

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université des Sciences et de la Technologie d'Oran « Mohamed Boudiaf »
Faculté des Sciences
Département d'Informatique

Spécialité : Informatique

Option : Reconnaissance des Formes - Intelligence Artificielle

MOUFOK Souad

Mémoire en vue d'obtention du diplôme de Magistère

Thème :

**Génération de graphes par les
fourmis artificielles : exploration
des traces de navigation internet**

Soutenance prévu le : 01/02/2012 devant le jury composé de :

M ^{me} BELBACHIR Hafida	Professeur	USTO-MB	Présidente
M ^r BENYETTOU Abdelkader	Professeur	USTO-MB	Rapporteur
M ^{me} ZAOUI Linda	Maître de conférences	USTO-MB	Examinatrice
M ^{me} MEKKI Rachida	Maître de conférences	USTO-MB	Examinatrice
M ^r BENDAHMANE Abderrahmane	Maître-assistant	USTO-MB	Invité

2011-2012

Remerciements

Je tiens particulièrement à exprimer mes profonds sentiments à Mr le professeur BENYETTOU Abdelkader, mon directeur, mon encadreur, pour sa disponibilité malgré ses occupations tant que vice recteur de la post-graduation USTO MB. Son écoute et ses conseils, m'ont été toujours précieux, sa confiance, son investissement scientifique ont été essentiels à la réalisation de ce travail.

Je voudrais exprimer tout mon respect à Mme le professeur BELBACHIR Hafida, qui m'a fait l'honneur de présider le jury.

Je veux remercier Mme Dr ZAOUÏ Linda et Mme Dr MEKKI Rachida, en leur qualité d'examinatrices, qui m'ont fait l'honneur aussi d'examiner minutieusement ce travail. A travers elles, je remercie tous mes enseignants.

Les discussions que j'ai eues avec Mr BENDAHMANE Abderrahmane tout au long de mon travail ont été particulièrement enrichissantes, je le remercie de m'avoir aidé par ses critiques constructives.

Bien que je ne puisse tous les citer ici, j'adresse un grand merci à tous les membres de l'équipe de notre laboratoire SIMPA pour leur gentillesse, les longues journées de travail et les multiples discussions qui m'ont aidées dans mes travaux.

Egalement à tous ceux qui ont pris de leur temps pour une lecture attentive et m'ont apporté leurs remarques et judicieux conseils.

Dédicace

Je dédie ce travail à,

Mes parents

Mes frères

Mes sœurs

...

Table des matières

Introduction générale	I
Chapitre I: Exploration de traces de navigation sur internet	1
I.1 Introduction	1
I.2 Présentation du problème	1
I.3 Trace	2
I.3.1 Approches centrées serveur	3
I.3.2 Approches centrées utilisateur	4
I.3.3 Approches basée sur des logiciels spécifiques	4
I.4 L'utilité des traces	5
I.5 Traces internes	5
I.5.1 Exemple	5
I.5.2 Formes physiques que prennent les traces internes	6
I.6 Traces externes	7
I.7 Bavardages techniques	7
I.8 Exploration et extension de données	8
I.9 Les approches de l'étude de navigation sur internet	9
I.10 Conclusion	11
Chapitre II: Le Web Mining et Prétraitement des traces de navigation sur internet	12
II.1 Introduction	12
II.2 Definition du Web Mining	12
II.3 Les principaux objectifs du Web Mining	13
II.4 Processus du Web Mining	13
II.5 Les donnees du Web et leurs sources	14
II.6 Quelques définitions	14
II.7 Axe de développement du Web Mining	15
II.8 Web Usage Mining	16
II.8.1 Présentation des fichiers logs	16
II.8.2 La difficulté du décryptage de l'User-Agent	18
II.8.3 Problèmes spécifiques aux données des fichiers logs	20
II.9 Prétraitement de fichier log	21
II.10 Nettoyage des données	22
II.10.1 Suppression des requêtes pour les ressources Web non-analysées (N1)	22
II.10.2 Suppression des requêtes et visites provenant des robots Web (N2a, b, c)	24
II.11 Transformation des données	25
II.11.1 Fusionnement des fichiers logs ensemble (T1)	25
II.11.2 Rendre anonymes les IP des utilisateurs (T2)	25
II.11.3 Identification de l'utilisateur (T3)	25
II.11.4 Identification des visites (T4)	27
II.12 Mesure de similarité entre sessions Web	30
II.13 Conclusion	31
Chapitre III: Classification et Clustering par les Fourmis Artificielles	32
III.1 Introduction	32
III.2 Taxonomie des methodes de classification ou clustering	32
III.3 Algorithmes de classification	33
III.3.1 Algorithmes de fourmis artificielles avec l'utilisation de phéromones	33

