”StreamByFacet”, un nouvel outil interactif d’exploration de données temporelles par une navigation de type facettes et un affichage en streamgraph

Eric Languenou, Pascale Kuntz

"StreamByFacet", un nouvel outil interactif d'exploration de données temporelles par une navigation de type facettes et un affichage en streamgraph

Eric Languenou*, Pascale Kuntz**

*LS2N/Université de Nantes
eric.languenou@univ-nantes.fr
**LS2N/Polytech, Nantes
pascale.kuntz@univ-nantes.fr

Résumé. La profusion des données en provenance des appareils mobiles et de la numérisation de nos vies, conjuguée au besoin des sociologues et des chercheurs en général d'analyser ces données ne cesse de stimuler la création d'outils d'exploration et de visualisation des données temporelles multi-dimensionnelles communément appelées traces. Nous proposons dans cet article une nouvelle approche qui associe la métaphore d'exploration à base de facettes "Elastic List" à une visualisation du nombre d'événements en "streamgraph" afin de fournir une exploration manuelle intuitive des données brutes. Le modèle de données de type facettes, et particulièrement celui lié à "Elastic List", et son adéquation avec la production de "streamgraphs" associés est ainsi mise en avant.

1 Introduction

La profusion des données en provenance des appareils mobiles et de la numérisation de nos vies, conjuguée au besoin des sociologues et des chercheurs en général d'analyser ces données ne cesse de stimuler la création d'outils d'exploration et de visualisation des données temporelles multi-dimensionnelles communément appelées traces. De plus, l'investissement intellectuel important indispensable afin d'utiliser les outils d'analyse statistiques de type R motive le développement d'outils intuitifs. Dans ce cadre, nous présentons dans cette communication *StreamByFacet*, un prototype fonctionnel d'exploration manuelle des données sans pré-traitement qui combine le paradigme *click and see* utilisant la métaphore *Elastic List* introduite par Stefaner et Muller (2007), et son approche de type facettes associée à une visualisation en *streamgraph* du nombre d'événements issu des données filtrées. L'apport du travail présenté réside d'une part, dans le processus de découverte de connaissances qui s'appuie sur la complémentarité de l' *Elastic List* et de la visualisation des données temporelles en *streamgraph*, et d'autre part dans la facilité de prise en main de l'outil.

L'article est organisé comme suit. Après un bref état de l'art, la section 3 présente le modèle de données de type facettes et plus particulièrement celui attaché à l'interface *Elastic List*, son extension au domaine temporel et son adéquation avec la visualisation en *streamgraphs*. La section 4 détaille les interactions disponibles et les choix algorithmiques associés. Des résultats

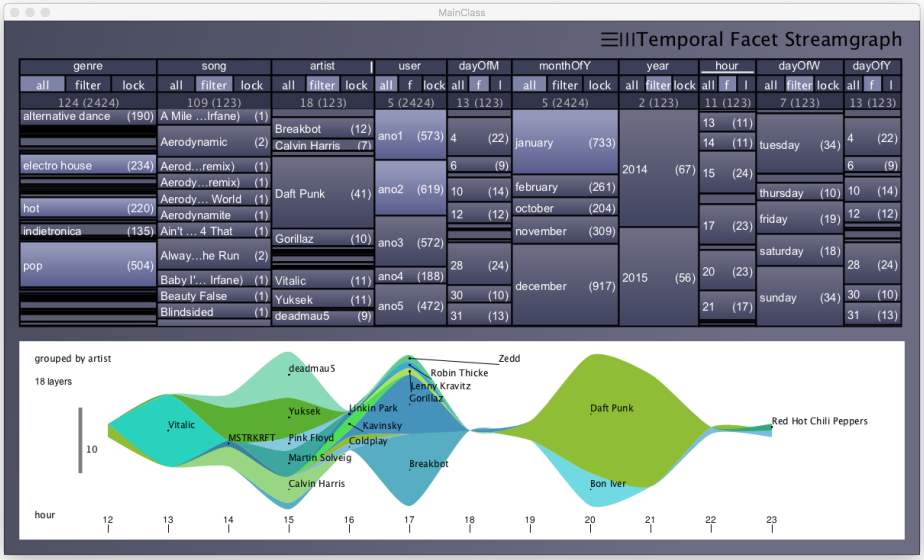

FIG. 1 – *Interface graphique du prototype "StreamByFacet"*

expérimentaux sur des données d'écoute musicale permettent d'illustrer le fonctionnement opérationnel du prototype *StreamByFacet* en section 4.

