

Anticipation de services dans un monde ouvert.



UMR 6285

Thomas ALVES

Encadrants : MAISEL Eric et DELOOR Pierre.
Avec la participation de : LE BODIC Laurent et LE GLEAU Sylvain

Table des matières

1	Introduction	4
1.1	Contexte	4
1.2	Objectifs	4
1.3	Problématiques	4
1.4	Déroulement	5
2	État de l'art	5
2.1	Interfaces plastiques	5
2.2	Ontologies	5
2.2.1	Un modèle de donnée	5
2.2.2	Taxonomies	5
2.2.3	Ontologies Souples, l'idée	6
2.3	ANTIMOINE et le profil utilisateur	6
3	Principes Généraux	8
4	Construction du Graphe de co-occurrence	10
4.1	Principes	10
4.2	Réalisation	12
5	Parcours des données	15
5.1	Profil Utilisateur	15
5.1.1	Principes : Phéromones	15
5.1.2	Formalisation	15
5.1.3	Réalisation	16
5.2	Échantillonnage de services et Mise à jours de l'interface	18
5.2.1	Services et paramétrage	18
5.2.2	Échantillonnage	18
5.2.3	Marqueurs d'utilisation	18
5.2.4	Pro-Activité de l'Application	18
6	Résultats	20
6.1	Scénario d'usage	20
6.2	Impressions	20
6.3	Complexité en temps	20
7	Conclusion et Perspectives	22
7.1	Conclusion	22
7.2	Perspectives d'améliorations	22
7.2.1	Changements de routine	22
7.2.2	Divers	22

Table des figures

1	[Rajaonarivo 2017]. Exemple de tags des 3 dimensions : Lieu=Brest ; Temps=XXème siècle ; Thème=Pont.	7
2	[Rajaonarivo 2017]. Le parcours de la base de donnée est modélisé par le parcours d'un environnement 3D peuplé d'objets de la base.	7
3	Principe global.	8
4	Principe général de notre solution.	9
5	Lorsque A et B sont cités, on s'attend à ce qu'ils s'attirent mutuellement et repoussent leurs autres voisins.	10
6	Modifications des poids portés par les arcs du modèle lorsque A B et C sont cités. . .	12
7	Liens de corrélation entre un nœud et chacun de ses voisins, tout au long de l'enrichissement du modèle à partir d'une base fixe de tweet.	12
8	Exemple d'un graphe réel obtenu via une requête simple sur Twitter. Ces graphes sont très rapidement super-massifs, et de plus en plus long et complexes à dessiner. (Ici le dessin a été réalisé par GraphViz)	13
9	Zoom sur une petite partie d'un graphe obtenu avec des données liées à Arkéa. (Ici le dessin a été réalisé par GraphViz)	14
10	Comportement des phéromones sur un graphe de test lorsqu'on dépose des phéromones à partir de "a" (et moins sur "a"). (Plus de phéromones → Plus de rouge.) .	17
11	Interface proposée.	21

Abstract

Dans un contexte ultra dynamique et très compétitif, les applications mobiles essayent de sortir du lot en proposant différentes innovations et toujours plus de personnalisation. Ce stage vient d'une idée d'Arkéa pour son application mobile : une application capable de s'adapter à chaque utilisateur. Ce stage s'articule autour de trois notions, le parcours de données dynamique dans un monde ouvert, la mise en place d'un profil utilisateur dynamique et auto adaptatif, et la co-construction en temps réel.

Afin de résoudre ces différentes problématiques, nous avons mis en place un système de graphe au maillage complet, s'inspirant du comportement des gaz, afin de classer les données du monde ouvert. Nous appliquons ensuite la métaphore des phéromones sur ce graphe afin de faire ressortir le profil de l'utilisateur, puis nous mettons à jours son interface afin qu'il puisse continuer son utilisation de l'application.

Mots clés : Co-Évolution, Interface, Clustering, Anticipation, Monde Ouvert.

1 Introduction

1.1 Contexte

À l'heure du Big Data, les quantités de données manipulées par nos logiciels et algorithmes explosent. De plus, ces données en grande quantités sont constamment en mouvement, nous poussant de plus en plus à adopter une façon de pensée beaucoup plus dynamique.

On se place dans le cadre de la consultation d'un réseau social, pour mieux comprendre un domaine ou un événement particulier. Pour cela l'utilisateur a à sa disposition un ensemble de **services**. Ces services sont des composants logiciels permettant de présenter des données à l'utilisateur, de modifier le modèle de donnée afin d'aller chercher des informations, de paramétrer d'autres services... Ces composants sont conçus selon une architecture MVC, l'interaction avec ces services se fait par leurs vues, et l'ensemble des vues présenté à l'utilisateur constituent notre interface.

1.2 Objectifs

L'objectif premier du stage est de fournir un prototype d'application dont l'interface s'adapte à l'utilisateur, en lui proposant les services paramétrés dont il a besoin en fonction de ses actions, notamment son parcours de nos données. Si les services élémentaires sont simples, des services composites paramétrés sont beaucoup plus intéressants pour l'utilisateur, une interface plastique devient alors nécessaire afin de s'adapter à chaque utilisateur pour anticiper ses besoins en terme de services.

Question Scientifique : Sommes nous en mesure de proposer un système permettant de co-construire en temps réel une interface en collaboration avec l'utilisateur, dans un monde ouvert, en tenant compte des besoins de l'utilisateur ainsi que de peu de connaissances a priori ?

1.3 Problématiques

Nous allons devoir proposer un prototype d'interface 2D s'adaptant à l'utilisateur, c'est donc tout naturellement que nous allons nous intéresser à différents travaux réalisés sur les interfaces plastiques.

La notion de monde ouvert nous apporte ses contraintes. Un monde ouvert est un monde où il est impossible de prévoir le contenu des données que nous allons recevoir, les possibilités sont potentiellement infinies. Nous savons seulement quelle forme auront les données que nous allons recevoir, ce qui constitue nos connaissances a priori. Nous allons donc avoir besoin de classer dynamiquement des données imprévisibles, et faire évoluer ce modèle au cours du temps. Nous allons donc nous intéresser à différents travaux réalisés sur le sujet.

La sensibilité des données bancaires d'Arkéa nous à empêché de nous en servir afin de développer notre prototype. Nous avons donc du trouver un autre cas d'application, respectant le cahier des charges initial. Nous avons donc choisi de travailler sur Twitter. La quantité de données est presque illimitée, il s'agit bien d'un monde ouvert et il nous est possible de proposer des services afin de manipuler ou observer les données, qui sont alors des tweets. Nous avons donc du mettre au points quelques services applicables aux données de Twitter.

1.4 Déroutement

Au fil de ce document, nous allons vous présenter les différentes étapes de réflexions, qui sont aussi les différentes étapes par lesquelles passe notre application dans son fonctionnement global. Premièrement nous étudierons ce qui a déjà été fait dans nos différents domaines, afin de se faire une idée des solutions couramment utilisées ainsi que des difficultés imposées par notre environnement. Nous aborderons ensuite nos questionnements sur le clustering dynamique, ainsi que les caractéristiques de notre solution. Ensuite, nous nous intéresserons à notre manière de construire le profil de l'utilisateur, et la sélection de termes qui sont supposés l'intéresser. Pour finir, nous discuterons de nos différents services, la manière de les paramétrer et de les échantillonner, puis du rafraîchissement de l'interface avec ces quelques nouveaux services.

2 État de l'art

2.1 Interfaces plastiques

Les interfaces plastiques sont un objectif théorique désignant des interfaces s'adaptant à leur environnement, problème posés par la mobilité de l'utilisateur, comme décrit dans l'article de G.Calvary[1]. Dans notre monde moderne, les supports d'applications sont innombrables et il semble vain d'essayer de développer et maintenir des interfaces dédiées pour chacun de ces supports.