III.3.2 Algorithmes de fourmis artificielles sans l'utilisation de phéromones	34
III.3.3 Le rassemblement d'objets	35
III.4 Classification par colonies de fourmis	36
III.4.1 Algorithme AntClass	36
III.4.2 Algorithme AntClust	37
III.4.3 Algorithme AntTree	38
III.5 Domaines d'application	38
III.6 Principes d'auto-assemblage chez les insectes sociaux	39
III.7 Modèle d'auto-assemblage chez les fourmis réelles	41
III.8 Conclusion	43
Chapitre IV : L'algorithme AntTree_{Stoch} et son extension AntTree_{sans-seuil}	44
IV.1 Introduction	44
IV.2 L'algorithme AntTree _{Stoch}	44
IV.2.1 Description de l'algorithme	46
IV.2.2 Propriétés de l'algorithme	51
IV.2.4 Tri initial des données	52
IV.3 Algorithme AntTree _{Sans-Seuil} avec décrochage de fourmis	53
IV.4 Conclusion	55
Chapitre V : Résultats du prétraitement des traces de navigation sur internet	56
V.1 Etude de cas	56
V.2 Implémentation	57
V.2.1 Fusionnement des fichiers logs (T1)	57
V.2.2 Chargement du fichier log et transformation en une table d'une BDD	57
V.3 Nettoyage de données	58
V.4 Identification des utilisateurs (sessions) et des visites (T3)	60
V.5 Pages populaires et impopulaires	61
V.6 Information sur les internautes	62
V.7 Classe Candidat	62
V.7.1 Regroupement des visites de la classe Candidat	62
V.7.2 Liens de provenance	63
V.7.3 Navigateurs	64
V.7.4 Système d'Exploitation	64
V.8 Classe Recruteur	65
V.8.1 Regroupement des visites de la classe Recruteur	65
V.8.2 Liens de provenance	66
V.8.3 Navigateurs	66
V.8.4 Système d'exploitation	67
V.9 Classe Administrateur	67
V.9.1 Regroupement des visites de la classe Administrateur	67
V.9.2 Liens de provenance	68
V.9.3 Navigateurs	69
V.9.4 Système d'exploitation	69
V.10 Résultats finale	70
V.11 Conclusion	70
Conclusion générale	72
Perspectives	72
Références bibliographique	73

Liste des figures

Figure I.1 Le mode d'extraction d'information du serveur d'un site.	10
Figure II.1 Le schéma du processus du WUM.	13
Figure II.2 Taxonomie du Web Mining proposés par Cooley en 1997.	16
Figure II.3 Enregistrement des requêtes adressées au serveur gérant un site web dans un fichier log.	17
Figure II.4 Exemple d'un fichier log.	18
Figure II.5 Schéma illustratif des champs d'une requête.	18
Figure II.6 Requêtes aux images.	23
Figure II.7 Organigramme d'identification des visites d'utilisateurs.	28
Figure II.8 Identification des sessions et des visites.	30
Figure III.1 Hiérarchie des méthodes de classification.	33
Figure III.2 a) construction de chaînes et b) construction d'un nid par les fourmis <i>Oecophylla</i> , c) et d) construction de grappes de fourmis par les <i>Linepithema humile</i> .	39
Figure III.3 Accrochage chez les fourmis du genre <i>Eciton</i> .	40
Figure III.4 a) colonie de fourmis construisant un radeau, b) rassemblement d'abeilles japonaises <i>Apis cerana japonica</i> en forme de boule, c) exemple d'essaim d'abeilles.	41
Figure IV.1 Construction de l'arbre par des fourmis : principe général. Les fourmis qui sont en déplacement sont représentées en gris et les fourmis connectées en blanc.	45
Figure IV.2 Calcul du voisinage d'une fourmi. Les voisines d'une fourmi <i>fi</i> en déplacement sont hachurées.	46
Figure IV.3 Règles de comportement de <i>fi</i> si elle est positionnée sur le support.	49
Figure IV.4 Règles de comportement de <i>fi</i> si elle est positionnée sur une fourmi <i>fpos</i> .	50
Figure IV.5 Effet de boucle dans $AntTree_{Stoch}$.	51
Figure IV.6 Transformation de l'arbre en classes (partitionnement), ici deux classes sont générées <i>C1</i> et <i>C2</i> .	52
Figure IV.7 Décrochage de fourmis dans $AntTree_{Sans-Seuil}$.	53
Figure V.1 Aperçu de la page d'accueil du site Web www.taseek.com .	56
Figure V.2 Fichier log à l'état brut.	58
Figure V.3 Base de données après import.	58
Figure V.4 Aperçu des différents contenus des pages de notre site.	59
Figure V.5 Table d'identification des sessions.	60
Figure V.6 Session d'un utilisateur.	61
Figure V.7 Nombre de visites des pages url.	61
Figure V.8 nombre de visiteur du site.	62
Figure V.9 Regroupement des visites de la classe candidat.	63
Figure V.10 Liens de provenance de la classe Candidat.	63
Figure V.11 Pourcentages des types de Navigateurs de la classe Candidat.	64
Figure V.12 Pourcentage des Systèmes d'Exploitation de la classe Candidat.	65
Figure V.13 Regroupement des visites de la classe recruteur.	65
Figure V.14 Liens de provenance de la classe Recruteur.	66
Figure V.15 Pourcentages des types de Navigateurs de la classe Recruteur.	66
Figure V.16 Pourcentage des Systèmes d'Exploitation de la classe Recruteur.	67
Figure V.17 Regroupement des visites de la classe Administrateur.	68
Figure V.18 Liens de provenance de la classe Administrateur.	68
Figure V.19 Pourcentages des types de Navigateurs de la classe Administrateur.	69
Figure V.20 Pourcentage des Systèmes d'Exploitation de la classe Administrateur.	70

Liste des Tables

Tableau II.1 Décryptage de l'User-Agent.	19
Tableau II.2 Identification du système d'exploitation.	20
Tableau II.3 Les étapes du prétraitement dans le WUM.	21
Tableau V.1 La taille des fichiers log a analysés.	57
Tableau V.2 Résultat de suppression des requêtes inutile (N1).	59

Liste des Algorithmes

Algorithme II.1 Algorithme d'identification des utilisateurs.	27
Algorithme II.2 Algorithme d'identification des visites des utilisateurs.	29
Algorithme III.1 Algorithme AntClass.	37
Algorithme IV.1 Algorithme principal de construction d'arbres par des fourmis artificielles.	47
Algorithme IV.2 Cas support : algorithme simulant le comportement d'une fourmi fi en déplacement sur le support f0.	48
Algorithme IV.3 Cas fourmis : algorithme simulant le comportement d'une fourmi fi en déplacement sur une autre fourmi fpos connectée à la structure.	49
Algorithme IV.4 AntTree(Sans-Seuil) (cas support et fourmis sont confondus)	54

Introduction générale

La forte croissance d'Internet ces dernières années, en terme de nombre d'utilisateurs visitant les sites web publiés, a renforcé la nécessité de posséder des outils capables d'extraire et de donner du sens aux navigations des internautes. Toute navigation sur internet laisse des traces, elle permet par exemple, aux responsables des sites de connaître l'adresse IP de l'ordinateur avec lequel nous naviguons.