2 État de l'art

Les travaux présentés dans cet article se situent à la croisée de la visualisation de données temporelles et de la navigation par facettes, Wong et Bergeron (1997). L'analyse des données temporelles a fait l'objet de nombreuses recherches (voir Aigner et al. (2007)) qui ont conduit au développement de nombreuses métaphores : *data cubes*, *nano cubes*, cercles concentriques, Zhao (2015), spirales ou encore *small multipiles*, Bach et al. (2015). Les interactions, le besoin de vision globale, de filtres et de détails ont motivé des propositions variées : lentilles, Zhao (2015), matrices multi-échelles ou encore *horizon graphs*, Perin et al. (2013). En parallèle, les travaux sur l'exploration des différentes facettes des bases de données ont suscité le développement de plusieurs approches qui s'inscrivent dans un continuum : (i) affichage des résultats sous forme de liste ou de grille, (ii) navigateurs de type facettes pour lesquels le nombre d'objets de la facette est communiqué graphiquement à l'utilisateur, comme *FacetMap*, Smith et al. (2006), qui présente un ensemble de bulles afin de montrer les facettes, les valeurs atteintes et les filtres utilisés, (iii) visualisation continuellement adaptée aux modifications des filtres comme la *table lens approach* reposant sur une liste adaptée dynamiquement en fonction des interactions en utilisant une approche *Context+Focus*. La construction et l'exploration de données hiérarchisées issus de classifications des données a également été explorée, Stolper et al. (2014). Notons que dans la plupart des applications le filtrage des données affichées est réa-

lisé via la sélection de noeuds dans un arbre issu de classifications des données ou à partir des données brutes, avec sélection des filtres par différents moyens graphiques comme dans l'application *TimeSlice*, Zhao (2015).

Dans le prototype *StreamByFacet*, la visualisation est effectuée via un affichage basé sur les *streamgraphs*, qui sont dérivés des *stacked graphs*, Harris (1999), popularisé par une publication dans le *New York Times* montrant l'évolution des bénéfices du box office américain, Byron et Wattenberg (2008). Sur un *stacked graph*, chaque série temporelle est associée avec une bande dont la hauteur est proportionnelle à la valeur temporelle, les bandes étant empilées les unes sur les autres. Les *streamgraphs*, quant à eux, présentent des bandes dont les contours sont adoucis et dont la ligne de base, sur laquelle les bandes sont empilées au dessus et en dessous, peut évoluer sur l'axe y afin de limiter les distorsions (*themeRiver*). Leur design attractif provient de leur esthétique et de la facilité de voir l'évolution de la somme des épaisseurs de bandes et celle de chaque bande individuellement.

L'exploration interactive combinée à l'affichage de *streamgraphs* est ainsi adaptée aux données temporelles et plusieurs variations du modèle initial ont été proposées dans la littérature. *MultiStream*, Cuenca et al. (2018), est un outil où la sélection des valeurs à filtrer est réalisée à travers un arbre et la visualisation *via* un *streamgraph* avec un effet loupe et des niveaux de détails automatiquement adaptés. *TouchWave*, Baur et al. (2012), est une application interactive pour tablette affichant des *streamgraphs* et pour laquelle des interactions directes de modification des données visualisées ont été développées.

Notre outil *StreamByFacet* combine la représentation en *streamgraph* avec un modèle de navigation de type facettes interactif basé sur la métaphore *Elastic List*, Stefaner et Muller (2007). Ce modèle permet de restituer les distributions des valeurs atteintes, dans des colonnes correspondant aux dimensions de la base de données, composées de rectangles dont les hauteurs sont dépendantes des cardinaux des valeurs atteintes (voir la partie supérieure de la figure 1). À travers un paradigme *click and see*, l'utilisateur sélectionne des valeurs comme filtre en cliquant sur les rectangles correspondants. Toutes les colonnes sont alors mises à jour, les rectangles apparaissent ou disparaissent ou encore changent de hauteur pour refléter les nouvelles données des facettes de la base de données. Notons que dans la métaphore *Elastic List*, les objets résultant du filtrage sont affichés dans un deuxième espace sous forme de liste. Le résultat du filtrage des données pour les rectangles/valeurs sélectionnés correspond à une conjonction de disjonctions de valeurs. Les objets résultants doivent posséder, sur chacune des dimensions, une des valeurs sélectionnées dans la dimension. Citons, parmi les applications d'*Elastic List*, les travaux de Laube (voir Laube et al. (2008)) qui utilisent certains des principes d'*Elastic List* pour filtrer les données, mais pour lesquelles la visualisation des résultats ne semble pas être intégrée dans l'outil.

