TRAITEMENT DES FLUX DE DONNÉES

Baptiste Csernel^{1,2} & Fabrice Clerot² & Georges Hébrail¹

csernel@enst.fr fabrice.clerot@francetelecom.com hebrail@enst.fr

1 :École Nationale Supérieure des Télécommunications de Paris

2 :France Télécom R&D

Résumé

Le monde comprenant de plus en plus de systèmes générant des flux de données, leur étude a généré ces dernières années beaucoup d'activité et commence donc à émerger comme un domaine à part entière au carrefour de l'algorithmique, des bases de données et des statistiques. Cet article se propose de présenter brièvement un état de l'ensemble des travaux actuellement en cours dans le domaine, afin de fournir une vision globale de ses problématiques et ses enjeux.

Abstract

After being the subject of numerous innovations and developments in recent years while the number of data flow generating applications kept growing, the area of data streams has started to form an identity of its own at the crossroad of algorithmics, data bases and statistics. This article's purpose is to present a brief overview of the various work currently being researched in the field in order to provide a global vision of the current problematics and aims.

Mots clefs

flux de données, fouille de flux de données, systèmes de management de flux de données.

Introduction

Un flux de données est une séquence infinie d'éléments générés de façon continue à un rythme rapide. Le terme "rapide" signifie ici que la vitesse d'arrivée des nouveaux éléments est grande devant les capacités de traitement et de stockage disponibles.

Les applications produisant des flux de données ou une sortie pouvant être modélisée comme telles ont connus une véritable explosion ces dernières années. On peut par exemple citer les logs de sites web, les tickets de communications fixes ou mobiles, ou bien encore les données de capteurs, de trafic routier par exemple mais aussi d'indices boursiers ou de météorologie. En réponse à ce nouveau besoin, des algorithmes pour traiter ces flux de données ont donc été développés. Ces derniers se trouvant au carrefour de trois champs disciplinaires, l'algorithmique, les bases de données, et bien sûr, les statistiques.

Quel que soit le domaine duquel ils sont issus, les algorithmes de traitement de flux de données doivent néanmoins tous respecter un certain nombre de contraintes. En effet, compte tenu de la nature d'un flux de données, il est impossible de stocker toutes les informations disponibles. Les algorithmes de traitements n'ont donc droit qu'à au plus une passe sur chaque élément et, malgré l'infinité du flux, doivent s'exécuter avec une quantité de mémoire fixe. Enfin, étant donné le rythme élevé d'arrivée des données, le temps de traitement par élément doit être impérativement court.

Cette communication s'attachera à présenter une vision globale et actuelle de l'ensemble des travaux en cours touchant aux flux de données. Une première partie présentera les différents modèles utilisés pour représenter formellement des flux de données ainsi que quelques opérations d'approximation classiques que l'on retrouve dans de nombreux algorithmes. Les problèmes touchant aux requêtes sur les flux de données, que ce soit leur syntaxe, leur réalisation ou encore le traitement de leurs résultats, sont ensuite abordés avant de traiter la question de la fouille des flux de données comme par exemple la classification ou l'agglomération de flux. Enfin, les différentes perspectives majeures de développement dans ce domaine encore jeune seront évoquées.

Modèles et Techniques de base

Bien que né de problèmes pratiques, le domaine des flux de données a rapidement fait l'objet de diverses modélisations par la communauté algorithmique, on distingue globalement deux modèles de flux de données qui peuvent être ordonnés ou non selon que leurs éléments arrivent ou non avec un ordre connu pour leur étiquette de temps.

Prenons l'exemple d'une suite de tickets de communications téléphoniques émis par une centrale d'appels que l'on souhaite analyser. On peut alors décider de considérer chaque ticket comme un objet à analyser. Dans ce cas, chaque élément du flux est un nouvel objet à analyser, on parle alors du modèle de la série temporelle. Mais on peut aussi considérer que les objets à analyser ne sont pas les tickets de communications mais les utilisateurs. Dans ce cas, chaque ticket représente non pas un nouvel élément à analyser mais bien une mise à jour des informations de l'utilisateur qui est l'objet sous jacent sur lequel porte l'analyse. On parle alors du modèle du tourniquet. Ces modèles ont permis à la communauté algorithmique de prouver des bornes de complexité sur les problèmes de flux de données ainsi que d'y rattacher un certain nombre d'autres problèmes pouvant être modélisés comme tels (S. Muthukrishnan, 2003).