Généralement, les traces de navigation des internautes sur un site Internet sont extraites d'un fichier log, présent sur le serveur Web, qui recense, pour simplifier et de manière idéale, l'ensemble des demandes de pages du site de la part des internautes. Ces requêtes clientes sont ensuite fusionnées, triées, nettoyées, prétraitées et regroupées en sessions qui constituent pour chaque internaute, l'ensemble des informations issues de leur navigation sur un site à un moment donné, donc chaque session représente l'ensemble des pages visitées par un seul utilisateur.

Pour améliorer le site, il est alors important d'analyser les comportements de ses utilisateurs. Une méthode d'analyse dirigée par l'usage consiste à réaliser une classification du contenu du site à partir des navigations enregistrées dans les logs du serveur. Les classes ainsi obtenues sont constituées de pages qui ont tendance à être visitées ensembles. Elles traduisent donc les préférences des utilisateurs.

Nous présentons dans ce travail une représentation des sessions de navigation des internautes reposant sur l'utilisation de traces de navigation sur internet, issues du fichier log, pour trouver la séquence des pages visitées, les dates d'accès aux pages, la durée totale de la session, l'adresse IP et l'identifiant de l'internaute, ... Cette représentation de sessions a été utilisée avec l'algorithme de classification non-supervisée AntTree(Sans Seuil) qui travaille sur le principe d'auto-assemblage des fourmis réelles. L'avantage de cet algorithme est qu'il n'utilise aucun paramètre ou seuil, en dehors du tri initial des données. Alors il s'agit de modéliser la manière dont les fourmis forment des structures vivantes et d'utiliser ce comportement pour organiser les données selon un arbre qui se construit de manière distribuée.

Cette, méthode est appliquée sur les fichiers log du site Internet "www.taseek.com" pour essayer d'en extraire des comportements de navigation des utilisateurs, les pages populaires et les pages impopulaires, les pays qui visitent le plus le site web,...

Chapitre I :

Exploration de traces de navigation sur internet

I.1 Introduction

Toute navigation sur internet laisse des traces, elle permet par exemple, aux responsables des sites de connaître l'adresse IP de l'ordinateur avec lequel nous naviguons. Ces traces sont comme nos empreintes de doigts que nous laissons volontairement ou involontairement lors de tout acte de notre vie. De même, lorsque nous marchons, nous laissons des empreintes de pas qui permettent de nous suivre à la trace. Parfois, nous nous passerions bien. Ainsi n'importe qui peut suivre notre parcours, connaître la pointure et la marque de nos chaussures.

Sur internet, c'est pareil. Une simple connexion permet de donner des informations que l'on ne soupçonne pas. Donc lorsqu'on visite une page web, nous laissons des traces qu'on appelle traces de navigation sur internet, donc nous donnons inconsciemment notre adresse IP, notre système d'exploitation (user agent), la page qui nous amène à visiter celle-ci (referré), des informations sur le site à collecter sur nous lors de notre dernière visite (cookies). Ainsi, quoi qu'on fasse sur le web, nos profils caractéristiques et historiques de navigation sont autant d'empreintes de pas que certains peuvent les identifier, suivre, stocker. Ces traces de navigation sont enregistrées dans des fichiers spéciaux appelés fichiers Logs, contenant une trace des requêtes reçues et des opérations effectuées par le serveur.

I.2 Présentation du problème

Le problème qui nous intéresse peut être vu comme la classification des usagers d'un site web en plusieurs catégories. En d'autres termes, nous cherchons à identifier et à qualifier des groupes d'utilisateurs par rapport à leurs motifs de navigation sur un site donné ou des traits représentant des centres d'intérêts. Ce problème de classification a été traité dans de nombreux travaux en appliquant différentes méthodes. Ainsi suivant le profil, le site devra mettre en avant les points intéressants pour la visite de cette catégorie de personnes.

Les itérées sont multiples :

- Améliorer la navigation et donc la satisfaction des utilisateurs ;

- Adapter le site a leurs besoins de manière dynamique ;

Notre propos dans cette étude consiste à transformer les données présentes dans les fichiers Logs d'un site Web en des connaissances utiles en procédant à un prétraitement de ces fichiers et à une classification non supervisée des visites effectuées par les internautes, basée sur des algorithmes de Fourmis Artificielles.

Pour ce faire, il nous faut récupérer les données concernant les utilisateurs. Les données sont stockées soit sur le serveur, soit du coté client. Sur le serveur, les données sont brut sans appartenance, il nous faudra les trier, les formater pour qu'elles soient utilisable.

I.3 Trace

D'après N.Bousbiha citée dans [1], La définition de la notion de trace diffère selon son rôle et son utilisation dans chaque domaine de recherche.

Dans [2], Cram et al. définissent une « trace d'interactions » comme « *tout objet informatique dans lequel s'accumulent des données à propos des interactions entre un système informatique et son utilisateur* ».

