flink. We presented a theoretical and experimental comparative study of the presented frameworks on a cluster of machines.

REFERENCES

[1] Amir Gandomi and Murtaza Haider. 2015. Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management* 35, 2 (2015), 137 – 144.

[2] Wissem Inoubli, Sabeur Aridhi, Haithem Mezni, Mondher Maddouri, and Engelbert Mephu Nguifo. 2018. An experimental survey on big data frameworks. *Future Generation Computer Systems* 86 (2018), 546–564.

[3] A Oguntimilehin and EO Ademola. 2014. A Review of Big Data Management, Benefits and Challenges. *A Review of Big Data Management, Benefits and Challenges* 5, 6 (2014), 433–438.

On the optimization of recursive relational queries

Louis Jachiet
Nils Gesbert
Pierre Geneves
Nabil Layaida

ABSTRACT

Graph databases have received a lot of attention recently as they are particularly useful in many applications such as social networks or for the semantic web. Various languages have emerged to query such graph databases. At the heart of many of those query languages, there is a construction to navigate through the graph which allows some form of recursion.The relational model has benefited from a huge body of research in the last half century and that is why many graph databases either rely on (or have adopted the techniques of) relational based query engines. Since its introduction, the relational model has seen various attempts to extend it with recursion and it is now possible to use recursion in several SQL or Datalog based database systems. The optimization of recursive queries remains, however, a challenge.In this paper, we introduce μ-RA, a variation of the Relational Algebra that allows for the expression of relational queries with recursion. μ-RA is strictly more expressive than the union of conjunctive regular path queries. We will present its syntax,semantics and the rewriting rules we specifically devised to tackle the optimization of recursive queries. As we will demonstrate,our μ-RA can generate query execution plans that can not be considered by other existing approaches. Taking the example of SPARQL, we will show why these newly considered plans are particularly relevant for the efficient evaluation of graph query languages.

Fouille de séquences et analyse formelle de concepts pour l'étude de trajectoires de visiteurs

Nyoman Juniarta, Miguel Couceiro, Amedeo Napoli
Université de Lorraine, CNRS, Inria, LORIA
F-54000, Nancy, France
nyoman.juniarta@loria.fr,miguel.couceiro@inria.fr,amedeo.napoli@loria.fr

RÉSUMÉ

Cet article concerne la fouille des trajectoires de visiteurs dans le cadre du projet européen CrossCult sur le patrimoine culturel à l'échelle européenne. Plus précisément, nous travaillons sur une caractérisation des trajectoires des visiteurs et la fouille de ces trajectoires en tant que séquences. Le processus de fouille repose sur deux approches, la fouille de sous-séquences fréquentes avec et sans lacunes (avec et sans contiguïté des éléments). Les deux approches sont définies dans le cadre de l'analyse formelle de concepts. De plus, en utilisant une mesure de similarité, nous construisons une classification hiérarchique utilisée pour interpréter et classer les trajectoires par rapport à quatre styles de visiteurs bien connus dans les études muséales.

KEYWORDS

Analyse formelle de concepts, fouille de motifs séquentiels avec et sans lacunes, classification hiérarchique, styles de visites.

1 INTRODUCTION

Ce travail de recherche se place dans le cadre du projet européen CrossCult sur le patrimoine culturel européen (https://www.crosscult.eu/). L'idée générale est de favoriser l'émergence d'un patrimoine culturel commun européen en permettant aux personnes d'envisager une visite (musée, ville, site archéologique) à l'échelle européenne à l'aide de technologies (mobiles) informatiques.

Dans ce projet, nous nous intéressons principalement à l'analyse et la recommandation des trajectoires. Nous avons notamment deux objectifs principaux : (i) la découverte de trajectoires de visiteurs sur la base de la fouille de données séquentielle, et (ii) la caractérisation des trajectoires en termes de sous-séquences découvertes. Nous supposons que les sous-séquences sont reliées à des styles de visites, au contenu de la visite et à l'environnement. Ainsi, les sous-séquences peuvent être utilisées pour analyser la trajectoire d'un visiteur et faire des recommandations tout au long de la visite. e plus, l'occurrence de certaines sous-séquences à une étape dans une trajectoire peut être témoin d'un changement de comportement et déclencher un changement dans les recommandations.

Dans cet article, nous discutons des travaux théoriques et pratiques concernant la définition de trajectoires des visiteurs et la fouille de ces trajectoires considérées comme des séquences. Le processus de fouille s'appuie sur deux approches de fouille de séquences en analyse formelle de concepts (AFC [7]) : `MRGS` pour "Mining Rare General Subsequences" [2] et `MFCS` pour 'Mining Frequent Contiguous Subsequences" [1]. La première approche fouille des sous-séquences rares d'une manière générale, c'est-à-dire que des lacunes peuvent apparaître dans les sous-séquences. La seconde approche recherche des des sous-séquences fréquentes où ne figure aucune lacune (une sorte de sous-chaînes). Si l'article sur `MRGS` [2] portait sur les sous-séquences rares, ce n'est pas plus le cas ici et nous considérons aussi des sous-séquences fréquentes. Nous réutilisons également la mesure de similarité sim_{ACS} développée pour analyser des trajectoires de patients entre hôpitaux [5, 6]. Avec sim_{ACS}, nous construisons une classification hiérarchique qui joue le rôle de "classification de référence".

Pour analyser et interpréter les trajectoires des visiteurs, nous comparons les résultats des algorithmes `MRGS` et `MFCS` avec la classification de référence. Nous analysons ces résultats à la lumière de quatre styles théoriques de visiteurs, à savoir la *fourmi*, le *papillon*, le *poisson* et la *sauterelle* [11]. Plus globalement, cet article met en valeur la fouille de données séquentielles complexes et multidimensionnelles dans le cadre de l'AFC selon deux approches, l'analyse de trajectoires à l'aide de "Jumping Emerging Patterns" (JEPs) et la classification hiérarchique.

2 DES DONNÉES SUR LES VISITES DE MUSÉE

Nous avons considéré un jeu de données sur les trajectoires de 254 visiteurs au Musée Hecht à Haïfa, en Israël [3, 10]. Après un pré-traitement pour éliminer des trajectoires (trop) courtes, nous avons obtenu 216 trajectoires. Chaque trajectoire est alors modélisée comme une séquence d'articles présents dans le musée.

3 LES QUATRE STYLES THÉORIQUES DE VISITES DE MUSÉE

Dans [8], des styles théoriques de visiteurs dans un musée ont été étudiés et quatre styles principaux ont émergés :

- La *fourmi* correspond à un visiteur qui va voir toutes les œuvres en suivant leur ordre de localisation dans le musée.
- La *sauterelle* correspond à un visiteur qui ne voit que certaines œuvres, en sautant de l'une à l'autre.
- Le *papillon* correspond à un visiteur désireux de découvrir certaines œuvres mais pas toutes, sans montrer de préférences exactes.
- Le *poisson* correspond à un visiteur qui n'est pas vraiment intéressé par les œuvres et qui reste la plupart du temps au centre des salles sans montrer aucun objectif précis.

Un visiteur peut changer de style au cours d'une visite, tandis que d'autres éléments peuvent être d'importance et intervenir dans ce changement de style. Ainsi, la quantité de visiteurs dans une salle du musée ou plus simplement la fatigue du visiteur peuvent jouer un rôle tangible. Ces éléments qui mériteraient une attention spéciale ne sont pas pris en compte dans cette étude.

4 L'ANALYSE DES TRAJECTOIRES DE VISITEURS

Dans ce qui suit, un objectif est d'étudier la correspondance entre certaines sous-séquences spécifiques incluses dans les trajectoires de visiteurs et les quatre styles théoriques de visite. Pour caractériser le comportement d'un visiteur, nous avons proposé le "workflow" suivant :

(1) Nous appliquons un "clustering hiérarchique" aux 216 séquences en utilisant sim_{ACS} comme mesure de distance. Dans le dendrogramme résultant, nous avons retenu 5 clusters désignés par *A*, *B*, *C*, *D* et *E*. Les 4 premiers correspondent aux 4 styles de visite, c'est à dire *fourmi*, *papillon*, *poisson*, et *sauterelle*. Le dernier cluster rassemble les trajectoires non caractérisées.

(2) Indépendamment du clustering, nous créons deux treillis de concepts en utilisant les algorithmes MFCS et MRGS sur tout le jeu de données. Chaque concept dans le treillis se compose d'un ensemble de visiteurs ("extent") et d'un ensemble de séquences communes ("intent").

(3) Sur la base des deux treillis, nous recherchons les JEPs [4] pour chaque cluster. Ici un JEP pour un cluster *X* correspond à l'"intent" d'un concept dont l'"extent" comprend uniquement des visiteurs classés dans le cluster *X*.

(4) Nous mettons en correspondance chaque cluster à un style de visite représenté par un ou plusieurs JEPs.

5 RÉSULTATS

Avec MFCS et MRGS, nous ne trouvons aucun concept satisfaisant pour caractériser le JEP du cluster *E*. Il apparaît que parmi tous les concepts dont l'extent est exclusivement composé d'éléments du cluster *E*, aucun de ces extents ne compte plus d'un visiteur. Ainsi, parmi les 33 membres du cluster *E*, 32 d'entre eux visitent moins de deux articles pendant toute leur visite du musée. Nous pouvons supposer que ces visiteurs ne sont pas vraiment intéressés par la visite du musée et donc considérer ce cluster comme celui du *poisson*.

Le cluster *D* est plus facile à interpréter. Sur la base des JEPs associés, nous remarquons que beaucoup des visiteurs sautent quelques articles lors de la visite. En outre, une partie de ces visites ne suit pas l'ordre proposé par le musée, par exemple la visite d'un élément situé loin de l'entrée précède celle d'un article près de l'entrée. Nous pouvons donc interpréter ce cluster comme celui de la *sauterelle*, puisque ces visiteurs sautent d'un élément à un autre.

Les clusters *A*, *B* et *C* sont relativement semblables et plus difficiles à discerner. Les visiteurs dans ces clusters suivent un comportement de *fourmi* avec un comportement "ordonné", les articles près de l'entrée sont visités au début et presqu'aucun élément n'est sauté. Néanmoins, dans le cluster *C*, certains membres visitent à plusieurs reprises un ensemble d'éléments dans une salle, indiquant plutôt un comportement de *papillon*.

6 CONCLUSION

Dans cet article, nous avons présenté un cadre original de fouille de trajectoires de visiteurs d'un musée qui sont modélisées comme des séquences d'éléments. Les résultats mettent en évidence un certain nombre de motifs intéressants qui peuvent caractériser des comportements de visiteurs.

Plus précisément, nous avons appliqué deux méthodes de fouille de séquences qui s'appuient sur l'analyse formelle de concepts, à savoir MFCS et MRGS, pour découvrir des sous-séquences caractéristiques et typiques d'un style de visiteur. Ces deux algorithmes nous ont permis de regrouper les visiteurs sur la base de treillis de concepts.

En parallèle, nous avons aussi construit une classification ("clustering") des visites en utilisant une mesure de similarité qui prend en compte les sous-séquences communes à deux visites. Les clusters sont utilisés conjointement avec les treillis pour caractériser quatre styles théoriques de visiteurs en fonction des sous-séquences découvertes avec MFCS et MRGS. En plus, ces clusters peuvent être réutilisés pour construire un système de recommandation pour les visiteurs, mais ce sujet reste encore à traiter plus spécifiquement.

Des résultats plus complets pourraient être attendus si d'autres éléments de connaissances étaient considérés, comme la durée et l'heure de la visite, ou encore des connaissances sur l'histoire et la géographie relatives aux œuvres du musée. Enfin, l'étude des "concepts stables" [9] tout comme des informations continues ou "instantanées" telles que les commentaires et l'état du visiteur pendant la visite pourraient également jouer un rôle dans l'analyse et la recommandation en ligne.

RÉFÉRENCES

[1] Aleksey Buzmakov, Elias Egho, Nicolas Jay, Sergei O Kuznetsov, Amedeo Napoli, and Chedy Raïssi. 2016. On mining complex sequential data by means of FCA and pattern structures. *International Journal of General Systems* 45, 2 (2016), 135–159.

[2] Victor Codocedo, Guillaume Bosc, Mehdi Kaytoue, Jean-François Boulicaut, and Amedeo Napoli. 2017. A Proposition for Sequence Mining Using Pattern Structures. In *Proceedings of the 14th International Conference on Formal Concept Analysis (LNCS 10308)*. Springer, 106–121.

[3] Eyal Dim and Tsvi Kuflik. 2014. Automatic Detection of Social Behavior of Museum Visitor Pairs. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 4, 4 (2014), 17 :1–17 :30.

[4] Guozhu Dong and Jinyan Li. 1999. Efficient mining of emerging patterns : Discovering trends and differences. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 43–52.

[5] Elias Egho, Nicolas Jay, Chedy Raïssi, Dino Ienco, Pascal Poncelet, Maguelonne Teisseire, and Amedeo Napoli. 2014. A contribution to the discovery of multidimensional patterns in healthcare trajectories. *Journal of Intelligent Information Systems* 42, 2 (2014), 283–305.

[6] Elias Egho, Chedy Raïssi, Toon Calders, Nicolas Jay, and Amedeo Napoli. 2015. On measuring similarity for sequences of itemsets. *Data Mining and Knowledge Discovery* 29, 3 (2015), 732–764.

[7] Bernhard Ganter and Rudolf Wille. 1999. *Formal Concept Analysis : Mathematical Foundations*. Springer.

[8] Tsvi Kuflik, Zvi Boger, and Massimo Zancanaro. 2012. Analysis and prediction of museum visitors' behavioral pattern types. In *Ubiquitous Display Environments*. Springer, 161–176.

[9] Sergei O. Kuznetsov. 2007. On stability of a formal concept. *Annals of Mathematics and Artificial Intelligence* 49 (2007), 101–115.

[10] Joel Lanir, Tsvi Kuflik, Eyal Dim, Alan J Wecker, and Oliviero Stock. 2013. The influence of a location-aware mobile guide on museum visitors' behavior. *Interacting with Computers* 25, 6 (2013), 443–460.

[11] Eliséo Véron and Martine Levasseur. 1983. *Ethnographie de l'exposition*. Bibliothèque Publique d'Information, Centre Georges Pompidou, Paris.

Extension et Partitionnement de Requêtes SQL pour l'Exploration de Données

Marie Le Guilly
INSA Lyon, CNRS, LIRIS UMR5205
Villeurbanne, France
marie.le-guilly@insa-lyon.fr

Ihab F. Ilyas
University of Waterloo
Waterloo, Canada
ilyas@uwarterloo.ca

Jean-Marc Petit
INSA Lyon, CNRS, LIRIS UMR5205
Villeurbanne, France
jean-marc.petit@insa-lyon.fr

Marian Scuturici
INSA Lyon, CNRS, LIRIS UMR5205
Villeurbanne, France
marian.scuturici@insa-lyon.fr

1 CONTEXTE ET PROBLÉMATIQUE

Le SQL et les bases de données relationnelles sont toujours très utilisés pour stocker et accéder à des jeux de données, dans de nombreuses applications: systèmes commerciaux de gestion de données, projets de recherche scientifiques, entreprises ...La nature déclarative du SQL permet à ses utilisateurs d'explorer de vastes jeux de données, d'y découvrir de nouvelles informations, ou d'y sélectionner des sous-ensembles servant de base à la construction de nouveaux modèles d'apprentissage. Cependant, si les vues et les mots-clés classiques du SQL sont généralement suffisants pour écrire la majorité des requêtes, plusieurs difficultés peuvent freiner des utilisateurs dans l'exploration de leur données. Localiser les données désirées, trouver la bonne jointure, ou encore faire face à des tables ou des attributs nommés de manière peu claire, sont des défis qui peuvent rendre l'écriture de requêtes SQL très ardue.

En pratique, les requêtes nécessitent parfois des tâtonnement et plusieurs itérations avant de parvenir au résultat désiré. Un problème récurrent au début du leur conception est qu'elles retournent souvent trop de tuples: l'analyste est alors submergé de données, et doit parvenir à déterminer comment compléter sa requête pour diminuer la taille du résultat. Ce problème a été abordé dans d'autres articles, tel que [5] qui se base sur un objectif de cardinalité, ou les approches de type *top-k*[2].

Dans ce papier, nous proposons une solution basée sur le SQL, pouvant ainsi être intégrée à n'importe quel SGBD. L'objectif est d'aider un analyste à comprendre les nombreux tuples de sa requête en les résumant, et en lui donnant plusieurs pistes d'exploration possibles à partir de sa requête initiale. Ainsi, étant une requête initiale Q, nous proposons n extensions de cette requête, avec des prédicats de sélection supplémentaires permettant de raffiner la requête initiale. Les propriétés de ces extensions sont les suivantes:

- Les résultats des extensions forment une partition des résultats de Q.
- Une extension raffine la requête initiale Q en lui ajoutant un ou plusieurs prédicats de sélection supplémentaires.

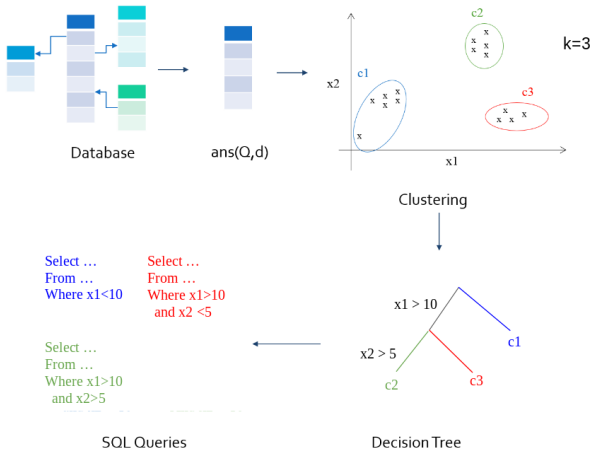

Figure 1: Principe de la solution proposée pour la construction d'extension de requête SQL

- Chaque extension mène à un sous-ensemble de résultat de Q qui est entièrement différent de celui des autres extensions.

Nous proposons un algorithme pour déterminer de telles extensions, avec une implémentation dotée d'une interface homme-machine pour les utilisateurs.

2 PROPOSITION

Afin de calculer un ensemble de k extensions pour une requête Q donnée, nous proposons un algorithme en deux étapes, qui commence par diviser les tuples de la requête initiale en k sous-ensembles, puis détermine une requête décrivant chacun d'eux. A notre connaissance, ce problème d'extension est nouveau, même s'il se rapproche de problématiques rencontrée dans les domaines du *query reverse engineering* [8] ou du *redescription mining* [6].

Le principe général de notre proposition est schématisé sur la figure 1. L'avantage est que le processus est applicable pour une requête SQL quelconque, car une requête produira toujours le même type de sortie, à savoir un tableau de tuples, qui est considéré dans notre approche comme un jeu de données.

Afin de diviser les tuples, nous proposons d'utiliser un algorithme de clustering [3], afin de regrouper ensemble les tuples les plus

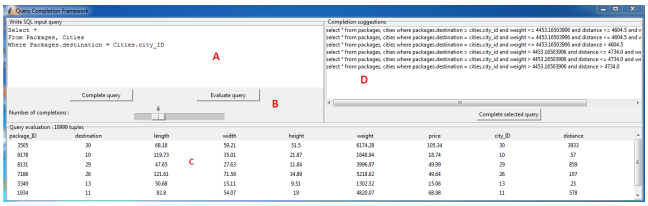

Figure 2: Prototype de l'interface d'extension de requêtes

proches, et donc d'obtenir par la suite une description à la fois significative et compacte. L'algorithme utilisé dans ce travail est *k-means* [4], qui permet de donner le nombre de clusters en paramètre de l'algorithme, et donc de jouer sur le nombre et le contenu des extensions obtenues, afin que l'utilisateur puisse choisir celui qui lui convient le mieux. Chaque tuple est ainsi assigné à un cluster, que nous considérons par la suite comme une sorte de label, qui est ajouté à notre jeu de données.

Une fois le jeu de données labellisé obtenu, nous proposons de décrire chaque cluster en utilisant un arbre de décision binaire [1]. L'objectif est de discriminer entre les différents clusters, et d'utiliser les branches de l'arbre comme description de chacun.

Cependant, tous ces algorithmes sont cachés à l'utilisateur, qui se contente de donner le nombre k d'extensions à obtenir. Par conséquent, nous proposons de construire un arbre de décision binaire comportant exactement k feuilles.

Une fois l'arbre obtenu, il est aisé d'obtenir les extensions: en effet, chaque parcours de l'arbre de la racine à une de ses feuilles, correspond à une extension, c'est à dire la conjonction de chacun des prédicats rencontrés.

3 PROTOTYPE ET EXPÉRIMENTATIONS

Nous avons implémenté notre algorithme en Python, à l'aide de la librairie scikit-learn [7]. Nous avons également développé une interface utilisateur, sous la forme d'un éditeur SQL classique, agrémenté de la fonction d'extension de requête. Le prototype de cette interface est présenté sur la figure 2. Nous avons utilisé cette interface afin de procéder à des expérimentations avec des utilisateurs, dans le but de répondre à deux questions:

- Les utilisateurs parviennent-ils à obtenir leurs résultats plus facilement ?
- Les utilisateurs parviennent-ils à s'habituer à l'outil ?

70 étudiants en informatique ont été réunis, et divisés en deux groupes équilibrés. Le premier groupe avait accès à notre outil, tandis que le deuxième bénéficiait d'une interface similaire, mais sans l'option d'extension. Il leur a été posé 10 questions. Les trois premières avait pour but de s'assurer que les deux groupes étaient homogènes, et obtenait des résultats similaires lorsque l'extension n'est pas utile. En revanche, elle pouvait servir sur toutes les questions suivantes, qui étaient de type exploratoire. Le matériel de cette expérimentation est disponible en ligne [1].

Les résultats sont présentés en détail sur la figure 3, qui présente le temps de réponse à chacune des questions, pour les deux groupes.

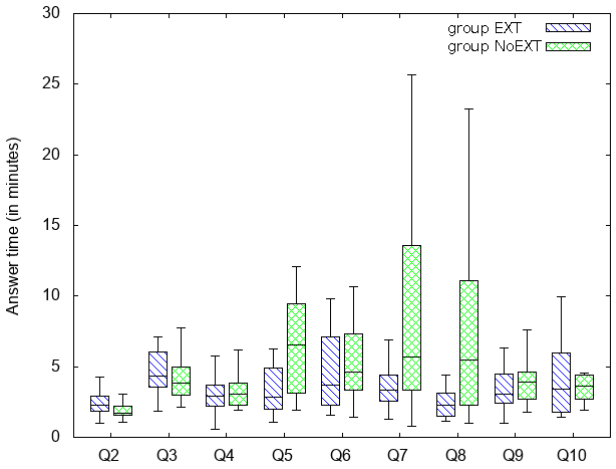

Figure 3: Principe de la solution proposée pour la construction d'extension de requête SQL

Si les performances des deux groupes sont similaires sur les questions sans extension, l'écart est significatif sur les autres, et est particulièrement marqué pour les questions 7 et 8 ou le groupe avec extension a répondu bien plus rapidement. Ainsi, les extensions semblent avoir permis d'arriver plus rapidement aux résultats désirés.

Nos résultats ont également montré qu'une fois qu'un étudiant avait utilisé l'extension pour une question, il avait tendance à l'utiliser pour toutes les questions suivantes, montrant à la fois l'utilité et l'acceptabilité de l'extension pour les utilisateurs.

4 ACKNOWLEDGMENTS

Ce travail a été en partie financé par le projet ContentCheck ANR-15-CE23-0025 par l'Association Nationale Recherche Technologie (ANRT)

REFERENCES

[1] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. 1984. *Classification and regression trees.* CRC press.
[2] Ronald Fagin, Amnon Lotem, and Moni Naor. 2003. Optimal aggregation algorithms for middleware. *Journal of computer and system sciences* 66, 4 (2003), 614–656.
[3] Jiawei Han. 2005. *Data Mining: Concepts and Techniques.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
[4] S. Lloyd. 2006. Least Squares Quantization in PCM. *IEEE Trans. Inf. Theor.* 28, 2 (Sept. 2006), 129–137. https://doi.org/10.1109/TIT.1982.1056489
[5] Chaitanya Mishra and Nick Koudas. 2009. Interactive query refinement. In *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology.* ACM, 862–873.
[6] Laxmi Parida and Naren Ramakrishnan. 2005. Redescription mining: Structure theory and algorithms. In *AAAI*, Vol. 5. 837–844.
[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.
[8] Quoc Trung Tran, Chee-Yong Chan, and Srinivasan Parthasarathy. 2014. Query Reverse Engineering. *The VLDB Journal* 23, 5 (Oct. 2014), 721–746. https://doi.org/10.1007/s00778-013-0349-3

[1] https://marielgy.github.io/sql_experimentation/

Parallel Computation of PDFs on Big Spatial Data Using Spark

Ji Liu
Inria, LIRMM and Univ. of Montpellier, France
jiliuwork@gmail.com

Noel Moreno Lemus
LNCC Petrópolis, Brazil
nmlemus@gmail.com

Esther Pacitti
LIRMM, Univ. of Montpellier and Inria, France
esther.pacitti@lirmm.fr

Fabio Porto
LNCC Petrópolis, Brazil
fporto@lncc.br

Patrick Valduriez
Inria, LIRMM and Univ. of Montpellier, France
patrick.valduriez@inria.fr

ABSTRACT

We consider big spatial data, which is typically produced in scientific areas such as geological or seismic interpretation. The spatial data can be produced by observation or numerical simulation programs and correspond to points that represent a 3D soil cube area. However, errors in signal processing and modeling create some uncertainty, and thus a lack of accuracy in identifying geological or seismic phenomenons. Such uncertainty must be carefully analyzed. To analyze uncertainty, the main solution is to compute a Probability Density Function (PDF) of each point in the spatial cube area. However, computing PDFs on big spatial data can be very time consuming (from several hours to even months on a parallel computer). In this paper, we propose a new solution to efficiently compute such PDFs in parallel using Spark, with three methods: data grouping, machine learning prediction and sampling. We evaluate our solution by extensive experiments on different computer clusters using big data ranging from hundreds of GB to several TB. The experimental results show that our solution scales up very well and can reduce the execution time by a factor of 33 (in the order of seconds or minutes) compared with a baseline method.

KEYWORDS

Spatial data, big data, parallel processing, Spark.

1 INTRODUCTION

Big spatial data is now routinely produced and used in scientific areas such as geological or seismic interpretation [4]. These spatial data allow identifying some phenomenon over a spatial reference [6]. The spatial data can be produced by observation or numerical simulation programs and correspond to points that represent a 3D soil cube area. However, errors in signal processing and modelling create some uncertainty, and thus a lack of accuracy when identifying phenomenons. In order to understand uncertainty, several simulation runs with different input parameters are usually conducted, thus generating multiple spatial data sets that can be very big, *e.g.* hundreds of GB or TB. Uncertainty quantification of spatial data is of much importance for geological or seismic scientists [13], [16]. It is the process of quantifying the uncertainty error of each point in the spatial cube space, which requires computing a Probability Density Function (PDF) of each point [10]. The PDF is composed of the distribution type (*e.g.* normal, exponential) and necessary statistical parameters (*e.g.* the mean and standard deviation values for normal).

In this paper, we propose a new solution to efficiently compute PDFs in parallel by taking advantage of Spark [17] framework for computer clusters (see [11]). To validate our solution, we use the spatial data generated from simulations based on the models from the seismic benchmark of the HPC4e project [1]. The problem we address is how to efficiently compute PDFs under bounded error constraints. In addition to deploying PDF computation over Spark, we propose three methods to efficiently compute PDFs: data grouping, machine learning (ML) prediction and sampling. Data grouping consists in grouping similar points to compute the PDF. ML prediction uses ML classification methods to predict the distribution type of each point. Sampling method enables to efficiently compute statistical parameters of a region by sampling a fraction of the total number of points to reduce the computation space.

The problem we address is new, and thus, there is no relevant related work. There are some MATLAB libraries [12][14], compatible with the Message Passing Interface (MPI), for calculating PDFs, using a baseline method without optimization for redundant PDF calculation and at the expense of difficult parallel programming. Machine Learning (ML) techniques are widely used to process big data [2, 5] for the sake of dimension reduction [3, 9], prediction, classification [15] and knowledge discovering based on big data [8]. In our approach, we also exploit ML techniques, *i.e.* decision trees, to classify the distribution types of the observation values at different points so as to reduce useless calculation [7]. However, unlike the traditional use of ML, we do not reduce the dimension since the PDF computation depends on the whole set of observation values of a point. But we do discover new knowledge, which is used to reduce redundant PDF computation, using ML techniques and reduce the quantity of data using data aggregation.

The paper is organized as follows. Section 2 presents our solution to compute PDFs in parallel, including the architecture and three methods, *i.e.* data grouping, ML prediction and sampling, to compute PDFs with Spark. Section 3 presents our experimental evaluation on different computer clusters with different data sizes, ranging from hundreds of GB to several TB. Section 4 concludes.

2 METHODS TO COMPUTE PDFS

In this section, we present the architecture, a baseline method and our methods to compute PDFs efficiently.