Par ailleurs, étant donné qu'un flux de données est infini, on ne peut pas le traiter dans son ensemble. Il faut donc définir la portion du flux sur laquelle portera un traitement. On utilise pour ce faire des fenêtres qui marquent la portion du flux à traiter au moyen d'une date de départ et d'une date de fin. Si ces dates sont fixes on parle de fenêtre fixe, par exemple une fenêtre entre le 1er mars et le 1er avril. Si en revanche l'une ou l'autre des dates est relative on parle de fenêtre point de repère, une fenêtre entre le 1er mars et le temps courant en est un exemple. Enfin si les deux dates sont relatives, on parle de

fenêtre défilante, comme par exemple une fenêtre portant sur les trois derniers jours. On peut aussi exprimer ces dates grâce à l'ordre d'arrivée des éléments du flux comme par exemple une fenêtre entre le 657^èm élément et le 2000^èm ou portant sur les trois derniers éléments. On parle alors de fenêtre logique par opposition aux fenêtres physiques dont les exemples précédents font partie (L. Golab & M.T. Özsu, 2003).

Des approximations classiques ont également vu le jour pour répondre aux pressions sur la mémoire et la puissance de calcul exercées par les flux de données. La plus courante est sans doute l'échantillonnage, une méthode simple mais efficace et générale pour diminuer rapidement les problèmes de capacité de traitement. Elle pose néanmoins des problèmes de représentativité du fait de la temporalité du flux dont l'échantillon doit rester représentatif. Pour répondre à ces besoins, différentes techniques ont été développées selon la nature logique ou physique du fenêtrage. On retiendra notamment parmi les plus simples et efficaces les échantillonnages en réservoir, en chaîne et avec réserve pour les fenêtres logiques et l'échantillonnage par priorité pour les fenêtres physiques. (B. Babcock et al., 2002)

Une autre méthode de base pour économiser des ressources informatique est de traiter non pas les éléments du flux mais des structures de résumés ou synopsis qui permettent de calculer à moindre coût de mémoire ou de temps de calcul les réponses aux problèmes posés, parfois au prix d'une certaine approximation. C'est sans nul doute une des méthodes ayant fait l'objet du plus d'attention car elle permet en théorie de maîtriser la qualité des réponses en fonction du temps de calcul et de l'espace mémoire disponible. On distingue plusieurs techniques principales, les vaguelettes, les sketches, le décomptage d'éléments et les histogrammes (L. Golab & M.T. Özsu, 2003).

Moteurs de Requêtes

La première famille d'applications à avoir émergé pour répondre au problème posé par les flux de données est celle des moteurs de requêtes. Leur but est de permettre à l'utilisateur d'interroger le flux via une série de requêtes, tout comme s'il interrogerait une base de données traditionnelle. Naturellement, ces requêtes ne peuvent porter que sur le contenu du flux postérieur à l'exécution de la requête, en effet, seules les informations nécessaires au calcul des requêtes en cours sont conservées, le reste étant définitivement perdu.

Plusieurs langages de requêtes sur flux de données ont été proposés en se basant sur différents principes bien établis en base de données classiques. On distingue tout d'abord les langages relationnels tels que CQL, StreaQuel ou AQuery qui utilisent des syntaxes proches de SQL. Un exemple de requête calculant la longueur moyenne des dix plus récents appels longue distance de chaque client est donné ci-dessous dans le langage CQL :

```
SELECT AVG(S.longueur_appel)
FROM S [PARTITION BY S.id_client
        ROWS 10
```

```
WHERE S.type = 'Longue Distance']
```