Dans [3], Champin et al. (définissent la trace comme étant une séquence d'états et de transitions représentant l'activité de l'utilisateur : « *la séquence temporelle des objets et opérations mobilisés par l'utilisateur lorsqu'il utilise le système est appelée trace d'utilisation* ».

Dans [4], Jermann et al. définissent la trace d'apprentissage comme « *une observation ou un enregistrement de l'interaction de l'apprenant avec un système en vue d'une analyse* ».

Dans [5], pour Pernin, la trace représente « *un indice de l'activité des acteurs d'une situation d'apprentissage qu'elle soit ou non instrumentée* ». Il précise aussi qu'il s'agit « *d'un résultat obtenu au cours ou au terme d'une activité, d'un événement ou d'un ensemble d'évènements relatif au déroulement de la situation d'apprentissage* ».

Une définition plus générale, est celle donnée dans [6]: « *la trace est définie comme une séquence temporelle d'observés* ». Le terme « *séquence temporelle* » dénote l'existence d'une relation d'ordre organisant les données de la trace par rapport à un repère de temps. Le terme « *observé* » dénote que les données de la trace sont issues d'une observation.

Dans notre travail, nous considérons cette même définition de la trace car elle nous semble la plus complète. Cependant, nous nous intéressons uniquement aux traces numériques.

Notons enfin que l'absence d'une définition commune et explicite de cette notion de trace, amène souvent à des confusions entre les traces et les informations issues des traces. Par conséquent, suite à la définition considérée, nous considérons dans notre travail, qu'une trace est une trace d'activité, d'utilisation, d'interaction. Il ne lui est pas associé d'interprétation sur la situation d'apprentissage. On parle alors de traces primaires, brutes, de base ou de « bas niveau » comme certains travaux les désignent.

Alors une trace peut être présentée sous différents mots comme (« log machine », « log des connexions », « éléments de preuve »...). Chacun de ces mots est considéré comme étant simplement le constat d'une action qui permet de déterminer la suite des événements produite par un utilisateur d'un site web. Ces traces sont créés et stockés comme l'écriture des logs sur un serveur, de manière distribuée.

On dispose alors de la liste horodatée des adresses des pages visitées par chaque internaute, qui constitue le matériau premier de l'étude.

Cette position d'observation particulière ainsi que la durée et le volume inédits d'observation nous permettent d'observer finement et fidèlement les pratiques des internautes à domicile et d'en suivre l'évolution.

La collecte des traces consiste à collecter l'ensemble de toutes les traces enregistrables. Nous pouvons identifier trois modes :

- *Une collecte manuelle* : procédée par un observateur humain, acteur ou non dans la situation d'internaute, ou par des questionnaires remis aux acteurs.
- *Collecte audiovisuelle* : exécutée par un outil d'enregistrement visuel et audio de la situation d'internaute.
- *Collecte numérique* : menée en utilisant un environnement informatique sauvegardant l'activité de l'internaute. Les résultats de la collecte numérique constituent une trace numérique.

Environ 78 % des environnements de formation ouverte et à distance intègrent un outil de capture des traces numériques [7]. Ces systèmes utilisent différentes approches de collecte selon les sources d'observation : le serveur, le poste client, ou des mécanismes d'observation spécifiques au système d'enregistrement.

I.3. 1 Approches centrées serveur

L'approche centrée serveur s'intéresse à la recherche des motifs de navigation des utilisateurs sur un site donné en se basant sur l'analyse des *logs* des serveurs Web. Ces *logs* contiennent l'ensemble des actions effectuées sur le serveur. La création des traces à partir de ces *logs* est un processus complexe qui nécessite de nombreuses opérations (filtrage, recomposition en sessions, etc.). De nombreuses informations peuvent ainsi être déterminées, telles que le navigateur et le système d'exploitation utilisés, le moment auquel une certaine activité a été effectuée, la durée passée sur une certaine page, le nombre de fois qu'une page spécifique a été visitée, ou encore l'adresse IP correspondant à chacun des événements.

Cette approche est généralement utilisée dans les sites à vocation commerciale, souhaitant disposer de données précises sur leur fréquentation, afin de savoir quelles pages sont les plus visitées, comment les utilisateurs y arrivent, et comment les garder plus longtemps sur le site.

Par exemple, dans le contexte de notre formation en ligne, si la plate-forme d'apprentissage est hébergée sur un serveur apache, les données présentes dans le fichier de *log*, peuvent être utilisées pour :

- évaluer l'efficacité d'un cursus en ligne ;
- quantifier les interactions entre les utilisateurs et les pages du cours ou de l'environnement de formation, etc.

Parmi les travaux qui emploient cette démarche, nous pouvons citer ceux présentés dans :

Dans [8], Iksal et al., s'appuient sur les fichiers de *log* pour l'acquisition des traces. Dans le même ordre d'idée, dans [9], Stermsek et al., cherchent à exploiter les fichiers de *log* des apprenants et les métadonnées des pages HTML pour déduire leurs profils et ainsi proposer des mécanismes de filtrage d'informations.

Ces approches sont intéressantes. Cependant, elles tiennent pour acquis que le contenu des pages est connu. En effet, les informations enregistrées dans les fichiers *log*, côté serveur, ne renseignent que sur les activités liées au serveur en question. Les données produites par les applications s'exécutant sur le poste client, ou celles issues d'applications provenant du contenu dynamique, ne sont pas collectées, empêchant ainsi de connaître les informations réellement consultées par les utilisateurs [10]. De plus, ces fichiers ne sont généralement pas disponibles à n'importe quel utilisateur (juste les administrateurs réseau et système).