The **architecture** has four layers, *i.e.* infrastructure, basic process to compute PDFs, memory management and methods to compute PDFs. The higher layers take advantage of the lower layers' services to implement their functionality. The infrastructure layer provides the basic execution environment, including Spark, HDFS and Network File System (NFS) in a computer cluster. The basic processing layer provides guiding principles to load the big spatial data and to compute PDFs. The memory management layer allows optimizing the execution of the basic process by caching data and managing sliding windows on big data. The methods to compute PDFs are presented as follows.

The **baseline method** computes the PDF of each point in a slice in three steps. First, it loads the spatial data sets and calculates the mean and standard deviation values of each point. Then, it computes the PDF and the corresponding error for each point, which can be executed in parallel using the Map operation in Spark. The calculation of PDF is realized using a brute-force method. Finally, the data is persisted in the storage resources and the average error of all the point in the slice is calculated.

Some points may have the same distribution of observation values. Thus, using the **data grouping** method, we can execute the PDF computation process only once and use the result to represent all the corresponding points. In this method, we use two steps to group data. First, the points with exactly the same mean and standard deviation values are aggregated into a group. The grouping can be realized by the *aggregation* operation in Spark. Then, one point in each group is selected to represent the group of points. Afterwards, the data corresponding to selected points are processed to compute the PDFs.

We propose an **ML prediction** method based on a decision tree to compute PDFs. In the brute-force method, the execution of calculating the PDF of a point is repeated several times, which is very inefficient. We assume that we can learn the correlation among statistical features, *e.g.* mean and standard deviation values, and the distribution type. Then, we can directly predict the distribution type based on the relationship and use the predicted distribution type to calculate the PDF for each point or group.

In order to choose a slice to compute PDFs, we need to compute the features of a slice very quickly. We propose a **sampling** method that samples the points and uses ML prediction to generate the distribution type of each point in order to reduce execution time.

3 EXPERIMENTAL EVALUATION

We evaluate and compare the different methods presented in Section 2. We use two different computer clusters (from 6 nodes to 64 nodes) and perform experiments with three different data sets (235 GB, 1.9 TB and 2.4 TB). The experimental results show that our solution is efficient and scales up very well compared with Baseline. Data grouping outperforms Baseline by up to 92% (more than 10 times) without introducing extra error. ML prediction can be up to 91% (more than 9 times) better than Baseline with very slight acceptable error (up to 0.017). The combination of data grouping and ML prediction can be up to 97% (more than 33 times) better than Baseline. As the number of nodes exceeds 10 nodes, ML prediction outperforms the combination. Thus, in order to compute PDFs, the combination of data grouping and ML prediction has good performance when each point corresponds to a small number of observation values, e.g. 1000, and when there is small number of nodes (less than 20). Otherwise, ML is the best option. We also showed that sampling is very efficient to calculate general statistics information in order to choose a slice for calculating PDFs. Finally, Sampling should be used with the aforementioned best option in order to efficiently compute PDFs.

4 CONCLUSION

In this paper, we addressed the problem of efficiently computing PDFs under bounded error constraints. We proposed a parallel solution using a Spark cluster with three new methods: data grouping, ML prediction and sampling. To validate our solution, we implemented these methods in a Spark cluster and performed extensive experiments on two different computer clusters using big spatial data ranging from hundreds of GB to several TB. The experimental results show that our solution is efficient and scales up very well compared with Baseline.

REFERENCES

[1] Hpc geophysical simulation test suite. https://hpc4e.eu/downloads/hpc-geophysical-simulation-test-suite.

[2] O. Y. Al-Jarrah, P. D. Yoo, S. Muhaidat, G. K. Karagiannidis, and K. Taha. Efficient machine learning for big data: A review. *Big Data Research*, 2(3):87–93, 2015.

[3] B. Bohn, J. Garcke, R. Iza-Teran, A. Paprotny, B. Peherstorfer, U. Schepsmeier, and C. Thole. Analysis of car crash simulation data with nonlinear machine learning methods. In *Int. Conf. on Computational Science ICCS*, pages 621–630, 2013.

[4] R. Campisano, F. Porto, E. Pacitti, F. Masseglia, and E. S. Ogasawara. Spatial sequential pattern mining for seismic data. In *Simpósio Brasileiro de Banco de Dados (SBBD)*, pages 241–246, 2016.

[5] T. Condie, P. Mineiro, N. Polyzotis, and M. Weimer. Machine learning on big data. In *29th IEEE Int. Conf. on Data Engineering, ICDE*, pages 1242–1244, 2013.

[6] S. Fotheringham, C. Brunsdon, and M. Charlton. *Quantitative Geography: Perspectives on Spatial Data Analysis*. 2000.

[7] M.A. Friedl and C.E. Brodley. Decision tree classification of land cover from remotely sensed data. *Remote Sensing of Environment*, 61(3):399 – 409, 1997.

[8] L. O. Hall, N. V. Chawla, and K. W. Bowyer. Decision tree learning on very large data sets. In *IEEE Int. Conf. on Systems, Man and Cybernetics*, pages 2579–2584, 1998.

[9] G E Hinton and R R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[10] F. Kathryn, J. T. Oden, and D. Faghihi. A bayesian framework for adaptive selection, calibration, and validation of coarse-grained models of atomistic systems. *Journal of Computational Physics*, 295:189 – 208, 2015.

[11] J. Liu, E. Pacitti, and P. Valduriez. A survey of scheduling frameworks in big data systems. *International Journal of Cloud Computing*, page 27, 2018.

[12] S. Marelli and B. Sudret. *UQLab: A Framework for Uncertainty Quantification in MATLAB*. ETH-Zürich, 2014.

[13] C. Michele, T. Stefano, and S. Andrea. Sensitivity and uncertainty analysis in spatial modelling based on gis. *Agriculture, Ecosystems & Environment*, 81(1):71 – 79, 2000.

[14] E. E. Prudencio and K. W. Schulz. The parallel C++ statistical library 'queso': Quantification of uncertainty for estimation, simulation and optimization. In *Euro-Par: Parallel Processing Workshops*, pages 398–407, 2011.

[15] S. Suthaharan. Big data classification: Problems and challenges in network intrusion prediction with machine learning. *ACM SIGMETRICS Performance Evaluation Review*, 41(4):70–73, 2014.

[16] G. Trajcevski. Uncertainty in spatial trajectories. In *Computing with Spatial Trajectories*, pages 63–107, 2011.

[17] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica. Spark: Cluster computing with working sets. In *USENIX Workshop on Hot Topics in Cloud Computing (HotCloud)*, 2010.

Une exploration désagrégée de corpus d'archives Web pour étudier des collectifs en ligne disparus

Quentin Lobbé

ABSTRACT

Le Web est un environnement instable. Alors que les sites Web grandissent jours après jours, des communautés entières peuvent venir à disparaître en ne laissant plus que des traces incomplètes voire inexistantes sur le Web vivant. Les corpus d'archives Web préservent l'histoire du Web et ont pour vocation de limiter la perte de notre patrimoine numérique. Les archives Web demeurent essentielles à la compréhension des cycles de vie des collectifs en ligne disparus. Dans cet article, nous proposons un framework pour suivre les dynamiques internes des communautés Web disparues, basé sur l'exploration de corpus d'archives Web existants. Pour atteindre cet objectif, nous introduisons une nouvelle unité d'analyse appelée fragment Web : un sous-ensemble sémantique et syntaxique d'une page Web donnée, conçu pour améliorer la précision historique des recherches à venir. Cette contribution a une valeur pratique pour ceux qui mènent des explorations d'archives à grande échelle (en termes de temps et de volume) ou qui sont intéressés par l'approche computationnelle de l'histoire du Web et des sciences sociales. En appliquant notre framework aux archives marocaines de l'Atlas e-diasporas, nous observons d'abord l'effondrement d'une communauté établie de blogs de migrants marocains. Nous montrons sa mutation progressive vers les plateformes émergentes du Web social, entre 2008 et 2018. Ensuite, nous étudions la création soudaine d'un collectif éphémère de membres d'un forum motivés par la vague révolutionnaire du printemps arabe, au début de l'année 2011. Nous proposons enfin de nouvelles pistes pour étudier le Web d'un point de vue historique en introduisant le concept de moment de pivot du Web.

Traitement Distribué des Requêtes Top-k en Préservant la Confidentialité des Données

Sakina Mahboubi
INRIA & LIRMM, Univ. Montpellier
sakina.mahboubi@inria.fr

Reza Akbarinia
INRIA & LIRMM, Univ. Montpellier
reza.akbarinia@inria.fr

Patrick Valduriez
INRIA & LIRMM, Univ. Montpellier
patrick.valduriez@inria.fr

1 INTRODUCTION

Nous considérons un système distribué où les utilisateurs peuvent externaliser leurs données sensibles et émettre des requêtes top-k. Une requête top-k est un type de requête important qui permet à l'utilisateur d'obtenir les éléments de données k les plus pertinents pour la requête. Les données utilisateurs sont cryptées (pour des raisons de confidentialité) et distribuées (pour des raisons de performances) sur plusieurs nœuds. Dans ce contexte, nous abordons le problème du traitement des requêtes top-k dans le respect de la vie privée.

La protection de la vie privée lors du traitement des requêtes top-k est essentielle pour de nombreuses applications qui externalisent des données sensibles. Prenons l'exemple d'une université qui externalise la base de données des étudiants dans un cloud public, en mode Infrastructure-as-a-Service (IaaaS), avec des nœuds non fiables. La base de données est partitionnée verticalement (pour des raisons de performance) et cryptée. Ensuite, une requête top-k intéressante sur les données distribuées cryptées est la suivante : retourner les k étudiants qui ont les pires moyennes dans certains cours donnés.

Il existe différentes approches pour traiter les requêtes top-k sur données *plaintext* (non cryptées). Une des approches les plus connues est TA [1] qui fonctionne sur des listes triées de valeurs d'attributs. Cependant, il n'existe pas de solution efficace capable d'évaluer efficacement les requêtes top-k sur les données cryptées dans les systèmes distribués.

Dans ce résumé étendu [1], nous décrivons brièvement notre système, appelé SD-TOPK (Secure Distributed TOPK), qui crypte et stocke les données utilisateur dans un système distribué, et est capable d'évaluer les requêtes top-k sur les données cryptées. SD-TOPK fournit un nouvel algorithme de traitement des requêtes top-k qui trouve un ensemble de données cryptées dont il est prouvé qu'il contient les éléments top-k. Ceci se fait sans avoir à décrypter les données dans les nœuds où elles sont stockées. De plus, nous proposons un puissant algorithme de filtrage qui élimine autant que possible les faux positifs sans déchiffrement des données.

Le reste de ce résumé étendu est organisé comme suit. La section 2 donne la définition du problème. La section 3 décrit le système SD-TOPK.

[1] Le papier complet a été publié dans [4].

2 DÉFINITION DU PROBLÈME

Dans ce travail, nous abordons le problème du traitement des requêtes top-k dans les systèmes distribués en préservant la confidentialité des données.

Par une requête top-k, l'utilisateur spécifie un nombre k, et le système doit retourner les k réponses les plus pertinentes. Le degré de pertinence des réponses à la requête est déterminé par une *fonction de notation*. Une méthode courante pour le traitement efficace des requêtes top-k est d'exécuter les algorithmes sur des listes *sorted lists* (aussi appelées *inverted lists*) [1]. Définissons-les formellement.

Soit D un ensemble d'éléments de données n, alors les listes triées sont des listes m $L_1, L_2, ...L_m$, de sorte que chaque liste L_i contient chaque élément de données $d \in D$ sous la forme d'une paire $(id(d), s_i(d))$ où $id(d)$ est l'identification d et $s_i(d)$ est une valeur qui indique le *local score* (valeur d'attribut) de d en L_i. Les éléments de données de chaque liste L_i sont triés par ordre décroissant de leurs scores locaux. Par exemple, dans une table relationnelle, chaque liste triée représente une colonne triée de la table où le score local d'une donnée est sa valeur d'attribut dans cette colonne.

Soit f une fonction de notation donnée par l'utilisateur dans la requête top-k. Pour chaque élément de données $d \in D$ une *note globale*, montrée par $ov(d)$, est calculée en appliquant la fonction f sur les notes locales de d. Formellement, nous avons $ov(d) = f(s_1(d), s_2\ (d), ..., s_m(d))$.

Dans ce travail, nous supposons que la fonction de notation est dans la classe des fonctions linéaires à coefficients positifs (notées par *LFPC*). Formellement, une fonction f est LFPC si $f = a_1x_1 + a_2x_2 + \cdots + a_mx_m$ où chaque coefficient $a_i \geq 0$ pour $1 \leq i \leq m$. Des fonctions importantes comme SUM, COUNT, AVG et MAX sont dans la classe des fonctions LFPC.

Nous supposons que les listes triées sont stockées dans les nœuds d'un système distribué. Nous ne faisons pas d'hypothèse spécifique sur l'architecture du système distribué qui peut être très générale, *i.e.*, un cluster de noeuds.

Nous considérons le modèle de l'adversaire *honest-but-curious* pour les noeuds du système distribué. Dans ce modèle, l'adversaire est curieux d'apprendre les données sensibles sans introduire aucune modification dans les données ou les protocoles. Ce modèle est largement utilisé dans de nombreuses solutions de traitement de la confidentialité [3].

3 SD-TOPK

Dans cette section, nous décrivons d'abord notre méthode de chiffrement des données et de stockage dans le système distribué. Ensuite, nous présentons notre algorithme de traitement des requêtes top-k sur les données cryptées.

Dans notre système, le chiffrement des données est effectué par un composant appelé *trusted-client*. Ce composant est installé sur la machine de l'utilisateur.

Avant de chiffrer une base de données, le trusted-client partitionne chaque liste triée en paquets (buckets an anglais) en utilisant la technique de bucketisation. Cette technique consiste à partitionner les données (*e.g.*, valeurs d'attribut) en paquets. Il existe plusieurs méthodes pour partitionner les valeurs d'un attribut, par exemple en divisant le domaine d'attribut en intervalles presque égaux ou en créant des paquets de tailles égales [2]. Dans l'implémentation actuelle de notre système, nous utilisons cette dernière méthode.

Soit b_1, b_2, ..., b_t les paquets créés pour une liste L_j. Chaque paquet b_i a une borne inférieure, notée $min(b_i)$, et une borne supérieure, notée $max(b_i)$. La limite inférieure du paquet b_i est égale à la limite supérieure du paquet suivant, *i.e.*, $min(b_i) = max(b_{i+1})$. Un élément de données d est dans le paquet b_i, si et seulement si son score local dans la liste L_j est entre les limites inférieure et supérieure du paquet, *i.e.*, $min(b_i) \leq s_j(d) < max(b_i)$.

Nous utilisons deux types de schémas de chiffrement (méthodes) pour crypter les éléments de données et leurs scores dans les paquets : *déterministe* et *probabiliste*. Le chiffrement *déterministe* est un schéma de chiffrement qui, pour deux entrées égales, génère les mêmes chiffres. Nous utilisons ce schéma pour chiffrer l'ID des éléments de données dans chaque liste de base de données. Cela nous permet d'avoir le même ID chiffré pour chaque donnée dans toutes les listes triées.

Le second schéma de chiffrement, c'est-à-dire *probabiliste*, est utilisé pour chiffrer les scores locaux des éléments de données dans les paquets. Avec le chiffrement *probabiliste*, pour les mêmes valeurs, différents cryptogrammes sont générés, mais la fonction de déchiffrement renvoie le même texte en clair pour eux. Ainsi, par exemple, si deux éléments de données ont les mêmes scores dans une liste, leurs scores chiffrés peuvent être différents. Le chiffrement probabiliste est le type de chiffrement le plus puissant.

Après avoir chiffré les ID des données et les scores locaux, le trusted-client place les scores locaux chiffrés dans le paquet correspondant et envoie les paquets et les données chiffrées concernées au serveur.

3.1 Algorithme Top-k

L'idée principale derrière le traitement des requêtes top-k dans SD-TOPK est d'utiliser les limites du paquet et une nouvelle technique pour décider quand arrêter la lecture des données chiffrées des listes. Pour chaque requête top-k, l'un des nœuds du système distribué assure la coordination entre les nœuds pour exécuter la requête. Nous appelons ce nœud *coordinateur*. Le coordinateur peut être le nœud qui reçoit initialement la requête de l'utilisateur ou être choisi au hasard parmi les nœuds du système.

Ci-dessous, nous décrivons notre algorithme de traitement des requêtes top-k. Étant donné une requête top-k avec un nombre k et une fonction de notation f qui est linéaire avec des coefficients positifs. SD-TOPK choisit un nœud comme *coordinateur*, puis les étapes suivantes sont effectuées pour répondre à la requête :

(1) Le coordinateur diffuse la requête en parallèle aux nœuds, et demande à chaque nœud de renvoyer les paquets qui contiennent les k premiers éléments de données dans sa liste. Chaque nœud renvoie l'identifiant chiffré des k premiers éléments de données, ainsi que la limite inférieure de leurs paquets.
(2) Pour chaque élément de données retourné d, le coordinateur calcule son *note globale minimale* défini comme suit : $ov_{min}(d) = f(v_1(d), v_2(d), ..., v_m(d))$ où $v_i(d)$ est la limite inférieure du paquet qui contient d dans la liste L_i. Si d n'a pas été retourné au coordinateur par le nœud qui stocke la liste L_j alors $v_j(d) = 0$.
(3) Le coordinateur trie les éléments de données reçus en fonction de leur score global minimum et choisit l'élément de données d' pour lequel le score global minimum k^{th} est indiqué par δ. Puis, il utilise le score global minimum de d' pour calculer un seuil θ comme suit : $\theta = \frac{\delta}{\sum_{i=1}^{m} a_i}$ où $a_1, \ldots, a_m$ sont les coefficients dans la fonction de notation.
(4) Le coordinateur diffuse θ en parallèle aux nœuds. Chaque nœud renvoie au coordinateur les paquets dont la limite supérieure est supérieure ou égale à θ.
(5) Soit Y l'ensemble de toutes les données qui sont envoyées au coordinateur par au moins un nœud. Nous appelons Y l'ensemble de *candidats*. Le coordinateur envoie l'identifiant chiffré de tous les éléments de données contenus dans Y aux nœuds, et ils retournent les scores locaux chiffrés de chaque élément de données contenu dans Y.
(6) Enfin, le coordinateur retourne au trusted-client les éléments candidats et leurs scores locaux cryptés.

Lorsque le trusted-client reçoit les éléments candidats, il les décrypte à l'aide des clés secrètes, extrait les k éléments qui ont les notes globales les plus élevées et les retourne à l'utilisateur.

4 CONCLUSION

Dans ce résumé étendu, nous avons présenté SD-TOPK, un système efficace qui crypte et externalise les données utilisateur dans un système distribué, et qui est capable d'évaluer les requêtes top-k sur des données chiffrées, sans les déchiffrer dans les nœuds du système.

Nous avons évalué les performances de notre solution sur des bases de données synthétiques et réelles. Les résultats montrent un excellent temps de réponse et un excellent coût de communication pour SD-TOPK. Ils montrent que le temps de réponse de SD-TOPK peut être de plusieurs ordres de grandeur supérieur à celui des algorithmes TA. Ceci est principalement dû à ses algorithmes optimisés de traitement et de filtrage des requêtes top-k. Les résultats montrent également un gain significatif en coût de communication de SD-TOPK par rapport aux autres algorithmes.

REFERENCES

[1] R. Fagin, A. Lotem, and M. Naor. 2003. Optimal aggregation algorithms for middleware. *J. Comput. Syst. Sci.* 66, 4 (2003), 614–656.
[2] B. Hore, S. Mehrotra, M. Canim, and M. Kantarcioglu. 2012. Secure multidimensional range queries over outsourced data. *J. VLDB* 21, 3 (2012), 333–358.
[3] R. Li, Alex X. Liu, Ann L. Wang, and B. Bruhadeshwar. 2014. Fast Range Query Processing with Strong Privacy Protection for Cloud Computing. *PVLDB* 7, 14 (2014), 1953–1964.
[4] S. Mahboubi, R. Akbarinia, and P. Valduriez. 2018. Privacy-Preserving Top-k Query Processing in Distributed Systems. In *24th Europ. Conf. on Parallel and Distributed Computing (Euro-Par).*

Une étude expérimentale de la largeur d'arbre de données graphe du monde réel

Silviu Maniu
Pierre Senellart
Suraj Jog

ABSTRACT

La largeur d'arbre est un paramètre qui mesure combien une instance relationnelle est proche d'un arbre, si cette instance peut raisonnablement être décomposée sous la forme d'un arbre. Beaucoup de tâches de calcul sont connus pour être tractables sur les bases de données de petite largeur d'arbre, mais calculer la largeur d'arbre d'une instance donnée est intractable. Cet article est la première étude à grande échelle de la largeur d'arbre et des décompositions arborescentes d'instances de graphes issus du monde réel (25 jeux de donnés issus de 8 domaines différentes, avec des tailles variant de quelques milliers à quelques millions de nœuds). Le but est de déterminer quelles données, s'il y en a, peuvent bénéficier de la pléthore d'algorithmes pour les bases de données de petite largeur d'arbre. Pour chaque jeu de données, nous obtenons des estimations de borne supérieure et inférieure de leur largeur d'arbre, et étudions les propriétés de leurs décompositions arborescentes. Notre principal résultat est que les graphes de largeur d'arbre relativement petite existent en pratique : c'est le cas des réseaux d'infrastructure, qui les rend compatibles avec certaines techniques basés sur la largeur d'arbre. Nous montrons également que, même quand la largeur d'arbre est haute, l'utilisateur de décompositions arborescentes partielles peut résulter en des structures de données permettant d'assister les algorithmes.

Intelligent clients for replicated Triple Pattern Fragments*

Thomas Minier
LS2N, University of Nantes
Nantes, France
thomas.minier@univ-nantes.fr

Hala Skaf-Molli
LS2N, University of Nantes
Nantes, France
hala.skaf@univ-nantes.fr

Pascal Molli
LS2N, University of Nantes
Nantes, France
pascal.molli@univ-nantes.fr

Maria-Esther Vidal
TIB Leibniz Information Centre For Science and Technology University Library & Fraunhofer IAIS
Bonn, Germany
Maria.Vidal@tib.eu

ABSTRACT

Following the Triple Pattern Fragments (TPF) approach, intelligent clients are able to improve the availability of the Linked Data. However, data availability is still limited by the availability of TPF servers. Although some existing TPF servers belonging to different organizations already replicate the same datasets, existing intelligent clients are not able to take advantage of replicated data to provide fault tolerance and load-balancing. In this paper, we propose Ulysses, an intelligent TPF client that takes advantage of replicated datasets to provide fault tolerance and load-balancing. By reducing the load on a server, Ulysses improves the overall Linked Data availability and reduces data hosting cost for organizations. Ulysses relies on an adaptive client-side load-balancer and a cost-model to distribute the load among heterogeneous replicated TPF servers. Experimentations demonstrate that Ulysses reduces the load of TPF servers, tolerates failures and improves queries execution time in case of heavy loads on servers.

KEYWORDS

Semantic Web, Triple Pattern Fragments, Intelligent client, Load balancing, Fault tolerance, Data Replication

1 INTRODUCTION

The Triple Pattern Fragments (TPF) [7] approach improves Linked Data availability by shifting costly SPARQL operators from servers to intelligent clients. However, data availability is still dependent on servers' availability, *i.e.*, if a server fails, there is no failover mechanism, and the query execution fails too. Moreover, if a server is heavily loaded, performances can be deteriorated.

The availability of TPF servers can be ensured by cloud providers and consequently servers' availability are depended on the budget of data providers. An alternative solution is to take advantage of datasets replicated by different data providers. In this case, TPF clients can balance the load of queries processing among data providers. Using replicated servers, they can prevent a single point of failure server-side, improves the overall availability of data, and distributes the financial costs of queries execution among data providers.

Some data providers already replicate RDF datasets produced by other data providers [5]. Replication can be total, *e.g.*, both DBpedia [1] and LANL Linked Data Archive [2] publish the same versions of DBpedia datasets. Replication can also be partial, *e.g.*, LOD-a-lot [3] [1] gathers all LOD Laundromat datasets [4] into a single dataset, hence each LOD Laundromat dataset is a partial replication of LOD-a-lot.

Existing TPF clients allow to process a federated SPARQL query over a federation of TPF servers replicating the same datasets. However, *existing TPF clients do not support replication nor client-side load balancing* [7]. Consequently, the execution time of queries are severely degraded in presence of replication.

Additionnaly, distributing the load of query processing across replicated requires to know servers capabilities. As TPF servers are heterogeneous, *i.e*, they do not have the same processing capabilities and access latencies, poorly distributed load further deteriorates query processing.

In this paper, we propose Ulysses, a replication-aware intelligent TPF client providing load balancing and fault tolerance over heterogeneous replicated TPF servers. Managing replication in Linked Data has been already addressed in [4–6]. These approaches consider SPARQL endpoints and not TPF servers. Moreover, they focus on minimizing intermediate results and do not address the problems of load-balancing and fault-tolerance. The load balancing problem with replicated datasets is addressed in [3] but without considering heterogeneous servers. The main contributions of this paper are:

- A replication-aware source selection for TPF servers.
- A light-weighted cost-model for accessing heterogeneous TPF servers.
- A client-side load balancer for distributing SPARQL query processing among heterogeneous TPF servers hosting replicated datasets.

*This work has already been accepted for publication at the 15th Extended Semantic Web Conference (ESWC 2018)

[1] http://fragments.dbpedia.org/
[2] http://fragments.mementodepot.org/
[3] http://hdt.lod.labs.vu.nl/?graph=LOD-a-lot
[4] http://lodlaundromat.org/wardrobe/

2 ULYSSES APPROACH

Ulysses proposes intelligent clients for replicated TPF servers. In order to balance the load on heterogeneous servers hosting replicated datasets, Ulysses relies on three key ideas.

2.1 Replication-aware source selection for replicated Triple Pattern Fragments

First, Ulysses uses a *replication-aware source selection* algorithm to identify which TPF servers can be used to distribute evaluation of triple patterns during SPARQL query processing. The source selection algorithm relies on the replication model introduced in [4, 5]. When processing a SPARQL query, Ulysses loads a *catalog* that describes *replicated fragments* and the servers that provide access to them, *i.e.*, the fragment localities. The results of the source selection algorithm are used to identify the TPF servers that replicate the same relevant fragments. These servers can be used to distribute the evaluation of triple patterns. However, as TPF servers are heterogeneous, *i.e.*, they do not exhibit the same processing capabilities, we must ensure that servers with weaker processing capabilities are less requested in favor of more powerful servers, to maintain good query processing performance. To this end, we define a cost-model to compute and evaluate servers capabilities.

2.2 A cost-model for evaluating TPF servers capabilities

The cost model uses server capability factors to evaluate servers capabilities. A *server capability factor* depends on: (i) The access latency of the TPF client for this server. (ii) The processing capabilities of the server, *i.e.*, in terms of CPU, RAM, etc. (iii) The impact of the server loads on its processing capabilities.

As a triple pattern is evaluated in constant time [2], a server capability factor can be deduced from its access time. If an HTTP request is not resolved in the server cache, a *server access time* is the time to receive one page of RDF triples matching a triple pattern from the TPF server [5]. However, as the size of pages could be different among servers, two servers with the same access times do not necessarily produce results at the same rate. Thus, we choose to rely on a *server throughput*, *i.e.*, the number of results served per unit of time, to evaluate more precisely its processing capabilities. Capabilities are normalized with respect to other servers used to evaluate the query. Ulysses keeps these estimations updated in real-time, using HTTP requests sent by the client during query processing as *probes* to update each server throughput. Hence, Ulysses can dynamically react to failures or heavy load of TPF servers without any additional probing messages.

2.3 Ulysses adaptive client-side load balancing with fault tolerance

Last, Ulysses uses an *adaptive load-balancer* to perform load balancing among replicated servers. Instead of simply selecting the server with the best access latency, Ulysses performs its selection using a weighted random algorithm: the probability of selecting a server is proportional to its processing capabilities. Thus, the load of query processing will be distributed across all replicated servers, *minimizing* the individual cost of query processing for each data provider. This allows for quick adaptation to variations in server loads: if a server throughput is deteriorated, its capability will decrease and it will be less frequently accessed. This load-balancer also provides *fault-tolerance*, by re-scheduling failed HTTP requests using available replicated servers.

[5] We suppose that an HTTP client is able to detect if an HTTP request has been resolved in the cache.

3 CONCLUSION AND FUTURE WORKS

In this paper, we presented Ulysses, a replication-aware intelligent TPF client providing load balancing and fault tolerance over heterogeneous replicated TPF servers. Ulysses accurately evaluates processing capabilities of TPF servers using only HTTP responses times observed during query processing. Experimental results demonstrate that Ulysses reduces the individual load per server, speeds up query execution time under heavy load, tolerates faults, and adapts to the load of servers in real-time. Moreover, by distributing the load among different data providers, Ulysses distributes the financial costs of queries execution among data providers without impacting query execution times for end-users.

Ulysses opens several perspectives. First, we do not address how the catalog of replicated fragments can be acquired. It could be provided by TPF servers as additional metadata or built collaboratively by TPF clients as they evaluate SPARQL queries. Building and maintaining the catalog of replicated data over the web is challenging. Another perspective is to consider divergence over replicated data. Executing queries over weakly-consistent replicated datasets raises interesting issues about the correctness of results.