Remplacer `ROWS 10` par `RANGE 10 DAYS` dans la requête ferait porter la moyenne non pas sur les dix appels longue distance les plus récents mais sur ceux passés dans les dix derniers jours. Les langages objets comme Tribeca et COUGAR ont eux aussi une syntaxe proche de SQL mais supportent en plus des flux de types abstraits de données (ADT) possédant chacun leurs propres méthodes. Enfin des langages procéduraux tels qu'Aurora proposent, à l'opposé des langages déclaratifs, de définir des requêtes en construisant graphiquement des arbres d'opérateurs. Tous ces langages supportent tout, ou une grande partie des types de fenêtres présentés précédemment, ainsi qu'un certain nombre d'opérateurs de base qui varient selon le langage et son champ d'utilisation supposé (L. Golab & M.T. Özsu, 2003).

Une fois le langage défini, il reste encore à écrire des algorithmes adaptés aux flux de données pour chaque opérateur ainsi que le système qui leur permettra de travailler de concert. Des solutions ont été apportées pour les schémas de requêtes les plus simples en utilisant des techniques incrémentales utilisant parfois les approximations présentées dans la partie précédente. De nombreux problèmes restent cependant encore ouverts, rendre incrémentales des requêtes SQL plus complexes par exemple, ou encore coordonner des requêtes sur des flux multiples, ou bien optimiser la coordination de requêtes en mémoire limitée dans le cadre des flux de données (B. Babcock et al., 2002).

Fouille de Flux de Données

Bien que les moteurs de requêtes sur flux de données répondent à une grande partie des besoins des utilisateurs, ils souffrent cependant de limitations. Comme par exemple leur incapacité à formuler de nouvelles requêtes sur des événements passés si de nouveaux besoins apparaissent ou à rendre compte du comportement global du flux. C'est pourquoi, tout comme en fouille de données classique, des algorithmes de fouilles de flux de données ont été développés.

Confrontés à l'infinité des flux de données, les algorithmes de fouille de données n'ont d'autre choix que de stocker des résumés s'ils veulent permettre une analyse portant sur une partie quelconque du flux. Reste à savoir quoi stocker et comment adapter la capacité de stockage limitée au caractère infini du flux à conserver sans pour autant laisser les résumés être dominés par le passé du flux. Aggarwal (2002) a proposé une réponse à ces deux questions qui s'est montrée populaire depuis dans le cadre d'un algorithme de classification automatique (clustering) de flux de données.

Cette réponse s'appuie sur deux idées, la première consiste à utiliser de micro-classes pour résumer le flux de donnée en mémoire centrale en ne gardant que les centroides de chacune d'elles. A son arrivée, un nouvel élément est affecté au centroïde le plus proche dont les statistiques sont mises à jour. Si aucun centroïde proche n'est trouvé, un nouveau est créé et un ancien détruit ou absorbé pour tenir compte de l'évolution du flux. Le devenir de chaque micro-classe est conservé via des identifiants qui permettent de conserver la

trace de leur évolution

La seconde idée est la mise en place d'un système pour photographier à intervalles réguliers l'état du système en sauvegardant sur disque l'ensemble des centroides des micro-classes. Ces clichés sont ensuite conservés selon une structure géométrique ou pyramidale de façon à en conserver un plus grand nombre pour les temps proches du temps courant que pour les périodes plus anciennes. Les propriétés mathématiques des micro-classes permettent de plus d'effectuer des opérations de soustractions sur les clichés pour obtenir le contenu du flux entre deux instants et observer ainsi l'évolution des micro-classes.

Enfin, la classification finale est réalisée sur les micro-classes qui sont agglomérées à leur tour en utilisant un algorithme classique de classification automatique pouvant fonctionner à partir de leurs statistiques. Ce sont ces agglomérats finaux, réalisés à partir de clichés bien choisis pour représenter une période précise du flux qui sont réellement représentatifs.

Ce principe d'un traitement en deux parties basé sur des micro-classes et la prise régulière de clichés a séduit par sa flexibilité et les vastes possibilités de traitements possibles sur les micro-classes mémorisées. De nombreuses variantes ont depuis vu le jour expérimentant d'autres structures de résumés ou adaptant cette structure en deux temps à d'autres algorithmes (N. Hun Park & W. Suk Lee, 2004).