3 Modélisation

Afin de montrer l'adéquation entre la modélisation de type facettes, et particulièrement la navigation *Elastic List*, et l'affichage en *streamgraph*, cette section détaille le modèle de donnée de type facettes, l'extension au domaine temporel puis la manière dont la génération de *streamgraphs* est aisément réalisée à travers les méthodes décrites pour le calcul des facettes.

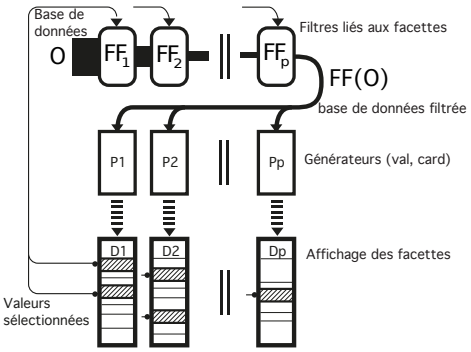

FIG. 2 – *Serveur de facette et colonnes*

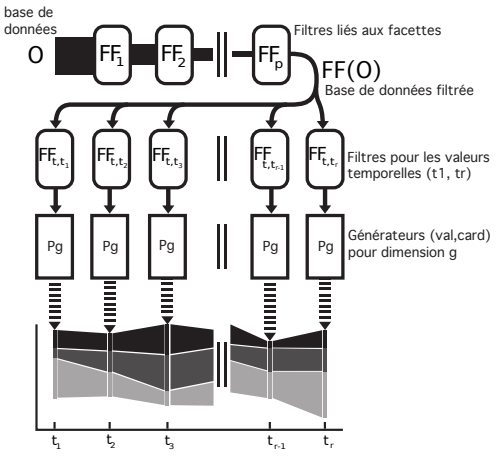

FIG. 3 – *Connexion au streamgraph*

3.1 Modélisation en facettes

Soit DB une base de données, contenant O un ensemble de nb objets o_i, $i \in [1, nb]$ chaque objet possédant p valeurs $(x_1, x_2, ..., x_p)$ et S un sous-ensemble de O. Soit x_{ik}, la valeur de l'objet o_i de S sur la k-ième dimension et $V_k(S)$, la facette k, soit l'ensemble des valeurs atteintes sur la dimension k. Cet ensemble peut être décrit par :

$$V_k(S) = \{x / \exists i \in [1, nb] \; et \; x_{ik} = x\}$$

La modélisation des données de type facettes fournit, pour chaque dimension, la liste des valeurs atteintes associées au nombre d'objets possédant chaque valeur (voir la partie supérieure de la figure 1). Ainsi, dans l'interface *Elastic List*, quand il n'y a pas de valeur sélectionnée par l'utilisateur, la facette k correspond à la liste ordonnée contenant $cardinal(V_k(O))$ couples valeur-cardinal $(v_j; n_j)$ où :

$$\begin{cases} v_j \in V_k(O) \\ n_j = cardinal\{o_i \in O / x_{ik} = v_j\} \end{cases} \quad (1)$$

L'utilisateur peut sélectionner des valeurs parmi celles affichées sur les différentes dimensions. Définissons U_k comme la liste des valeurs sélectionnées sur la dimension k et u_{ki} où $i \in [1, m_k]$ les m_k valeurs de cet ensemble. Ces valeurs vont opérer comme des filtres, réduisant la base de données initiale aux objets possédant une des valeurs de U_k sur k.