La notion d'adaptation à l'utilisateur n'est pas nouvelle. Nous pouvons notamment citer J.Simonin[2] qui distingue deux catégories d'adaptation. L'adaptation à l'utilisateur, et l'adaptation à l'environnement. Il situe la plasticité dans cette seconde catégorie. La première catégorie se divise elle-même en deux grandes familles d'interfaces s'adaptant à l'utilisateur. Les interfaces personnalisables, offrant à l'utilisateur différents moyens de la modifier selon ses goûts sont dites **Adaptables**. Les interfaces s'adaptant à l'utilisateur grâce à des métriques mises en places par l'interfaces, sont appelées **Adaptatives**. Ainsi, l'utilisateur n'aura pas conscience, ou du moins ne sera pas à la source de la décision d'adaptation. C'est ce genre d'interface que nous souhaitons mettre en place.

2.2 Ontologies

2.2.1 Un modèle de donnée

Selon Gruber[3], « l'ontologie est une spécification explicite d'une conceptualisation ». Il s'agit de regrouper un ensemble de termes qui serviront à définir le sujet étudié. L'ontologie est donc un modèle de donnée, qui représente les concepts d'un domaine, ainsi que les relations entre ces concepts.

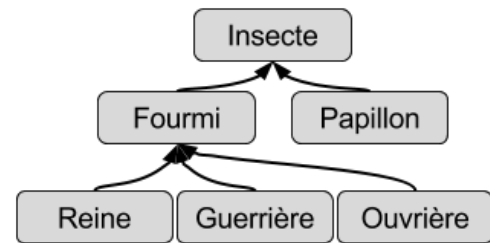
Une ontologie peut être qualitative ou numérique. Par exemple, une ontologie qualitative simpliste pourrait se composer de la manière suivante : Une fourmière est un ensemble de fourmis, et une fourmi peut être soit Reine, soit Guerrière soit Ouvrière. Nous pourrions aussi définir une ontologie de couleur, en associant à chaque couleur sa fréquence, et ainsi obtenir une ontologie numérique.

2.2.2 Taxonomies

Les taxonomies sont une des implémentations possibles d'ontologies. Elles permettent de représenter des relations de subsumption (notion de hiérarchie) entre des concepts du domaine considéré. Typiquement un arbre hiérarchique.

Ces taxonomies sont souvent utilisées pour indexer des collections d'objets ou de documents. Bonis et al[4] proposent d'utiliser ces taxonomies afin d'adapter une interface 3D (basée sur la métaphore de musée 3D) entre des utilisateurs et une base de données.

Dans notre cas, le monde ouvert nous empêche de correctement utiliser un système hiérarchique, puisque toutes les données auront la même forme. De plus, nous avons besoin d'un modèle beaucoup plus dynamique, afin de pouvoir s'adapter à des données possiblement très différentes, sans lien entre elles.



2.2.3 Ontologies Souples, l'idée

Les ontologies souples, développées par A.Collao[5] et présentées dans l'article de M.Kaipainen [6], apportent aux ontologies vues précédemment la souplesse de s'adapter à notre monde moderne hautement dynamique, et semblent toutes indiquées pour notre cas. Le principe clé d'une ontologie souple est d'offrir la possibilité de rajouter et d'enlever des concepts de l'ontologie, à la volée, pendant son utilisation. La gestion des données en est alors changée, largement assouplie et permet le traitement d'une grande quantité d'information. Les ontologies souples trouvent leur intérêt notamment du côté des bases de données collaboratives, où chaque utilisateur va rajouter les données dont il dispose, et les décrire au moyen de mots-clés qu'il aura lui-même choisis. Ces mots-clés n'auront peut-être jamais été saisis auparavant, et vont venir enrichir l'ontologie souple de cette base de données.

Notre monde ouvert nous impose une vision dynamique de la gestion de nos données. Nous allons donc fortement nous inspirer des principes apportés par ces ontologies souples.

2.3 ANTIMOINE et le profil utilisateur

La thèse de Landy Rajaonarivo et notamment cet article[7] ont été une grande source d'inspiration tout au long de la réalisation du stage. Dans son travail, du clustering est utilisé sur des données statiques afin de proposer à l'utilisateur les données qui l'intéressent le plus. Le parcours de la base de données est réalisé en utilisant la métaphore d'un musée aux multiples pièces (voir *figure 2*), où chaque pièce contient un cluster de la BDD, avec lequel l'utilisateur peut interagir. Ce qui nous intéresse particulièrement dans cette application, c'est qu'on y observe la jonction entre une **base de données importante** (avec des connaissances a priori), et un **profil utilisateur**, dont nous allons aussi avoir besoin.

Dans cet exemple, les éléments de la base de données sont des artefacts du patrimoine breton. Ainsi, chaque objet s'est vu attribuer une date, une position géographique et un thème (exemple *figure 1*). Les différentes mesures de distances ont ensuite été définies avec un expert. Les distances temporelle et géographique sont évidentes à calculer, mais la dimension du Thème est moins triviale. Ainsi une échelle entre 0 et 1 a été mise en place, 1 signifiant une concordance parfaite entre les deux thèmes des objets.



FIGURE 1 – [Rajaonarivo 2017]. Exemple de tags des 3 dimensions : Lieu=Brest ; Temps=XXème siècle ; Thème=Pont.

La génération des salles du musée est entièrement dynamique et réalisée à la volée, afin de s'adapter aux actions de l'utilisateur. Dans une salle, l'utilisateur parcourt les éléments présents et peut interagir avec eux. Selon ses interactions, dont des outils permettant directement de cibler une des 3 dimensions, l'algorithme de clustering va échantillonner un nouvel ensemble d'éléments pertinents pour l'utilisateur, et va les disposer dans une des salles suivantes. Chaque salle offre la possibilité d'aller dans 3 salles "successeurs", qui seront de plus en plus pertinentes pour l'utilisateur.



FIGURE 2 – [Rajaonarivo 2017]. Le parcours de la base de donnée est modélisé par le parcours d'un environnement 3D peuplé d'objets de la base.

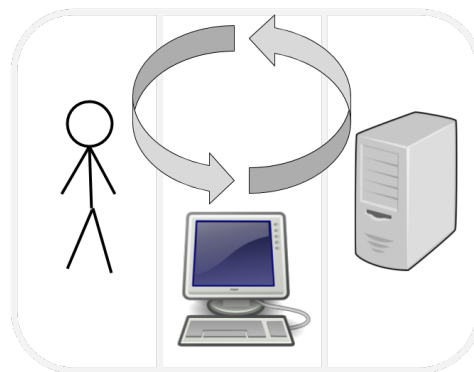
L'environnement 3D de ce projet ne sera pas nécessaire pour le stage car nous avons décidé de ne travailler qu'en 2D, afin de limiter la complexité et de se concentrer sur l'essentiel. Nous pourrions tout de même utiliser les notions soulevées par cette méthode, ainsi que d'autres notions fondamentales soulevées par B.Bonis[4] dans son article traitant de l'importance de la présentation des informations à l'utilisateur. Il met en avant la capacité d'adaptation de la présentation ainsi que la possibilité de personnalisation, importantes du point de vue de l'utilisateur.

3 Principes Généraux

L'objectif est de proposer à l'utilisateur des services sont supposés intéresser l'utilisateur, sans connaissance a priori. Pour ceci, nous avons besoin de déterminer quels messages sont plus susceptibles que d'autres d'intéresser l'utilisateur courant de l'application. Tout ceci en utilisant la co-construction, afin que l'utilisateur et notre application évoluent ensemble, de manière personnalisée et dynamique.