Reste encore le problème du traitement relativement lourd des clichés pour obtenir de réelles informations sur l'évolution du flux, mais ce travail, si il reste à faire, relève plus de l'analyse de séries temporelles traditionnelles que du domaine des flux de données.

Perspectives

Bien que de plus en plus d'algorithmes soient développés pour traiter de façon générale les problèmes liés aux flux de données, beaucoup de domaines conservent, historiquement ou par nécessité de performance, des traitements qui leurs sont propres.

On peut ainsi voir des algorithmes spécialisés dans les traitements de flux de données produits par des logs de sites web (D. Tanasa & B. Trousse, 2004) ou d'autres encore spécialisés dans le traitement des données spatiales telles les données issus de l'astronomie ou de l'épidémiologie. On voit aussi émerger des algorithmes développés pour traiter un et un seul type de requête bien particulier comme détecter les objets les plus fréquents par exemple (G. Cormode et al., 2004).

Parallèlement à cela, les travaux sur les traitements généraux se poursuivent sur plusieurs fronts. Du côté fondamental, les algorithmiciens ont de plus en plus de résultats à la fois positifs et négatifs à leur actif qui permettent de mieux cerner les possibilités du domaine, bien que les heuristiques restent encore plus populaires.

Du côté applicatif, de nombreux développements sont aussi en cours notamment dans le domaine du traitement de flux multiples ou dans le but d'étendre le type des données traitées. En effet, les algorithmes de traitement pour les données qualitatives, mixtes ou symboliques restent inexistantes ou moins aboutis, de même que pour le traitement de

données de forte dimensionalité.

Il reste aussi tout un travail de tests et de perfectionnements à effectuer sur les outils déjà développés jusqu'à maintenant. En effet, ces derniers utilisent souvent des critères empiriques pour certains de leurs traitements dont l'impact et la pertinence a encore été peu évaluée. Bien peu d'études comparatives ont été faites sur l'efficacité relative des outils déclarés généraux en fonction de la nature des flux qu'ils analysent.

Enfin, tout un pan de la recherche est naturellement dédiée à perfectionner l'efficacité et la précision des outils existants en proposant de nouveaux algorithmes ou de nouvelles observations. Ainsi, bien qu'ayant fait l'objet d'une grande attention ces dernières années le domaine des flux de données reste riche et présente encore bien des défis à la communauté scientifique.

Références

- [1] C.C. Aggarwal, J. Han, J. Wang, and P.S. Yu (2003), *A framework for clustering evolving data streams*, Proceedings of the 29th VLDB Conference (Berlin, Germany).
- [2] R. Motwani B. Babcock, M. Datar (2002), *Sampling from a moving window over streaming data*, Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms (San Francisco, California), ACM SIAM, ACM Press, pp. 633–634.
- [3] B. Badcock, S. Babu, M. Datar, R. Motwani, and J. Widom (2002), *Models and issues in data stream systems*, Symposium on Principles of Database Systems, ACM SIGACT-SIGMOD, pp. 1–16.
- [4] G. Cormode, F. Korn, S. Muthukrishnan, and D. Srivastava (2004), *Diamond in the rough, finding hierarchical heavy hitters in multi-dimensional data*, International Conference on Management of Data 2004 (New York, USA), ACM SIGMOD, ACM Press, pp. 155–166.
- [5] L. Golab and M.T. Özsu (2003), *Data stream management issues - a survey*, Technical report cs 2003-08, University of Waterloo, Waterloo, Canada.
- [6] S. Muthukrishnan (2003), *Data streams algorithm and applications*, Internal review, Rutgers University AT&T Labs Research, Waterloo, Canada.
- [7] N. Hun Park and W. Suk Lee (2004), *Statistical grid based clustering over data streams*, SIGMOD Record, vol. 33, ACM.
- [8] D. Tanasa and B. Trousse, *Intelligent Systems* **19** (2004), *Preprocessing for intersites web usage mining*, 59–65, Issue 2.