Définissons $FF_{k,v}$ une fonction de filtrage sur S qui ne garde que les objets possédant la valeur v sur la dimension k.

$$FF_{k,v}(S) = \{o \in S / x_k = v\} \; avec \; o(x_1, .., x_p)$$

En utilisant U_k, et pour étendre à l'ensemble des valeurs sélectionnée sur k, une nouvelle fonction de filtrage est définie : FF_k ne garde de l'ensemble S que les objets possédant une des valeurs de U_k sur la dimension k et rend l'ensemble S si U_k est vide.

$$\begin{cases} FF_k(S) = S & si\ U_k = \emptyset \\ FF_k(S) = \bigcup_{i\in[1,m_k]} FF_{k,u_{ki}}(S) & sinon \end{cases} \quad (2)$$

Elastic List filtre la base de données présentée de telle sorte que les objets retenus doivent posséder sur chaque dimension, une des valeurs sélectionnées par l'utilisateur. Le filtrage pour l'ensemble des dimensions est alors exprimé par une nouvelle fonction FF qui calcule l'intersection des résultats de filtrage sur chaque dimension.

$$FF(S) = \bigcap_{k\in[1,p]} FF_k(S) \text{ ou encore, } FF(S) = (\bigcirc_{k\in[1,p]} FF_k)(S)$$

Une fois les objets de la base de données filtrés, les valeurs atteintes et leurs cardinaux associés sont affichés via des métaphores graphiques (les hauteurs de rectangles). Pour ce faire, définissons P_k un générateur prenant un ensemble d'objets et produisant l'ensemble des couples valeur-cardinal pour la dimension k, ordonnés suivant l'ordre associé au type de la dimension k.

$$P_k(S) = \{(x, card)/x \in V_k(S)\ et\ card = cardinal(FF_{k,x}(S)\}$$

En appliquant $FF(O)$ pour filtrer la base de données complète et en utilisant $P_k(FF(O))$ sur chaque dimension k, les métaphores graphiques liées aux couples valeur-cardinal sont alors affichables (processus visible sur la figure 2).

3.2 Extension au temporel et à la visualisation en streamgraph

Dans le cas de la visualisation d'événements par *streamgraph*, le nombre d'événements à chaque pas temporel et pour chaque valeur atteinte doit être communiqué visuellement par les épaisseurs de bandes. Pour étendre le traitement au domaine temporel des dimensions sont ainsi ajoutées au modèle : année, mois, quantième, heures, minutes, secondes. Supposons que l'utilisateur désire grouper les valeurs suivant la dimension g et afficher le temps avec une granularité correspondant à la dimension t. Les fonctions définies précédemment sont naturellement utilisées afin d'obtenir les informations nécessaires à l'affichage du *streamgraph*, autrement dit, les valeurs et les cardinaux associés pour les valeurs de regroupement atteintes et ce, pour chaque pas temporel issu du filtrage.

$FF(S)$ réalise le filtrage suivant les valeurs sélectionnées par l'utilisateur. L'expression $V_t(FF(S))$ fournit $(t_1, .., t_r)$ l'ensemble des valeurs temporelles atteintes. Afin de calculer les valeurs rencontrées sur la dimension g, pour chaque temps t_i de cet ensemble, on utilise d'abord le filtre par valeur FF_{t,t_i} puis le générateur P_g qui fournit les valeurs atteintes sur g et leurs cardinaux respectifs. Finalement pour chaque valeur t_i comprise entre t_1 et t_r, on obtient :

$$P_g(FF_{t,t_i}(FF(S)))$$

Le processus complet est décrit sur la figure 3.

4 Visualisation et exploration

Le prototype *StreamByFacet* est développé en java et utilise la bibliothèque Processing (`www.processing.org`). Il intègre deux composantes majeures. La composante "facettes"

communique en elle seule des informations, via les métaphores graphiques, sur la répartition des données suivant les dimensions et pour des valeurs sélectionnées. À tout moment, l'utilisateur peut choisir une première dimension de granularité temporelle et une deuxième dimension de regroupement (bandes) afin d'engendrer le *streamgraph* correspondant. Ces deux dimensions, et donc colonnes, particulières sont identifiables dans l'interface par l'affichage d'un trait blanc horizontal pour le temps et d'un trait blanc vertical pour la dimension de groupe, dans l'entête de colonne (voir figure 1).