Pour cela, deux grandes parties distinctes du stage se profilent. Toute une partie que nous pourrions appeler "**clustering**", qui se chargera de déterminer la proximité entre chacun de nos éléments, et une partie "**parcours de données**", basée sur les résultats du clustering, qui devra nous renseigner sur les intérêts de l'utilisateur. À ces deux parties majeures viennent s'ajouter deux parties moins développées dans ce stage, l'**échantillonnage** de service et la mise à jours de l'interface avec ces nouveaux services.

Le principe de Co-construction de l'interface, entre le programme et l'utilisateur est fondé sur le paradigme de l'énaction. L'énaction est une vision de la cognition où différents systèmes s'influencent mutuellement pour évoluer au mieux. Dans notre cas, le parcours de l'utilisateur est guidé par les services que nous lui proposons, eux même guidés par les choix et actions de l'utilisateur. Le système est alors naturellement récursif.



Comme illustré sur la *figure 3*, nous disposons de deux mécanismes en parallèle, la gestion des données et la gestion du profil utilisateur. Nous recevons des données, nous pouvons alors construire nos méta-données ainsi que des relations entre elles. Ensuite, avec les actions utilisateurs, et en se basant sur nos méta-données, nous pouvons construire et modifier un profil utilisateur. Un échantillonnage est alors réalisé et l'interface est mise à jours, l'utilisateur est alors libre d'interagir avec les nouveaux services qui lui sont proposés, générant de nouvelles actions utilisateurs.

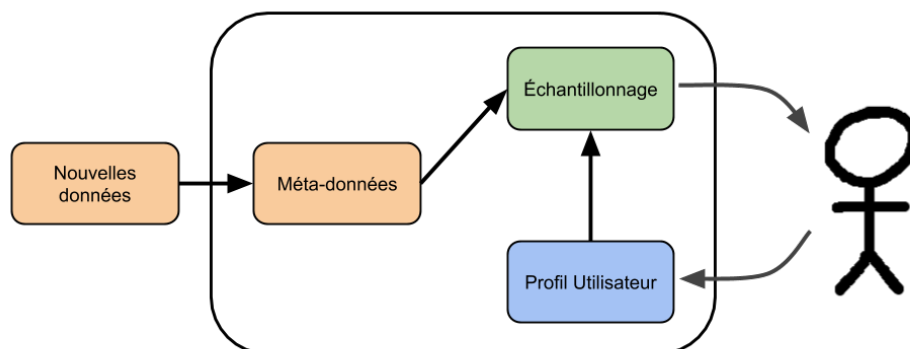


FIGURE 3 – Principe global.

Sur la *figure 4* nous voyons le même schéma de principe mais étoffé de nos propositions. Nous allons construire nos connaissances sur les données que nous recevons grâce à un graphe de **co-occurrence** que nous allons étoffer au court de l'utilisation de l'application. Ce graphe est dit "de co-occurrence" car il se base sur l'apparition des termes dans les mêmes messages. Afin de tenir compte des intérêts de l'utilisateur, nous allons utiliser la métaphore de phéromones que nous allons déposer sur le graphe précédent. Ainsi nous pouvons marquer les termes qui intéressent l'utilisateur, et oublier des choix passés qui ne sont plus d'actualité pour lui. Ensuite l'échantillonnage est réalisé et de nouveaux services sont présentés à l'utilisateur via l'interface, fermant la boucle de récursivité, et de co-construction.

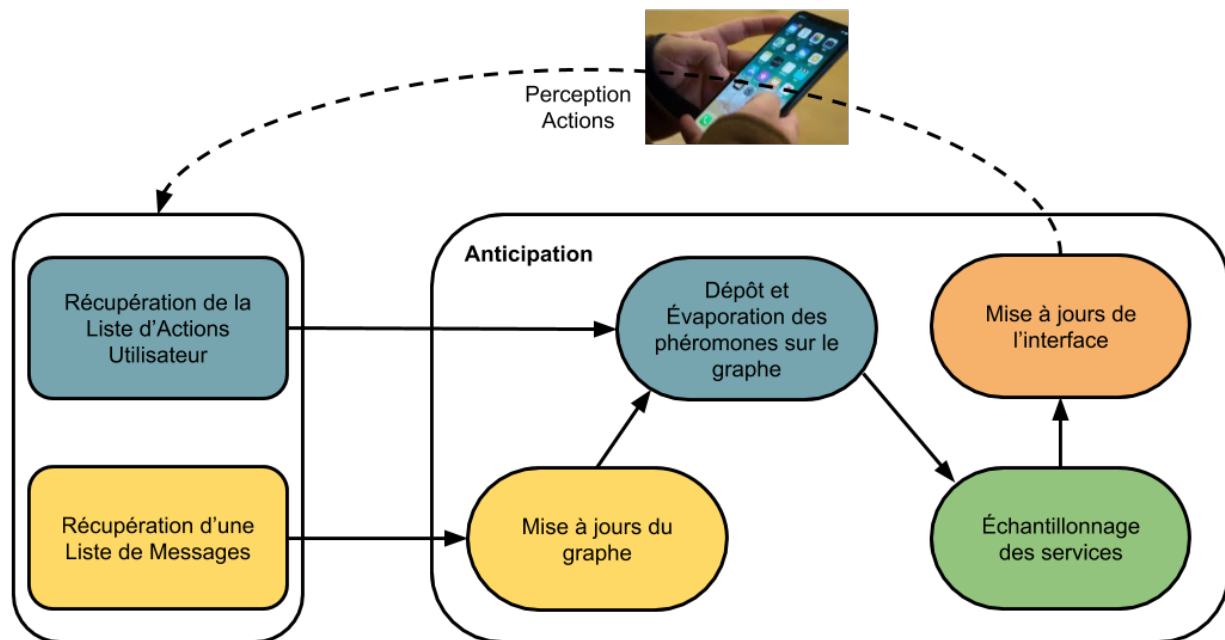


FIGURE 4 – Principe général de notre solution.

4 Construction du Graphe de co-occurrence

4.1 Principes

On doit pouvoir comparer des éléments provenant d'un monde ouvert de manière dynamique afin de pouvoir calculer des "distances" entre ces éléments.

Idée fondatrice Afin de trier les différents termes entre eux, le postulat de base est de dire que plus ils auront un sens proche, plus ils seront cités en même temps. Nous allons donc obtenir un résultat statistique.

Inspiration GNG L'idée du Growing Neural Gas [8][9] contenant des nœuds et pouvant se contracter et s'étirer afin de remplir un espace nous a inspiré l'idée du modèle de co-occurrence des étiquettes.

Ainsi on va utiliser un Graphe au maillage complet. Les étiquettes que nous voulons comparer vont être associées à des nœuds du graphe. Chaque arc portera alors une valeur de corrélation. Les arcs sont directionnels, ainsi deux arcs lient deux nœuds, sans forcément être symétrique ($A \rightarrow B \neq B \rightarrow A$). La valeur de ses arcs est comprise entre 0 et 1 et indique le taux de corrélation des tags. Le graphe ainsi créé va alors "s'étirer et se contracter" en fonction des séries d'étiquettes que nous allons lui fournir. Deux nœuds possédant une connexion directe non nulle sont appelés voisins. La figure 5 illustre le comportement général voulu d'un nœud envers ses voisins lors de l'arrivée d'un message. Le nœud central auquel nous nous intéressons a été cité avec un de ses voisins. Il va alors se rapprocher de ce-dernier et repousser tout ses autres voisins.