ACKNOWLEDGMENTS

This work is partially supported through the FaBuLA project, part of the AtlanSTIC 2020 program.

REFERENCES

[1] Javier D Fernández, Wouter Beek, Miguel A Martínez-Prieto, and Mario Arias. 2017. LOD-a-lot. In *International Semantic Web Conference*. Springer, 75–83.
[2] Javier D Fernández, Miguel A Martínez-Prieto, Claudio Gutiérrez, Axel Polleres, and Mario Arias. 2013. Binary RDF representation for publication and exchange (HDT). *Web Semantics: Science, Services and Agents on the World Wide Web* 19 (2013), 22–41.
[3] Thomas Minier, Gabriela Montoya, Hala Skaf-Molli, and Pascal Molli. 2017. Parallelizing Federated SPARQL Queries in Presence of Replicated Data. In *The Semantic Web: ESWC 2017 Satellite Events, Revised Selected Papers*. 181–196.
[4] Gabriela Montoya, Hala Skaf-Molli, Pascal Molli, and Maria-Esther Vidal. 2015. Federated SPARQL queries processing with replicated fragments. In *International Semantic Web Conference*. Springer, 36–51.
[5] Gabriela Montoya, Hala Skaf-Molli, Pascal Molli, and Maria-Esther Vidal. 2017. Decomposing federated queries in presence of replicated fragments. *Web Semantics: Science, Services and Agents on the World Wide Web* 42 (2017), 1–18.
[6] Muhammad Saleem, Axel-Cyrille Ngonga Ngomo, Josiane Xavier Parreira, Helena F. Deus, and Manfred Hauswirth. 2013. DAW: Duplicate-AWare Federated Query Processing over the Web of Data. In *The Semantic Web - ISWC 2013*. 574–590.
[7] Ruben Verborgh, Miel Vander Sande, Olaf Hartig, Joachim Van Herwegen, Laurens De Vocht, Ben De Meester, Gerald Haesendonck, and Pieter Colpaert. 2016. Triple Pattern Fragments: A low-cost knowledge graph interface for the Web. *Web Semantics: Science, Services and Agents on the World Wide Web* 37 (2016), 184–206.

CaLi: A Lattice-Based Model for License Classifications

Benjamin Moreau
Nantes University, LS2N, CNRS
OpenDataSoft
Benjamin.Moreau@ls2n.fr
Benjamin.Moreau@opendatasoft.com

Patricia Serrano-Alvarado
Nantes University, LS2N, CNRS
Patricia.Serrano-Alvarado@ls2n.fr

Emmanuel Desmontils
Nantes University, LS2N, CNRS
Emmanuel.Desmontils@ls2n.fr

1 INTRODUCTION AND MOTIVATION

Web applications facilitate combining resources (linked data, web services, source code, documents, etc.) to create new ones. For a resource producer, choosing the appropriate license for a combined resource involves choosing a license compliant with all the licenses of combined resources and controlling the reusability of the resulting resource through the compatibility of its license. The risk is either, to choose a license too restrictive making the resource difficult to reuse, or to choose a not enough restrictive license that will not sufficiently protect the resource.

We consider that a license l_j is compliant with a license l_i if a resource licensed under l_i can be licensed under l_j without violating l_i. If a license l_j is compliant with l_i then we consider that l_i is compatible with l_j

Finding the right trade-off between compliance and compatibility is a difficult process. An automatic classification over licenses protecting resources would facilitate this task.

Our research question is: *given a license l_i, how to automatically position l_i over a set of licenses in terms of compatibility and compliance?*

We propose CaLi, a lattice-based model to classify licenses. CaLi is able to answer questions like, "what are the licenses with which l_i is compliant?" and "what are the licenses with which l_i is compatible?". We show the usability of a CaLi classification through a prototype of a license-based search engine for the Web of Data.

The remainder of this abstract is as follows. Section 2 discuses related works, Section 3 introduces the CaLi model, Section 4 shows the implementation of a license-based search engine for linked data and Section 5 concludes.

2 RELATED WORK

There exist some tools to facilitate the creation of license compliant resources such as TLDRLegal[1], CC Choose[2], ChooseALicense[3], CC search[4], Web2rights [5] and Licentia[6]. Unfortunately, one of them is able to order licenses in terms of compatibility or compliance.

The problem of license compatibility and license combination are addressed in Web services [3], in collaborative environments [6] and in the Web of Data [4, 8, 9]. These works focus on combining operators for automatic license combination but do not propose to position a license over a set of licenses.

Concerning the problem of license classification to facilitate the selection of a license, [1, 5] propose a compatibility graph to classify licenses. However, such graph is build from a manual interpretation of each license, so its generalization and automation is not possible.

In the domain of access control, [2] proposes a lattice model of secure information flow to automatically classify security classes with associated resources. [7] describes several models based on this approach but none focuses on classifying licenses.

None of these works answers our research question. In our work we propose a lattice-based model inspired by [2]. Existing combining operators like [3, 4, 8, 9] make feasible the automatic generation of a lattice. This model is independent on any license combination approach, license description language and licensed resources so that it can be used in a wide variety of domains.

3 CALI: A LATTICE-BASED LICENSE MODEL

CaLi is a model that allows to express license classifications with a lattice structure. The model considers a finite set of licenses, a combining operator, a compatibility relation over licenses, resources and their association with licenses, and a set of constraints. The combining operator applied to the set of licenses produces a lattice that expresses the compliance and compatibility among licenses. This model is introduced next.

Definition 3.1. CaLi $= \langle L, \bigoplus, \rightarrow, R, \mapsto, C \rangle$.

$L = \{l_1, ..., l_n\}$ is a set of licenses.
$\bigoplus$ is a license combining operator.
$\rightarrow$ is the *compatible with* relation defined on pairs of licenses.
$R = \{r_1, ..., r_m\}$ is a set of resources.
$\mapsto$ is the *protected by* relation between R and L.
$C = \langle C_L, C_{\rightarrow} \rangle$ is a finite set of constraints to express *viability*. C_L is a set of constraints over L and $C_{\rightarrow}$ is a set of constraints over $\rightarrow$ in $L \times L$.

A license is considered as a set of actions. Actions used in L are taken from $A = \{a_1, ..., a_n\}$, a set of actions that can be applied on resources.

[1] https://tldrlegal.com/
[2] https://creativecommons.org/choose/
[3] https://choosealicense.com/
[4] https://ccsearch.creativecommons.org/
[5] http://www.web2rights.com/creativecommons/
[6] http://licentia.inria.fr/

$\bigoplus$ is an associative and commutative binary operator that specifies for any pair of licenses, the combined license that is compliant with both licenses. For licenses l_i and l_k, if $l_i \bigoplus l_k = l_j$ then l_j is compliant with l_i and l_k.

For licenses l_i and l_j, we write $l_i \rightarrow l_j$ iff l_i is compatible with l_j (or l_j is compliant with l_i). $l_i \rightarrow l_j$ implies $l_i \bigoplus l_j = l_j$, i.e., l_j overcomes l_i.

We write $r_i \mapsto l_i$ when the resource r_i is protected by l_i. For $r_i \mapsto l_i$, if l_i is compatible with l_j then r_i can be protected by l_j.

In addition we introduce the concepts of viability of a license and viability of a $\rightarrow$ relation.

- *A viable license is a license that allows the licensed resource to be used.* A license l_i is viable iff it respects all C_L constraints.
- *A viable compatible with relation, $l_i \rightarrow l_j$, is a relation that allows a resource licensed under l_i to be licensed under l_j.* A $\rightarrow$ relation between two licenses is viable iff it respects all $C_{\rightarrow}$ constraints.

l_i is compatible with l_j if there exists a path from l_i to l_j where all *compatible with* relations and all licenses are viable.

We demonstrate that the set of licenses L, the *compatible with* relation $\rightarrow$ and the combining operator $\bigoplus$ form a lattice where:

(1) $\langle L, \rightarrow \rangle$ is a partially ordered set.
(2) L is finite.
(3) L has an infimum (or greatest lower bound) I such that $I \rightarrow l_i$ $\forall l_i \in L$.
(4) $\bigoplus$ is a totally defined least upper bound operator on L.

4 A LICENSE-BASED LINKED DATA SEARCH ENGINE

We experiment the usefulness of our work with a license-based search engine for the Web of Data. This search engine is based on the following CaLi classification:

$L = \{\langle P, O, Pr \rangle\}$; *$P, O, Pr$ are sets of permissions, obligations and prohibitions containing actions $\in A$. A is the set containing the 72 actions considered by ODRL*[7].

$\bigoplus$ *is the license combining operator, $l_i \bigoplus l_j \equiv \langle P_{l_i} \cap P_{l_j}, O_{l_i} \cup O_{l_j}, Pr_{l_i} \cup Pr_{l_j} \rangle$ where P_{l_i} is the permission set of l_i, O_{l_i} is the obligation set of l_i, and Pr_{l_i} is the prohibition set of l_i.*

$\rightarrow$ *is the compatible relation stating that $l_i \rightarrow l_j$ iff $P_{l_i} \supseteq P_{l_j}$ and $O_{l_i} \subseteq O_{l_j}$ and $Pr_{l_i} \subseteq Pr_{l_j}$.*

$C = \langle C_L, C_{\rightarrow} \rangle$ *where $C_L = \{\omega_{L_1}, \omega_{L_2}\}$ and $C_{\rightarrow} = \{\omega_{\rightarrow_1}, \omega_{\rightarrow_2}\}$.*

Concerning the set of constraints that characterizes viable licenses and viable $\rightarrow$ relations:

ω_{L_1} specifies that viable licences have sets of permissions, obligations and prohibitions that have to be mutually disjoint.

ω_{L_2} specifies that if an action a_i is prohibited, all the actions included in a_i must not be permitted nor obligated to be a viable license. For example, a license that prohibits *Use* is not viable if it permits or obligates *Commercial Use*.

$\omega_{\rightarrow_1}$ specifies that cc:ShareAlike term obligates the distribution of derivative works only under the same license.

$\omega_{\rightarrow_2}$ specifies that if cc:DerivativeWorks is prohibited in a license l_i then resources protected by l_i should not be protected by l_j.

[7] https://www.w3.org/TR/odrl-vocab/#actionConcepts

The generation of a lattice has a complexity of $O(2^n)$. In our case, $n = 2|A|$ and the number of nodes in the lattice is 2^{144}. The number of nodes grows exponentially with the number of terms resulting in a combinatorial explosion. Generating the complete lattice is not suitable for a real scenario with a large number of terms. Thus we propose algorithms to generate part of the lattice *on the fly*. Our approach is in $O(n)$ where n is the number of existing nodes.

Answering our questions with the CaLi classification is straightforward. *Finding all licensed datasets whose licenses are compliant with a particular license l_i* means to retrieve datasets protected by l_i and all datasets protected by licenses that are above l_i in the graph. Similarly, *finding all licensed datasets whose licenses are compatible with a specific license l_i* means to retrieve datasets protected by l_i and all datasets protected by licenses that are below l_i in the graph.

The prototype of this search engine is available at http://cali.priloo.univ-nantes.fr[8].

5 CONCLUSIONS

In this paper we proposed CaLi, a model to express license classifications. The goal of our work is to encourage the publication and reuse of resources in a license compliant web.

CaLi can be used to classify licenses in different contexts where resources need to be reused. It considers a set of licenses, a combining operator, a compatibility relation over the set of licenses and a set of constraints. The combining operator applied automatically to the set of licenses produces a lattice that expresses both compliance and compatibility among licenses. This lattice garantees a partial order over licenses.

ACKNOWLEDGMENTS

Authors thank Matthieu Perrin for his helpful comments on this work.

REFERENCES

[1] Enrico Daga, Mathieu d'Aquin, Enrico Motta, and Aldo Gangemi. October 11-15, 2015. A Bottom-up Approach for Licences Classification and Selection. In *Legal Domain And Semantic Web Applications (LeDA-SWAn) in International Semantic Web Conference (ISWC), Bethlehem, USA.*
[2] Dorothy E Denning. 1976. A Lattice Model of Secure Information Flow. *Commun. ACM* 19, 5 (1976), 236–243.
[3] GR Gangadharan, Michael Weiss, Vincenzo D'Andrea, and Renato Iannella. September 17-20, 2007. Service License Composition and Compatibility Analysis. In *International Conference on Service-Oriented Computing (ICSOC), Vienna, Austria.* 257–269.
[4] Guido Governatori, Antonino Rotolo, Serena Villata, and Fabien Gandon. October 21-25, 2013. One License to Compose Them All. A Deontic Logic Approach to Data Licensing on the Web of Data. In *International Semantic Web Conference (ISWC), Sydney, Australia.*
[5] Georgia M Kapitsaki, Frederik Kramer, and Nikolaos D Tselikas. 2017. Automating the License Compatibility Process in Open Source Software With SPDX. *Journal of Systems and Software* 131 (2017), 386–401.
[6] Marco Mesiti, Paolo Perlasca, and Stefano Valtolina. August 26-29, 2013. On the Composition of Digital Licenses in Collaborative Environments. In *Conference on Database and Expert Systems Applications (DEXA), Prague, Czech Republic.*
[7] Ravi S. Sandhu. 1993. Lattice-Based Access Control Models. *Computer* 26, 11 (1993), 9–19.
[8] Valeria Soto-Mendoza, Patricia Serrano-Alvarado, Emmanuel Desmontils, and Jose Antonio Garcia-Macias. October 11-15, 2015. Policies Composition Based on Data Usage Context. In *Consuming Linked Data (COLD) in International Semantic Web Conference (ISWC), Bethlehem, USA.*
[9] Serena Villata and Fabien Gandon. November 11-15, 2012. Licenses Compatibility and Composition in the Web of Data. In *Consuming Linked Data (COLD) in International Semantic Web Conference (ISWC), Boston, USA*, Vol. 905.

[8] A video demonstration is available at https://youtu.be/YkSWHSiD-Ps.

Benchmarking Top-K Keyword and Top-K Document Processing with T^2K^2 and $T^2K^2D^2$

Le calcul des *top-k* mots clés et documents au banc d'essais avec T^2K^2 et $T^2K^2D^2$

Ciprian-Octavian Truică
Université Polytechnique de Bucarest
Roumanie
ciprian.truica@cs.pub.ro

Jérôme Darmont
Université de Lyon, Lyon 2, ERIC EA 3083
France
jerome.darmont@univ-lyon2.fr

Alexandru Boicea
Université Polytechnique de Bucarest
Roumanie
alexandru.boicea@cs.pub.ro

Florin Rădulescu
Université Polytechnique de Bucarest
Roumanie
florin.radulescu@cs.pub.ro

Avec le développement continu et croissant des contenus textuels sur le Web, notamment (mais pas uniquement) dans les médias sociaux, l'analyse de texte est devenue un tendance lourde qui soulève encore aujourd'hui de nombreux problèmes. Parmi ces méthodes d'analyse, la recherche des *k* mots clés ou documents les plus fréquents (*top-k keywords/documents*) est très courante. Par exemple, extraire les *k* termes les plus fréquents d'un corpus permet de déterminer des tendances [5, 14] ou de détecter des événements [10] ; et découvrir les *k* documents les plus similaires à une requête est bien sûr la tâche principale des moteurs de recherche. Par ailleurs, calculer des *top-k* mots clés et documents nécessite un vocabulaire pondéré, qui peut aussi être utilisé dans de nombreux autres types d'analyse de texte comme la recherche de modèles thématiques (*topic modeling*) [2] et le regroupement (*clustering*) [1].

Afin de comparer des combinaisons de méthodes de pondération, des stratégies de calcul, ainsi que l'efficacité de différentes implémentations, les approches par bancs d'essais (*benchmarking*) sont courantes. Toutefois, les bancs d'essais dédiés aux mégadonnées (*big data*) [11, 15, 19] se focalisent essentiellement sur des opérations MapReduce et ne modélisent pas souvent des scénarios basés sur des documents textuels. De plus, les rares qui le font se restreignent au paradigme Hadoop et demeurent à des stades de méthodologie [8] ou de spécifications [7].

Par ailleurs, le calcul de pondérations au niveau de la couche applicative peut se révéler inefficace sur de gros volumes de données, car toute l'information doit être lue et traitée à différents niveaux en plus de celui du stockage. Une meilleure approche consiste à traiter l'information au niveau du stockage, en utilisant des fonctions d'agrégation et en retournant le résultat à la couche applicative.

C'est pourquoi nous avons proposé, dans un premier temps, le banc d'essais T^2K^2 (*Twitter Top-K Keywords Benchmark*), qui s'appuie sur le stockage en base de données d'un corpus de *tweets* et y associe des requêtes de complexité et de sélectivité variées [16]. Nous avons conçu T^2K^2 de manière la plus générique possible. Il permet en effet de comparer différentes méthodes de pondération, des implémentations logiques et physiques de bases de données, ainsi que des plateformes complètes d'analyse de texte, en termes d'efficacité de calcul. Dans cet article, nous poursuivons ce travail par les contributions suivantes [17].

(1) Comme le modèle de données de T^2K^2 est suffisamment générique pour prendre en compte tout type de document textuel (et pas seulement des *tweets*), nous complétons son modèle de charge par des requêtes de calcul de *top-k* documents.

(2) Nous illustrons la pertinence et la généricité de T^2K^2 par la comparaison des méthodes de pondération TF-IDF et Okapi BM25, d'une part, et leur emploi dans des implémentations relationnelles (Oracle, PostgreSQL) et orientée document (MongoDB), d'autre part.

(3) Puisque les requêtes de T^2K^2 sont analytiques par nature, nous faisons l'hypothèse qu'un modèle en étoile d'entrepôt de données peut améliorer les temps de réponses aux requêtes. En conséquence, nous proposons une évolution de T^2K^2 appelée $T^2K^2D^2$, les deux derniers « D » signifiant *Dimensional* et *Documents*, respectivement.

(4) Nous complétons nos premières expériences sur les *top-k* mots clés et documents avec le banc d'essais $T^2K^2D^2$, toujours en comparant les méthodes de pondération TF-IDF et Okapi BM25 sur des implémentations Oracle, PostgreSQL et MongoDB.

Nous avons conçu le modèle de charge de nos bancs d'essais en journalisant le travail de spécialistes en linguistique computationnelle sur une plateforme d'analyse de texte [18] et des données réelles. Après analyse et regroupement (*clustering*) des requêtes similaires, nous avons sélectionné 8 requêtes (4 pour T^2K^2, 4 pour $T^2K^2D^2$) que nous considérons suffisamment génériques pour pouvoir comparer des systèmes similaires d'analyse de texte.

Par ailleurs, il est important de remarquer que nos bancs d'essais ne s'appliquent pas à l'évaluation de systèmes de recherche d'information (RI). Ils ciblent en effet des systèmes utilisant des bases de données et construits comme des surcouches de systèmes de RI, le but étant de faciliter des tâches de fouille de texte (*text mining*) comme la classification, la recherche de modèles thématiques ou la détection d'événements. Les requêtes de nos bancs d'essais ciblent des tâches d'apprentissage automatique (*machine learning*) pour

lesquelles il est important d'analyser différents sous-ensembles d'un jeu de données pour en extraire de la connaissance.

A contrario, les systèmes de RI ne gèrent pas bien les sous-ensembles d'un corpus initial, ni les jeux de données dont la taille varie avec le temps, les poids n'étant pas recalculés. Cela induit deux problèmes de reproductibilité des résultats [12]. Premièrement, l'absence de recalcul des poids induit des erreurs dans les fonctions de classement (*ranking*) utilisées pour le calcul des *top-k* mots clés et documents. Deuxièmement, les poids doivent être recalculés à chaque fois que la taille du corpus change, ce qui provoque de nombreuses opérations d'écriture coûteuses.

L'approche base de données permet de résoudre ces problèmes de reproductibilité en calculant les poids de manière dynamique au moment de la recherche, en utilisant des champs de données (*fielded data*). Bien que les systèmes de RI utilisent aussi des champs de données [6], ils ne calculent les poids qu'une seule fois et les fonctions de classement ne les mettent à jour ni quand des sous-ensembles du corpus sont utilisés, ni quand le volume de donnée croît [3], ce qui provoque des erreurs dans le calcul des *top-k* mots clés et documents.

En revanche, calculer les poids automatiquement est parfaitement adapté à l'analyse de sous-ensembles du corpus initial ou à des corpus en évolution, comme par exemple des flux de données [4]. Les systèmes de gestion de bases de données (SGBD) étant conçus pour gérer de grands volumes de données avec un haut débit et capables de manipuler facilement des volumes de données croissants grâce à des opérations CRUD (*Create, Read, Update, Delete*) optimisées, ils sont plus à même de prendre en charge des jeux de données dont la taille évolue et d'analyser des sous-ensembles du corpus.

Nos résultats expérimentaux sont synthétisés dans le Tableau 1, qui indique le SGBD qui a obtenu les meilleures performances de calcul des *top-k* mots clés et documents en fonction du banc d'essais et de la méthode de pondération utilisés. Les résultats détaillés sont discutés dans [17].

Table 1: Meilleur système pour les calculs *top-k*

Système de pondération	TF-IDF	Okapi BM25
T^2K^2 *top-k* mots clés	MongoDB	Oracle
T^2K^2 *top-k* documents	MongoDB	Oracle
$T^2K^2D^2$ *top-k* mots clés	Oracle	Oracle
$T^2K^2D^2$ *top-k* documents	Oracle	Oracle

Pour terminer, nous avons conçu nos bancs d'essais selon les critères de Jim Gray [9] : pertinence, portabilité, simplicité et scalabilité. Dans nos travaux futurs, nous prévoyons d'étendre de manière significative le jeu de données de T^2K^2 et $T^2K^2D^2$ pour parvenir à un volume à l'échelle des mégadonnées (scalabilité). Nous allons également adapter nos bancs d'essais pour qu'ils s'exécutent dans les environnements distribués tels qu'Hadoop et Spark (portabilité). De plus, nous allons poursuivre nos efforts de preuve de concept et de validation en comparant d'autres SGBD NoSQL que MongoDB (portabilité), afin de déterminer leurs points forts et leurs limites (pertinence). Enfin, nous avons considéré dans cet article que les méthodes de pondération TF-IDF et Okapi BM25 étaient suffisamment représentatives pour des tâches d'apprentissage automatique. Cependant, la prochaine version de nos bancs d'essais devrait en inclure d'autres (pertinence), comme la KL-divergence [13].

RÉFÉRENCES

[1] Charu C. Aggarwal and ChengXiang Zhai. 2012. A Survey of Text Clustering Algorithms. In *Mining Text Data*. Springer, 77–128. https://doi.org/10.1007/978-1-4614-3223-4_4

[2] Rubayyi Alghamdi and Khalid Alfalqi. 2015. A Survey of Topic Modeling in Text Mining. *International Journal of Advanced Computer Science and Applications* 6, 1 (2015), 147–153. https://doi.org/10.14569/IJACSA.2015.060121

[3] P. Bellot, A. Doucet, S. Geva, S. Gurajada, J. Kamps, G. Kazai, M. Koolen, A. Mishra, V. Moriceau, J. Mothe, M. Preminger, E. SanJuan, R. Schenkel, X. Tannier, M. Theobald, M. Trappett, A. Trotman, M. Sanderson, F. Scholer, and Q. Wang. 2013. Report on INEX 2013. *SIGIR Forum* 47, 2 (2013), 21–32. https://doi.org/10.1145/2568388.2568393

[4] Albert Bifet and Eibe Frank. 2010. Sentiment Knowledge Discovery in Twitter Streaming Data. In *Discovery Science*. Springer Berlin Heidelberg, 1–15. https://doi.org/10.1007/978-3-642-16184-1_1

[5] Sandra Bringay, Nicolas Béchet, Flavien Bouillot, Pascal Poncelet, Mathieu Roche, and Maguelonne Teisseire. 2011. Towards an On-Line Analysis of Tweets Processing. In *International Conference on Database and Expert Systems Applications (DEXA)*. 154–161. https://doi.org/10.1007/978-3-642-23091-2_15

[6] Matt Crane, J. Shane Culpepper, Jimmy Lin, Joel Mackenzie, and Andrew Trotman. 2017. A Comparison of Document-at-a-Time and Score-at-a-Time Query Evaluation. In *10th ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 201–210. https://doi.org/10.1145/3018661.3018726

[7] Jaume Ferrarons, Mulu Adhana, Carlos Colmenares, Sandra Pietrowska, Fadila Bentayeb, and Jérôme Darmont. 2014. PRIMEBALL : a Parallel Processing Framework Benchmark for Big Data Applications in the Cloud, In 5th TPC Technology Conference on Performance Evaluation and Benchmarking (TPC-TC). *LNCS* 839, 109–124. https://doi.org/10.1007/978-3-319-04936-6_8

[8] Anne E. Gattiker, Fadi H. Gebara, H. Peter Hofstee, J. D. Hayes, and A. Hylick. 2013. Big Data text-oriented benchmark creation for Hadoop. *IBM Journal of Research and Development* 57, 3/4 (2013), 10 :1–10 :6. https://doi.org/10.1147/JRD.2013.2240732

[9] Jim Gray. 1993. *The Benchmark Handbook for Database and Transaction Systems (2nd Edition)*. Morgan Kaufmann.

[10] Adrien Guille and Cécile Favre. 2015. Event detection, tracking, and visualization in Twitter : a mention-anomaly-based approach. *Social Network Analysis and Mining* 5, 1 (2015), 18. https://doi.org/10.1007/s13278-015-0258-0

[11] Shengsheng Huang, Jie Huang, Jinquan Dai, Tao Xie, and Bo Huang. 2010. The HiBench benchmark suite : Characterization of the MapReduce-based data analysis. In *Workshops Proceedings of the 26th International Conference on Data Engineering (ICDE)*. 41–51. https://doi.org/10.1109/ICDEW.2010.5452747

[12] Jimmy Lin, Matt Crane, Andrew Trotman, Jamie Callan, Ishan Chattopadhyaya, John Foley, Grant Ingersoll, Craig Macdonald, and Sebastiano Vigna. 2016. Toward Reproducible Baselines : The Open-Source IR Reproducibility Challenge. In *Advances in Information Retrieval*. Springer, 408–420. https://doi.org/10.1007/978-3-319-30671-1_30

[13] Fiana Raiber and Oren Kurland. 2017. Kullback-Leibler Divergence Revisited. In *Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval*. ACM, 117–124. https://doi.org/10.1145/3121050.3121062

[14] Franck Ravat, Olivier Teste, Ronan Tournier, and Gilles Zurfluh. 2008. Top_Keyword : an Aggregation Function for Textual Document OLAP. In *10th International Conference on Data Warehousing and Knowledge Discovery (DaWaK)*. 55–64. https://doi.org/10.1007/978-3-540-85836-2_6

[15] Transaction Processing Performance Council. 2016. TPC Express Benchmark HS Standard Specification Version 1.4.2. http://www.tpc.org

[16] Ciprian-Octavian Truică and Jérôme Darmont. 2017. T^2K^2 : The Twitter Top-K Keywords Benchmark, In 21st European Conference on Advances in Databases and Information Systems (ADBIS). *CCIS*, 21–28. https://doi.org/10.1007/978-3-319-67162-8_3

[17] Ciprian-Octavian Truică, Jérôme Darmont, Alexandru Boicea, and Florin Rădulescu. 2018. Benchmarking Top-K Keyword and Top-K Document Processing with T^2K^2 and $T^2K^2D^2$. *Future Generation Computer Systems* 85 (August 2018), 60–75. Special issue on Big Data Benchmarking.

[18] Ciprian-Octavian Truică, Jérôme Darmont, and Julien Velcin. 2016. A Scalable Document-based Architecture for Text Analysis, In International Conference on Advanced Data Mining and Applications (ADMA). *LNAI* 10086, 481–494. https://doi.org/10.1007/978-3-319-49586-6_33

[19] Lei Wang, Jianfeng Zhan, Chunjie Luo, Yuqing Zhu, Qiang Yang, Yongqiang He, Wanling Gao, Zhen Jia, Yingjie Shi, Shujie Zhang, Chen Zheng, Gang Lu, Kent Zhan, Xiaona Li, and Bizhu Qiu. 2014. BigDataBench : A big data benchmark suite from internet services. In *20th IEEE International Symposium on High Performance Computer Architecture (HPCA)*. 488–499. https://doi.org/10.1109/HPCA.2014.6835958

Des données particulières : les données de la recherche en Sciences Humaines et Sociales

Tiphaine Van De Weghe
Marie-Noëlle Bessagnet
Philippe Roose

ABSTRACT

Les données de la recherche en Sciences Humaines et Sociales (SHS) sont au cœur de tous travaux des chercheurs. Gérer ces données tout au long du cycle de vie nécessite un travail commun entre chercheurs en SHS produisant et exploitant les données qui ont pour leur part été structurées par les chercheurs en Informatique. Afin de répondre aux problématiques posées par les traitements sur les données de la recherche en SHS depuis leur recueil jusqu'à leur valorisation, nous proposons dans cet article un cadre conceptuel et méthodologique, une chaîne de traitements ainsi que des outils de traitement. Nous aborderons deux types de données de la recherche : des documents textuels, le plus souvent anciens, que les chercheurs en SHS doivent retranscrire en premier lieu avant tout traitement informatique et/ou statistique et des données plus hétérogènes issues du Patrimoine Culturel Immatériel (PCI).