I.3.2 Approches centrées utilisateur

Si, pendant un exercice, l'internaute effectue des recherches sur le Web, cette interaction n'est pas observée sur le serveur. Pourtant cette interaction peut être un élément important d'explication du parcours de l'internaute. Il est donc intéressant d'instrumenter le poste client afin d'observer toutes les interactions propres à l'internaute. Cette démarche n'est pas largement utilisée. En effet, si le recueil de données de trafic centrées-serveur est maintenant relativement standardisé, la collecte d'informations au niveau des postes utilisateurs est encore un domaine d'activité, du fait des différents choix technologiques possibles et du degré de finesse des informations que l'on souhaite recueillir [11]. Parmi les travaux adoptant cette approche, nous pouvons citer :

- *WebIC* [12], qui est un système de recommandation de liens, focalisé sur la recherche d'information. Son objectif est de proposer à l'utilisateur les liens qui semblent le mieux convenir à ses attentes, en fonction des recherches qu'il a effectuées dans des moteurs de recherche et des liens qu'il suit. Le système - installé sur le poste client - repère les mots importants au fur et à mesure de la navigation et propose des pages qui ont un contenu susceptible de contenir une information correspondant aux mots récoltés lors du parcours.
- Dans [13], pour Blanchard et al., le système trace les actions des utilisateurs, en enregistrant les pages visitées par ces derniers et le mode de navigation. De plus, le système mesure la diversité thématique des dernières pages visitées. Cette dernière caractéristique est calculée à partir d'une distance entre textes.
- Dans les travaux de [11], les données utilisées proviennent de sondes de recueil de trafic Internet, installées sur les postes des utilisateurs à domicile. La liste des URLs visitées par chaque internaute ainsi obtenue, constitue le matériau premier de l'étude. Sur cette base, Beauvisage propose une description des parcours des internautes de page en page et de site en site centrée sur la session.
- Dans [14], Loghin propose un agent *keylogger* capable d'enregistrer tous les événements clavier et souris produits par l'internaute, les processus qui sont exécutés ainsi que les titres et le contenu des boîtes de dialogue affichées sur l'écran. Néanmoins, très peu de travaux implémentent cette approche. Généralement, ils tracent uniquement les actions des internautes dans l'environnement de navigation en utilisant les *logs* serveur ou des outils spécifiques intégrés à l'environnement tracé.

I.3.3 Approches basées sur des logiciels spécifiques

Au-delà des approches qui distinguent la source d'observation, selon que cela soit fait sur le poste client ou sur le serveur, d'autres approches se focalisent sur l'identification de l'interaction au moment de la collecte à travers un outil spécifique à l'environnement tracé.

Parmi ces travaux, nous pouvons citer eMédiathèque, un outil de travail collaboratif où toutes les interactions de l'utilisateur sont observées et traitées à partir de deux modèles [15]:

- l'un définit les objets observables (les outils mis à disposition tels que le navigateur Internet, la messagerie instantanée, ou les ressources telles que les textes, les images, etc.) ;
- l'autre spécifie les actions réalisables par l'utilisateur (création, modification, suppression de contenus, etc.). Les traces sont ensuite créées en fonction des activités des utilisateurs, stockées dans un composant interne de l'application, et affichées en temps réel à l'utilisateur.

I.4 L'utilité des traces

Les traces de navigation sur internet sont utilisées pour :

- Analyser les statistiques d'un site, les fichiers qui n'ont pu être chargés, les fichiers qui intéressent les visiteurs et ceux qui ne les intéressent pas, ainsi que le référencement du site dans les moteurs de recherche et annuaires.
- Détecter des anomalies de fonctionnement d'un système.
- Analyser et/ou améliorer l'utilisation du système.
- Détecter une tentative d'usage prohibé du système, de preuve de bon fonctionnement du système, de preuve d'imputabilité (attribuer à un acteur la réalisation d'une action sur une information) [16].

Donc ces traces nous servent à améliorer un site d'une part, et comprendre son trafic d'autre part. Comme elles permettent également d'établir des profils d'utilisateurs ainsi que de connaître leurs centres d'intérêt.

Toute activité sur internet laisse des traces dont n'importe qui localement ou à distance peut prendre connaissance et permettent de savoir exactement qui a fait quoi à quel moment.

On peut distinguer trois types de traces :

I.5 Traces internes

Ce sont celles laissées par nos activités sur nos ordinateurs dans des fichiers de logs (des journaux d'activité), dans la base de registre, dans des fichiers temporaires, dans des antémémoire (caches), ..., quoi que nous fassions, que nous soyons connectés ou non.

I.5.1 Exemples

Windows, notre explorateur de Windows, notre navigateur Internet (quelle que soit leur version) et chacun des logiciels que nous utilisons, aussi infime et anodin soit-il, conservent des traces de notre activité.

Les traces les plus connues sont:

- Les "journaux" (fichiers dits "logs" annotant notre activité et les événements

survenus).

- Toutes les listes de saisie dites "rapides" (autocomplete - remplissage automatique) qui se "souviennent" de nos frappes au clavier.
- Toutes les listes des derniers fichiers ouverts dans chaque application (fichiers récemment manipulés).
- Les historiques de notre activité comme, par exemple, toutes les dernières adresses des pages web consultées (historique de notre navigation conservée jusqu'à 999 jours soit presque 3 ans).
- Les cookies et leurs contenus.
- Les inscriptions dans la base de registre de Windows.
- Tous les fichiers contenant nos divers mots de passe.
- Tous les fichiers et répertoires des "caches" de tous les navigateurs.
- Tous les cookies sous tous les navigateurs.
- La base de registre contient plusieurs traces (de nos historiques de navigation etc.).
- etc.