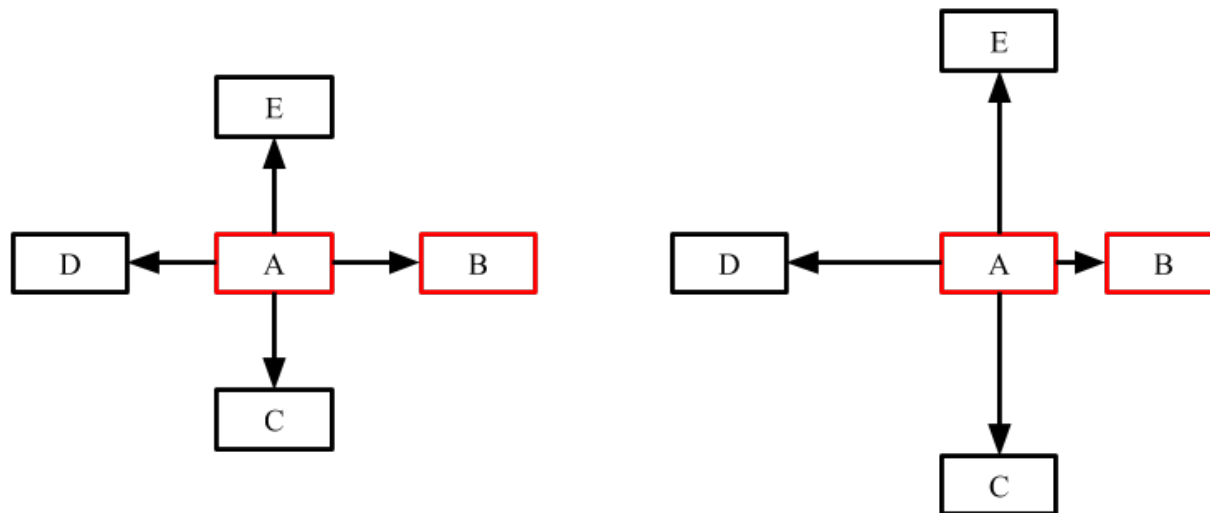


FIGURE 5 – Lorsque A et B sont cités, on s'attend à ce qu'ils s'attirent mutuellement et repoussent leurs autres voisins.

Formalisation

Notre graphe de co-occurrence est noté G_{co} :

$G_{co} = \{X, \Gamma, v\}$ avec :

X : L'ensemble des sommets, associés aux étiquettes connues.

Γ : L'ensemble des arcs : $\Gamma = X^2$.

$v : v(x1, x2)$, représentation de la corrélation entre les sommets $x1$ et $x2$. Tous initialisés à 0.

Calcul de v : Le modèle est modifié à chaque réception de message. On a alors :

m : un message. α et β deux paramètres.

$C(m)$ l'ensemble des étiquettes citées dans m .

$\forall x1, x2 \rightarrow x1 \in X \wedge x2 \in X$:

Si les deux termes sont cités,

soit $x1 \in C(m) \wedge x2 \in C(m)$: Contraction.

$v(x1, x2) \leftarrow \min(1, v(x1, x2) + \alpha)$

Sinon, si un seul des termes est cités,

soit $x1 \in C(m) \wedge x2 \notin C(m)$: Dilatation.

$v(x1, x2) \leftarrow \beta \times v(x1, x2)$

Sinon, si aucun des termes ne sont cités,

soit $x1 \notin C(m) \wedge x2 \notin C(m)$: Conservation.

$v(x1, x2) \leftarrow v(x1, x2)$

Dans notre cas nous avons fixé α à 0.5 et β à 0.9.

Manipuler α permet de régler la vitesse à laquelle un nœuds devient proche d'un autre. Une valeur trop importante saturerait le modèle qui deviendrait inexploitable, et une valeur trop faible ne permettrait pas de faire ressortir des relations, tant le graphe est volumineux.

Manipuler β permet de régler la vitesse à laquelle deux nœuds s'éloignent l'un de l'autre lors de la dilatation. Une valeur trop importante écraserait trop rapidement de nouvelles relations, les oubliant trop rapidement. Une valeur trop faible saturerait le modèle de lien extrêmement forts, car trop peu de liens seraient oubliés.

Nous pouvons observer *figure 6* un exemple numérique de l'évolution du graphe lors de la réception d'un message contenant 3 des 5 termes connus par le graphe. Les liens non représentés sont nuls. Puisque A, B et C sont cités, nous voyons que les liens entre ces trois termes se contractent, se voyant appliquer une addition de 0.5, notre premier paramètre. En revanche, B à un lien non nul vers E, de même que A vers D, or ces deux termes n'ont pas été cités. Ainsi les liens de B vers E et de A vers D subissent une dilatation, le poids est multiplié par 0.9.

Il est important de noter que les liens de D vers A et de E vers B sont inchangés, car ni D ni E n'ont été cités. Cette asymétrie est utile pour conserver l'information. Prenons un cas extrême pour exemple, nous avons deux étiquettes, A et B. A est ***toujours*** cités avec B, mais B est cité deux fois plus souvent avec une autre étiquette. A aura alors un lien très fort vers B, mais B aura un lien un peu plus faible vers A.

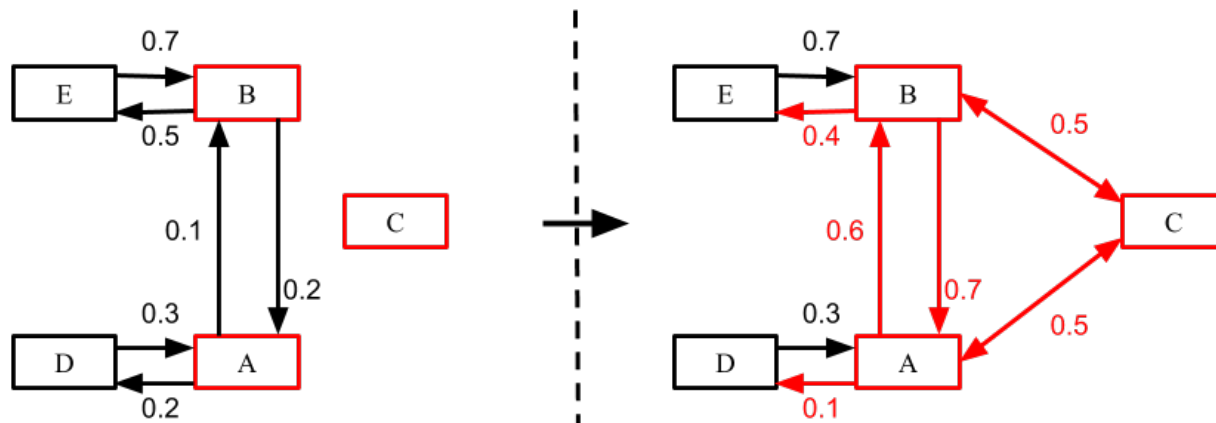


FIGURE 6 – Modifications des poids portés par les arcs du modèle lorsque A, B et C sont cités.