Les cardinaux sont communiqués à l'utilisateur via des hauteurs de rectangles à travers une fonction de transfert, la valeur et le cardinal correspondant étant indiqués en texte dans le rectangle. Lorsque la place dans les rectangles est insuffisante pour l'affichage textuel, un mode d'affichage *détail à la demande* peut être activé qui montre ces informations au survol de la souris dans les différentes colonnes et un dégradé de couleur est posé sur chaque rectangle afin d'indiquer ceux qui ne sont pas entièrement visibles. Les colonnes sont munies d'ascenseurs graphiques et l'échelle de représentation peut être modifiée. De plus, cette échelle peut être adaptée pour que la totalité des rectangles soit visible, ce qui communique en un regard les pourcentages occupés par les différentes valeurs (action *échelle adaptée*). À chaque clic de souris sur un rectangle, celui-ci est sélectionné ou dé-sélectionné et la totalité des autres colonnes est modifiée afin de présenter les nouvelles facettes résultant de la modification des valeurs filtrées. *Elastic List* dispose de 3 modes de contrôle sur chaque colonne : "tout" (*all*), "filtre" (*filter*) et "verrouillé" (*lock*). Le mode "tout" affiche toutes les valeurs de la base de données, le mode "filtre" n'affiche que les valeurs filtrées par les valeurs sélectionnées dans toutes les colonnes et enfin le mode "verrouillé" qui gèle les mises à jour et conserve l'affichage quelles que soient les sélections futures.

À tout moment, il est possible pour l'utilisateur d'engendrer le *streamgraph* correspondant aux deux dimension choisies, de regroupement et temporelle, et aux données filtrées, *streamgraph* pour lequel une navigation en déplacement et changement d'échelle est disponible ainsi qu'une exportation en image vectorielle. Les bandes sont organisées suivant l'ordre naturel de la dimension de regroupement. Les labels des bandes du *streamgraph* sont placés *via* un algorithme génétique et la palette de couleurs est choisie en fonction des limites fixées par l'utilisateurs sur les luminances, teintes et saturations. Une échelle placée sur la gauche de la figure informe sur le nombre d'enregistrements et au besoin, l'utilisateur peut, via un clic de souris, afficher les valeurs spécifiques pour une valeur temporelle donnée. Les bandes pouvant être étroites, un mode *détail* affiche la valeur correspondant à la bande située sous le pointeur de souris. L'utilisateur peut exporter la base filtrée, ainsi que les filtres actifs afin de conserver ces données pour un autre usage.

5 Application

Afin d'évaluer l'apport de notre outil d'exploration de données temporelles, différents jeux de données ont été testés (jusqu'à $500K$ enregistrements) et cette section examine les résultats sur une base de données issue d'une étude, Le Guern (2016), sur les usages musicaux à travers les outils numériques par les jeunes adultes.

Pour chaque personne, la base contient les morceaux écoutés par les principaux appareils de diffusion musicale ainsi que par les services de diffusion en ligne. Les titres des morceaux ainsi que les artistes et les dates d'écoute ont été enregistrés avec l'accord des participants.

Comme la musique est généralement référencée par les genres musicaux, la base de données *EchoNest* (à présent *Spotify*) a fourni ces derniers autorisant ainsi une exploration plus riche. Une recherche spécifique resultant de plusieurs valeurs filtrées sur différentes dimensions est donnée dans la suite, montrant ainsi la facilité de filtrage et d'exploration.

La figure 1 présente l'aspect de notre prototype après une exploration visant à afficher la distribution des écoutes au long de la journée regroupées par artistes des genres *electroHouse*, *hot* et *indietronica*, pour deux des personnes et pour le mois de janvier. Cette visualisation montre la facilité de sélection des filtrages sur les différentes dimensions. Pour l'obtenir, les genres musicaux sont sélectionnés dans la facette *genre*, les deux personnes *ano1* et *ano2* également, ainsi que le mois de janvier. La dimension "artiste" (*artist*) est placée en mode "filtre" et en zoom adapté. La partie du prototype montrant les facettes fournit de premières informations. On remarque qu'il y a 22 artistes pour 202 écoutes et que *Daft punk* est le groupe le plus écouté. La facette heure (*hour*) est placée en mode "filtre" et en "échelle adaptée", montrant ainsi la répartition des écoutes au sein de la journée. Trois pics sont visibles sur 15h, 17h et 20h. Quant à la dimension "jour de semaine" (*dayOfWeek*), elle indique par exemple que le mercredi, il y a moins d'écoutes. Le *streamgraph* est engendré après réglage de la granularité temporelle sur la dimension "heure" et le regroupement sur la facette "artiste" (les traits blancs sont visibles sur les entêtes des colonnes). Les trois pics sont confirmés, la part importante de *Daft Punk* également. Une exportation du *streamgraph* résultat sous forme d'image vectorielle (type *svg*), avec la légende de couleurs associée, est possible par l'intermédiaire d'un menu. Remarquons qu'il n'y a pas de limitation sur le type de dimension choisie pour le regroupement des valeurs et ainsi il est possible de regrouper suivant la dimension "jour de semaine". En choisissant alors la dimension "heure" pour granularité temporelle, le prototype affichera le nombre d'écoutes réparties en jour de semaine au long des heures de la journée.