Decomposition and Sharing for User-defined Aggregation: from Theory to Practice

Chao Zhang
Farouk Toumani
Emmanuel Gangler

ABSTRACT

We study the problems of decomposition and sharing for user-defined aggregate functions in distributed and parallel computing.Aggregation usually needs to satisfy the distributive property to compute in parallel, and to leverage optimization in multidimensional data analysis and conjunctive query with aggregation.However, this property is too restricted to allow more aggregation to benefit from these advantages. We propose for user-defined aggregation functions a formal framework to relax the previous condition, and we map this framework to the MRC, an efficient computation model in MapReduce, to automatically generate efficient partial aggregation functions. Moreover, we identify the sound conditions for sharing the result of practical user-defined aggregation without scanning base data, and propose a hybrid solution, the symbolic index, pull-up rules, to optimize the sharing process.

Quality Metrics For RDF Graph Summarization

Mussab Zneika
ETIS, UMR 8051 Paris Seine University, University of Cergy-Pontoise, ENSEA, CNRS
Pontoise, France
Mussab.Zneika@ensea.fr

Dan Vodislav
ETIS, UMR 8051 Paris Seine University, University of Cergy-Pontoise, ENSEA, CNRS
Pontoise, France
Dan.Vodislav@u-cergy.fr

Dimitris Kotzinos
ETIS, UMR 8051 Paris Seine University, University of Cergy-Pontoise, ENSEA, CNRS
Pontoise, France
Dimitrios.Kotzinos@u-cergy.fr

ABSTRACT

In the recent years, RDF (Resource Description Framework) has been widely adopted to represent diverse datasets. Thus, the amount of RDF data available increases fast both in size and complexity, making available RDF Knowledge Bases (KBs) with millions or even billions of triples something usual, e.g. more than 1000 datasets are now published as part of the Linked Open Data (LOD) cloud, which contains more than 62 billion RDF triples, forming big and complex RDF data graphs. This explosion of size, complexity and number of available RDF Knowledge Bases (KBs) and the emergence of Linked Datasets made querying and understanding of the data in these KBs difficult and led to the need of summarization methods in order to extract concise and meaningful information from RDF Knowledge Bases which produces a succinct overview of these RDF KBs. But these proposed methods come from various scientific backgrounds ranging from generic graph summarization to explicit RDF graph summarization and from using rules to using the graph structure and thus and they produce different results while applied on the same KB. Additionally, there is no clear single definition of RDF summary, and not a single but many approaches to build RDF summaries. And given that RDF graph summarization plays a critical role in many different applications, including graph exploration and visualization, highlighting communities and query optimization, we need to have a way to compare the results that we get and decide which is the most suitable for our case.

One fundamental difficulty towards understanding the quality of the different generated summaries is a lack of widely accepted evaluation criteria or an extensive empirical evaluation. This leads to the necessity of a method to compare and evaluate the quality of the produced summaries. This method would allow a better understanding of the quality of the different summaries and facilitate their comparison and decide on their quality and best-fitness for specific tasks. Despite the existence of a good number of RDF summarization approaches, there is a very little effort in the literature into addressing in a comprehensive and coherent way the problem of evaluating these summaries against different criteria and have some mathematical metrics to describe the quality of the results. Only sparse efforts have been reported, usually tailored to a specific method or algorithm. So in this work, we provide a comprehensive Quality Framework for RDF Graph Summarization to cover the gap that exists in the literature. This framework would allow a better, deeper and more complete understanding of the quality of the different summaries and facilitate their comparison.

The framework is independent of the way RDF summarization algorithms work and makes no assumptions on the type or structure neither of the input nor of the final results. We provide metrics that help us understand not only if this is a valid summary but also if a summary is better than another in terms of the specified quality characteristics. In order to achieve this, we compare the summaries against two levels of information possibly available for a RDF KB: the level of the ideal summary of the KB and the level of the instances contained by the KB. For the first level where an ideal summary is available, either because it has been proposed by a human expert or because we can assume that an existing schema represents perfectly the data graph, we compute how close the proposed summary is to the ideal solution by computing its precision, recall and F-measure against the ideal solution using novel customized definitions for precision and recall. We compute the precision and recall for each class and each neighborhood (properties and attributes having as domain that class) of the produced summary against the ideal one. We also compute the precision and recall of the whole summary against the ideal one. The first will capture the quality of the summary at the local (class) level, while the second will give us the overall quality in terms of classes' and properties/attributes' precision and recall. For the second level, if the ideal summary is not available or usually in addition to it, we are computing the coverage, i.e. if the existing instances (including both class and property instances) are covered (i.e. can be retrieved) and at which degree by the proposed summary. Again we define and compute the summary precision, recall and F-measure against the data contained in the original KB. One more important aspect that we also consider is the connectivity of the summary i.e is the summary a connected graph? So, we propose a new metric to compute the connectivity of the proposed summary compared to the ideal one, so we will know how the compare and it is up to the user to decide if something is better or worse. The connectivity metric can provide an indication of the quality of the graph summary by measuring whether or not the summary is a connected graph, since in many cases (for example, when we want to query the KB) this is an important factor. Ideally, a summary should be a connected graph.

We evaluated the Quality Framework using a set of ten different and diverse datasets (RDF KBs) with different characteristics like size (in number of triples), number of classes and properties, number of instances, etc. We used three different algorithms for RDF Graph Summarization picked from the literature (that work in substantially different ways) and used the Quality Framework to

understand the different behavior of each algorithm when summarizing each KB. Results show that the Quality Framework captures different behaviors even at the detail level and thus provides to the user adequate information to decide which algorithms is more suitable for each use case.

In summary we can say that the proposed framework allows for understanding the quality of the different summaries at different levels. The users can look in all metrics or in the specific metrics that better fit to the task for which they need to pick a summary and make an informed decision on which algorithm to use. The paper can cited as "Mussab Zneika, Dan Vodislav, Dimitris Kotzinos (2019). *"Quality Metrics For RDF Graph Summarization"*, Special Issue on Intelligent Exploration of Semantic Data, Semantic Web Journal, to appear." and the corresponding software is available under an open source license at https://github.com/ETIS-MIDI/Quality-Metrics-For-RDF-Graph-Summarization.

KEYWORDS

Quality framework; Quality metrics; RDF Summarization; Linked Open Data; RDF Query processing

Résumés des articles courts

L'approche des Versions de Bases de données: vue d'ensemble et domaines d'application

En hommage à Geneviève Jomier (1948 - 2018)

Talel Abdessalem
LTCI, Télécom ParisTech, Université Paris-Saclay, France
nom.prenom@telecom-paristech.fr

Claudia Bauzer Medeiros
University of Campinas, Institute of Computing (IC - UNICAMP)
Campinas, Brazil
cmbm@ic.unicamp.br

Wojciech Cellary
Poznan University of Economics
Poznan, Poland
cellary@kti.ue.poznan.pl

Stéphane Gançarski
LIP6 - U.P.M.C
Paris, France
nom.prenom@lip6.fr

Khaled Jouini
MARS Research Lab LR17ES05, ISITCom, University of Sousse
Sousse, Tunisia
nom.prenom@isitc.u-sousse.tn

Maude Manouvrier
Université Paris-Dauphine, PSL Research University, CNRS UMR [7243] LAMSADE, France
nom.prenom@dauphine.fr

Marta Rukoz
Université Paris-Dauphine, PSL Research University, CNRS UMR [7243] LAMSADE, France
nom.prenom@dauphine.fr

Michel Zam
Université Paris-Dauphine, PSL Research University, CNRS UMR [7243] LAMSADE, France
zam@dauphine.fr

1 INTRODUCTION

En 1990, W. Cellary et G. Jomier ont proposé l'approche des VBD (Version de Bases de Données) [5], qui permet de gérer des bases de données multiversions, c-à-d celles dans lesquelles plusieurs versions d'un ensemble de données coexistent. Depuis lors, ce modèle, sa théorie et ses algorithmes ont été adoptés dans une multitude de projets de recherche et de publications, et ont été appliqués à divers domaines d'applications, en particulier ceux dans lesquels il est nécessaire de garder la trace de l'évolution parallèle ou (spatio) temporelle des états du monde modélisé. Cet article présente une vue d'ensemble de l'approche des VBD et de certains des projets de recherche associés au cours des trois dernières décennies. Cet article a été écrit en hommage à Geneviève Jomier, professeur émérite de l'Université de Paris-Dauphine, qui nous a quittée en mars 2018.

2 L'APPROCHE DES VBD

Classiquement, une base de données est monoversion, représentant un seul état du monde modélisé; chaque donnée y a une unique valeur. Dans l'approche des VBD, une base de données multiversion regroupe plusieurs états du monde, qui sont des variantes ou des évolutions dans le temps du monde modélisé. Chaque donnée a, dans ce cas, plusieurs versions, la valeur de la donnée pouvant changer d'une version à l'autre. Chaque version de base de données – i.e., chaque VBD – représente un état cohérent ou une configuration du monde modélisé [7].

La figure 1 donne un exemple simplifié d'une base de données multiversion dont les données (la photo, les coordonnées, les paragraphes décrivant la formation et les expériences professionnelles) modélisent le contenu d'un Curriculum Vitae (CV). La base contient 4 VBD, identifiées de v_1 à v_4, chacune correspondant à une version du CV. La VBD v_2 a été créée à partir de v_1, en ajoutant une photo N&B et en étoffant la partie expériences professionnelles ; elle correspond à la version longue du CV, la VBD v_1 correspondant à la version courte. La VBD v_3 a été créée à partir de v_2, en modifiant la photo N&B en couleur. La VBD v_4, créée à partir de v_1, correspond à la version anglaise du CV.

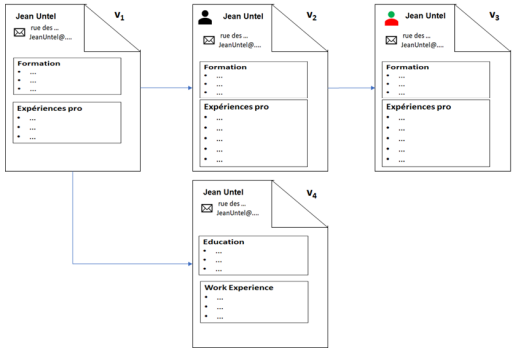

Figure 1: Une base de données multiversion de CV (reprise de [3]).

Une VBD est identifiée par un identificateur nommé v. Une donnée multiversion est associée à un identificateur immuable, d, permettant d'identifier de manière unique la donnée. Une VBD

contient exactement une version logique de chaque donnée multiversion. Une version logique possède un identificateur et une valeur. L'identificateur de la version logique de la donnée d_j dans une VBD v_i est le couple (d_j, v_i).

Une VBD est créée par dérivation d'une VBD existante, *i.e.* par copie logique. Après dérivation il est possible de mettre à jour la valeur d'une donnée dans une VBD. Les liens de dérivation entre les VBD sont enregistrés dans un arbre de dérivation.

On peut distinguer deux niveaux d'une base de données multiversion : le niveau logique (ce que voit l'utilisateur) et le niveau physique (ce qui est rèellement stocké). Au niveau physique, les versions logiques, qui partagent la même valeur *val* dans plusieurs VBD, partagent la même version physique, contenant la valeur *val*. Une table d'association fait le lien entre chaque version logique (*d*,*v*) et la version physique *p* associée qui contient la valeur *val*. Lorsqu'une donnée n'existe pas dans une VBD, sa valeur est $\perp$, signifiant "n'existe pas".

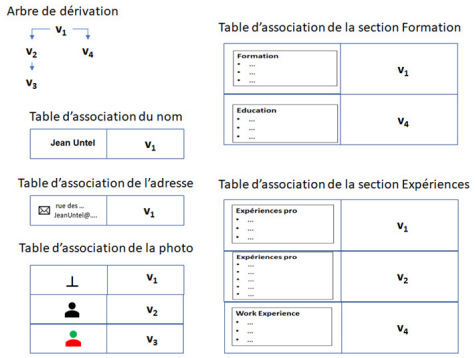

Figure 2: Le niveau physique de la base de données de la figure 1 (reprise de [3]).

La Figure 2 représente le niveau physique de la base multiversion de la Figure 1. La table d'association de la "section Formation" indique que la version physique, explicitement associée à la VBD v_1, est implicitement partagée par les VBD v_2 et v_3. La version anglaise est en revanche explicitement associée à v_4. La table d'association de la photo montre que celle-ci n'existait pas dans la VBD v_1 (sa valeur est $\perp$), et donc implicitement dans la VBD v_4, puis qu'elle est apparue en N&B dans la VBD v_2, et est devenue en couleur dans la VBD v_3.

3 DOMAINES D'APPLICATION

Le modèle des VBD est à l'origine de plusieurs initiatives de recherche, y compris celles présentées dans divers articles - par exemple, [2, 4, 7, 10, 12, 13, 15–18, 22, 23, 23, 25]. Plusieurs thèses de doctorat en informatique [1, 8, 9, 14, 19, 24], dirigées par G. Jomier, sont basées sur ce modèle. Le modèle a également inspiré des thèses et des projets de recherche - par exemple, [6, 11, 21], dans d'autres institutions.

Les VBD peuvent être appliquées à tout type de données, qu'il s'agisse d'images [14, 15], de documents [4], de données spatio-temporelles [19, 20, 22], de tracabilité de processus de construction de logiciel [24, 25].

4 CONCLUSION

Le modèle des VBD offre donc des mécanismes puissants de gestion de bases de données multiversions. La puissance du modèle VBD réside dans sa simplicité et sa généricité, s'appliquant à des situations où on veut traiter l'évolution du monde (changement climatique, migrations, santé).

Il existe encore de nombreux problèmes ouverts de recherche qui peuvent bénéficier de son utilisation pour maintenir la cohérence entre plusieurs versions du monde, en particulier dans le contexte des bases de données orientées graphe, utilisées aujourd'hui dans la modélisation des réseaux sociaux.

REFERENCES

[1] T. Abdessalem. 1997. *Approche des versions de bases de données : représentation et interrogation des versions.* Ph.D. Dissertation. Univ. Paris-Dauphine (France).
[2] T. Abdessalem and G. Jomier. 1997. VQL: A Query Language for Multiversion Databases. In *DBPL*. 160–179.
[3] T. Abdessalem, C. Bauzer Medeiros, W. Cellary, M. Manouvrier, M. Rukoz, and M. Zam. 2019. Les Versions de Bases de Données. *Livre Recherche des 50 ans de Dauphine* (2019). To appear.
[4] M. L. Ba, T. Abdessalem, and P. Senellart. 2013. Merging Uncertain Multi-Version XML Documents. In *Int. Workshop on Doc. Changes: Modeling, Detection, Storage and Visualization.*
[5] W. Cellary and G. Jomier. 1990. Consistency of Versions in Object-Oriented Databases. In *VLDB*. 432–441.
[6] L. del Val Cura. 1997. *Processing versions in geographic databases (in Portuguese).* Master's thesis. Inst. of Comp., University of Campinas – Unicamp (Brazil).
[7] S. Gançarski and G. Jomier. 2001. A framework for programming multiversion databases. *Data Knowl. Eng.* 36, 1 (2001), 29–53.
[8] S. Gançarski. 1994. *Versions et bases de données; modèle formel, supports de langage et d'interface utilisateur.* Ph.D. Dissertation. Univ. Paris XI Orsay (France).
[9] K. Jouini. 2008. *Optimisation de la localité spatiale des données temporelles et multiversions.* Ph.D. Dissertation. Univ. Paris-Dauphine (France).
[10] K. Jouini and G. Jomier. 2007. Indexing multiversion databases. In *ACM CIKM*. 915–918.
[11] J. S. C. Longo. 2013. *Management of Integrity Constraints for Multi-scale geospatial Data.* Master's thesis. Inst. of Comp., University of Campinas – Unicamp (Brazil). Supervisor C. B. Medeiros.
[12] J. Savio C. Longo, L. Camargo, C.Bauzer Medeiros, and A. Santanche. 2012. Using the DBV model to maintain versions of multi-scale geospatial data. In *SeCoGIS, LNCS 7518.*
[13] J. Savio C. Longo and C. Bauzer Medeiros. 2013. Providing multi-scale consistency for multi-scale geospatial data . In *SSDBM.*
[14] M. Manouvrier. 2000. *Objets Similaires de Grande Taille dans les Bases de Données.* Ph.D. Dissertation. Univ. Paris-Dauphine (France).
[15] M. Manouvrier, M. Rukoz, and G. Jomier. 2002. Quadtree representations for storage and manipulation of clusters of images. *Image Vision Comput.* 20, 7 (2002), 513–527.
[16] C. Bauzer Medeiros, M.-J. Bellosta, and G. Jomier. 1996. Managing Multiple Representations of Georeferenced Elements. In *DEXA*. 364–371.
[17] C. Bauzer Medeiros, M. Joliveau, G. Jomier, and F.de Vuyst. 2010. Managing sensor traffic data and forecasting unusual behaviour propagation. *Geoinformatica* 14 (2010), 279–305.
[18] C. Bauzer Medeiros and G. Jomier. 1993. Managing Alternatives and Data Evolution in GIS. In *ACM Workshop on Advances in Geographic Info. Sys.* 36–39.
[19] M.-A. Peerbocus. 2001. *Gestion de l'évolution spatiotemporelle dans une base de données géographiques.* Ph.D. Dissertation. Univ. Paris-Dauphine (France).
[20] M. A. Peerbocus, C; Bauzer Medeiros, G. Jomier, and A. Voisard. 2004. A System for Change Documentation Based on a Spatiotemporal Database. *GeoInformatica* 8, 2 (2004), 173–204.
[21] M. S. Pierre. 2007. *Access Control in Multiversion Databases (in Portuguese.* Master's thesis. Inst. of Comp., University of Campinas – Unicamp (Brazil).
[22] A. Santanchè, J. Sávio C. Longo, G. Jomier, M. Zam, and C. Bauzer Medeiros. 2014. Multi-focus Research and Geospatial Data - anthropocentric concerns. *JJIDM* 5, 2 (2014), 146–160.
[23] A. Voisard, C. Bauzer Medeiros, and Genevieve Jomier. 2000. Database Support for Cooperative Work Documentation. In *COOP*. 275–290.
[24] M. Zam. 1998. *Contribution à la traçabilité du processus de conception en génie logiciel.* Ph.D. Dissertation. Univ. Paris IX Dauphine (France).
[25] M. Zam, G. Dodinet, and G. Jomier. 2011. Software objects fairy tales: merging design and runtime objects into the cloud with mydraft. In *ACM SIGPLAN OOPSLA.*

Comptage des triangles avec mises à jour

Ahmet Kara
Hung Ngo
Milos Nikolic
Dan Olteanu
Haozhe Zhang

ABSTRACT

Nous considérons le problème de la maintenance incrémentale du nombre de triangles dans une base de données lorsque celle-ci peut être mise à jour. Les approches de maintenance incrémentale existantes nécessitent un temps linéaire en la taille de la base de données. Nous proposons une approche qui présente un compromis espace-temps et où le temps de maintenance du nombre de triangles peut être proportionnel à la racine carrée de la taille de la base de données. Ce temps de maintenance peut être prouvé optimal dans le pire des cas, moyennant des hypothèses standards de complexité algorithmique. Les approches incrémentales existantes peuvent être récupérées en tant que cas particuliers, mais sous optimaux, de notre approche.

Réparation des données IoT manquantes avec une régression linéaire multiple incrémentale

Tao Peng
Aix Marseille Univ, Université de Toulon, CNRS, LIS, Marseille, France
tao.peng@etu.univ-amu.fr

Sana Sellami
Aix Marseille Univ, Université de Toulon, CNRS, LIS, Marseille, France
sana.sellami@univ-amu.fr

Omar Boucelma
Aix Marseille Univ, Université de Toulon, CNRS, LIS, Marseille, France
omar.boucelma@univ-amu.fr

ABSTRACT

Dans cet article, nous traitons le problème lié des données manquantes dans le domaine IoT(Internet of Things) [4]. Plus spécifiquement, nous proposons le modèle ISTM (Incremental Space-Time-based Model) pour la réparation des valeurs manquantes dans un flux de données IoT en temps réel. ISTM est basé sur la régression linéaire multiple incrémentale [2, 5], qui traite les données comme suit : lors de l'arrivée de nouvelles données, ISTM met rapidement à jour le modèle en modifiant des matrices intermédiaires (avec les données arrivées) au lieu d'accéder à tout l'historique. Si une valeur manquante est détectée, ISTM fournit une estimation de cette valeur manquante en fonction de ses données références. Les données références consistent en des données récentes historiques du capteur responsable de la valeur manquante ainsi que des données issues des capteurs voisins [3]. Cet article évalue et compare les performances de ISTM aves des techniques existantes, en utilisant des données de trafic réelles [1].

CCS CONCEPTS

• **Information systems → Sensor networks**; **Data streaming**;
• **Computing methodologies → Learning linear models**.

KEYWORDS

Qualité des Données, Réparation des Données, Régression Linéaire

REFERENCES

[1] Muhammad Intizar Ali, Feng Gao, and Alessandra Mileo. 2015. CityBench: A Configurable Benchmark to Evaluate RSP Engines Using Smart City Datasets. In *In proceedings of ISWC 2015 - 14th International Semantic Web Conference*. W3C, Bethlehem, PA, USA, 374–389.

[2] Wang Huiwen, Wei Yuan, and Huang Lele. 2014. Incremental algorithm of multiple linear regression model. *Journal of Beijing University of Aeronautics and Astronsutics* 40, 11 (2014), 1487–1491.

[3] PAN Liqiang, LI Jianzhong, LUO Jizhou, et al. 2010. A Multiple-Regression-Model-Based Missing Values Imputation Algorithm in Wireless Sensor Network. *Journal of Computer Research and Development* 33, 1 (2010), 1–11.

[4] Somasundaram RS and Nedunchezhian R. 2011. Evaluation of three simple imputation methods for enhancing preprocessing of data with missing values. *International Journal of Computer Applications, Vol21* 21, 10 (2011).

[5] Maximilian Schleich, Dan Olteanu, and Radu Ciucanu. 2016. Learning linear regression models over factorized joins. (2016), 3–18.

Indexation et interrogation de séries temporelles massivement distribuées

Djamel-Edine Yagoubi
djamel-edine.yagoubi@inria.fr
Inria - Univ. Montpellier - Lirmm
France

Reza Akbarinia
reza.akbarinia@inria.fr
Inria - Univ. Montpellier - Lirmm
France

Florent Masseglia
florent.masseglia@inria.fr
Inria - Univ. Montpellier - Lirmm
France

Themis Palpanas
themis@mi.parisdescartes.fr
Univ. Paris Descartes
France

RÉSUMÉ

L'indexation est cruciale pour de nombreuses tâches d'analyse de données qui reposent sur un traitement efficace des requêtes de similitude. Par conséquent, l'indexation de grands volumes de séries temporelles, ainsi que le traitement des requêtes de similitude à haute performance, sont devenus des sujets d'intérêt majeur. Pour de nombreuses applications dans divers domaines, la quantité de données à traiter peut cependant être difficile à traiter pour une seule machine, ce qui rend les solutions d'indexation centralisée existantes inefficaces. Nous proposons une solution d'indexation parallèle qui s'adapte à des milliards de séries temporelles (ou de vecteurs à haute dimension, en général), et une stratégie de traitement des requêtes parallèle qui exploite efficacement l'index à partir d'un ensemble de requêtes. Nos expériences, tant sur des données synthétiques que sur des données du monde réel, illustrent que notre algorithme de création d'index fonctionne sur 4 milliards de séries temporelles en moins de 5 heures, alors que les algorithmes centralisés de la littérature ne passent pas à l'échelle et sont limités à 1 milliard de séries temporelles, pour lesquelles ils ont besoin de plus de 5 jours pour construire l'index. De plus, notre algorithme d'interrogation distribué est capable de traiter efficacement des millions de requêtes sur des collections de milliards de séries temporelles, grâce à un mécanisme efficace d'équilibrage de charge.

KEYWORDS

Séries temporelles, Indexation, Interrogation, Calcul distribué, Parallélisme

1 INTRODUCTION

De nos jours, les individus sont en mesure de surveiller divers indicateurs pour leurs activités personnelles (par exemple, grâce à des compteurs ou des robinets connectés pour la consommation d'électricité ou d'eau), ou professionnelles (par exemple, grâce aux capteurs installés sur les plantes par les agriculteurs). La technologie des capteurs s'améliore également avec le temps et le nombre de capteurs augmente, par exemple, dans les études financières et sismiques. Il en résulte une production de données volumineuses et complexes, généralement sous la forme de séries temporelles (Time Series ou *TS* en bref) [4, 6, 9, 10, 13–15, 18] qui représentent un défi pour la découverte de connaissances. Avec des séries temporelles aussi complexes et massives, la recherche rapide et précise de similitude est la clé pour effectuer de nombreuses tâches d'exploration de données comme la découverte de Shapelets, de Motifs, la Classification ou le Clustering [8, 13, 22].

Afin d'améliorer les performances de ces requêtes de similitude, l'indexation est l'une des techniques les plus populaires [3], qui a été utilisée avec succès dans divers contextes et applications [1, 2, 5, 7, 16, 17, 20]. Bien que des études récentes aient montré que les scans séquentiels peuvent être très efficaces dans certains cas [13, 19], ces techniques ne sont utiles que lorsque la base de données est constituée d'une seule série temporelle longue, et les réponses aux requêtes ne sont que de petites sous-séquences de cette longue série. De telles approches, cependant, ne sont pas utiles dans le cas général

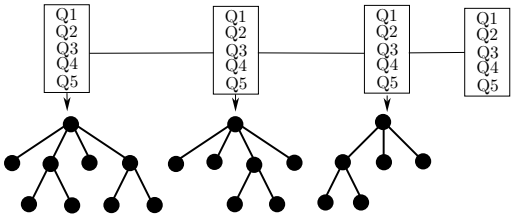

Figure 1: Implémentation simple : le lot de requêtes est dupliqué sur tous les nœuds de calcul.

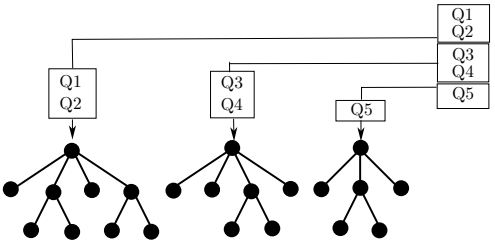

Figure 2: Répartition idéale des séries temporelles dans les nœuds de l'index : chaque requête est envoyée uniquement à la partition concernée.

d'interrogation d'une base de données mixte de nombreuses petites séries temporelles [21] (*e.g.*, en neurosciences, ou applications industrielles [10]), ce qui est l'objet de cette étude. Par conséquent, l'indexation est nécessaire afin de soutenir efficacement les tâches d'analyse de données qui impliquent des requêtes ad hoc.

Dans ce travail, nous nous concentrons sur le problème de la recherche de similitude dans de telles masses de séries temporelles [1] au moyen de la construction et de l'utilisation d'indices qui passent à l'échelle.

Malheureusement, créer un index sur des milliards de séries temporelles en utilisant des approches centralisées traditionnelles prend beaucoup de temps [11, 12]. De plus, une construction naïve de l'indice sur l'environnement parallèle peut conduire à de mauvaises performances d'interrogation. Ceci est illustré dans les figures 1 et 2, où un index est calculé et stocké sur un système de fichiers distribué. L'index se présente sous la forme d'un arbre, où chaque feuille contient des séries temporelles (ou id/adresses des séries temporelles sur le disque). Considérons maintenant que l'ensemble de données des séries temporelles est naïvement divisé sur les W nœuds distribués (figure 1). Ensuite, pour une nouvelle requête q, nous ne savons pas quelle partition peut contenir la meilleure réponse à cette requête (c'est-à-dire quelle série temporelle, dans l'ensemble de données distribué, est la plus similaire à q). Ce n'est pas un problème avec une seule requête. Mais lorsque nous traitons un lot de requêtes, alors la puissance de calcul parallèle est juste sous exploitée par une approche aussi naïve. Fondamentalement, un lot de requêtes B doit être dupliqué et traité séquentiellement sur chaque noeud.

Cependant, au moyen d'une stratégie dédiée, où chaque requête de B pourrait être orientée vers la bonne partition (la partition qui doit correspondre à la requête), la charge de travail pour répondre à cette requête peut être considérablement réduite. La figure 2 montre un cas idéal, où B est divisé en W sous-ensembles qui sont réellement traités en parallèle. Notre objectif est d'atteindre une distribution aussi idéale de la construction d'index et du traitement des requêtes dans des environnements massivement distribués.

Nous proposons une solution parallèle pour construire l'indice iSAX de l'état de l'art [2] sur des milliards de séries temporelles en tirant le meilleur parti de l'environnement parallèle en distribuant soigneusement la charge de travail. Notre solution tire parti de la puissance de calcul des systèmes distribués en utilisant des frameworks parallèles tels que MapReduce ou Spark. Nous fournissons des stratégies et des algorithmes dédiés pour une combinaison de parallélisme et de techniques d'indexation, pour de meilleures performances de requêtes.