4.2 Réalisation

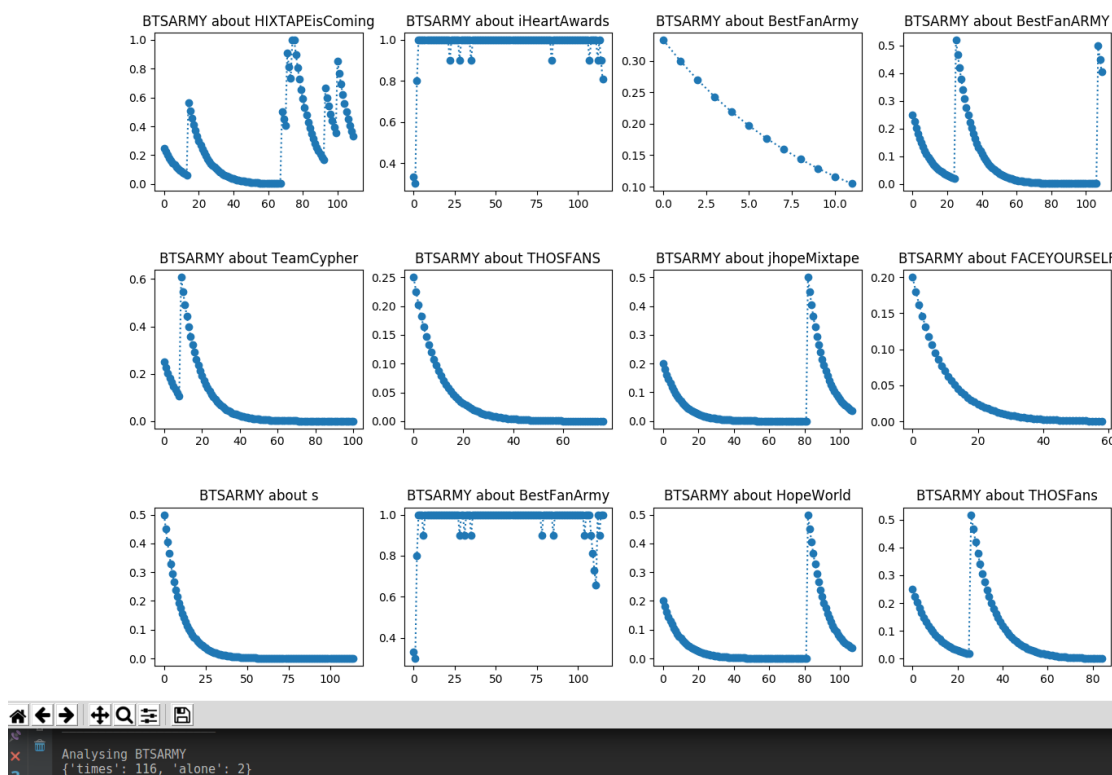


FIGURE 7 – Liens de corrélation entre un nœud et chacun de ses voisins, tout au long de l'enrichissement du modèle à partir d'une base fixe de tweet.

On observe sur la *figure 7* l'évolution dans le temps de la force des différents liens qui lient un nœuds à ses voisins. L'abscisse correspond aux modifications des poids en fonction du temps, et l'ordonnée la force du lien. Ce graphe a été obtenu avec des données diverses provenant de Twitter, et ce nœuds a été choisi pour démonstration car il possédait un nombre de voisin et une fréquence d'apparition intéressante. En effet, la grande majorité des termes reçu lors de requêtes générales ne

sont cités qu'une fois. On observe trois grands profils types sur cette figure, résumés dans les trois premiers graphes de la première ligne.

Profil 1 - **Lien sémantique** : La force de la corrélation fluctue énormément au fil du temps, ce qui veut dire que les termes sont cités ensemble quelques fois mais ont aussi un sens séparément.

Profil 2 - **Égalité sémantique** : Le terme observé est systématiquement cité avec ce terme voisin, la corrélation est donc très élevée. On peut donc aisément imaginer que ces deux termes ont un sens très proche.

Profil 3 - **Lien faible** : Les deux termes ont été cités une seule ou très peu de fois ensemble, le poids de la corrélation reste très faible. On peut conclure que si les termes ont eu un lien à un moment, ils sont désormais éloignés.

Nous pouvons observer *figure 8 et 9* deux exemples de graphes de co-occurrence obtenus via l'application.

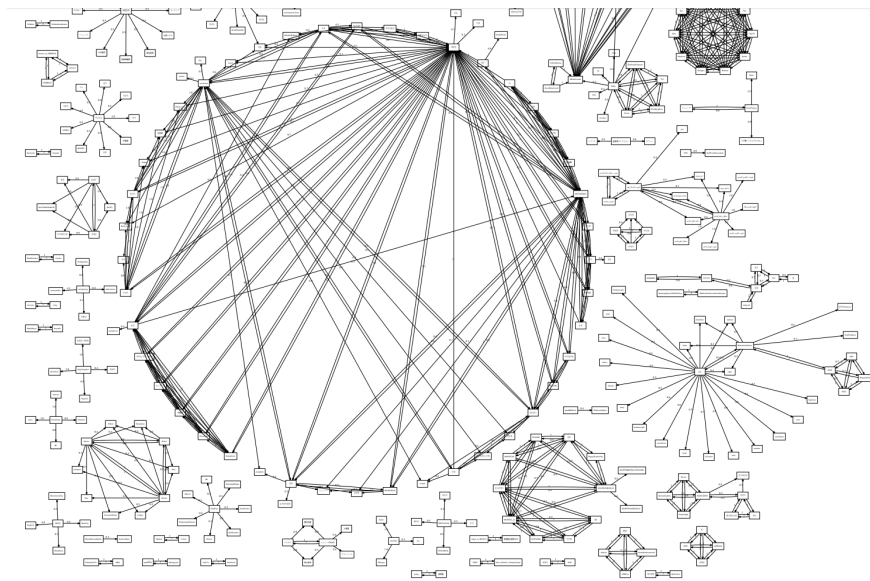


FIGURE 8 – Exemple d'un graphe réel obtenu via une requête simple sur Twitter. Ces graphes sont très rapidement super-massifs, et de plus en plus long et complexes à dessiner. (Ici le dessin a été réalisé par GraphViz)

On observe sur la *figure 9* l'apparition de deux termes extrêmement proches : "Arkéa" et "Arkea". Ces deux termes différents ont la même racine, mais il est important de conserver l'information de leur différence, car en rechercher un sur Twitter ne donnera pas les mêmes résultats que de rechercher l'autre. Ainsi, pour aider l'algorithme à tout de même trouver un lien entre ces deux termes, nous ajoutons un liens artificiel vers tout les noeuds qui ont la même base, sans tenir compte des accents et des majuscules. Ce lien artificiel permet le regroupement de plus de concept et une exploration plus simple de nos données.

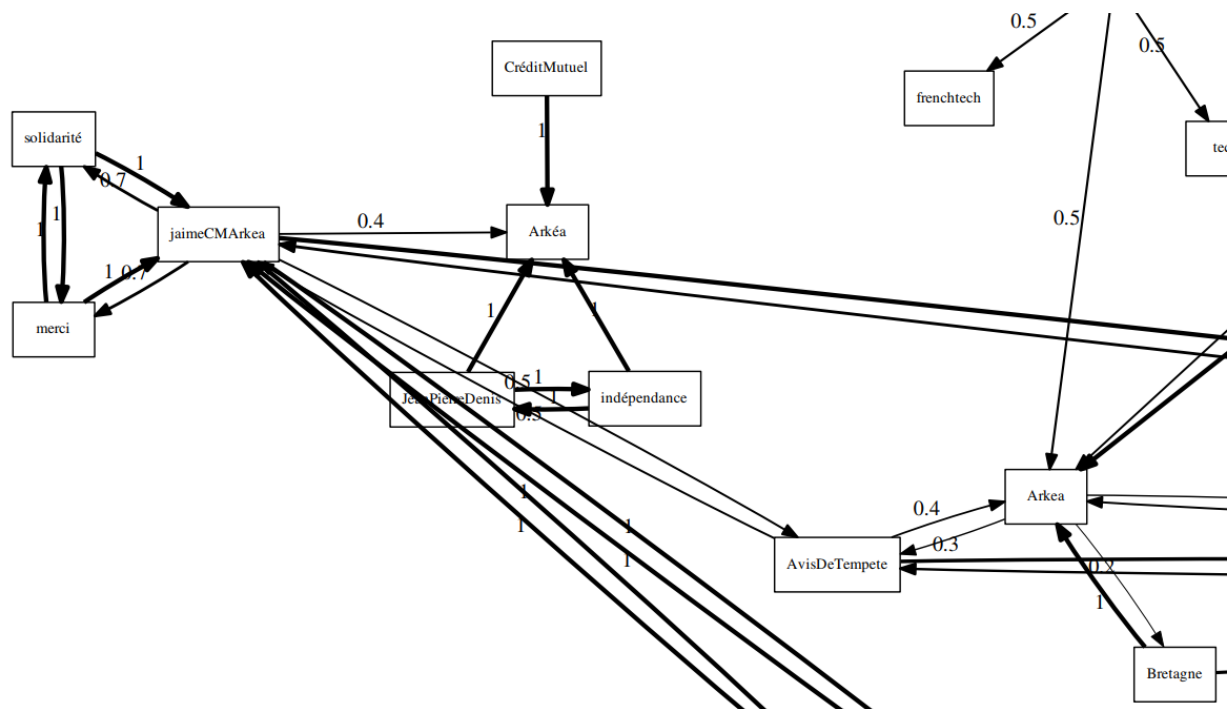


FIGURE 9 – Zoom sur une petite partie d'un graphe obtenu avec des données liées à Arkéa. (Ici le dessin a été réalisé par GraphViz)