I.5.2 Formes physiques que prennent les traces internes

Les formes que peuvent prendre les traces internes sont comme suit:

- Fichiers journaux (les "logs") dont on peut plus ou moins maîtriser la taille (donc limiter la quantité de données contenues). Ces journaux peuvent être dédiés aux traces d'activité des utilisateurs ou aux traces d'activités techniques dédiées aux administrateurs.
- Fichiers temporaires, très nombreux. Mémorisés par les applications dans des sous-répertoires du compte utilisateur (comptes justement appelés, par une ironie du sort, "profile").
- Fichier de mémoire virtuelle de Windows (le swappfile) comportant la trace de tous les programmes et de tous les documents ouverts, raison pour laquelle nous recommandons avec insistance de ne jamais utiliser les "Error Reporting Tools" (Rapports d'erreurs) qui envoient aux éditeurs de logiciels une copie de cette mémoire virtuelle.
- Fichier ou répertoires de cookies (selon le navigateur utilisé).
- Les fichiers index.dat qui, malgré l'effacement des traces, continuent à inventorier l'intégralité des traces effacées avec des informations complémentaires, en plus !
- Les historiques de chaque application, destinés à nous faciliter la vie : la liste des derniers documents manipulés par chaque application : les MRUs (derniers utilisés - Most Recently Used). Ce sont, physiquement, d'interminables listes de clés dans la base de registre ou des séries de liens dans des répertoires spécialisés. Chaque application propose, dans ses options, la possibilité de rappeler automatiquement un nombre, généralement paramétrable entre 0 et n, de noms de documents dernièrement utilisés.
- Une trace souvent oubliée : le presse-papiers (clipboard) qu'il faut vider.

- Une autre trace souvent oubliée : la corbeille. Rien ne sert de mettre des documents de traces à la poubelle si celle-ci n'est pas vidée et que ce qui s'y trouve peut en être retiré, intact.
- Une autre trace, plus subtile et qui ne peut être révélée qu'avec des moyens hors de portée du premier venu : la rémanence magnétique des disques qui, même après effacement d'un fichier, nettoyage de la corbeille et défragmentation, permet encore à des outils très spécialisés de reconstituer les données. Seuls des effaceurs dit "de sécurité" peuvent rendre définitivement irrécupérable des données sensibles.

I.6 Traces externes

Ce sont celles laissées entre autres chez les FAI (Fournisseurs d'Accès Internet), dans d'autres logs, durant une navigation sur internet, les émissions et réceptions de courriers, les séances de chat etc. Contre ces dernières, on se tournera vers l'anonymisation de la connexion, la cryptographie etc.

C'est, par exemple, la fameuse trace appelée "Referrer" (référant). Pour un Webmaster le référant est le site qui envoie un internaute sur son site. En gros, c'est "Qui pointe sur moi ?". Pour les marketeurs et autres organismes traquant les internautes (dont toutes les sociétés de statistiques pour Webmasters), la collection de nos référants constitue notre histoire, notre vie sur Internet. Nos millions d'URL visitées sont conservées afin d'établir notre profil comme un psychologue ou un policier le ferait : "Dis moi qui tu fréquentes et je te dirais qui tu es".

I.7 Bavardages techniques

Ce sont les informations techniques que les navigateurs donnent à tous les serveurs sur lesquels on se branche et que n'importe quel webmaster peut récupérer. Généralement, les webmasters ne récupèrent pas eux-mêmes ces données mais ils truffent leurs sites de tags de statistiques sous traitées à des sociétés spécialisées qui s'en délectent au point d'offrir ces statistiques, gratuitement, aux Webmasters, afin d'espionner tous leurs visiteurs. Ces tags sont généralement invisibles (ce sont des web-bug) et dotés de scripts. Quelque fois ils sont visibles. C'est ce type de traces qu'on va utiliser lors de notre application, se présentant sous formes de fichiers logs d'un site précis.

Il y a des traces que nous ne pouvons effacer facilement et, pour certaines, que nous ne pouvons pas effacer du tout. Sachant qu'il est facile de collecter certaines informations sur les internautes qui visitent des sites, informations dont nous ignorons même l'existence. Ces informations ne proviennent pas du pillage d'informations secrètes. Elles sont envoyées volontairement par notre navigateur (par notre Internet Explorer, Netscape, Opra, Mozilla etc.) au serveur du site que nous visitons. On les appelle les "variables d'environnement".

Par exemple, savoir quel navigateur nous utilisons, l'adresse IP, le nom de l'utilisateur, les dates de début et de fin de connexion, ainsi que la page d'où nous venons avant d'être arrivé, permettent de suivre notre navigation et de nous profiler [46].

I.8 Exploration et extension des données

Connue sous les noms fouille de données, Data Mining (forage de données) ou encore extraction de Connaissances à partir de Données, a pour objet l'extraction d'un savoir ou d'une connaissance ou la découverte d'informations intéressantes et utile à partir de grandes quantités de données, par des méthodes automatiques ou semi-automatiques.

Les données peuvent être de plusieurs types, comme des données comportementales (gratuit et disponible), données produits, données démographique etc.

Notre travaille sera réalisé sur des données comportementales qui sont les traces de navigation sur internet.

Pour pouvoir apprendre un profil utilisateur pour un site défini, il nous faut un maximum d'informations sur le mode de navigation de l'utilisateur sur ce site. La tâche principale de l'extraction et du prétraitement est donc de rassembler toutes les données de cet individu dans le laps de temps de sa visite : c'est ce que nous appellerons une session utilisateur.

Le champ de données sur lesquelles nous pouvons travailler est assez vaste. La première étape est donc d'explorer les plus exhaustivement possible les types de données disponibles. La seule catégorie de données que nous avons décidée de traiter est les traces laissées par les utilisateurs de manière passive lors d'une navigation sur internet. Les formulaires qui demandent de répondre à telle ou telle questions concernant la navigation sont exclus pour des raisons de facilité : le processus se veut universel alors que l'utilisation des formulaires demande du développement pour chaque site.