6 Conclusion et travaux futurs

Des extensions sont prévues dans un futur proche. Elles concernent à la fois l'interactivité dans le *streamgraph* et le traitement des données de grande taille. Pour l'interactivité, il sera nécessaire de compléter les modalités de sélection pour restreindre les champs pris par les valeurs, en particulier pour les variables temporelles. Pour le passage à l'échelle, les expérimentations ont montré que l'analyse sans pré-traitement des données originales était parfois associée à une cardinalité très importante des ensembles de valeurs observées sur les variables. Un regroupement à la demande faciliterait l'interprétation. En sus, nous souhaitons approfondir les expérimentations en situation réelle afin de pouvoir évaluer plus précisément les qualités d'usage du prototype *StreamByFacet*.

Références

Aigner, W., S. Miksch, W. Müller, H. Schumann, et C. Tominski (2007). Visualizing time-oriented data-a systematic view. *Comput. Graph. 31*(3), 401–409.

Bach, B., N. H. Riche, T. Dwyer, T. M. Madhyastha, J.-D. Fekete, et T. J. Grabowski (2015). Small multipiles : Piling time to explore temporal patterns in dynamic networks. *Comput. Graph. Forum 34*(3), 31–40.

Baur, D., B. Lee, et S. Carpendale (2012). Touchwave : Kinetic multi-touch manipulation for hierarchical stacked graphs. In *Proceedings of ITS 2012*. ACM.

Byron, L. et M. Wattenberg (2008). Stacked graphs–geometry & aesthetics. *Visualization and Computer Graphics, IEEE Transactions on 14*(6), 1245–1252.

Cuenca, E., A. Sallaberry, F. Y. Wang, et P. Poncelet (2018). Multistream : A multiresolution streamgraph approach to explore hierarchical time series. *IEEE Trans. Vis. Comput. Graph. 24*(12), 3160–3173.

Harris, R. L. (1999). *Information graphics : A comprehensive illustrated reference*. Oxford University Press.

Laube, V., C. Moewes, et S. Stober (2008). Browsing music by usage context. In *In Proc. of 2nd Workshop on Learning the Semantics of Audio Signals (LSAS*.

Le Guern, P. (2016). *Où va la musique ?* Presses des Mines.

Perin, C., F. Vernier, et J.-D. Fekete (2013). Interactive Horizon Graphs : Improving the Compact Visualization of Multiple Time Series. In S. Brewster, S. Bødker, P. Baudisch, et M. Beaudouin-Lafon (Eds.), *Proceedings of the 2013 Annual Conference on Human Factors in Computing Systems (CHI 2013)*, Paris, France, pp. 3217–3226. ACM : ACM.

Smith, G., M. Czerwinski, B. Meyers, D. C. Robbins, G. G. Robertson, et D. S. Tan (2006). Facetmap : A scalable search and browse visualization. *IEEE Trans. Vis. Comput. Graph. 12*(5), 797–804.

Stefaner, M. et B. Muller (2007). Elastic lists for facet browsers. In *Proceedings of the 18th International Conference on Database and Expert Systems Applications*, DEXA '07, Washington, DC, USA, pp. 217–221. IEEE Computer Society.

Stolper, C. D., A. Perer, et D. Gotz (2014). Progressive visual analytics : User-driven visual exploration of in-progress analytics. *IEEE Trans. Vis. Comput. Graph. 20*(12), 1653–1662.

Wong, P. C. et R. D. Bergeron (1997). 30 years of multidimensional multivariate visualization. *Scientific Visualization, Overviews, Methodologies, and Techniques. Washington, DC, USA : IEEE Computer Society*, 3–33.

Zhao, J. (2015). *Interactive Visual Data Exploration : A Multi-focus Approach*. Ph. D. thesis, University of Toronto, Computer Science Dpt.

Summary

The abondance of temporal data incoming from mobile devices and the digitalisation of our lives, associated with the need for sociologists and researchers in general, has resulted in a continuous creation of tools to explore/visualize multi-dimensional temporal data. We propose in this article a novel approach which combines the facet based Elastic List metaphor to a streamgraph visualization in order to provide an intuitive manual exploration tool of raw data. Moreover, the article shows that the data model of Elastic List is specially suited to the production of streamgraphs.