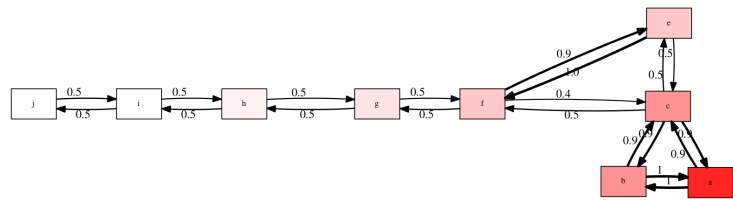
5 Parcours des données

5.1 Profil Utilisateur

5.1.1 Principes : Phéromones

Maintenant que nous avons à disposition un graphe de co-occurrence, nous savons évaluer des distances entre chaque terme. Nous avons désormais besoin de savoir quoi montrer à l'utilisateur, et contre connaître ses points d'intérêts, afin de pouvoir proposer des services paramétrés avec des termes proches de ses centres d'intérêts.

Pour cela, nous allons nous inspirer de la métaphore des phéromones, puis l'adapter à notre cas. L'idée est simple, nous allons déposer des phéromones sur des nœuds du graphe lors d'actions utilisateurs, puis nous allons répandre ces phéromones récursivement de voisin en voisin, en prenant en compte la force des liens reliant chacun. Les phéromones présentent deux intérêts majeurs pour notre application.



Marquer l'intérêt : Des phéromones sont déposés sur un nœud afin de noter l'intérêt que l'utilisateur porte à ce nœud. En propageant cet intérêt dans notre graphe de co-occurrence, nous pouvons proposer d'autres nœuds intéressants à l'utilisateur.

Oubli dynamique Ces phéromones seront alors périodiquement "évaporées". Diminuer régulièrement la quantité de phéromones sur tout le graphe, tout en continuant d'en déposer sur les termes que l'utilisateur consulte nous permet d'oublier les termes qui ne l'intéressent tout simplement plus, et de découvrir des nouveaux termes qui l'intéressent.

Désormais, avec cet ensemble de phéromone, il nous est possible de choisir au hasard un terme supposé intéresser l'utilisateur courant. Pour ceci, nous allons séparer notre tirage en deux parties. Une partie des termes seront tirés purement aléatoirement, afin de maintenir un taux d'exploration, l'autre partie est un tirage SoftMax simplifié, une "*roue de la fortune*", où la chance d'un terme d'être tiré est proportionnelle à son taux de phéromones.

5.1.2 Formalisation

Le modèle est modifié par deux événements, le dépôts de phéromones sur un nœud ou l'évaporation de phéromones sur tout les nœuds.

On a :

N : La liste de tout les nœuds du graphe.

$\phi(n)$: Le taux de phéromones d'un nœud. Toujours dans l'intervalle $[0 : 1]$

$\Delta(n1, n2)$: La distance entre le nœud 1 et le nœud 2. Toujours dans l'intervalle $[0 : 1]$

Lors de l'évaporation :

$$\forall n \in N : \phi(n)_{+1} \leftarrow \phi(n) * \alpha * \beta$$

Avec $\alpha \in [0 : 1]$ et β aléatoire $\in [0 : 1]$

Lors du dépôt de phéromones sur un nœud particulier, que nous allons appeler n_i :

$$\forall n \in N :$$

si $n \neq n_i$:

$$\phi(n)_{+1} \leftarrow \min(1, \phi(n) + \varrho * \Delta(n, n_i))$$

Avec ϱ le taux de propagation des phéromones, compris dans l'intervalle $[0 : 1]$.
 On considère ensuite le paramètre de dépôt initial noté $p()$, valant 0 si on ne souhaite pas déposer des phéromones sur n_i , et 1 dans le cas contraire. On a alors :

$$\phi(n_i)_{+1} \leftarrow \min(1, \phi(n_i) + \min(0.2, p()) * \varrho)$$

Paramètres Empiriquement nous avons fixés les différents paramètres afin d'avoir des résultats concluants. Nous avons fixés le taux d'évaporation α à 0.97, et la part d'aléatoire β à $[0.9 : 1.0]$. Une évaporation trop lente ne permet pas l'oubli d'ancien concept et sature le modèle, alors qu'une évaporation trop rapide ne permettrait à aucun concept de ressortir suffisamment longtemps afin d'être proposé à l'utilisateur.

La propagation ϱ a été fixée à 0.7, afin d'atténuer le dépôt de phéromones, pour ne pas saturer le modèle.

5.1.3 Réalisation

Puisque nous souhaitons utiliser ces phéromones pour choisir le terme le plus susceptible d'intéresser l'utilisateur, nous souhaitons éviter que le système sature, et que quelques termes soit toujours systématiquement proposés. Afin d'éviter cela, deux mécanismes légèrement différents ont été créés pour déposer des phéromones. Lors du dépôt, nous avons alors le choix de déposer ou non des phéromones sur le nœuds initial.

Si l'utilisateur consulte un terme, il marque son intérêt pour ce terme mais n'a absolument aucun besoin qu'on le lui propose à nouveau lors de la prochaine itération des propositions. Ne pas déposer de phéromones supplémentaire sur ce terme, mais sur ses voisins permet de prioriser ses voisins directs pour de futures propositions. Par contre, si l'utilisateur interagit avec un terme sans le consulter, c'est qu'il souhaite qu'on le lui propose. Alors des phéromones seront déposées sur ce nœud initial, en plus de ses voisins.

De plus, nous avons ajouté une part d'aléatoire dans l'évaporation des phéromones, afin que les proportions entre termes ne soient pas conservées lorsqu'ils s'évaporent tous, et ainsi permettre d'explorer un peu plus notre ensemble de terme.

Nous pouvons voir sur la *figure 10* un exemple de la répartition des phéromones dans un graphe de démonstration. L'utilisateur a consulté le terme 'a', nous propageons alors des phéromones à partir de ce nœud. Les prochaines suggestions logiques seront alors 'c', 'd', 'e' ou 'g' qui sont en effet les plus proches de 'a' lorsque nous prenons en compte la force des liens du graphe de co-occurrence.

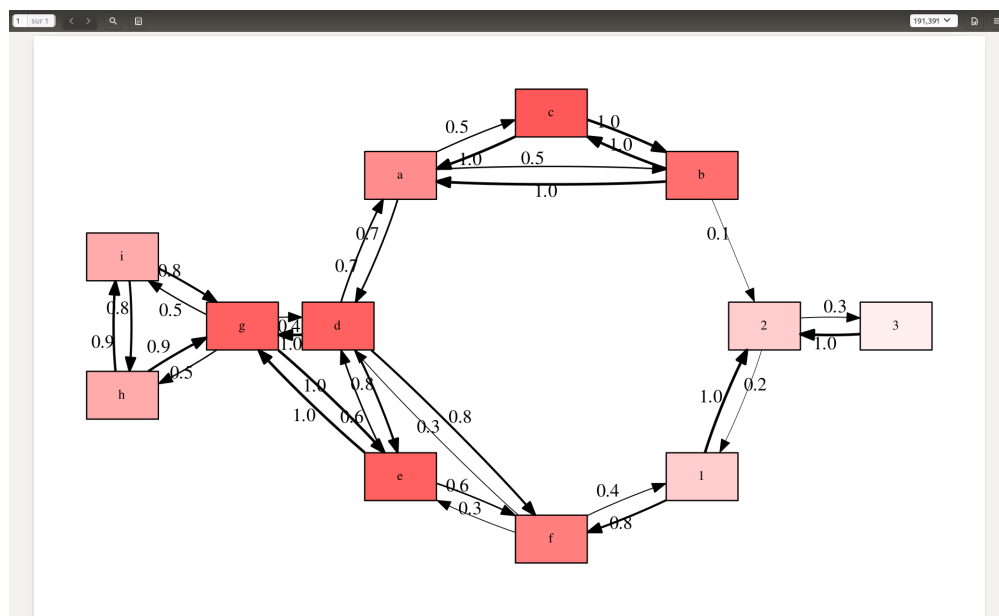


FIGURE 10 – Comportement des phéromones sur un graphe de test lorsqu'on dépose des phéromones à partir de "a" (et moins sur "a"). (Plus de phéromones → Plus de rouge.)