Nous proposons un algorithme de construction d'indices parallèles qui tire parti des environnements distribués pour construire efficacement des indices basés sur iSAX sur de très grands volumes de séries temporelles (ou de vecteurs à haute dimension, en général).

Nous avons implémenté nos algorithmes de construction d'index et de traitement des requêtes, et évalué leur performance sur de grands volumes de données (jusqu'à 4 milliards de séries, pour un volume total de 6 Téraoctets), en utilisant des données synthétiques et réelles avec des séquences et des vecteurs. Nos expériences illustrent les avantages de notre algorithme avec un temps d'indexation inférieur à 2 heures pour plus d'un milliard de séries, alors que l'algorithme centralisé de l'état de l'art nécessite plus de 5 jours.

Nous proposons également un algorithme de traitement des requêtes parallèles qui, dans le cadre d'une requête, exploite les processeurs disponibles du système distribué pour répondre à la requête en parallèle en utilisant l'index parallèle construit. Comme l'illustrent nos expériences, et grâce à notre stratégie d'interrogation distribuée, notre approche est capable de traiter 10 millions de requêtes en moins de 140 secondes, alors que l'algorithme centralisé de l'état de l'art nécessite presque 2300 secondes.

1. Notons que les techniques proposées s'appliquent également aux vecteurs de grande dimension.

RÉFÉRENCES

[1] Ira Assent, Ralph Krieger, Farzad Afschari, and Thomas Seidl. 2008. The TS-tree : Efficient Time Series Search and Retrieval. In *EDBT*.

[2] A. Camerra, J. Shieh, T. Palpanas, T. Rakthanmanon, and E. J. Keogh. 2014. Beyond one billion time series : indexing and mining very large time series collections with i SAX2+. *Knowl. Inf. Syst.* (2014).

[3] Philippe Esling and Carlos Agon. 2012. Time-series Data Mining. *ACM Comput. Surv.* 45, 1, Article 12 (Dec. 2012), 34 pages. https://doi.org/10.1145/2379776.2379788

[4] Pablo Huijse, Pablo A. Estévez, Pavlos Protopapas, Jose C. Principe, and Pablo Zegers. 2014. Computational Intelligence Challenges and Applications on Large-Scale Astronomical Time Series Databases. *IEEE Comp. Int. Mag.* 9, 3 (2014), 27–39.

[5] Haridimos Kondylakis, Niv Dayan, Kostas Zoumpatianos, and Themis Palpanas. 2018. Coconut : A Scalable Bottom-Up Approach for Building Data Series Indexes. *PVLDB* 11, 6 (2018).

[6] Michele Linardi and Themis Palpanas. 2018. ULISSE : ULtra Compact Index for Variable-Length Similarity Search in Data Series. In *ICDE*.

[7] Michele Linardi and Themis Palpanas. 2019. Scalable, Variable-Length Similarity Search in Data Series : The ULISSE Approach. *PVLDB* (2019).

[8] Michele Linardi, Yan Zhu, Themis Palpanas, and Eamonn J. Keogh. 2018. Matrix Profile X : VALMOD - Scalable Discovery of Variable-Length Motifs in Data Series. In *SIGMOD*.

[9] Michele Linardi, Yan Zhu, Themis Palpanas, and Eamonn J. Keogh. 2018. VALMOD : A Suite for Easy and Exact Detection of Variable Length Motifs in Data Series. In *SIGMOD*.

[10] Themis Palpanas. 2015. Data Series Management : The Road to Big Sequence Analytics. *SIGMOD Record* 44, 2 (2015), 47–52.

[11] Themis Palpanas. 2016. Big Sequence Management : A glimpse of the Past, the Present, and the Future. In *SOFSEM*.

[12] Themis Palpanas. 2017. The Parallel and Distributed Future of Data Series Mining. In *International Conference on High Performance Computing & Simulation, HPCS*.

[13] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh. 2012. Searching and Mining Trillions of Time Series Subsequences Under Dynamic Time Warping. In *KDD*.

[14] Usman Raza, Alessandro Camerra, Amy L. Murphy, Themis Palpanas, and Gian Pietro Picco. 2015. Practical Data Prediction for Real-World Wireless Sensor Networks. *TKDE* (2015).

[15] W.H.Baumgartner et al. G.Ponti C.R.Shrader P. Lubinski H.A.Krimm F. Mattana J. Tueller S. Soldi, V. Beckmann. 2014. Long-term variability of AGN at hard X-rays. *Astronomy & Astrophysics* (2014).

[16] J. Shieh and E. Keogh. 2009. iSAX : Disk-aware mining and indexing of massive time series datasets. *DMKD* 19, 1 (2009), 24–57.

[17] Yang W., Peng W., Jian P., Wei W., and Sheng H. 2013. A Data-adaptive and Dynamic Segmentation Index for Whole Matching on Time Series. *PVLDB* (2013).

[18] Lexiang Ye and Eamonn J. Keogh. 2009. Time series shapelets : a new primitive for data mining. In *KDD*.

[19] C. C. M. Yeh, Y. Zhu, L. Ulanova, N. Begum, Y. Ding, H. A. Dau, D. F. Silva, A. Mueen, and E. J. Keogh. [n. d.]. Matrix Profile I : All Pairs Similarity Joins for Time Series : A Unifying View That Includes Motifs, Discords and Shapelets. In *ICDM, 2016*.

[20] K Zoumpatianos, S Idreos, and T Palpanas. 2014. Indexing for Interactive Exploration of Big Data Series. In *SIGMOD Conf.*

[21] Kostas Zoumpatianos, Stratos Idreos, and Themis Palpanas. 2016. ADS : the adaptive data series index. *VLDB J.* 25, 6 (2016), 843–866.

[22] Kostas Zoumpatianos and Themis Palpanas. 2018. Data Series Management : Fulfilling the Need for Big Sequence Analytics. In *ICDE*.

Résumé des démonstrations

EGG: Framework pour la Génération de Graphes Temporels

Karim Alami
Université de Bordeaux & CNRS
LaBRI, France
karim.alami@u-bordeaux.fr

Radu Ciucanu
INSA Centre Val de Loire
radu.ciucanu@insa-cvl.fr

Engelbert Mephu Nguifo
Université Clermont Auvergne &
CNRS LIMOS, France
mephu@isima.fr

ABSTRACT

Les graphes modélisent naturellement des scénarios du monde réel où il existe des interactions entre des entités, et souvent, ces graphes ne sont pas statiques mais changent avec le temps. Ainsi, il est nécessaire de concevoir des systèmes qui opèrent sur ce type de graphes. De ce fait, la communauté a besoin de graphes de données comme ensemble de test pour évaluer leurs systèmes. Cependant, il est difficile de trouver des graphes de données réels qui correspondent exactement aux besoins de la communauté. Ainsi, nous avons développé EGG (**E**volving **G**raph **G**enerator), un framework open-source[a] permettant de générer des graphes RDF temporels basés sur des contraintes temporelles définies par l'utilisateur. Concrètement, EGG prend en entrée un graphe statique généré par gMark [2] puis génère plusieurs snapshots en gardant le même schéma (voir Figure 1). Durant la démonstration, nous présenterons les différentes contraintes que l'utilisateur peut spécifier dans EGG afin de générer des graphes temporels qui représentent des scénarios réels,la précision et l'évolutivité du générateur, ainsi que la manipulation de EGG pour comparer les performances de systèmes de traitement de graphes temporels. Nous avons déjà présenté EGG lors de la session de démonstration d'ISWC 2017 [1].

[a] https://github.com/karimalami7/EGG

REFERENCES

[1] K. Alami, R. Ciucanu, and M. N. Engelbert. Egg: A framework for generating evolving rdf graphs. In *ISWC 2017, system demo*, 2017.
[2] G. Bagan and et al. gMark: Schema-driven generation of graphs and queries. *IEEE TKDE*, 29(4):856–869, 2017.

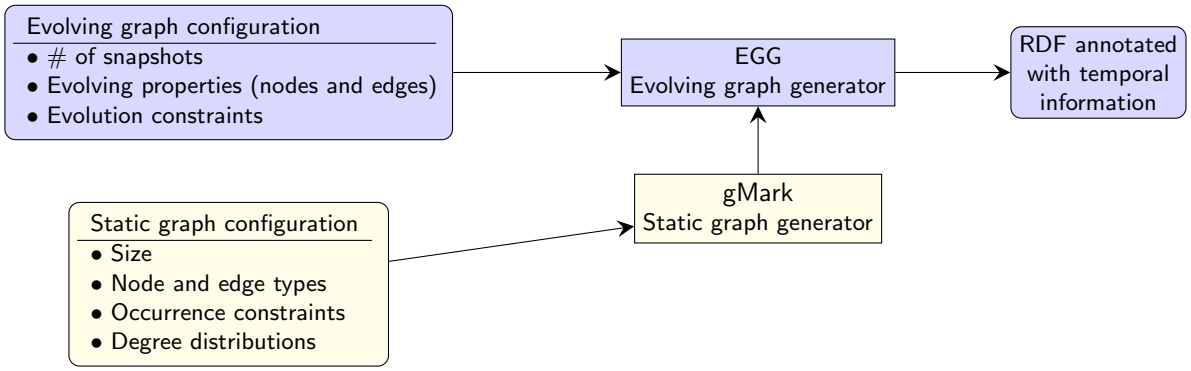

Figure 1: Architecture de EGG. Les composants en jaune font partie de gMark [2] générateur de graphes statiques. Les composants en violets font partie de EGG.

ConnectionLens: Finding Connections Across Heterogeneous Data Sources

Camille Chanial
Rédouane Dziri
Helena Galhardas
Julien Leblay
Minh-Huong Le Nguyen
Ioana Manolescu

ABSTRACT

In this work, we focus on a core data journalism problem: identifying connections across a set of heterogeneous, independently produced data sources. We are inspired by investigative journalism work, which seeks out and explores connections between individuals, organizations, companies etc. Such work requires, first, getting access to data sources, through any means (from verbal or phone communication, to paper mail, fax, and any electronic transmission method); second, analyzing this data, to find out what valuable information it may contain. For instance, consider an example raised by journalists at Le Monde, a leading French national newspaper, with which we currently collaborate: France's national elections in 2017 brought about an unprecedented ratio of first-time National Assembly elected members. Journalists scrambled to learn as much as possible about the nation's new representatives, in particular asking e.g., "what connections the new representatives have (or have had) with companies?" This question is interesting to identify areas of economic competence, but also possible conflicts of interest. By law, French representatives must disclose direct financial interests, but more indirect connections (e.g., being, for many years, a close collaborator of a company's current CEO) must be dug out by the press.

TeamBuilder: D'un moteur de recommandation de CV notés et ordonnés à l'analyse sémantique du patrimoine informationnel d'une société*

Patrice Darmon, Rabah Mazouzi, Otman Manad et Mehdi Bentounsi
Umanis, Levallois-Perret, France
{pdarmon;rmazouzi;omanad;mebentounsi}@umanis.com

RÉSUMÉ

La démonstration présente les fonctionnalités de Teambuilder, une plateforme big data couplée à un moteur de recommandation permettant d'apparier, de manière sécurisée, les meilleurs talents avec une fiche de mission.

1 INTRODUCTION

De nombreuses sociétés et en particulier les Entreprises de Services du Numérique (ESN) sont confrontées à l'enjeu de la recherche et de l'appariement automatisé et surtout fiable de Curriculum Vitae (CV), aux multiples formats[1], avec des fiches de mission, elles mêmes pouvant être de formes et de structures très diverses. A ceci doit s'ajouter: *(i)* L'expérience de l'utilisateur avec des exigences d'ergonomie et de temps de réponse des IHM ; *(ii)* La nécessité de prendre en compte des référentiels de compétences métiers et d'entreprises actualisés ; *(iii)* La nécessité de fusionner des données hétérogènes issues du système d'information de l'entreprise (Données RH, CRM, etc.) ; *(iv)* Des contraintes de sécurité de plus en plus poussées (cf. RGPD[7]).

Du point de vue des approches de traitement et en partant du postulat que les CV et fiches de missions prennent souvent la forme de documents textuels et de données non-structurées ou semi-structurées, utiliser des techniques d'intelligence artificielle dans un tel cadre constitue un challenge pour la communauté des chercheurs en fouille de texte [4].

1.1 Contexte et approches existantes

Les modes de recherche proposés par les CVthèques ou sur les sites de recherche d'emploi sont souvent syntaxiques et à base de mots clés sur le titre du CV et les compétences techniques des profils. En effet, les outils du marché ne proposent pas de moteur de recherche sémantique des compétences détaillées avec des appariements, scalables et en temps réel, et intégrant les contraintes de sécurité sur les données personnelles de profils de candidats, aux fiches de mission clients [5]. Certains travaux proposent des approches notamment à base d'ontologies [3, 6]. Cependant, ils ne présentent pas des cas d'utilisation permettant un passage à l'échelle et une évolution vers d'autres domaines métiers.

*La version longue du papier est accessible sur le lien https://hal.archives-ouvertes.fr/hal-01980539

[1]Les CV peuvent être créés à partir d'outils ou de sites Web dédiés (via, e.g., onlinecv.fr).

1.2 Motivations et contributions

Suite à ce constat quant aux solutions du marché, le projet "TeamBuilder" a été lancé début 2016 avec dès son origine, des objectifs forts *(i)* d'évolutivité vers d'autres domaines métiers, *(ii)* de scalabilité et *(iii)* de sécurité. TeamBuilder vise à créer une architecture NoSQL de type Data Lake, alimenté par plusieurs sources de données (bases de CV, fiches de missions clients, etc.). L'architecture implémentée est à base de microservices et utilise des ontologies combinées à des techniques de fouille de texte, d'apprentissage automatique et de pseudonymisation (Figure 1).

L'un des défis majeurs du projet a été de concevoir et d'intégrer de manière itérative un flux de traitement de données non-structurées à base de microservices évolutifs, et d'automatiser le déploiement avec Docker. Aujourd'hui, TeamBuilder facilite grandement la mission des équipes commerciales et des chargés de recrutement. Il propose *(i)* un moteur de recommandation sémantique de CV à base de scores statiques et dynamiques, pour *(ii)* l'appariement avec des fiches de missions clients, en temps réel basé sur une ontologie propriétaire ; Avec *(iii)* des fonctionnalités de visualisation, de chiffrement et de pseudonymisation des données.

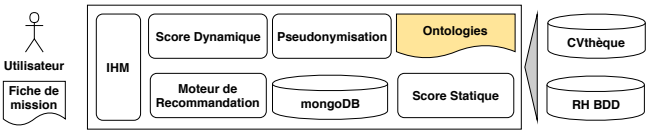

Figure 1: Architecture Fonctionnelle de TeamBuilder.

2 LA PLATEFORME TEAMBUILDER

Teambuilder propose un moteur de recommandation scalable et sécurisé de CV notés et ordonnés. Il inclut également des fonctions de désambiguïsation sémantique à base d'ontologies (compétences avec alias, clients sectorisés, fonctions, certifications, diplômes, etc.). Par exemple, l'ontologie des compétences, implémentée en collaboration avec les experts Umanis, dispose de plus de 900 compétences avec une dizaine d'attributs par compétence, en comparaison avec la norme européenne EN 16234-1:2016 qui dispose de seulement 40 compétences pour les professionnels des Technologies de l'Information.

La plateforme TeamBuilder permet de fusionner plusieurs sources de données hétérogènes dans une base de données orientée documents (i.e., MongoDB [2]). Elle est aussi constituée de plusieurs processus indépendants et faiblement couplés sous la forme d'images constituant un environnement microservices basé sur l'outil Docker.

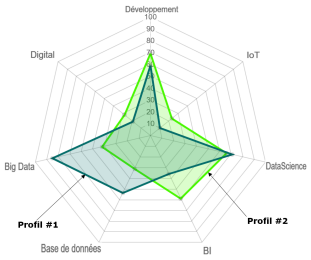

Figure 2: Radars des compétences

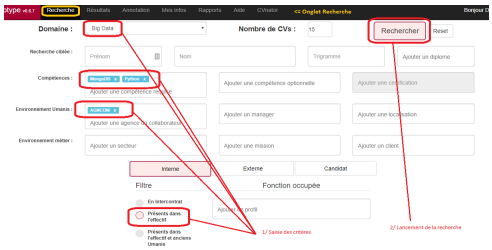

Figure 3: IHM de recherche de CVs avec filtres

2.1 Scoring

On distingue trois types de scores implémentés par TeamBuilder: **Scores Statiques :** calculés à partir des données obtenues par l'annotation des CV, notamment la fréquence et la position de la compétence dans le document et basés sur l'ontologie des compétences proposée. **Scores Radars :** scores statiques particuliers, calculés durant l'étape de préparation. Ils représentent une agrégation de différents scores statiques des compétences par domaine (e.g., Big Data, BI, etc.) et permettant de comparer visuellement plusieurs profils. La figure 2 montre les résultats de scores radars de deux profils, où une branche représente un domaine de compétences.

Scores Dynamiques : scores calculés au moment de l'exécution d'une requête de recherche en effectuant une pondération des scores statiques afin d'obtenir le nouveau score dynamique, selon certains critères de recherche (compétences requises, compétences optionnelles, etc.). Les résultats sont triés pour retourner les N meilleurs profils.

2.2 Préservation de la vie privée

En plus des bonnes pratiques de sécurité informatique, la *"Privacy by Design"* (exigé par le RGPD) rajoute une nouvelle couche de sécurité sur les données elles mêmes, permettant ainsi leur gestion et traitement de manière anonyme. Dans ce contexte, Teambuilder intègre *(i)* Un service de pseudonymisation à base de chiffrement réversible (i.e., RSA) pour préserver la confidentialité des données personnelles ; *(ii)* Un Service de génération de pseudonymes aléatoires permettant d'identifier les profils lors des échanges entre utilisateurs sans obligation de révéler les vraies identités des personnes; *(iii)* Un service de contrôle de l'intégrité des données en générant des signatures, et *(iv)* un flux complet de traitement des données implémentant une architecture *Privacy by Design* [1].

3 LES FONCTIONNALITÉS DE TEAMBUILDER

La Figure 3 présente l'IHM TeamBuilder permettant d'exécuter plusieurs types de requêtes dans le but de trouver les profils adéquats. De plus, chaque CV, compétence du CV et catégorie de compétences fait l'objet d'un score dédié. Les requêtes proposées sont:

(1) **Recherche par domaine de compétences et/ou par mots clés:** L'utilisateur peut choisir un domaine de compétences dans une liste prédéfinie, afin d'obtenir une liste de N CV notés, ordonnés pour le domaine choisi. La requête peut aussi être composée d'un filtre sur plusieurs paramètres (domaine, secteur d'activité, etc.).

(2) **Recherche par fiche de mission:** L'utilisateur a la possibilité de déposer, en mode *drag and drop*, une fiche de mission client, qui constitue la requête pour un appariement sémantique facile[2]. La recherche est lancée ensuite en vue de trouver les profils les plus adéquats.

$$distMatching(P_i, M) = |SkillsPerson_i \cap SkillsMission| + w \quad (1)$$

$$w = \frac{|SkillsPerson_i| * |(SkillsPerson_i \cap SkillsMission)|}{Nombre\ Total\ Skills} \quad (2)$$

$SkillsPerson_i$ = compétences d'une personne P_i.
$SkillsMission$ = compétences issues de fiche mission d'un client M.

(3) **Annotation de documents:** Permet à l'utilisateur de déposer, en mode drag and drop, n'importe quel(s) document(s) pour annotations en temps réel, en regard des ontologies de TeamBuilder.

4 CONCLUSION ET PERSPECTIVES

Au delà de son domaine d'application initial présenté dans ce document, les atouts de notre plate-forme sont: *(i)* son architecture innovante, *(ii)* sa scalabilité, *(iii)* sa capacité à analyser sémantiquement différents domaines métiers via des ontologies évolutives, et *(iv)* sa richesse en termes de service (scoring, annotation automatique, recommandations, pseudonymisation, etc.) et son évolutivité liées à son approche par microservices. Nos futurs travaux incluent notamment la gestion des droits et des consentements des personnes en lien avec le règlement RGPD et l'analyse du patrimoine informationnel d'une société via des ontologies adaptées.

RÉFÉRENCES

[1] M. Bentounsi et C. S. Deme. 2017. *Procédé sécurisé d'analyse externe de données d'exploitation d'une infrastructure de traitement de données.* Brevet FR3043809.

[2] Kristina Chodorow et Michael Dirolf. 2010. *MongoDB - The Definitive Guide: Powerful and Scalable Data Storage.* O'Reilly.

[3] Maryam Fazel-Zarandi et Mark S Fox. 2012. An ontology for skill and competency management. In *FOIS*, 89–102.

[4] Patrick Föll et Frédéric Thiesse. 2017. Aligning is curriculum with industry skill expectations: a text mining approach. In *25th European Conference on Information Systems, ECIS 2017, Guimarães, Portugal, June 5-10, 2017.*

[5] Viet Ha-Thuc et al. 2016. Search by ideal candidates: next generation of talent search at linkedin. In *Proceedings of the 25th International Conference on World Wide Web, Montreal, Canada, April 11-15, 2016, Companion Volume*, 195–198.

[6] Vinaya R Kudatarkar, Manjula Ramannavar et Nandini S Sidnal. 2014. An unstructured text analytics approach for qualitative evaluation of resumes. *International Journal of Innovative Research in Advanced Engineering*, 2349–2163.

[7] Colin Tankard. 2016. What the GDPR means for businesses. *Network Security*, 2016, 5–8.

[2] À noter que nous avons lancé des expérimentations en utilisant plusieurs mesures de similitude (Jaccard, Edition, Hamming, etc.) qui ne correspondent pas à nos attentes.

Spark-parSketch: A Massively Distributed Indexing of Time Series Datasets

Oleksandra Levchenko
Djamel-Edine Yagoubi
Reza Akbarinia
Florent Masseglia
Boyan Kolev
Dennis Shasha

ABSTRACT

A growing number of domains (finance, seismology, internet-of-things, etc.) collect massive time series. When the number of series grow to the hundreds of millions or even billions, similarity queries may take excessive time on a single machine. Further, naive parallelization requires time proportional to the number of time series times the size of the window for a simple query. So, we need both efficient indexing and parallelization. We propose a demonstration of Spark-parSketch, a complete solution based on sketches / random projections to efficiently perform both the parallel indexing of large sets of time series and a similarity search on them. Because our method is approximate, we explore the tradeoff between time and precision. A video showing the dynamics of the demonstration can be found by the link http://parsketch.gforge.inria.fr/video/parSketchdemo_720p.mov.

Détection d'événements historiques depuis des corpus d'archives Web

Quentin Lobbé

ABSTRACT

Corpora of web archives are wide and sparse. As the living web expands, worldwide volumes of web archives constantly increase, making difficult to identify archived webpages relevant to a specific sociological or historical study. We propose an application for detecting historical events out of a web archives corpus and discovering pertinent archived digital contents. We introduce here the usage of a new entity called Web Fragment to reduce issues related to corpus quality and consistency, and effectively guide researchers through exploration of web archives. A web fragment is defined as a semantic and syntactic subset of a given webpage and has the particularity to be indexed by its edition date (the time when the web fragment was written) instead of its archiving date (the time when its parent webpage was crawled and saved). Building on top of web fragments, we show how this application can be used to study a large archived Moroccan forum and to understand how this online collective reacted to the Arab Spring at the end of 2010.

Ulysses: an Intelligent client for replicated Triple Pattern Fragments*

Thomas Minier
LS2N, University of Nantes
Nantes, France
thomas.minier@univ-nantes.fr

Hala Skaf-Molli
LS2N, University of Nantes
Nantes, France
hala.skaf@univ-nantes.fr

Pascal Molli
LS2N, University of Nantes
Nantes, France
pascal.molli@univ-nantes.fr

Maria-Esther Vidal
TIB Leibniz Information Centre For Science and Technology University Library & Fraunhofer IAIS
Bonn, Germany
Maria.Vidal@tib.eu

ABSTRACT

Ulysses is an intelligent TPF client that takes advantage of replicated datasets to distribute the load of SPARQL query processing and provides fault-tolerance. By reducing the load on a TPF server, Ulysses improves the Linked Data availability and distributes the financial costs of queries execution among data providers. This demonstration presents the Ulysses web client and shows how users can run SPARQL queries in their browsers against TPF servers hosting replicated data. It also provides various visualizations that show in real-time how Ulysses performs the actual load distribution and adapts to network conditions during SPARQL query processing.

CCS CONCEPTS

• **Information systems → Search engine architectures and scalability**; **Distributed retrieval**;

KEYWORDS

Semantic Web, Triple Pattern Fragments, Intelligent client, Load balancing, Fault tolerance, Data Replication

1 INTRODUCTION

We proposed Ulysses [1], a replication-aware intelligent TPF client that distributes the load of SPARQL query processing across heterogeneous replicated TPF servers. Ulysses relies on a light-weighted cost-model for computing servers processing capabilities and a client-side load balancer to distribute SPARQL query processing and provides fault tolerance during query processing.

Using replicated servers, Ulysses prevents a single point of failure server-side, improves the overall availability of data, and distributes the financial costs of queries execution among data providers.

This demonstration presents the Ulysses web client. It details which informations are collected by Ulysses about servers in real-time, how the cost model is recomputed, and how the load of SPARQL query processing is balanced among replicated servers through different real-time visualizations. Finally, Ulysses reactions in presence of servers failure are illustrated.

2 OVERVIEW OF ULYSSES CLIENT

The Ulysses web client is available online at http://ulysses-demo.herokuapp.com. In order to distribute the load of SPARQL query processing across heterogeneous TPF servers hosting replicated data, it relies on three key ideas detailed in [1]. In next sections, we provides a brief overview of key ideas and how they are integrated in the Ulysses web client[1].

2.1 Replication-aware source selection

Ulysses uses a *replication-aware source selection* algorithm to identify which TPF servers can be used to distribute evaluation of triple patterns during SPARQL query processing, based on the replication model introduced in [2, 3].

This replication model allows to describe replicated datasets using *replicated fragment* and a *fragment mapping*. A fragment is defined as 2-tuple : the authoritative source of the fragment, and a triple pattern met by the fragment's triple. A fragment mapping is a function that maps each fragment to a set of TPF servers. Using these information, Ulysses is able to compute relevant sources for all triple pattern in a SPARQL query. For simplicity, in this demonstration we only consider the scenario with total replication.

2.2 A cost-model for estimating servers processing capabilities

Ulysses uses response times of HTTP requests performed against TPF servers during query processing as probes to accurately estimate the *processing capabilities* of a server. The response time of each request is used to compute the throughput of a server, *i.e.*, the number of results server per unit of time by a server. As SPARQL query processing with the TPF approach requires to send many requests to a server in order to evaluate triple patterns, Ulysses

*This work has already been accepted for publication as a demonstration at the 15th Extended Semantic Web Conference (ESWC 2018)

[1]The open-source Ulysses client is available at https://github.com/Callidon/ulysses-tpf, under MIT license.

Real-time statistics

Estimated load

Server	Access time	Page size	Throughput	Server capability factor	Estimated load
http://fragments.dbpedia.org/2015-10/en	139ms	100 triples	0.714 triple/ms	1	25%
http://fragments.mementodepot.org/dbpedia_201510	264ms	500 triples	2.392 triple/ms	3	75%

Figure 1: Ulysses cost-model, updated in real-time

can keep the servers throughputs updated in real-time without additional probing. This can also easily detect load spikes or server failures.

Servers' throughputs are used to compute a *cost-model* that define a *capability factor* of each TPF server. This capability factor determines the load distribution among servers: a server with a high capability factor has more chance to be selected to evaluate a triple pattern as detailed in Section 2.3.

Figure 1 shows a real-time estimation of servers loads during execution of a SPARQL query against two replicated TPF servers, denoted S_1 and S_2 respectively. S_1 is slightly faster to access than S_2, but as the latter serves five times more results per access (**Page size** column), S_2 has a better throughput than S_1. As, S_2 has a better capability factor than S_1, it will receive approximately 75% of the query load, while S_1 will approximately receive the remaining 25% (**Estimated load** column)

2.3 Adaptive client-side load balancing with fault tolerance

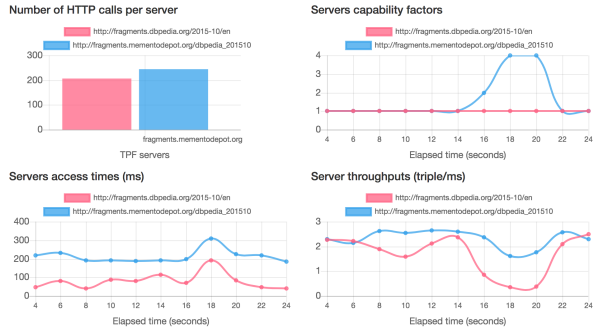

Figure 2: Metrics recorded by Ulysses and used to perform load-balancing during SPARQL query processing

Ulysses uses an *adaptive load-balancer* to perform load balancing among replicated servers. Each evaluation of a triple pattern scheduled by the client is sent to a server selected using a weighted random algorithm, inspired by the Smart clients approach [4]. The probability of selecting a server is proportional to its processing capabilities, according to Ulysses cost-model.