Le type de données le plus répondu est évidemment les logs des pages visitées et stockées sur le serveur http. Mais d'autres types de traces sont accessibles :

- La position, les clics de la souris et toutes les informations traitées uniquement sur le navigateur web de l'utilisateur.
- La technologie de l'occulométrie nous permet pour un site précis de savoir en plus où l'utilisateur regarde pendant sa navigation.

Les traces peuvent être récupérées par plusieurs méthodes citées ci dessous :

- **Les informations du navigateur**

Du côté client, une quantité non négligeable d'information sur l'utilisateur circule. Elle se compose des interactions directes de l'utilisateur avec sa machine : ce sont les entrées du clavier, les mouvements et clics de souris.

Inconvénients:

- Récupération du côté client difficile pour des raisons de compatibilité de navigateurs.
- Augmentation du trafic réseau.

- **Les données oculométriques**

La seule méthode de récupération de telles données est la mise en place d'un protocole de navigation par des personnes volontaires avec des contraintes physiques souvent pesantes : en effet, les usagers doivent rester quelques vingt minutes la tête bloquée

par le menton et les tempes. Après les testes effectués, les résultats consistent en des nuages de points représentant les parties regardées par le sujet.

Inconvénients:

- Temps de récupération gigantesque, coût humain important.
- Protocole d'expérience lourd.
- **Les logs des serveurs http**

Ces données sont composées d'une suite de requêtes http sur un serveur. Quand l'utilisateur demande une page à un serveur et que le serveur lui renvoi cette page, le serveur ajoute dans le fichier de log la requête qu'il vient d'exécuter. Ce processus est appelée hit. Cette première information a été et encore utilisée dans le domaine du web, en effet le nombre de hits rencontrés sur un site donne une approche de ce qu'est l'utilisation de celui-ci. Mais cette approche est loin d'être satisfaisante. En effet une page web peut être composée d'objets externes : l'exemple le plus simple est une page contenant une image. Dans ce cas, l'image ne se trouve pas réellement dans la page mais le navigateur de l'utilisateur va demander après avoir lu la page « contenant » l'image, le fichier de l'image : ce qui nous fait deux hits pour une page demandée. En suivant ce principe, la comparaison de deux sites suivant le nombre de hits enregistrés dépend très fortement de la structure même des pages des sites en question. La première action consiste donc à ne prendre en compte dans ces fichiers de logs que seulement les entrées correspondantes à des pages HTML (ou PHP, ou etc).

Avantages:

- Coûts de récupération physique quasi nul.
- Homogénéité des formats de logs grâce aux recommandations du web.

Inconvénient:

- Manques des données à cause des proxy-caches.

I.9 Les approches de l'étude de navigation sur internet

Quand un utilisateur navigue dans un site Web, il génère plusieurs transactions par les requêtes qu'il formule durant sa navigation. Il peut accéder aux différentes pages par des hyper - liens et via la recherche par mots clés. Ses traces sont enregistrées dans des fichiers générés par le serveur du site. Un fichier contient en général, des enregistrements des transactions des pages demandées (ex : index.asp, home.htm), les images qui vont avec (ex : bouton.gif, backgroug.jpg) et pleins d'autres fichiers. Presque tous les serveurs enregistrent les fichiers de transactions sous format texte et incorporent au minimum deux types de fichiers : un fichier qui informe sur ce que voit le visiteur du site et un fichier erreur qui donne des informations sur une éventuelle déconnexion. Le premier type de fichier contient toutes les transactions des visiteurs du site, et le cheminement de ces transactions représente l'aspect comportemental des données [17].

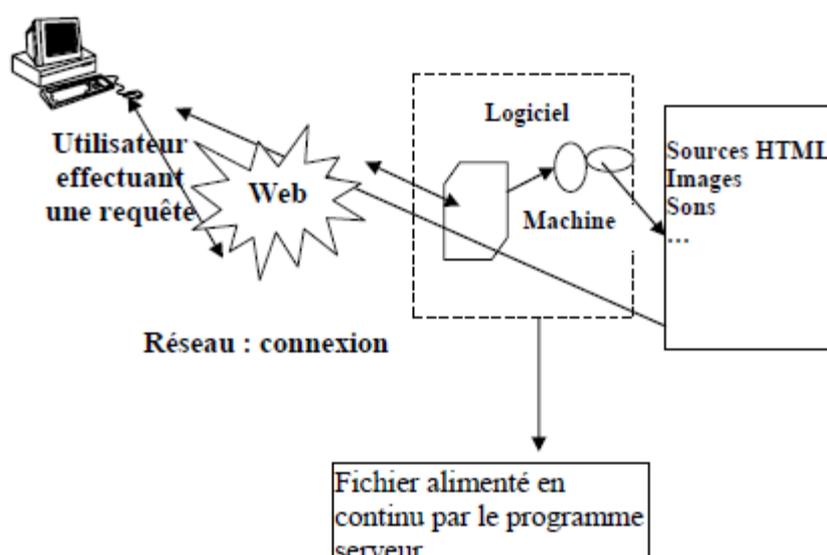


Figure I.1 Le mode d'extraction d'information du serveur d'un site.

L'étude de comportement de navigation des internautes sur des différents sites a été étudiée selon plusieurs chercheurs de manières différentes, on peut citer les approches utilisées dans ce domaine comme suite :

Dans [18], W.Gao, O. Sheng identifient les utilisateurs d'un site Web à visée commerciale à l'aide de réseaux Bayésiens, afin de retenir au maximum les clients éventuels.