5.2 Échantillonnage de services et Mise à jours de l'interface

5.2.1 Services et paramétrage

Il nous est difficile d'imaginer une grande quantité de services adaptés a Twitter, en gardant en tête les besoins d'Arkéa. Ainsi nous avons isolé deux services paramétrables par des mots clés :

Ces deux services sont :

- Liste : Présente une dizaine de tweet avec une thématique commune. Contenant le même mot-clés, ou provenant du même auteur (le paramètre).
- Cloud : Présente un nuage de mot-clés, sélectionné parmi les voisins proche du mot-clé paramètre.

Ces paramètres nous permettent d'obtenir une grande quantité de services différents avec la même base abstraite. Deux même services paramétrés différemment sont alors considérés comme des services différents, ce qui nous amène à un nombre de service largement suffisant pour notre application.

5.2.2 Échantillonnage

À chacun de ces services est associée une valeur d'exploration, qui servira à les discriminer lors du choix. L'idée est d'utiliser un tirage aléatoire de type SoftMax en se basant sur ce score d'exploration. Ensuite, nous avons besoin de quelques données caractéristiques des interactions de l'utilisateur, afin de déterminer un taux d'exploration lui correspondant. Le SoftMax est alors réalisé sur la différence entre cette exploration voulue et le taux d'exploration de chaque service.

Ce score permet de tirer un service aléatoirement en fonction de son taux d'exploration, peu importe le nombre de services disponibles. En effet, l'ajout de services au cours de la vie de l'application est une demande d'Arkéa. Dans notre application, la Liste a une valeur d'exploration à 0 et le Cloud à 1.

5.2.3 Marqueurs d'utilisation

Afin de déterminer quel taux d'exploration est préférable pour l'utilisateur courant, nous utilisons deux marqueurs, l'activité et la concentration de l'intérêt. L'activité se mesure à l'aide d'un compteur, qui s'incrémente au fil du temps et ne se remet à zéro qu'en cas d'interaction utilisateur. Ainsi, plus ce compteur est élevé, moins l'utilisateur trouve ce qu'il souhaite dans ce que nous lui proposons. Nous allons donc privilégier un fort taux d'exploration. La concentration de l'intérêt dérive légèrement de l'activité, mais ajoute de la précision. Elle est mesurée en prenant le taux de phéromone présent sur le terme le plus "phéromoné". Si beaucoup de phéromones sont présentes au même endroit, l'utilisateur s'intéresse à un faible voisinage, c'est donc qu'il a trouvé un sous ensemble de terme qui l'intéresse, nous allons alors avoir un taux d'exploration faible.

5.2.4 Pro-Activité de l'Application

Afin de donner à l'application les moyens de s'adapter rapidement et efficacement à l'utilisateur, nous lui donnons la possibilité de modifier l'interface même lorsque l'utilisateur ne fait rien. Ainsi, notre programme est en mesure de proposer un certain nombre de nouveaux services régulièrement, permettant à l'utilisateur de trouver plus facilement ce qui l'intéresse.

De plus, en nous servant du marqueur d'utilisation d'activité de l'utilisateur, nous pouvons directement ajuster le nombre de propositions "rafraîchies" par l'application en fonction du client.

S'il n'a pas interagit depuis un certains temps, nous pouvons considérer qu'aucun services proposés ne l'intéressent, nous allons donc en remplacer d'autant plus.

Ergonomie : Pour des raisons d'ergonomie, une animation à été implémentée, permettant à l'utilisateur de mieux voir qu'une proposition de service à été mise à jours. Lors du rafraîchissement, le rectangle entourant le nouveau service est dessiné Rouge et revient progressivement à la couleur normale, attirant l'œil à coup sûr.

6 Résultats

6.1 Scénario d'usage

Afin d'illustrer le fonctionnement de notre application, nous vous présentons ici un cas d'utilisation.

Tout d'abord, nous arrivons sur l'application vide, seuls les carrés rouges extérieurs proposent deux services fixes. L'un permet d'initialiser son parcours de donnée à partir d'une des tendances Twitter en France en ce moment, l'autre permet de rechercher n'importe quel mot clé. Nous décidons alors de rechercher le mot clé "**UBO**". Après cette première interaction, l'application nous propose ses premiers résultats.

La recherche étant vague, la majorité des résultats ne concernent pas l'Université de Bretagne Occidentale, mais parmi les différentes suggestions étrangères nous trouvons un service de Liste permettant d'observer les Tweet d'un compte "**@UBO_UnivBrest**". Nous décidons d'ouvrir ce service, qui vient alors s'afficher au centre de notre écran, dans les cases gris clair. Là nous pouvons lire quelques tweet provenant de l'UBO, et observer un nouveau set de services qui nous sont proposés. À mesure que nous lisons, certaines des propositions sont mise à jours avec leur animation rouge. Là nous pouvons ouvrir un service de Liste à propos d'un compte "**@GPiegeais**" (doctorant en histoire à l'UBO), ou encore "**@CNRS_DR17**", mais nous décidons de cliquer sur le service de Liste de "**@Océanopolis**".

Nous découvrons ici grace aux tweets d'océanopolis qu'un événement est en cours, la fête des océans. Intéressé, nous ouvrons alors le service de Liste paramétré par "**#FeteOcean2018**" qui vient d'apparaître pour en savoir plus. Nous découvrons ici, dans des tweets de "**@PLCommunication**" et "**@Respon_SEA_ble**" que dans le cadre de la fête internationale des océans 2018, un festival de 4 jours a eu lieu à l'aquarium tropical de Paris.

Nous observons bien sur ce court extrait d'utilisation de l'application que nous réussissons simplement à découvrir de nouvelles informations pertinentes pour nous avec l'aide de l'algorithme.

6.2 Impressions

À l'utilisation, notre application semble efficacement s'adapter à l'utilisateur. Les services fixes présent en bas servent régulièrement afin de réorienter l'algorithme dans une nouvelle direction, et les services proposés sont le plus souvent pertinents. L'application pourrait être soumise à une expérimentation scientifique et rigoureuse, ce que nous n'avons pas fait dans le cadre de ce stage.

Les quelques démonstrations que nous avons réalisés avec d'autres utilisateurs ont été concluantes. En suivant leurs indications, l'application faisait ressortir des concepts pertinent pour eux.

De plus, Twitter étant notre exemple, les services que nous avons mis en place ne sont pas optimisés pour une bonne utilisation de Twitter, mais servent de démonstrateurs. Ainsi, le parcours de Twitter n'est pas forcément optimal.