This probability distribution ensures that each TPF server will only process an amount of requests proportional to its processing capabilities, without concentrating all the load of query processing on the most performant servers. Ulysses load-balancer also provides *fault-tolerance*, by re-scheduling failed HTTP requests using available replicated servers.

Figure 2 shows the metrics displayed in real-time by the Ulysses web client during SPARQL query processing of Q_1, distributed among S_1 and S_2. We see that the server throughputs and capability factors of both servers remain close at the start of query processing (**Server access times** and **Servers capability factors**). However, after 18 seconds, S_1 access times increase, so S_2 became more efficient than S_1, causing its capability factor to rise. Thus, the load distribution is affected in real-time, and, at the end of query processing, we see that S_2 has received more HTTP requests (**Number of HTTP requests per server**).

3 CONCLUSION

In this demonstration, we presented the Ulysses web client that enables Web Browsers to perform client-side load balancing and provides fault tolerance when evaluating SPARQL queries against TPF servers hosting replicated data. Real-time visualizations allow to observe how Ulysses distributes the load of SPARQL query processing across replicated TPF servers according to their processing capabilities, and adapts to failures or variations in network conditions.

ACKNOWLEDGMENTS

This work is partially supported through the FaBuLA project, part of the AtlanSTIC 2020 program.

REFERENCES

[1] Thomas Minier, Hala Skaf-Molli, Pascal Molli, and Maria-Esther Vidal. 2018. Intelligent clients for replicated Triple Pattern Fragments. In *Proceedings of the 15th Extended Semantic Web Conference (ESWC 2018)*.

[2] Gabriela Montoya, Hala Skaf-Molli, Pascal Molli, and Maria-Esther Vidal. 2015. Federated SPARQL queries processing with replicated fragments. In *International Semantic Web Conference*. Springer, 36–51.

[3] Gabriela Montoya, Hala Skaf-Molli, Pascal Molli, and Maria-Esther Vidal. 2017. Decomposing federated queries in presence of replicated fragments. *Web Semantics: Science, Services and Agents on the World Wide Web* 42 (2017), 1–18.

[4] Chad Yoshikawa, Brent Chun, Paul Eastham, Amin Vahdat, Thomas Anderson, and David Culler. 1997. Using smart clients to build scalable services. In *Proceedings of the 1997 USENIX Technical Conference*. CA, 105.

ProvSQL : Gestion de provenance et de probabilités dans PostgreSQL

Pierre Senellart
Louis Jachiet
Silviu Maniu
Yann Ramusat

ABSTRACT

Cette démonstration présente ProvSQL, un module open-source pour le système de gestion de base de données PostgreSQL qui lui ajoute un support pour le calcul de la provenance et des probabilités des résultats de requêtes. Une large gamme de formalismes de provenance sont pris en charge, en particuclier tous ceux capturés par les semi-anneaux de provenance, les semi-anneaux avec monus et la where-provenance. L'évaluation probabiliste de requête est rendue possible par l'usage d'outils de compilation des connaissances, en plus d'approches standard comme l'énumération des mondes possibles et l'échantillonage de Monte-Carlo. ProvSQL prend en charge un sous-ensemble important des requêtes SQL sans agrégation.

Upsortable: Programming Top-K Queries Over Data Streams

Julien Subercaze
Christophe Gravier
Syed Gillani
Abderrahmen Kammoun
Frederique Laforest

ABSTRACT

Top-k queries over data streams is a well studied problem. There exists numerous systems allowing to process continuous queries over sliding windows. At the opposite, non-append only streams call for ad-hoc solutions, e.g. tailor-made solutions implemented in a mainstream programming language. In the meantime, the Stream API and lambda expressions have been added in Java 8, thus gaining powerful operations for data stream processing. However, the Java Collections Framework does not provide data structures to safely and conveniently support sorted collections of evolving data. In this paper, we demonstrate Upsortable, an annotation-based approach that allows to use existing sorted collections from the standard Java API for dynamic data management. Our approach relies on a combination of pre-compilation abstract syntax tree modifications and runtime analysis of bytecode. Upsortable offers the developer a safe and time-efficient solution for developing top-k queries on data streams while keeping a full compatibility with standard Java.

Résumés des articles de doctorant·e·s

Streaming saturation for large RDF graphs with dynamic schema information

Mohammad Amin Farvardin
LAMSADE, Université Paris-Dauphine
Paris, Île-de-France, France
mohammadamin.farvardin@dauphine.eu

Dario Colazzo
LAMSADE, Université Paris-Dauphine
Paris, Île-de-France, France
dario.colazzo@lamsade.dauphine.fr

Khalid Belhajjame
LAMSADE, Université Paris-Dauphine
Paris, Île-de-France, France
khalid.belhajjame@lamsade.dauphine.fr

ABSTRACT

In the Big Data era, RDF data, just like other kinds of data, is produced in high volumes. To enable reasoning over massive RDF datasets, a number of researchers have recently sought to leverage big data paradigms underlying big data platforms, notably Hadoop and Spark. While these solutions have shown to be effective, they are not instrumented to cope with RDF reasoning in environments where RDF data is produced continuously in streams.

With this in mind, we present in this paper the first solution for reasoning over large streams of RDF data. In doing so, we focus on the saturation operation. Unlike existing solutions which saturate RDF data in bulk, our solution carefully identifies the subset of the existing (and already saturated) RDF dataset that needs to be considered given the RDF statements that have recently delivered by the stream. Thereby, it performs the saturation in an incremental manner. Evaluation exercises that we performed show that our solution outperforms existing bulk-based saturation solutions, which we use as a baseline.

CCS CONCEPTS

• **Computing methodologies → Massively parallel algorithms**;

KEYWORDS

RDF saturation, RDF streams, Big Data, Spark

1 INTRODUCTION

To take full advantage of semantic data and turn it into actionable knowledge, the semantic web community has devised techniques for processing and reasoning over RDF data, see e.g., [1, 5, 7]. However, in the Big Data era, RDF data, just like many other kinds of data, is produced in high volumes. For instance, based on W3C, in City Data Fusion category, the frequency of update for *Anonymized Call Data Records* is in seconds but also it generates from 1,000 to 100,000 triples/minute/km^2 of a city[1].

A typical and fundamental operation for reasoning about RDF data is data saturation. This operation involves a set D of RDF data triples and a set of S of semantics properties, expressed in terms of either RDF Schema and/or OWL as goal to infer the implicit triples that can be derived from D by using properties in S. Data saturation is crucial in order to ensure that RDF processing and querying actually involve the *complete* informative content of an RDF dataset, without ignoring implicit information.

[1]https://www.w3.org/community/rsp/wiki/Use_cases#City_Data_Fusion

2 EXISTING APPROACHES

In order to deal with the problem of saturating massive RDF datasets, a handful of researchers sought to leverage big data paradigms (namely Map-Reduce [4]) underlying big data platforms, notably, Hadoop and Spark, (see e.g., [3, 6]). They assume that the RDF datasets subject to reasoning are given entirely prior to the saturation, and as such, are not instrumented to saturate RDF data produced continuously in streams. Indeed, using such solutions in contexts where RDF is produced in streams or in an incremental fashion, means that the saturation operation needs to be re-run from scratch every time the stream produces new RDF statements (triples).

3 SOLUTION TO OVERCOME THE STREAMING SATURATION LIMITATIONS

To overcome this limitation, in this PhD work we investigate and present new techniques for the problem of saturating streams of large volumes of RDF data. To the best of our knowledge this is the first work tackling streaming RDF saturations, and we rely on RDF Schema as a language to define property triples. As we will show, despite its simplicity, RDF Schema is rich enough to make the problem far from being trivial in order to ensure fast incremental saturation.

Specifically, we make the following contributions, obtained since the beginning of the thesis investigation.

(1) **An algorithmic solution for saturating RDF data in a stream-based fashion using Spark.** Unlike existing solutions [3, 6], which saturate RDF data in bulk, our solution carefully identifies the subset of the existing (and already saturated) RDF dataset that needs to be considered given new RDF triples that are delivered by the stream. Thereby, it performs the saturation in an incremental manner. In doing so, we leverage a technique that was proposed by Goasdoué *et al.* [2] for querying dynamic RDF datasets, in a centralized, batch fashion. Therefore, to the best of our knowledge, our approach is the first principled solution for saturating large volumes of RDF streams.
(2) **Handling dynamic RDF schema.** Our solution is equipped to gracefully deal with dynamic RDF schema. Indeed, we do

not assume that the RDF schema is entirely known a priori, but rather that schema triples can arrive at any moment in the input RDF stream, which in our case consists of a sequence RDF batches. So a newly batch delivered by the stream can convey RDF instance triples but also RDF schema triples. In this last case all the past data would need to be re-processed, but in order to limit the processing time in this case we have devised indexing techniques that store information allowing the system to reprocess a minimal amount of past microbatches in the presence of a new schema triple. Using Spark streaming, each received micro-batch is processed by multiple executors of multiple nodes in a parallel manner. The triples obtained by processing a micro-batch, therefore, yield multiple files, each produced by a different executor. Such files are stored under a time-stamped directory. The triples are organized in a way to efficiently perform RDFS entailment. In particular, the triples are organized into predicate-based and object-based triples. Predicate triples are those that may be used in subsequent iterations (micro-batches) to infer new triples based on their predicate. Whereas object-based triples are those that may be used in subsequent iterations to infer new triples based on their objects. Our solution keeps information that index those triples in the memory, thereby providing the means to efficiently locating the triples that may participate as premises to RDFS rules.

(3) **Preliminary experimental evaluation** whose results are illustrated by Figure 1, shows the effectiveness of the solution in the case of the DBpedia dataset. We assume an intial data set of 200 millions triples, both instance and schema, and then a stream of microbatches having each 160 thousands triples, including both instance and schema triples. We use state of the art bulk-based saturation, namely the one proposed by Gu *et al.* [3] as a competitor for the comparison. We conducted experiments on a portion of a local cluster with 4 nodes. Among these nodes, one is reserved to act as the master and the remaining nodes are used for computing. Each node has one Xeon CPU 2.40GHz processors with 16 cores. We used just 8 GB out of 48 GB memory of each node. Also, each node has 3 TB 7200 RPM SATA hard disks. The nodes are connected with 1 Gb/s Ethernet. The version of the underlying Spark for both systems is 2.3.0.
As it can be seen in the initial case (200M triples) Cichlid outperforms our approach as our approach needs to spend time on building our indexes. However, when the following micro batches arrive our approach is fast while Cichlid has to re-process the entire dataset. Note that for each X (size) value, the reported time consists of the time for the previous X value plus the additional time to perform saturation in order to take into account the new batch. So times are actually total times for the saturation.

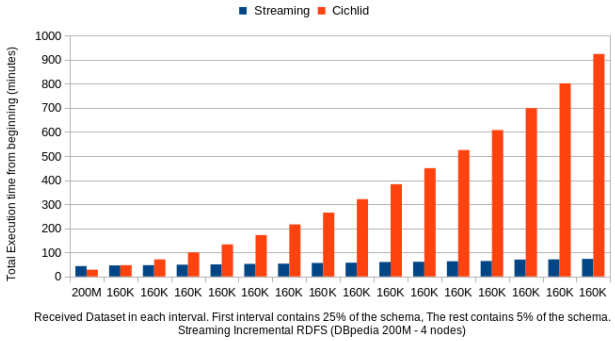

Figure 1: Incremental RDFS Streaming

4 CONCLUSION

So far, we have devised a particular incremental indexing technique to trace the streaming data regarding the RDFS rules. In contrast to the batch processing, our approach preserve a lot of computations and network communication. Currently, we are working on the proposed approach to make it stable for more RDF datasets. In the future, we have the plan to extend the indexing model for the OWL Horst rules.

REFERENCES

[1] Christian Bizer, Tom Heath, and Tim Berners-Lee. 2009. Linked data-the story so far. *International journal on semantic web and information systems* 5, 3 (2009), 1–22.
[2] François Goasdoué, Ioana Manolescu, and Alexandra Roatiş. 2013. Efficient query answering against dynamic RDF databases. In *Proceedings of the 16th International Conference on Extending Database Technology*. ACM, 299–310.
[3] Rong Gu, Shanyong Wang, Fangfang Wang, Chunfeng Yuan, and Yihua Huang. 2015. Cichlid: efficient large scale RDFS/OWL reasoning with spark. In *Parallel and Distributed Processing Symposium (IPDPS), 2015 IEEE International*. IEEE, 700–709.
[4] Jure Leskovec, Anand Rajaraman, and Jeffrey David Ullman. 2014. *Mining of massive datasets*. Cambridge university press.
[5] Markus Stocker and Evren Sirin. 2009. PelletSpatial: A Hybrid RCC-8 and RDF/OWL Reasoning and Query Engine.. In *OWLED*, Vol. 529.
[6] Jacopo Urbani, Spyros Kotoulas, Jason Maassen, Frank Van Harmelen, and Henri Bal. 2012. WebPIE: A web-scale parallel inference engine using MapReduce. *Web Semantics: Science, Services and Agents on the World Wide Web* 10 (2012), 59–75.
[7] Xiao Hang Wang, D Qing Zhang, Tao Gu, and Hung Keng Pung. 2004. Ontology based context modeling and reasoning using OWL. In *Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second IEEE Annual Conference on*. Ieee, 18–22.

Performance of Large Scale Data-Oriented Operations under the TEE Constraints

Robin Carpentier[1, 2], Nicolas Anciaux[1, 2], Iulian Sandu Popa[1, 2], Guillaume Scerri[1, 2]
[1] INRIA Saclay-IDF
Palaiseau, France
<Fname.Lname>@inria.fr
[2] DAVID Laboratory
Univ. of Versailles, France
<Fname.Lname>@uvsq.fr

1. INTRODUCTION

Trusted execution environments (TEE), like Intel SGX [3] or ARM TrustZone [1], are now embedded in end-users' personal computers and in cloud servers. TEEs offer an interesting set of security primitives, ranging from *isolation* from the host OS, *confidentiality* and *integrity* of the executed code, to *attestation* capabilities, all of this being combined with the efficiency of modern CPUs.

With such promises, TEEs obviously raise new hopes towards highly secure and efficient data management, as attested by the several recent contributions in the data management community to leverage these security properties in database processing.

EnclaveDB [8] and HardIDX [4] are examples of database works which leverage the SGX security properties to secure database operations. EnclaveDB considers a transactional database running inside an SGX enclave. EnclaveDB investigates the case of using Intel SGX to host an existing database engine (typically, the Hekaton engine) and thus convert it into a secure DBMS. HardIDX identifies the critical security operation subset of a search algorithm to be implemented within an SGX enclave to reach the same security level as the best searchable encryption schemes, but with much better performance. Such proposals mainly consider SGX as a secure black box, and identify the minimal codebase of existing database solutions to be included in SGX to improve the security level for the entire database system.

In this context, we follow a rather complementary approach consisting in opening the black box and studying the impact of Intel SGX specificities on the design of the underlying database structures and algorithms. The main limitations include the cryptographic overhead of accessing persistent data outside the TEE enclave [2], the limited RAM amount of each TEE enclave [3], the cost of external function calls [10] and memory access overheads, which can slow the computing performance by orders of magnitude compared to a regular environment, and have to be taken into account.

More precisely, our ongoing work focuses on (i) identifying the specific constraints of trusted execution environments (in particular Intel SGX) and understand how side channel attacks (timing, memory access patterns) affect classical data processing; and (ii) proposing appropriate design rules for data structures, algorithms and countermeasures, for efficient, scalable and secure personal data management using secure enclaves.

The rest of the paper is organized as follows: Section 2 introduces the main properties of Intel SGX, Section 3 gives a preliminary overview of the potential impact of these properties on database processing and the consequences on the underlying data structure and algorithms, and Section 4 sketches the outline of our ongoing work.

2. INTEL SGX PROPERTIES

Intel SGX [3] offers a mechanism for isolating processes from the rest of the system in *enclaves*. The security goals of SGX are twofold. First, under the assumption that the CPU is not compromised, programs running in enclaves cannot be observed nor affected by the rest of the system (including other programs in enclaves, and the untrusted OS). Second, enclaves should be able to prove their identity and the integrity of their code to third parties through a process called attestation. Our main concern in this work is the impact of SGX security on elementary data processing operations. However, the memory protections of enclaves have strong impact on memory management, as does the interaction of enclaves with the rest of the system. We give here an overview of the impact SGX has on code execution.

Note that the memory is protected from the rest of the system (including other enclaves) through cryptographic means, i.e., the memory belonging to an SGX enclave needs to be encrypted/decrypted before being usable. This process has a relatively low impact on program execution with respect to the more classical context. Indeed, the encryption is performed online by a dedicated hardware module and does not incur a big overhead.

A more important restriction is that enclave memory is set once and for all at enclave initialization. Additionally, enclave memory is restricted to 90MB currently. Therefore, if an enclave ever needs more memory than its initially allocated memory, it needs to store its additional memory outside of its dedicated part of the RAM. This can be done through encrypting a memory block using a key known only by the enclave and passing the page back to the OS encrypted. However, this process is much costlier than a typical RAM access or RAM copying. To some extent it is similar to running out of memory and paging memory to disk in more classical setups. In order to obtain efficient algorithms, this should be avoided or, at least, carefully controlled.

Finally, communication between an enclave and the rest of the system needs to be considered quite carefully. First, context switching between enclave mode and non-enclave mode in the CPU incurs additional cost compared to a normal context switch as the CPU has to perform a significant amount of bookkeeping during this context switch. Second, passing data between an enclave and the rest of the system requires such a context switch, and additionally the encryption/decryption of said data for the enclave, which are both non negligible costs.

3. IMPACT ON DATABASE RELATED PROCESSING

At first, we focus on benchmarking elementary database operations. While other works [8] require loading the entire data system within the RAM allocated to an enclave, we consider this approach as unrealistic in the general case since this requires allocating up to hundreds of GB of RAM to an enclave. Since the

RAM is statically allocated to an enclave, such solutions are too costly in practice.

For instance, it would not be acceptable (neither in the cloud nor home machine contexts) to permanently allocate large quantities of RAM (which cannot be easily reclaimed) to an enclave belonging to a specific user. Indeed, in the cloud context statically allocating amounts of RAM to mostly offline users is unrealistic as it would require much more RAM than effectively needed to run the system in an unsecure context.

For a home computer this assumption is even more unrealistic as user cannot effectively block large amounts of memory for a personal data management system. An alternative would be to page the enclave memory to external memory managed by the OS. However, as mentioned above, this would break optimizations made to data processing algorithm due to the high cost of paging. Therefore, this calls for data processing techniques where private data mostly resides encrypted outside the enclave and is consumed by the enclave on request within the limit of the enclave RAM.

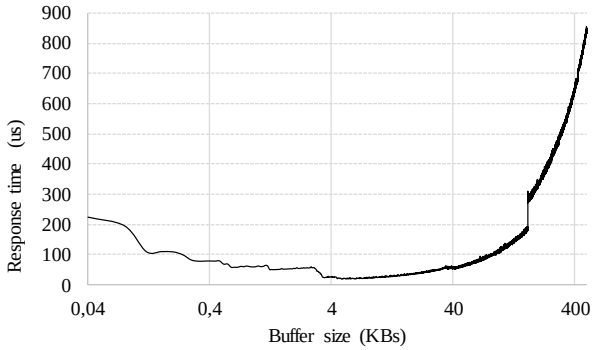

Figure 1. Binary search in a sorted list.

Hence, our first performance evaluation consists in measuring the data transfer cost in between the enclave RAM to the untrusted system RAM and conversely. We measured both the cost of importing/exporting large chunks of data at once, and the cost of importing/exporting large amounts of small pieces of data. The additional cost compared to classical RAM access comes from two main factors: the cost of context switching to the untrusted mode in order to request data, and the cost of encrypting the data for the enclave RAM. We observed that importing large chunks of memory at once is much faster than importing a large number of small pieces of memory summing up to the same size. Also, as soon as the size of the data imported is larger than 90MB (i.e., the maximum RAM enclave size in our test system), the overhead due to paging (managed automatically by the Linux SGX driver) is clearly visible.

But an important question is in which way the above noted constraints can impact a data-intensive processing. Let us consider the basic example of item search in a large sorted list in RAM. In the classical context, a binary search on the list offers the best performance. In the SGX context, each access to a list item from the enclave requires a context switch to load the item from the unprotected RAM, which is costly. Alternatively, the enclave could load multiple list items at each access (e.g., every 10^{th} element in the list), which would proportionally reduce the number of context switches but increase the data access cost. Figure 1 shows the average time search in a sorted list with increasing access buffer size. The left-most point corresponds to a buffer size of one item (i.e., a binary search). We can see that the optimum buffer size is around 4KB, which leads to search times one order of magnitude smaller than the binary search. On the other hand, increasing further the buffer size leads to increasing the search cost, especially if the buffer size exceeds the enclave RAM size (see right-most part of the graph).

4. FUTURE WORKS

The results presented here cover the processing efficiency within SGX enclaves, and derive design principles for SGX data processing. We aim at formalizing these insights and extend our results to a more extensive set of data processing algorithms. The next essential step is considering the security issues related to side channel attacks both on SGX and on the access patterns on external data.

First, vulnerabilities and side channel attacks exist on SGX. The most significant one being the recent Foreshadow [9] attack. Mostly, these attacks rely on out of order execution in a way somewhat similar to Spectre [5] and Meltdown [6]. The mitigation techniques for these attacks typically rely on forcing a L1 cache flush on each enclave enter/exit. An important study would be to examine how these cache flushes influence performance. This study will make our analysis somewhat robust to future versions of SGX which will include these mitigations. Additionally, one should consider timing attacks on the algorithms used for data management, as timing of enclave computations can be easily observed from the untrusted OS. In particular, we plan to study the dependency between sensitive data and computation time in classical data-oriented algorithms.

Second, an important leakage source resides in the interactions between the untrusted OS and the data processing enclave. For example, in our search example the OS gets the position of the searched element in the list from the position of the queried elements. Mitigations for this kind of leakage is an important research direction, as well as the impact of such mitigations on performance and algorithm designs.

5. REFERENCES

[1] Alves, Tiago, and Don Felton. TrustZone: Integrated Hardware and Software Security-Enabling Trusted Computing in Embedded Systems. 2004.

[2] Brenner, Stefan, et al. SecureKeeper: Confidential ZooKeeper using Intel SGX. Middleware, 2016.

[3] Costan, Victor and Srinivas Devadas. Intel SGX Explained. *IACR Cryptology ePrint Archive* 2016.086 (2016): 1-118.

[4] Fuhry, Benny, et al. HardIDX: Practical and secure index with SGX. *IFIP Annual Conference on Data and Applications Security and Privacy*, 2017.

[5] Kocher, Paul, et al. Spectre attacks: Exploiting speculative execution. *S&P*, 2019.

[6] Lipp, Moritz, et al. Meltdown: Reading Kernel Memory from User Space. *USENIX Security Symposium*, 2018.

[7] McKeen, Frank, et al. Innovative instructions and software model for isolated execution. *HASP@ ISCA 10*, 2013.

[8] Priebe, Christian, Kapil Vaswani, and Manuel Costa. EnclaveDB: A Secure Database using SGX. EnclaveDB: A Secure Database using SGX. *IEEE*, 2018.

[9] Van Bulck, Jo, et al. FORESHADOW: Extracting the Keys to the Intel SGX Kingdom with Transient Out-of-Order Execution. *USENIX Security Symposium*, 2018.

[10] Zhao, ChongChong, et al. On the Performance of Intel SGX. *IEEE Web Inf. Systems and Applications Conf.*, 2016.

2HD: Advanced processing methods for heterogeneous data in heterogeneous datacenters

Roxana Gabriela Stan
Computer Science and Engineering Department
Faculty of Automatic Control and Computers
University Politehnica of Bucharest
Bucharest, Romania
roxanagabrielastan@gmail.com

Florin Pop
Computer Science and Engineering Department
Faculty of Automatic Control and Computers
University Politehnica of Bucharest
Bucharest, Romania
florin.pop@cs.pub.ro

ABSTRACT

One major challenge of Big Data is reasonably justified by its heterogeneity property. Big Data heterogeneity requires a simultaneous management of structured, semi-structured and even entirely unstructured data. Heterogeneous data refer to a high variability of representation formats and data types and have serious implications for Big Data analytics.

The overwhelming amount of data generated today motivates the need for a platform to handle the properties of large and dynamic data sets, which is more emerging than ever before.

The main challenges addressed by the research thesis, whose doctorate proposal is presented in this paper, include the real-time processing, concurrent data processing and data types complexity handling, conducting to a heterogeneity-aware designed solution, which will also minimize the problems implied by managing heterogeneous data centers.

CCS CONCEPTS

• **Information systems** → **Data management systems**; Data mining; • **Computer systems organization** → *Cloud computing*; • **Networks** → Data center networks;

KEYWORDS

Big Data, Cloud Computing, Internet of Things, Big Data analytics, heterogeneity, datacenters, variety, data transformation

1 INTRODUCTION

In recent years, the rapid development of Internet, Internet of Things, the proliferation of the Cloud Computing [2] and the spread of smart devices have led to the explosive growth of data in almost every industry and business domain. Nowadays, large data volumes are generated on a daily basis at an unprecedented rate from various and heterogeneous sources, including areas such as health, government, social networks, financial and marketing. In this context, due to the rapid expansion in all science and engineering domains, Big Data presents a remarkable opportunity for organizations to gather critical intelligence to drive decisions and obtain enhanced insights as never before [1].

2 LITERATURE RESEARCH

Variety – diverse and dissimilar data representations – consisting of multiple formats for public or private, local or distant, directly accessible or restricted, complete or incomplete, structured or loosely unstructured text, image, audio, video, multimedia content, sensor data and noise, poses numerous challenges nowadays. The variety in data holds the most potential for exploitation. The axes of data variety are defined as follows: structural (related to models and formats), semantic (instructing how to proceed on interpreting and operating on available data) and media (represented by the medium in which data get delivered) [4].

3 HETEROGENEITY

The heterogeneity property rises extreme challenges to both Big Data and Cloud Computing domains.

3.1 Heterogeneous data

The enormous volume of data is not consistent nor does it follows a specific template or pattern – the captured data have diverse formats and originate from various sources, including: messages (text, emails, tweets, blogs), user-generated content, transactional data (Web logs, business transactions), scientific data (data coming from data-intensive experiments – genome and health-care data), Web data (multimedia content posted on social media, sensor data readings), graph data. These different forms clearly indicate that heterogeneity represents a natural property of Big Data and as a consequence, the comprehension and management of such data is a serious challenge [7].

Considering the growing data sets which are collected from autonomous sources, several characteristics such as inconsistency, ambiguity, redundancy, partial or lack of information cause difficulties to the process of data integration and correlation. A vital source for generating heterogeneous data is Internet of Things. Starting from the variety of embedded devices used for acquiring data, the data to be collected will result in a data types and formats variety as well.

The diverse representations of the gathered data particularly depend on the specific of each system. For any real-world application, the generation of data is regarded as a continuous and iterative process, considering that the existing data sets become obsolete as time progresses, being incrementally replaced by information relevant in the current context. Hence, new associations are established while considering the avoidance of noisy components interferences. Aided by the data mining

techniques, the relevant patterns resulted from the involved data correlations need to be extracted. By employing advanced data processing methods, these data correlations could not be limited only to the existing shared attributes. Instead a combination between structured and unstructured data is needed for leveraging the real potential of the data [8].

3.2 Heterogeneous datacenters

The heterogeneity of modern data centers is basically influenced by the hardware platform heterogeneity. Existing research rarely considers Cloud data centers heterogeneity.

A modern data center is represented as a heterogeneous environment containing many generations of servers with possibly different hardware configurations, especially operating at different speeds, with distinct processor capacities. In fact, these servers were incrementally added and provisioned to replace the legacy or the existing machines of the underlying infrastructure [3] [5].

Tackling the heterogeneity problem requires analyzing the data gathered in various contexts and subsequently looking for patterns and generalities within them. Current real-world applications such as sensor networks, credit card transactions, stock management, blog posts and network traffic produce tremendous data sets. Due to the high variety of data sources for the data acquisition phase of the Big Data processing, the applied filters for discarding information of no importance should avoid loss of any relevant data.

Data mining methods are essential to discover non-trivial patterns and to extract value hidden in such huge data sets. Traditional techniques such as association mining, clustering and classification lack of efficiency, scalability and accuracy when applied to such Big Data sets in a distributed environment with decentralized control. The variability of incoming streams brings unpredictable changes for a continuously evolving system. Therefore, new methods need to be identified to provide optimized analytical techniques, producing real-time accurate results while processing data instances with limited time and resources [6].

4 DOCTORATE THESIS OBJECTIVES

The data processing technology has limitations with regard to the heterogeneity challenges on Big Data and Cloud Computing domains, as previously described. Currently, the existing data analysis algorithms handle data in a homogeneous format, with no support to deal with the diverse data formats, the considered data being limited to a structured format. Furthermore, the regularly used software tools and frameworks are not capable of processing a large amount of heterogeneous data within a flexible time limit.

4.1 General Objective

Big Data – being generally unstructured and heterogeneous – is extremely complex to deal with via traditional approaches, and requires real-time or almost real-time analysis. It is difficult to state that one platform could solely resolve the problem or there could be some algorithm to synchronize the data varieties in a uniform format – it greatly depends on particular cases.

The major objective of the doctorate thesis is the research and design of a scalable and viable solution to address the problem of data variety by making sense of and adding context to the diverse data types and sources.

The solution will be delivered as a platform whose underlying algorithms will possess decision-making capacities, with the possibility to perform data correlations and extract more information from interpretable features (using, for example, time series). Based on various real life scenarios (Cloud platforms for robots, environment monitoring, maritime data analysis, air quality monitoring), the system could evolve to support consequent complex processing operations.

4.2 Specific Objectives

The specific objectives of the research include the following:

(1) Handling and pre-processing the received data, ensuring high efficiency and an accurate data classification. Data cleaning could be also employed for particular cases with data affected by noise.
(2) Building a reliable auto-scaling system capable to dynamically decide if the handled data have to be processed real-time or quasi-real-time. The resulting system will deliver fast execution times.
(3) Acquiring and discovering data knowledge from autonomous, distributed data sources and making intelligent decisions regarding the potential data correlations, when possible.
(4) Integrating the above mentioned phases into a platform available as a configurable dashboard to users.

In the end, the users will benefit from a compact platform handling and also aggregating considerably variable data sets.

5 CONCLUSIONS

In conclusion, the proposed solution will have a massive impact on currently loosely coupled areas, leveraging the full potential of Big Data in diverse industrial fields and research domains. Straightforward access to a broad variety of data is a key part of a platform for driving innovation and efficiency.

REFERENCES

[1] Jemal Abawajy. 2015. Comprehensive Analysis of Big Data Variety Landscape. *International Journal of Parallel, Emergent and Distributed Systems* 30, 1 (Jan. 2015), 5–14. https://doi.org/10.1080/17445760.2014.925548
[2] Alessio Botta, Walter de Donato, Valerio Persico, and Antonio Pescapé. 2016. Integration of Cloud computing and Internet of Things: A survey. *Future Generation Computer Systems* 56 (2016), 684 – 700. https://doi.org/10.1016/j.future.2015.09.021
[3] Christina Delimitrou and Christos Kozyrakis. 2013. Paragon: QoS-aware Scheduling for Heterogeneous Datacenters. *SIGARCH Comput. Archit. News* 41, 1 (March 2013), 77–88. https://doi.org/10.1145/2490301.2451125
[4] Salvador García, Sergio Ramírez-Gallego, Julián Luengo, José Manuel Benítez, and Francisco Herrera. 2016. Big data preprocessing: methods and prospects. *Big Data Analytics* 1, 1 (01 Nov 2016), 9. https://doi.org/10.1186/s41044-016-0014-0
[5] J. Mars, L. Tang, and R. Hundt. 2011. Heterogeneity in "Homogeneous" Warehouse-Scale Computers: A Performance Opportunity. *IEEE Computer Architecture Letters* 10, 2 (July 2011), 29–32. https://doi.org/10.1109/L-CA.2011.14
[6] Ahmed Oussous, Fatima-Zahra Benjelloun, Ayoub Ait Lahcen, and Samir Belfkih. 2017. Big Data technologies: A survey. *Journal of King Saud University - Computer and Information Sciences* (2017). https://doi.org/10.1016/j.jksuci.2017.06.001
[7] Uthayasankar Sivarajah, Muhammad Kamal, Zahir Irani, and Vishanth Weerakkody. 2016. Critical analysis of Big Data challenges and analytical methods. 70 (08 2016).
[8] Lidong Wang. 2017. Heterogeneous Data and Big Data Analytics. *Automatic Control and Information Sciences* 3, 1 (2017), 8–15. https://doi.org/10.12691/acis-3-1-3

Efficient re-execution of data intensive scientific workflows for high-throughput phenotyping

Article Doctorant

Gaetan Heidsieck
INRIA, Univ Montpellier, Montpellier, France
gaetan.heidsieck@inria.fr

Esther Pacitti
INRIA and LIRMM, Univ Montpellier, Montpellier, France
Esther.Pacitti@lirmm.fr

Christophe Pradal
CIRAD, Montpellier, France
christophe.pradal@inria.fr

Francois Tardieu
INRA, Montpellier, France
francois.tardieu@inra.fr

KEYWORDS

Scientific workflow, Provenance, Optimisation, Workflow re-execution

1 INTRODUCTION

Climate change is a major aspect to consider for the selection of crops for future environments. Plant scientists analyze specific traits evolution in order to identify the genetic variations and the genetic controls in response to environmental factors.

In the last decade, high-throughput phenotyping platforms have emerged to allow the acquisition of quantitative data on thousands of plants in well-controlled environmental conditions. These platforms produce huge datasets of heterogeneous data (images, environmental conditions and sensor outputs) and generate complex elaborated variables with in-silico data analyses. National infrastructure such as Phenome project produces 200 terabytes of data annually. Analysing automatically and efficiently these massive datasets is a major problem [7].

Such analysis can be represented, managed and shared in an efficient and reproducible way using scientific workflows, where compute- and data-based activities are linked by dependencies. Several workflow management systems use provenance to analyze and share executions and their results. These workflows are data-intensive due to the high volume and the size of the data to process. They are computed on distributed computational infrastructures [5].

As computational experiments, workflows are executed with different parameters, and their structure evolved through time. Moreover, workflows are reused and repurposed by other scientists and teams [1]. For instance, the OpenAlea Phenomenal workflow (fig 1) is composed of a set of tools for automatic analysis of 3D plant architecture from raw plant images. It is composed of six fragments (set of activities) : binarization, skeleton generation, 3D-reconstruction, mesh generation, stem detection and organ segmentation. Each fragment of the workflow produces intermediate data, such as B&W images, 3D volumes of plants, then organ geometry and topology.

2 PROBLEM

In this community, different teams of users, geographically distributed in different sites, may execute or re-execute the same or related workflows. For instance, one team A may propose an incremental analysis of a workflow WFo, for organ segmentation based on a previous workflow WFp that for instance provides 3D plant volume. Workflow WFp was computed by another team B on the same dataset, and the output is stored and available in one of the site (cluster, grid or cloud). Yet the team A may need to change parameters and settings of WFp to adapt the output data to their analysis workflow WFo. Normally, with different parameters, WFp needs to be re-executed from scratch.

The main challenge we address in this thesis is how to automatically benefit from some intermediate data of the workflow (WFp) that have been computed from previous executions, to speed-up the execution of workflows (WFp+WFo) in a distributed context. To the best of our knowledge , this work is an original research problem, and its solutions will bring a new approach that enables distributed sharing and reuse of intermediate data between users.

3 APPROACHES

To address this problem, we pursue three directions. First, we propose to investigate how provenance can be used to create an index, which allows retrieving efficiently previous data results [2, 4]. Second we will propose a distributed cache, to store intermediate data generated by the execution. Third, we will investigate a new scheduling algorithm to execute and re-execute workflows using the index and the cache. For this we will take into account two approaches.

With respect to the scheduling algorithm, we first take into account a greedy approach that always uses the cached data. Next, we will refine and propose solutions that take into account a cost function. There are many different costs related to the execution, the data, or the computational infrastructure [3]. For instance, it can be more efficient to re-execute an activity rather than to search if it has already been executed, and to retrieve the intermediate data already produced by the activity from the cache. To tackle this trade-off (execution vs data reuse), we define a cost function that takes into account the cost for : i) the cost for moving data in the distributed infrastructure ($C_{Datamovement}$), ii) for searching similarities in workflows ($C_{Datasearch}$) compared to the costs for

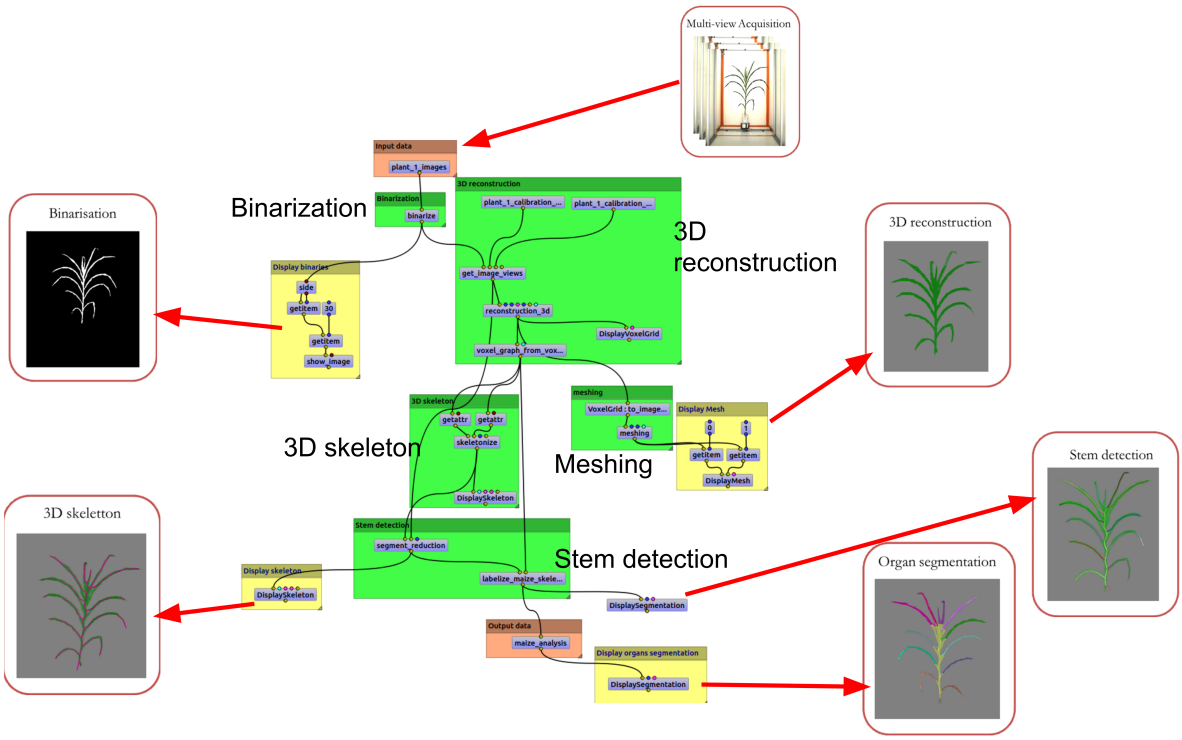

Figure 1: The 3D reconstruction workflow Phenomenal in the OpenAlea visual programming environment [6] is applied to one plant at a given time. The same workflow is run on thousands of plants through time, consuming several terabytes of data, and distributed transparently on the European Grid infrastructure (EGI).

iii) the execution of the activity ($C_{Execution}$), and for iv) storing the intermediate data ($C_{Persistence}$).

We propose to model the cost function (f) as:

$$f = \frac{C_{Data\ movement} + C_{Data\ search}}{C_{Execution} + C_{Persistence}}$$

Our goal is to propose several heuristics for the cost function over different types of architecture (Cluster, Cloud and Grid), with either centralized and distributed index and cache.

We will demonstrate this cost aware scheduling algorithm on a real use case for automatic 3D reconstruction and analysis of plants imaged in the high-throughput phenotyping platform, PhenoArch, located in Montpellier.

ACKNOWLEDGMENTS

This work is supported by #DigitAG in Montpellier, and represents an important challenge in an interdisciplinary research field involving agronomy and big data management.

REFERENCES

[1] Sarah Cohen-Boulakia, Khalid Belhajjame, Olivier Collin, Jérôme Chopard, Christine Froidevaux, Alban Gaignard, Konrad Hinsen, Pierre Larmande, Yvan Le Bras, Frédéric Lemoine, et al. 2017. Scientific workflows for computational reproducibility in the life sciences: Status, challenges and opportunities. *Future Generation Computer Systems* 75 (2017), 284–298.

[2] Daniel de Oliveira, Kary ACS Ocaña, Fernanda Baião, and Marta Mattoso. 2012. A provenance-based adaptive scheduling heuristic for parallel scientific workflows in clouds. *Journal of Grid Computing* 10, 3 (2012), 521–552.

[3] Ji Liu, Esther Pacitti, Patrick Valduriez, Daniel De Oliveira, and Marta Mattoso. 2016. Multi-objective scheduling of scientific workflows in multisite clouds. *Future Generation Computer Systems* 63 (2016), 76–95.

[4] Ji Liu, Esther Pacitti, Patrick Valduriez, and Marta Mattoso. 2016. Scientific workflow scheduling with provenance support in multisite cloud. In *International Conference on Vector and Parallel Processing*. Springer, 206–219.

[5] Christophe Pradal, Simon Artzet, Jerome Chopard, Dimitri Dupuis, Christian Fournier, Michael Mielewczik, Vincent Negre, Pascal Neveu, Didier Parigot, Patrick Valduriez, et al. 2017. InfraPhenoGrid: a scientific workflow infrastructure for plant phenomics on the grid. *Future Generation Computer Systems* 67 (2017), 341–353.

[6] Christophe Pradal, Samuel Dufour-Kowalski, Frédéric Boudon, Christian Fournier, and Christophe Godin. 2008. OpenAlea: a visual programming and component-based software platform for plant modelling. *Functional plant biology* 35, 10 (2008), 751–760.

[7] François Tardieu, Llorenç Cabrera-Bosquet, Tony Pridmore, and Malcolm Bennett. 2017. Plant phenomics, from sensors to knowledge. *Current Biology* 27, 15 (2017), R770–R783.

Blockchain privacy-preserving in healthcare

Liviu-Adrian Hîrțan

ABSTRACT

Blockchain is a relatively new technology whose potential is compared with Internet. Blockchain is the core of Bitcoin which has disrupted the financial sector allowing digital currency transfer without a centralized entity. This technology has been proposed to develop applications for smart city, smart government, internet of things, and more. In the medical field, data sharing, privacy and interoperability must be major concerns. Using blockchain technology, in this paper we describe the architecture of a healthcare system where we take into account all the aspects mentioned.

Study science evolution by using word embedding to build phylomemetic structures

Ian Jeantet

ABSTRACT

Understanding the evolution of various scientific fields is important for our society. Obtaining a general picture of important evolutions of entire scientific fields is rather challenging in the light of the proliferation of scientific publishing and in the presence of over-specialized scientific journals. Recent papers propose text analysis techniques to reconstruct important aspects of evolution, based on large corpora of scientific publications (such as Web of Science, PubMed) and the goal here is to propose automated tools that can assist (social) scientists to study empirically particular aspects of the social dynamics of science.

Secure and distributed computations for a personal data management system

Riad Ladjel
Inria & U. Versailles
Versailles, France
riad.ladjel@inria.fr

Nicolas Anciaux
Inria & U. Versailles
Versailles, France
nicolas.anciaux@inria.fr

Guillaume Scerri
U. Versailles & Inria
Versailles, France
guillaume.scerri@uvsq.fr

INTRODUCTION

A large amount of economically valuable data is produced every day. Whether they come from smartphones, connected devices, sensors or smart meters, this data is a gold mine for the people holding it. This data is often stored in centralized servers, servers that usually belong to large corporations that collected it in the first place (Google, Amazon, Facebook, insurance companies etc.) The result of this situation is that users lose control over their own data. This represents a real threat to privacy, whether it is intentional (misuse, malicious attack), or just by negligence (data leakage, mismanagement). These threats point to the need for personal platforms which allow their users to collect, manage and share their own data. This is the essence of the self-data movement. Thanks to smart disclosure initiatives, users can access their personal data from the companies or government agencies that collected them. Concurrently, personal data management system (*PDMS*) solutions are flourishing. Their goal is to empower users to leverage their personal data for their own good.

While storing data in *PDMS* increases user control over data, in the *PDMS* context collaborative use of data is often overlooked. The benefits derived from exploiting data are considerable. A user may want to share her GPS position to have accurate traffic prediction [8], or her medical records to train a shared neural network so that it can detect several diseases [5]. She may also want to adapt her energy contract based on her actual consumption without jeopardizing her privacy [3].A naive approach to this problem is to send personal data to a trusted third party who will perform said collaborative computations. This, however involves very strong trust in the third party's honesty. The goal of this work is to overcome this unrealistic trust assumption and propose a privacy preserving distributed computation framework for performing collaborative computations over a large number of *PDMS*.

Our objective is to propose a secure distributed computation protocol offering the same guaranties as secure multi-party computation (*SMC*) [6] while keeping a profit-to-cost ratio as low as possible in case of an attack. In other words, it should be possible for parties to compute any function $f(x_1, x_2, ..., x_n)$ over their inputs while keeping them private. An individual cannot infer any information other than what she can infer from her own input and the final result. The result should be exact and not approximate. Unlike *SMC*, the proposed protocol should be scalable, with respect to the number of parties and the complexity of the computation.

SHORTCOMINGS OF EXISTING SOLUTIONS

Secure multi-party computation (*SMC*) [6] allows N users to perform a secure computation on their personal data. The users involved learn nothing more than what they can infer from their input and the result. SMC is based on complex cryptographic techniques, which limit their scalability, in particular for very large number of parties. Ad-hoc *SMC* protocols have been proposed but these are not generic and do not allow all types of computations.

Fully Homomorphic Encryption (FHE) [4] schemes allow one to compute arbitrary functions over encrypted data without a decryption key. Given c_1 and c_2 ciphertexts of m_1 and m_2, one can compute $f(m_1, m_2)$ by decrypting $f(c_1, c_2)$. *FHE* is not suitable in our context due to its high cost and because users need to have the same encryption key, which creates serious security issues.

Schemes based on gossip protocols [7] Gossip protocols are based on a communication technique that works the same way a rumor spreads. On every round each node randomly selects a node from its neighborhood to exchange information, after a sufficient number of rounds, the result of each node converges to the final result. Gossip protocols scale well but are not generic in terms of possible computations, which make them unsuitable for our case.

OUR APPROACH

We propose a protocol for secure distributed computations based on *Trusted Execution Environments (TEE)*. A *TEE* is a secure part of the processor, which guarantees confidentiality by providing isolated enclaves and integrity provided by attestation mechanisms[1]. Code executed inside a *TEE* is protected against all elements outside it, including the operating system. Several vendors propose their own Trusted Execution Environment : *Platform Security Processor* for *AMD* , *TrustZone* for ARM and *Software Guard Extensions* for *Intel*, the latter one will be used for the implementation and evaluation of our protocol. To be usable in our protocol a *TEE* must be able to do *Attested Computation*. Our architecture is composed of a querier and several users who agree to contribute to the computation. They are all equipped with a *TEE*. The use of *TEE* increases the cost of an attack while distributing the computations over plenty of nodes decreases the benefits of attacking a single computation node.

Threat model. The threat model in our context is different from *SMC* one. In *SMC*, the adversary is honest-but-curious. But in our context, data involved in the computation is stored and managed inside an isolated enclave. This makes the computation node honest even in the presence of curious or malicious *PDMS* owner.

[1] *"Attestation is the process of demonstrating that a piece of software has been properly instantiated on the platform. In Intel SGX it is the mechanism by which another party can gain confidence that the correct software is securely running within an enclave on an enabled platform"* [1]

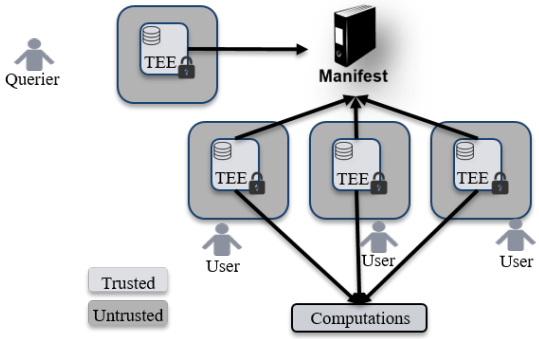

Figure 1: overview of the architecture

The first step in our study was to identify and propose a well-adapted threat model. We consider two different ones (1) honest computation node held by honest-but-curious participants and/or querier who may try to infer some information without breaking the security of the *TEE* and (2) malicious participants and/or querier who may perform a side channel attack (e.g., cache attack using technique presented in [2]) to retrieve some information from the TEE. They may also either participate to the computation with many *TEEs* to infer more information than they should or try to run another code then the expected one.

Problem statement. Based on the previous threat model and taking advantage of the confidentiality and the integrity guarantees provided by *TEEs*, we propose adopting a manifest based protocol. The manifest will be used to specify how the distributed computation will unfold. The protocol should ensure that the computation is respected as described in the manifest. This is not trivial, for the following reasons: (1) Each user should have the possibility to check the global correctness of the computation from her local view (*local correctness checking basis*). (2) The execution should be *zero knowledge*. In other words, in the case of data oriented operations where the dataflow depends on the data value (e.g., hashing or sorting as part of joins), an attacker able to observe the data exchanges should not be able to infer users' personal data. (3) Finally, the protocol should be *massive leak resilient*. More specifically, in case of a successful side channel attack, a user or a set of colluding users, should not be able to compromise a large amount of data nor targeting a specific user. Such attacks should be taken into account by the protocol by adding again a set of countermeasures to reduce the benefit-to-cost ratio.

THE PROTOCOL

Let us take as a simple example the case of a group by computation. A naive secure execution approach based on *TEEs* could consist in participants sending their personal data to a single node which perform the computation. This approach is not massive leak resilient nor can it enforce a local correctness checking basis. Another approach could be participants sending their data to different nodes, where a node is in charge of computing data belonging to the same group, this obviously is not a zero knowledge execution.

Our approach aims at designing a protocol that supports execution of any distributed computation, on a potentially large number of participants, while providing strong security guarantees. The protocol is organized in three phases (see Fig. 1). The first phase is to set up the environment; the querier broadcasts the function to be computed, together with the organization of the distributed computation. Users who want to participate to the computation register with the querier . When the number of participants is sufficient, the querier and participants collaborate to produce an execution table, which is a list of tasks to do for each participant (e.g. $user_1$ executes f_1 on $user_2$ data and sends her output to $user_3$). The list of participants, the topology and the execution table form a manifest. This manifest has to be public, so that everyone can verify it. Thanks to attestation, the manifest is generated with strong guarantees of integrity. In the second phase, participants establish links between them following the manifest. Users leverage attestation to prove that they are executing their assigned tasks from the manifest. Finally, the last phase is the actual computation and it depends of the algorithm implemented. This protocol provides strong integrity guarantees, as breaking integrity means compromising a tamper resistant *TEE*. Moreover, if a malicious participant performs a side channel attack, retrieving the content of its *TEE*, it only has access to a limited amount of data as prescribed by the manifest. As long as the distributed computation is organized in a sensible way, no party should have access to large amount of personal data, and the threat of large scale attacks is this mitigated. Additionally, note that all computations are done on clear data, the results produced are thus exact. Moreover, only simple cryptographic operations and bookkeeping inside *TEE* are required in addition to the actual computation. This allows for a limited overhead w.r.t. performing the computation in the clear between mutually trusted parties.

FURTHER WORK

As further work, we aim at designing a protocol solving the three problems defined above. In other words, a protocol giving guarantees on the global integrity of the computation while taking into account side channel attacks against *TEEs*. Moreover, the protocol should be validated for three different kinds of computation, computation with static built of the manifest, those where the manifest has to be built dynamically (group by query based on hash) and iterative computation (such as training a shared neural network).

ACKNOWLEDGMENTS

This research is supported by the ANR PersSoCloud grant n° ANR-16-CE39-0014.

REFERENCES

[1] Ittai Anatiet al. 2013. Innovative technology for CPU based attestation and sealing. In *HASP*.
[2] Johannes Götzfriedet al. 2017. Cache attacks on Intel SGX. In *Proceedings of the 10th European Workshop on Systems Security*. ACM.
[3] Andrés Molina-Markham et al. 2010. Private memoirs of a smart meter. In *Embedded sensing systems for energy-efficiency in building*.
[4] Craig Gentry et al. 2009. *A fully homomorphic encryption scheme*. Stanford University Stanford.
[5] Joseph A Cruz et al. 2006. Applications of machine learning in cancer prediction and prognosis. *Cancer informatics* (2006).
[6] Ronald Cramer et al. 2015. *Secure multiparty computation*. Cambridge University Press.
[7] Tristan Allard et al. 2016. Lightweight privacy-preserving averaging for the internet of things. In *M4IOT*.
[8] Yisheng Lv et al. 2015. Traffic flow prediction with big data: a deep learning approach. *IEEE Transactions on Intelligent Transportation Systems* (2015).

Privacy-Preserving Queries on Highly Distributed Personal Data Management Systems

Julien Loudet[1,2,3]
[1]Cozy Cloud
France
julien@cozycloud.cc

Luc Bouganim[2,3]
[2]INRIA Saclay
France
<fname.lname>@inria.fr

Iulian Sandu-Popa[2,3]
[3]University of Versailles
France
<fname.lname>@uvsq.fr

1 INTRODUCTION

The time of individualized management and control over one's personal data is upon us. Thanks to smart disclosure initiatives [4] and new regulations, we can access our personal data from the companies or government agencies that collected them. Concurrently, Personal Data Management System (PDMS) solutions are flourishing in academia [1] and industry [3]. Their goal is to offer a data platform allowing users to easily store into a single place any personal data: data directly generated by user devices (e.g., quantified-self data, smart home data, photos, etc.) and user interaction data (e.g., user preferences, social interaction data, health, bank, telecom, etc.). Users can then leverage the power of their PDMS to use their personal data for their own good and in the benefit of the community. Thus, the PDMS paradigm holds the promise of unlocking new innovative usages while preserving the current ones developed around personal data. A prominent example of novel usages is related to the computations between a large number of PDMSs (e.g., recommendations, participative studies, collective decisions).

However, these exciting perspectives should not eclipse the security issues raised by the PDMS paradigm. Indeed, each PDMS can store potentially the entire digital life of its owner, thereby proportionally increasing the impact of a leakage. Hence, centralizing all users' data into few powerful servers is risky since the data servers become genuine honeypots: huge amounts of personal data belonging to millions of individuals could be leaked or lost as illustrated by recent massive attacks [5]. Besides, such a centralized solution makes little sense in the PDMS context in which data is naturally distributed at the users' side.

Fortunately, Trusted Execution Environments (TEE) [6] are also rising and employing them in PDMS context leads to trustworthy computational data platforms. Hence, in this work, we assume that all PDMSs are secured thanks to a TEE. A PDMS can be considered to offer high connectivity and availability and can establish peer-to-peer (P2P) connections with other PDMSs. As such, we envision a fully distributed architecture of PDMS devices in which participants can create large communities, contribute with their personal data and issue queries over the globally contributed data. In this context, an important issue needs to be addressed: how to query this massively distributed data in a pertinent, efficient and privacy-preserving way?

2 DESCRIPTION OF THE SOLUTION

To achieve a high degree of pertinence, in our system each query only targets the subset of PDMSs exposing a given *user profile*: a structured description indicating the user's attributes (e.g. location, age, interests). Besides pertinence, a second benefit of user profiles is to increase query processing efficiency by avoiding flooding the entire network with each query. The queries can be, for instance, aggregate queries allowing users to compute generic statistics (e.g., recommendations of films). While the data and query model definitions and expressivity are significant issues, the global system security is paramount for a large system adoption, i.e., gaining users' trust and encouraging them to contribute with their data. We thus focus on privacy preservation during query execution in the abovementioned system.

As previously stated, TEEs provide a high level of data confidentiality against malicious PDMS owners. However, since no security measure can be considered as unbreakable, we cannot exclude having some corrupted nodes in the system and, even worse, those corrupted nodes might very well be undistinguishable from honest nodes. As we consider a fully distributed system, the query processing relies exclusively on PDMS nodes. This implies some data disclosure risk whenever a corrupted node is selected as a query actor. Therefore, to maximize the system security, we need to minimize the benefit of corrupting a node. This translates into two requirements:

R1: Minimize the private information any node could have access to whenever it is assigned with a data related task.
R2: Ensure that an attacker controlling several corrupted nodes cannot influence the selection of the nodes processing a query.

To efficiently index and retrieve node profiles, we leverage the classical Distributed Hash Table (DHT) P2P overlay [7, 9]: each node is responsible for a set of use profile attributes and indexes all the node IP addresses that match one of them. To query our system in a secure and efficient manner, we build a distributed protocol on top of this P2P overlay.

Our protocol relies on three design principles, namely *knowledge dispersion*, *task compartmentalization* and *imposed randomness*, which, once combined, answer both requirements.

Knowledge dispersion. This principle aims at protecting the *data-at-rest* (i.e. the distributed index used to retrieve the nodes corresponding to a given user profile) and can be summed up by: no single node (or few nodes) should store a significant amount of sensitive data, unless it owns that data.

In practice, this design principle is realized through the use of *Shamir's Secret Sharing technique* [8]. Even though the DHT uniformly distributes the knowledge among the nodes, if one node

were to be corrupted it could access the entire list of IP addresses corresponding to the set of user attributes it indexes. By applying this technique the entries in the distributed index become "shards" that, taken alone, cannot be decrypted, thus protecting the data.

Task compartmentalization. The second principle concerns the query execution. Thanks to the first design principle the *data-at-rest* is protected, but the *data-in-use* — the data that is used to compute the final result — is still open to attacks. To protect it we perform task compartmentalization: we split the query execution in elementary tasks assigned to distinct nodes, so that each actor has only access to the minimal information it needs to know to perform its task.