6.3 Complexité en temps

Plus nous utilisons notre application, plus le graphe de co-occurrence se charge, et ainsi l'application devient de plus en plus lente, car le parcours de ce graphe devient de plus en plus coûteux. Empiriquement et d'après l'algorithmie et la structure du graphe, nous pouvons dire qu'il s'agit

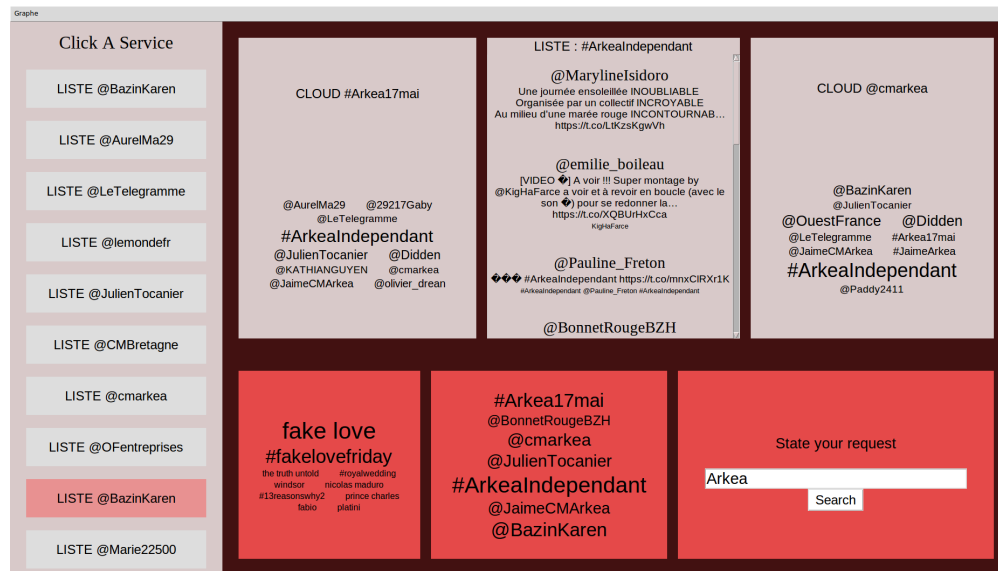


FIGURE 11 – Interface proposée.

d'une complexité au moins **quadratique**, sûrement **exponentielle**. Une réduction de ce coût serait bénéfique, mais l'optimisation nécessaire sort du cadre du stage (parcours de graphe au maillage complet).

7 Conclusion et Perspectives

7.1 Conclusion

Nous prenons nos données chez Twitter, ensuite insérées dans le graphe de co-occurrence. En parallèle, les actions utilisateur déposent des phéromones sur ce même graphe, phéromones servant à échantillonner les termes que nous allons proposer. Pour finir, quelques métriques sur l'utilisateur nous permettent d'échantillonner nos services, que nous allons proposer à l'utilisateur en les paramétrant avec les termes déjà échantillonnés. Ces services vont alors venir charger de nouvelles données, modifiant le graphe, mais aussi déposer d'autres phéromones, et fermer la boucle récursive de la co-évolution.

La co-construction en ligne de notre application est donc réelle, s'adaptant à l'utilisateur, qui lui-même va être guidé par l'application. Notre graphe de co-occurrence gère correctement les données de notre monde ouvert, et nos phéromones rendent compte des intérêts de l'utilisateur pour n'importe quel terme. Nous pouvons donc que la question scientifique à été correctement traitée, les 3points majeurs ayant été résolus.

7.2 Perspectives d'améliorations

7.2.1 Changements de routine

Notre prototype permet aujourd'hui de proposer des services adaptés à un utilisateur. Détecter différentes habitudes était un objectif optionnel que le temps nous a empêché de réaliser. Par différentes habitudes, on entend que l'application serait capable de détecter la différence entre par exemple une routine "semaine" et une routine "week end", ou encore entre "travail" et "vacances". Pour ceci un algorithme de détection de motif, capable de déceler des variations dans les actions de l'utilisateur aurait été nécessaire, et sortait du cadre de ce stage.

Cependant, une fois doté d'un tel algorithme, nous avons pensé qu'il suffirait de posséder plusieurs graphes de phéromones superposés. Un actif à la fois, et dédié à une routine précise, que l'algorithme de détection de pattern pourrait sélectionner. Ainsi, "congeler" un graphe de phéromone et en sortir un autre à chaque changement de routine devrait modifier naturellement le comportement de l'application et proposer des services adaptés aux différentes routines.

7.2.2 Divers

Tout d'abord, l'ergonomie de l'application n'a pas du tout été dans nos priorités, il reste donc une grande marge d'amélioration de ce côté. Placement d'éléments visuels, couleurs et d'autres paramètres ont été choisis initialement pour permettre de répondre à la question scientifique sans passer la majorité du stage à du développement d'IHM.

Ensuite, il reste un travail d'intégration par rapport aux données et aux attentes d'Arkéa. Twitter n'était qu'un cas d'exemple afin de substituer les données Arkéa, mais la finalité reste d'utiliser leurs données et services.

D'un point de vue algorithmique, la présence de notre graphe de co-occurrence de très grande taille apporte une grande possibilité d'optimisation des algorithmes de parcours utilisés pour l'instant. Toutefois notre maillage complet risque de poser problème à quelques heuristiques.

Juger la co-construction est une chose relativement difficile, il est certains que quelques expérimentations rigoureuses seraient bénéfiques au projet.

Pour finir, notre solution contient quelques paramètres, il est très probable que nos valeurs expérimentales ne soit pas les meilleurs pour l'exécution du modèle, des ajustements sont sûrement à faire.

Références

- [1] G. CALVARY et J. COUTAZ, "Plasticité des interfaces : une nécessité", *Information-interaction-intelligence, Actes des deuxièmes Assises nationales du GDR I*, t. 3, p. 247–261, 2002.
- [2] J. SIMONIN et N. CARBONELL, *Interfaces adaptatives adaptation dynamique à l'utilisateur courant*, 2007.
- [3] T. R. GRUBER, "Toward principles for the design of ontologies used for knowledge sharing?", *International journal of human-computer studies*, t. 43, n° 5, p. 907–928, 1993.
- [4] B. BONIS, J. STAMOS, S. VOSINAKIS, I. ANDREOU et T. PANAYIOTOPOULOS, "A platform for virtual museums with personalized content", *Multimedia tools and applications*, t. 42, n° 2, p. 139–159, 2009.
- [5] A. COLLAO, L. DIAZ-KOMMONEN, M. KAIPAINEN et J. PIETARILA, "Soft ontologies and similarity cluster tools to facilitate exploration and discovery of cultural heritage resources", in *Database and Expert Systems Applications, 2003. Proceedings. 14th International Workshop on*, IEEE, 2003, p. 75–79. DOI : <http://doi.ieeecomputersociety.org/10.1109/DEXA.2003.1232001>.
- [6] M. KAIPAINEN, P. NORMAK, K. NIGLAS, J. KIPPAR et M. LAANPERE, "Soft ontologies, spatial representations and multi-perspective explorability : soft ontologies, spatial representations and multi-perspective explorability", *Expert Systems*, t. 25, n° 5, p. 474–483, 21 oct. 2008, ISSN : 02664720. DOI : [10.1111/j.1468-0394.2008.00470.x](https://doi.org/10.1111/j.1468-0394.2008.00470.x).
- [7] L. RAJAONARIVO, M. COURGEON, E. MAISEL et P. DE LOOR, "Inline co-evolution between users and information presentation for data exploration", ACM Press, 2017, p. 215–219, ISBN : 978-1-4503-4348-0. DOI : [10.1145/3025171.3025226](https://doi.org/10.1145/3025171.3025226).
- [8] K. S. THOMAS MARTINETZ, "A "neural gas" network learns topologies", *University of Illinois at Urbana-Champaign*, n° 5, p. 397–402, 1991.
- [9] B. FRITZKE, "A growing neural gas network learns topologies", *Institut fur Neuroinformatik Ruhr-Universitat Bochum*, 1994.