In practice, this principle is enforced through the use of different actors having different roles. Also, whenever possible, several nodes are assigned to the same role to further split the access to sensitive data. The roles are: the *Concept Indexors* indexing the attributes, the *Target Finders* determining the relevant nodes based on a *target profile*, and the *Data Aggregators* aggregating the individual results to obtain the answer to the query.

Imposed randomness. The third and last design principle stipulates how the query actors should be selected. Indeed, if an attacker controls enough nodes in the network and manages to execute a query where all the actors are corrupted, then all our previous precautions would be nullified. This reason is why we force *imposed randomness*: the actors for each query must be randomly selected and the selection cannot be influenced by an attacker controlling several nodes.

In practice, this principle is applied by generating verifiable random numbers in a distributed fashion using an algorithm based on CSAR [2]. Those random numbers will designate a location on the DHT overlay, around which the actors will be selected (also randomly, using such number).

Thus, our protocol answers the requirement R1 by applying *knowledge dispersion* and *task compartmentalization* design principles to protect, respectively, the data-at-rest and the data-in-use during the query evaluation process. Requirement R2 is addressed by applying the *imposed randomness* design principle in the selection of the nodes processing a query.

This guarantees that an attacker cannot obtain more private information than what she can passively get from observing the information randomly reaching its corrupted nodes. Thus, the impact of a collusion attack remains proportional with the number of corrupted nodes, which is the best situation given our context.

To assess the security offered by our protocol we have implemented a simulation in which we measured the impact of collusion attacks at different scales: where an attacker controls 1, 50, 0.01%, 1% nodes in a network that contains 10k, 100k, 1M and 10M nodes. For each level of attack we repeatidly simulated the execution of a query and measured the latency and total effort in terms of cryptographic operations as well as the disclosure rate.

We realized the same measurements with two relaxed versions of our protocol: (i) a naïve version that does not respect any design principle (the querier is at the center of the execution and handles all the information), and (ii) a basic distributed version that only respects the first two (i.e. without *imposed randomness* — the information is split between different actors but they are not chosen randomly).

We finally compared the results we obtained: the naïve version only requires one corrupted node to leak all of the exchanged information and does not scale well as the Querier acts as a bottleneck; the basic distributed version requires an inconsiderate amount of actors to achieve an acceptable disclosure (i.e. their number has to be greatly superior to the number of corrupted nodes) which, in turn, greatly increases the cost of the execution of a query; finally, our protocol offers the best features: the private information disclosure is guaranteed to increase linearly with the number of corrupted nodes, while the enforcement cost only increases logarithmically.

3 CONCLUSION

Personal Data Management Systems arrive at a rapid pace allowing users to share their personal data within large P2P communities. While the benefits are unquestionable, the important risks of private personal data leakage and misuse represent a major obstacle on the way of the massive adoption of such systems. This work is one of the first efforts to deal with this important and challenging issue. To this end, we propose a fully-distributed P2P system laying the foundation for secure, pertinent and efficient evaluation of aggregate queries based on user profiles. By considering a realistic threat model and guided by three design principles (knowledge dispersion, task compartmentalization and imposed randomness), we proposed a secure and efficient distributed protocol to protect both the data-at-rest and the data-in-use against an attacker controlling many nodes in the system. Furthermore, our simulation-based experiments show that our solution offers interesting properties: the private information leakage increases linearly with the number of corrupted nodes, while the cost of the security mechanisms increases logarithmically.

ACKNOWLEDGMENTS

This research is supported by the ANR PersSoCloud grant n°ANR-16-CE39-0014.

REFERENCES

[1] Serge Abiteboul, Benjamin André, and Daniel Kaplan. 2015. Managing your digital life. *Commun. ACM* 58, 5 (2015), 32–35.
[2] Michael Backes, Peter Druschel, Andreas Haeberlen, and Dominique Unruh. 2009. CSAR: A Practical and Provable Technique to Make Randomized Systems Accountable. In *NDSS*, Vol. 9. 341–353.
[3] Cozy Cloud. 2013. Cozy allow you to control your personal data (pictures, bank statements, bills, health reinbursements) in a secure and private space. Retrieved September 3, 2018 from https://cozy.io/en
[4] Fing. 2013. The mesinfos project explores and implements the self data concept in france. Retrieved September 3, 2018 from http://mesinfos.fing.org/english
[5] Troy Hunt. 2013. ';–Have I been pwned? Check if you have an account that has been compromised in a data breach. Retrieved September 3, 2018 from https://haveibeenpwned.org
[6] Saliha Lallali, Nicolas Anciaux, Iulian Sandu Popa, and Philippe Pucheral. 2017. Supporting secure keyword search in the personal cloud. *Information Systems* 72 (2017), 1–26.
[7] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. 2001. *A scalable content-addressable network*. Vol. 31. ACM.
[8] Adi Shamir. 1979. How to share a secret. *Commun. ACM* 22, 11 (1979), 612–613.
[9] Ion Stoica, Robert Morris, David Karger, M Frans Kaashoek, and Hari Balakrishnan. 2001. Chord: A scalable peer-to-peer lookup service for internet applications. *ACM SIGCOMM Computer Communication Review* 31, 4 (2001), 149–160.

A distributed, object oriented run-time and storage system, framework proposal for edge computing

Dorin Palanciuc
Florin Pop

ABSTRACT

DOORS is a proposed distributed system which provides execution and storage services in the form of objects and aims to avoid "impedance mismatches" that may occur in multi-paradigm frameworks. We describe the class of problems that DOORS solves, the architectural principles, and the current state of the research. We considered concurrency control, replication and the choice between consistency and availability. In the context of this analysis, we identified three different ”kinds” of inter-node communication: messaging, replication, and migration. In spite of their apparent similarity, these scenarios need to be addressed by separate design features, such that we are able to provide a simpler and more consistent system implementation.

Advanced analysis and visualization techniques in Big Data In datacenters

Andrei Talaș
Florin Pop

ABSTRACT

Extracting valuable information from raw data is difficult because of annually data growth rate. The difficulty to deal with comes from the fact that most of the data is unstructured. Reports suggest the rate of growth is between 40 and 60%. Additionally, data sources are heterogeneous and come from various contexts and situations. Datacenters always have been an important data source for analysis. Organizations continuously track in-house data, equipment data, data logs, network traffic for better optimization, efficiency, self-protection, etc. with the purpose of having decision support.

A Privacy Aware Approach for Participatory Sensing Systems

Dimitrios Tsolovos
Inria & U. Versailles St-Q. en Yvelines
Versailles, France
dimitrios.tsolovos@inria.fr

Nicolas Anciaux
Inria & U. Versailles St-Q. en Yvelines
Versailles, France
nicolas.anciaux@inria.fr

Valérie Issarny
Inria
Paris, France
valerie.issarny@inria.fr

1 INTRODUCTION

Mobile Participatory Sensing (MPS) systems can be used to provide useful data gathered from mobile devices that would otherwise require expensive deployments of sensor networks which would also need to be maintained, thus, further adding to the overall cost. Data collected by these applications typically include the location and time as well as the actual measurement itself which can vary from environmental and health data, to any other sensor data that can be collected by the majority of modern smart phones. This data can be used purely for the personal benefit of the individual user (e.g. *"self quantification"*, systems) or they can be used to benefit every user of the application (e.g. traffic tracking).

This type of data is sensitive since it can be used to identify participants and infer important information such as their interests, location or possible medical conditions. Omitting directly identifiable data, such as the name of a user, from a data set is not enough to protect their privacy [12]. An adversary can extract behavior patterns which, when combined with external knowledge, can identify users with high accuracy. MPS systems are by nature distributed and yet in most applications, participants report the data they have collected to a central server and thus "re-centralizing" it. This approach assumes, that users do not question the honesty of the hosting company nor its capacity to defeat severe attacks, since centralization creates in essence a massive honeypot.

Our objective is to provide a novel architecture for MPS systems which protects its users' privacy without sacrificing the utility of the system. The architecture should be designed such that participants can retain complete control over their data throughout the lifetime of the system and ensure that, an adversary performing a successful attack, will not be able to get access to all user data.

In the following sections, we present current privacy aware approaches for MPS systems and discuss their limitations. We extract the objectives that a privacy aware distributed MPS system should achieve and we provide an initial architecture in that respect.

2 PRIVACY IN TRADITIONAL MPS SYSTEMS

MPS systems are generally comprised of the following sub-systems: **a)** user registration, **b)** tasking, which includes the process of defining a task and finding suitable users to participate (task assignment) and collect relevant data, **c)** sensing and processing of local data, **d)** reporting, during which participants will transfer their collected data back to the server, **e)** global data processing, **f)** long term storage, and finally, **g)** presentation and end-user queries.

Each sub-system has different actors and consequently different types of adversaries. Typical MPS systems, generally have four types of actors. The server administrators, the tasking entities, the participants and the end-users. During the tasking process, a tasker defines a task by describing the points of interest, the types of measurements, the temporal scope of the task and more. Moreover, they define a function that needs to be applied on the collected data. An adversarial tasking entity might create a task that targets specific people by providing a very restrictive task description. Once the task is distributed and the data collected by the users, they will be called to report it to the server. Generally, this is where Privacy Enhancing Techniques (PETs) such as k-anonymity, differential privacy and data aggregation techniques are applied. Data need to be anonymized before reaching the server. Once on the server, a corrupted administrator can use the data to expose the participants privacy. Once the final results are published, end-users might be given the ability to perform queries. Their own privacy can also be exposed simply by inferring their interests based on their queries.

Privacy threats in MPS systems can be grouped into three categories: **a)** *Data snooping*, where adversaries might attempt to gain access to sensitive participant data by targeting specific participants. **b)** *Data inference*, where an adversary might attempt to extract additional information from the participants by based on the collected data and external information. **c)** *integrity of the system*, where adversarial actors might attempt to corrupt the system.

In recent studies, a trusted third party, is used to anonymize user data. In [2] and [8] that is an anonymization server while in [6] that role is held by the mobile phone service provider. In [3], participants are called to exchange their measurements with each other before reporting them back to the server. This way, the link between the participant and the data is broken. This alone is not sufficient and it needs to be coupled with other PETs as this link can be reestablished by analyzing a participant's history and combining it with external knowledge. For a survey of relevant studies see [1].

Current studies assume that central points of the system are trusted with user data. This in many cases is not realistic since companies like Google collect massive amounts of user data while their incentives lie in giving advertisers more relevant views to their ads. Additionally, they usually employ techniques such as k-anonymity or differential privacy, which reduce data utility. In general, centralized approaches fail to provide users with guarantees that their data will not be misused either intentionally, by negligence or in case of a successful attack against the server.

3 OUR APPROACH

A privacy aware architecture for MPS systems should enforce that the benefit-to-cost ratio of a successful attack is reduced and additionally, the cost of attacking multiple user data should increase (at least) linearly with the number of users. It should enable users to

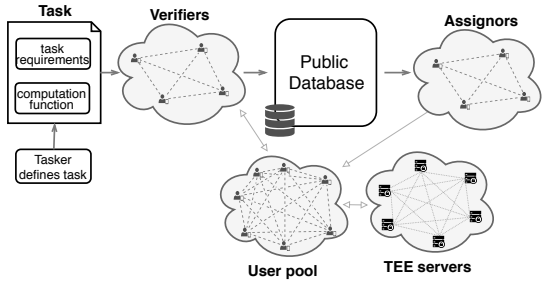

Figure 1: A high level overview of our proposal.

have complete control over their own data. The user should be the only one who has her own data in the clear and should have control over its deletion, usages and more. Finally, the architecture should not sacrifice main functionalities (such as optimal task assignment, rich computations, etc.) to achieve privacy.

Our approach relies on users keeping collected data on their own devices. This prevents an adversary from getting access to all user data with a single attack. This does not ensure that the benefit-to-cost ratio is reduced. An exploit that can "break" a device can be distributed to multiple devices without increasing the cost as seen in the recent Spectre and Meltdown attacks. The threats remain the same as in the centralized approach but their scope is different. The move to a decentralized setting ensures that users keep control of their data and can choose how they will be used. Additionally, on our approach we utilize devices equipped with a Trusted Execution Environment (TEE) [7, 9]. A TEE is a secure area of the main processor that guarantees code and data loaded in it will be protected with respect to confidentiality and integrity.

The main challenge when moving to a decentralized approach is achieving computations over global data. There are multiple techniques in the literature enabling privacy preserving distributed computations. These include Secure Multiparty Computations (SMC) [11] which does not scale well with the number of parties and gossip protocols [5] which can not be used for general computations. With the use of TEE enabled devices we can achieve fast and secure computations. Our objective is to ensure that an adversary breaking one device, will not be able to perform the same attack to break the other devices in the system. This will lead to the benefit-to-cost ratio being linearly reduced with the number of devices. Moreover, the attestation property that some TEEs offer can ensure that the result is a product of the specified function and which was not tampered with. In [4] the authors propose an architecture where users hold devices equipped with secure hardware with the goal of protecting the location privacy of the participants. Moreover, they propose protocols for efficient data aggregation. However, task assignment is not considered and location is not the only privacy sensitive information that participants provide in such systems.

In Figure 1 we present a high level architecture of our proposal. The key features of our architecture are the following: **a)** A user defines a task in the form of a smart contract and publishes it in an immutable, append and read only public database. **b)** A selection mechanism will select a subset of the available users (verifiers) who will test the task for well known tasking threats. **c)** If it passes the verification process, it can be distributed to the users. **d)** Another selection process selects a subset of users (assignors) who will assign the task to the available users based on their compatibility with the task description. **e)** The users will then perform the required measurements according to the task and save the collected data on their devices. **f)** With the assistance of a set of TEE enabled devices, the participants will collaborate to execute the defined function.

In this initial architecture user data is held on the user devices. When participants need to share their collected data in order to perform computations, the use of TEEs can help in keeping clear data away from adversaries. The two selection mechanisms we introduce are going to assume the role that a central server would have in a typical MPS system. Participants will need to collaborate in order to ensure that task verification and distribution is optimal while ensuring that user data is not collected on a single location. Ensuring the integrity and confidentiality of these two mechanisms is not trivial, even with the use of TEEs, since these can suffer from side channel attacks which lead to massive data leaks, thus impeding confidentiality, while the integrity of the result is not guaranteed when computations are distributed among devices.

4 FUTURE WORK

In this paper, we argued that, since MPS systems are inherently distributed, keeping them distributed can bring benefits to user privacy. Contrarily, it also brings a lot of challenges that will need to be overcome. Moving forward, we need to refine this initial architecture and prove that it can achieve our specific goals. Moreover we need to address data oriented tasks on TEEs. In particular, optimal task assignment and global computations over time series. An interesting challenge would also be to formulate the tasks as smart contracts, which, along with TEE attestation techniques, can provide integrity to the system. Finally, we plan to build part of this architecture and evaluate it over real user data collected by the Ambiciti project [10].

ACKNOWLEDGEMENTS

This work is part of a Ph.D. thesis funded by the Inria project CityLab@Inria co-supervised by Nicolas Anciaux and Valérie Issarny.

REFERENCES

[1] Delphine Christin. 2016. Privacy in mobile participatory sensing: current trends and future challenges. *Journal of Systems and Software* 116 (2016), 57–68.
[2] Cory Cornelius et al. 2008. Anonysense: privacy-aware people-centric sensing. In *6th international conference on Mobile systems, applications, and services*. ACM.
[3] Delphine Christin et al. 2011. Privacy-preserving collaborative path hiding for participatory sensing applications. In *Mobile Adhoc and Sensor Systems (MASS), 2011 IEEE 8th International Conference on*. IEEE, 341–350.
[4] Dai Hai Ton That et al. 2016. PAMPAS: Privacy-Aware Mobile Participatory Sensing Using Secure Probes. In *ACM SSDBM 2016*.
[5] David Kempe et al. 2003. Gossip-based computation of aggregate information. In *IEEE Symposium on Foundations of Computer Science*.
[6] Hien To et al. 2014. A framework for protecting worker location privacy in spatial crowdsourcing. *VLDB Endowment* 7, 10 (2014).
[7] Ittai Anati et al. 2013. Innovative technology for CPU based attestation and sealing. In *2nd international workshop on hardware and architectural support for security and privacy*, Vol. 13. ACM New York, NY, USA.
[8] Khuong Vu et al. 2012. Efficient algorithms for k-anonymous location privacy in participatory sensing. In *IEEE INFOCOM*. IEEE, 2399–2407.
[9] Tiago Alves et al. 2014. TrustZone: Integrated Hardware and Software Security-Enabling Trusted Computing in Embedded Systems (July 2004).
[10] Ventura Raphaël et al. 2017. Estimation of urban noise with the assimilation of observations crowdsensed by the mobile application Ambiciti. (2017).
[11] Oded Goldreich. 1998. Secure multi-party computation. *Manuscript*. (1998).
[12] John Krumm. 2007. Inference attacks on location tracks. In *International Conference on Pervasive Computing*. Springer, 127–143.

CWE Pattern Recognition Algorithm in Any-Language Source Code*

Sergiu Zaharia†
Security Center of Excellence
BearingPoint / Bucharest, Romania
sergiu.zaharia@bearingpoint.com

Published in the Proceedings of the BDA 2018 Conference (22–26 October 2018, Bucharest, Romania).

ABSTRACT

Source code became one of the backbones for business and personal processes, with significant growth rate. As applications are one of the most used attack surfaces against individuals and organizations from all sectors, their intrinsic vulnerability arising from the supporting source code must be reduced by design. Currently there are technology providers and open communities which provide Static Analysis Security Testing (SAST) solutions, able to detect vulnerabilities in code written in the most used programming languages and development frameworks.

The proposed solution consists of a Code Analysis Module that can identify vulnerability patterns in source code written in languages with less coverage, including code developed in languages which have not been previously learned by the solution. The ability of understanding and transforming unknown programming languages to the Intermediate Representation, which is then analyzed by a common machine learning algorithm for vulnerability patterns, is core idea for this research project.

CCS CONCEPTS

• Security and privacy / Software and application security

• Security and privacy / Software security engineering

KEYWORDS

Static Analysis, Software Vulnerabilities, Application Security

1. CONTEXT

Source code became one of the backbones for business and personal processes, with significant growth rate. GitHub, the development platform used by 28 million developers, declared that more than 2.9 trillion lines of code have been committed in 2017 alone [1]. As applications are one of the most used attack surfaces against individuals and organizations from all sectors in the last years, their intrinsic vulnerability arising from the supporting source code has to be reduced by design. Overall, there is a strong need for solutions being able to scan source code automatically and identify code level security vulnerabilities early in the software development phase. Currently, SAST solutions cover only Top 30 most used programming languages and development frameworks. The remaining programming languages (e.g. D, R languages) are not secured as result of the gap in both supporting technology and security experts. This situation opens a huge attack surface for hackers willing to compromise applications and consequently, organizations' or individuals' security. Today, an amount of 995 technical vulnerabilities - specific to different programming languages or common to all types of source code - is maintained by MITRE [3] as Common Weakness Enumeration (CWE) items.

2. PROPOSED SOLUTION

We recommend the Code Analysis Module which identifies CWE patterns in source code written in both popular programming languages or in languages with less coverage, including languages which have not been learned by the solution. From a functional perspective, this consists of the following two blocks:

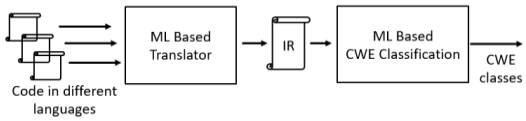

Figure 1: Code Analysis Module – Functional Blocks

The **ML Based Translator** is built on a Language-Agnostic scanner which transforms any language into an Intermediate Representation (IR) using similarities of lexical tokens within programming languages. Languages like C, Java and those inheriting them have quite similar keywords used by different but close grammars. This property can leverage the transformation of various languages in a common IR, using NLP-aware algorithms to choose the best IR keyword ($r_k^{(IR)}$, k=1..n, in Figure 2).

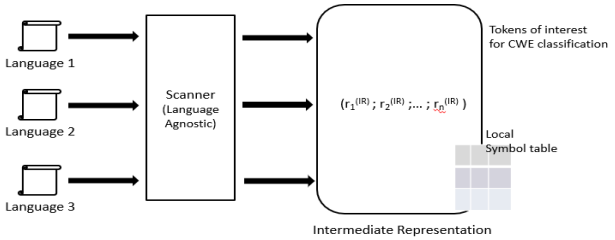

Figure 2: Transformation of programming languages in IR

For example, a snippet of CWE23 vulnerable code:

```
recvResult = recv(connectSocket, (char *)(data + dataLen), sizeof(char) * (FILENAME_MAX - dataLen - 1), 0); pFile = FOPEN(data, "wb+");
```

*Research topic under the PhD Program in University POLITEHNICA of Bucharest;
† Correspondence to: sergiu.zaharia@bearingpoint.com

is represented in IR format as below, where keyword are tokens not yet attributed to CWEs, which preserve program's context, and data flow is maintained through the relative positioning of tokens in the CWE classification input vector. The IR is practically an ordered set of CWE relevant tokens and generic identifiers, built using a specially designed Machine Learning algorithm.

IR for code snippet	IR for CWE classification algorithm
CWE_TOKENS['0'] = 'recv'	nr_cwe_tokens['0'] = '2'
CWE_TOKENS['1'] = 'FOPEN'	nr_cwe_tokens['1'] = '1'
CWE_TOKENS['2'] = '0'	nr_cwe_tokens['2'] = '0'
CWE_TOKENS['3'] = '0'	nr_cwe_tokens['3'] = '0'
CWE_KEYWORDS['0'] = 'char'	nr_cwe_keywords['0'] = '11'
CWE_KEYWORDS['1'] = 'sizeof'	nr_cwe_keywords['1'] = '13'
CWE_KEYWORDS['2'] = 'char'	nr_cwe_keywords['2'] = '11'
CWE_KEYWORDS['3'] = '0'	nr_cwe_keywords['3'] = '0'
CWE_KEYWORDS['4'] = '0'	nr_cwe_keywords['4'] = '0'
CWE_KEYWORDS['5'] = '0'	nr_cwe_keywords['5'] = '0'
IDENTIFIERS['0'] = 'recvResult'	generic_identifiers['0'] = '1'
IDENTIFIERS['1'] = 'connectSocket'	generic_identifiers['1'] = '2'
IDENTIFIERS['2'] = **'data'**	generic_identifiers['2'] = **'3'**
IDENTIFIERS['3'] = ''	generic_identifiers['3'] = '4'
IDENTIFIERS['4'] = 'dataLen'	generic_identifiers['4'] = '5'
IDENTIFIERS['5'] = 'FILENAME_MAX'	generic_identifiers['5'] = '6'
IDENTIFIERS['6'] = 'dataLen'	generic_identifiers['6'] = '5'
IDENTIFIERS['7'] = 'pFile'	generic_identifiers['7'] = '7'
IDENTIFIERS['8'] = **'data'**	generic_identifiers['8'] = **'3'**
IDENTIFIERS['9'] = '"wb'	generic_identifiers['9'] = '8'
IDENTIFIERS['10'] = '"'	generic_identifiers['10'] = '9'
IDENTIFIERS['11'] = '0'	generic_identifiers['11'] = '0'

Generic identifiers are abstract representations of real identifiers, whose position in the source code is maintained very loosely via an empirical symbol table (only generic name and relative positions are maintained). The ability of understanding and transforming unknown programming languages to IR - based on lexical similarities between programming languages - is core to this research project.

The **ML Based CWE Classification** block identifies CWE patterns in the IR, using machine learning algorithms for classification of non-linear patterns (e.g. SVM). The algorithm should be able to identify CWE classes for each source code snippet, using an "one versus all" approach. Code snippets may have more than one CWE vulnerability, consequently, the classifier may identify more than one class per each code snippet.

3. APPROACH

Research is planned in three main phases, as defined below,

Phase I: to design and build the training set for the CWE Classifier, starting from popular programming languages. The activities consist of identifying source code known as being vulnerable and the CWE-pattern relevant code snippets; designing the pre-processing algorithm applied to IR data sets, for later use in ML Based CWE classification algorithm; and finally, building the training data set for one programming language (e.g. C) and one vulnerability (e.g. CWE 23). We use NIST Juliet Test Suite [2] with vulnerable C, Java, C# and PHP source code. For C/C++, the repository consists of 8.67 million lines of code covering 118 CWEs. As today, the potential structure of IR has been designed considering the relevant tokens for specific CWEs, generalization of identifiers and relative positioning of tokens and identifiers, as a light data flow remanence.

Phase II: to identify the best model for the machine learning algorithm used for CWE pattern identification, using the training data sets from Phase I, and to enrich the solution with one more class (CWE pattern) and one more language (Java code snippets).

Phase III: to design and demonstrate the core concept of identifying CWE pattern in any-language source code. Includes Language Agnostic Scanner design for C and Java code translation to IR, adding new programming languages to adjust the algorithm, training the ML Based Translator to correctly represent the source code snippets in the IR format, and adjusting the accuracy of CWE pattern classification using the input resulted from the ML Based Translator, for C, Java and new languages.

4. RELATED WORK

Studies for source code splitting in smaller pieces like code snippets, logically mapped to vulnerabilities or code clone [4] do focus on specific languages, the Language-Agnostic Scanner not being in general addressed by previous work. One similar approach [5] considers deep learning for source code vulnerability detection. The authors use "code gadgets" to represent programs in a granular way, then vectorized as input to deep learning. The authors declare solution's limitations to C/C++ programs and to vulnerabilities dealing only with library/API calls.

Other studies [6][7][9][10][14][12] propose static analysis methods strongly related to one or two programming languages, even when the concept may be replicated for different languages. The drawback comes from the cumulated time and from the required expertise in both the new language scanner to be implemented and the static analysis concept itself defined in the respective study.

As an exception, ReDeBug [11] identifies latent security vulnerabilities in programs "written in different languages", as result of "a lightweight syntax-based code clone detection system" but limited to languages used in OS distributions. A different concept is implemented using signal processing techniques [13], which maintains the method language-independent, with the limitation that security vulnerabilities' localization is not realized.

ACKNOWLEDGMENTS

The author acknowledges Prof. Dr. Ştefan Trausan-Matu and Associate Professor, Dr. Traian Rebedea for their support and creative ideas provided under this PhD exercise.

REFERENCES

[1] https://github.com/ten, April 2018
[2] https://samate.nist.gov/SARD/testsuite.php
[3] https://cwe.mitre.org/
[4] Pham, Nam Hoai, *Detection of recurring software vulnerabilities* (2010). Graduate Theses and Dissertations, 11590, https://lib.dr.iastate.edu/etd/11590
[5] *VulDeePecker: A Deep Learning-Based System for Vulnerability Detection*, Z. Li, D. Zou, S. Xu, X. Ou, H. Jin, S. Wang, Z. Deng, Y. Zhong, 5 Jan 2018
[6] *Chucky: Exposing Missing Checks in Source Code for Vulnerability Discovery*, Nov. 2013, https://user.informatik.uni-goettingen.de/~krieck/docs/2013-ccs.pdf
[7] *Understanding Bag-of-Words Model: A Statistical Framework,* Yin Zhang, Rong Jin, Zhi-Hua Zhou
[8] *Generating robust parsers using island grammars*, IEEE2001, academia.edu/31982245/Generating_robust_parsers_using_island_grammars
[9] *Automated software vulnerability detection with machine learning*, J. Harer, L.Y. Kim, R.L. Russell, O. Ozdemir, L.R. Kosta, K. Rangamani, Lei H. Hamilton, Gabriel I. Centeno, Jonathan R. Key, Paul M. Ellingwood, Marc W. McConley, Jeffrey M. Opper, Peter Chin, Tomo Lazovich, 14 February 2018.
[10] *Automatic Inference of Search Patterns for Taint-Style Vulnerabilities*, F.Yamaguchi, A.Maier, H. Gascon, K. Rieck, Univ. of G¨ottingen, Germany
[11] J. Jang, A. Agrawal, and D. Brumley, *ReDeBug: Finding unpatched code clones in entire OS distributions,* in Proceedings of the 33th IEEE Symposium on Security and Privacy. IEEE, 2012, pp. 48–62.
[12] *SourcererCC: Scaling Code Clone Detection to Big Code*, H. Sajnani, V. Saini. J. Svajlenkoy, C. K. Royy, C.V. Lopes, 2015, USA
[13] *The use of machine learning with signal- and NLP processing of source code to fingerprint, detect, and classify vulnerabilities and weaknesses with MARFCAT*, Serguei A. Mokhov, Nov 2011, https://arxiv.org/pdf/1010.2511.pdfConference
[14] R.L. Russell, L. Kim, L.H. Hamilton, T. Lazovich, J.A. Harer, O. Ozdemir, P.M. Ellingwood, M.W. McConley, *Automated Vulnerability Detection in Source Code Using Deep Representation Learning*

Metadata based datasets placement in Smartgrid Ecosystems

Asma Zgolli
Christine Collet
Houssem Chihoub

ABSTRACT

In Smartgrid ecosystems it is important to choose the placement of datasets across different kind of data systems to achieve scalability and performance of workloads, our approach for datasets placement is based on datasets and workloads metadata but also on metadata describing the target data systems. Such metadata is the oil of our datasets placement module.