Dans [19] et [20], s'appuient sur des cartes topologiques de Kohonen. K.Benabdeslem et Y. Bennani proposent ces cartes de Kohonen afin de construire une cartographie d'un site Web tel qu'il est perçu par les utilisateurs et de projeter leur navigation sous la forme de trajectoires représentant leur comportement. M.Charrad, Y. Lechevallier [20] caractérisent les utilisateurs d'un site Internet en se basant sur leurs motifs de navigation. La découverte de motifs de navigation est effectuée en combinant une analyse des correspondances multiples et des cartes de Kohonen.

Dans [21], A.Büchner, M.Baumgarten, S.Anand, M.Mulverna et J.Hughes : présentent l'algorithme MiDAS pour la découverte de séquences d'actions dans des fichiers log. Les mécanismes proposés sont appliqués là aussi à un ensemble de sites de commerce en ligne, dans le but de proposer des promotions et des activités personnalisés.

Dans [22], R.Iváncsy et I. Vajk utilisent trois algorithmes différents fondés sur des ensembles de ressources fréquemment consultées, des séquences et des motifs sous forme d'arbres. Il préconise l'utilisation de chacun de ces algorithmes selon le but de l'application (publicités personnalisées, création dynamique de profil utilisateur, etc.).

Dans [23], Q.Yang Q., H.Zhang, T.Li applique des méthodes de Data Mining à des fichiers log de navigation sur un même serveur afin d'en extraire un modèle de prédiction des requêtes futures d'un utilisateur. Le but est de stocker en mémoire cache les ressources visées pour réduire le temps de latence entre la requête et l'envoi de la ressource. Le modèle s'appuie sur l'identification de séquences de requêtes fréquemment utilisées et l'algorithme

de prédiction utilise un ensemble de règles d'association. Ces travaux ont donné lieu à la mise en place d'un système de stockage de données appelé Netshark, [24].

I.10 Conclusion

Au cours de ce chapitre, nous avons présenté les traces de navigation sur internet qu'on peut exploiter à des fins multiples. Le chapitre suivant aborde quelques points touchant le domaine de la fouille de données d'usage du web (WUM), ainsi que les différentes phases de son processus. Comme nous allons donner les différentes heuristiques de prétraitement qui décrit le premier pas du processus de la fouille de données d'usage du web. Les travaux de recherche récents menés a ce sens dénotent l'importance sans cesse accrue de la portée d'un tel thème de recherche.

Chapitre II :

Le Web Mining et le Prétraitement des traces de navigation sur internet

II.1 Introduction

Les sites Web représentent actuellement une véritable source de production de grands volumes d'informations. La recherche d'information, la consultation de données et l'achat en ligne, sont des exemples de l'utilisation du Web. Dans le but d'améliorer ces services et d'autres nous nous intéressons aux informations liées au comportement des utilisateurs de cet environnement. C'est dans cet espace que nous pouvons récupérer une quantité d'informations pertinentes.

La caractérisation des internautes fréquentant un site Web et l'identification de leurs motifs de navigation est un problème incontournable pour les concepteurs de sites Web qui visent à assister l'internaute, prédire son comportement et personnaliser la consultation. Ces trois considérations ont motivé d'importants efforts dans l'analyse des traces de navigation sur internet pour les sites Web et l'adaptation des méthodes de classification aux données du Web durant ces dernières années.

La fouille de données (ou *Data Mining*) est un domaine de recherche encore jeune, qui fait suite au désir des propriétaires de grandes masses de données de mieux les connaître.

Compte tenu de la croissance exponentielle du web, les applications de la fouille de données aux fichiers *logs* (qui enregistrent les actions des utilisateurs sur un site web) sont rapidement apparues d'où la taille gigantesque que peuvent atteindre parfois ces fichiers logs.

Ce présent chapitre est structuré en deux sections, nous présentons dans la première partie la définition du Web Mining, en particulier ses objectifs et les axes de son développement. Nous nous intéressons au troisième axe de développement du Web Mining, le Web Usage Mining, en particulier les motifs du WUM, les données de l'usage. Dans la seconde, nous décrivons les étapes de prétraitement de fichiers logs.

II.2 Définition du Web Mining

Le Web Mining, défini comme l'application des techniques du Data Mining aux données du Web (documents, structure des pages, des liens...), s'est développé à la fin des

années 1990 afin d'extraire des informations pertinentes sur l'activité des internautes sur le Web. Le Web Mining est utilisé pour comprendre le comportement des clients, d'évaluer l'efficacité d'un site Web particulier, et d'aider à quantifier le succès d'une campagne de marketing, etc. L'exploration du Web peut être divisée en trois types différents, qui sont : *Web Content Mining*, *Web Structure Mining* et *Web Usage Mining*.

II.3 Les principaux objectifs du Web Mining

Le Web Mining poursuit deux principaux objectifs:

1. L'amélioration et la valorisation des sites Web : L'analyse et la compréhension du comportement des internautes sur les sites Web permettent de valoriser le contenu des sites en améliorant l'organisation et les performances des sites.
2. La personnalisation: Les techniques de Data Mining appliquées aux données collectées sur le Web permettent d'extraire des informations intéressantes relatives à l'utilisation du site par les internautes. L'analyse de ces informations permet de personnaliser le contenu proposé aux internautes en tenant compte de leurs préférences et de leur profil.

II.4 Processus du Web Mining

Le processus du Web Mining se déroule en trois étapes :

1. Collecte des données sur l'utilisateur,
2. Utilisation de ces données à des fins de personnalisation,
3. Présentation à l'utilisateur d'un contenu ciblé.

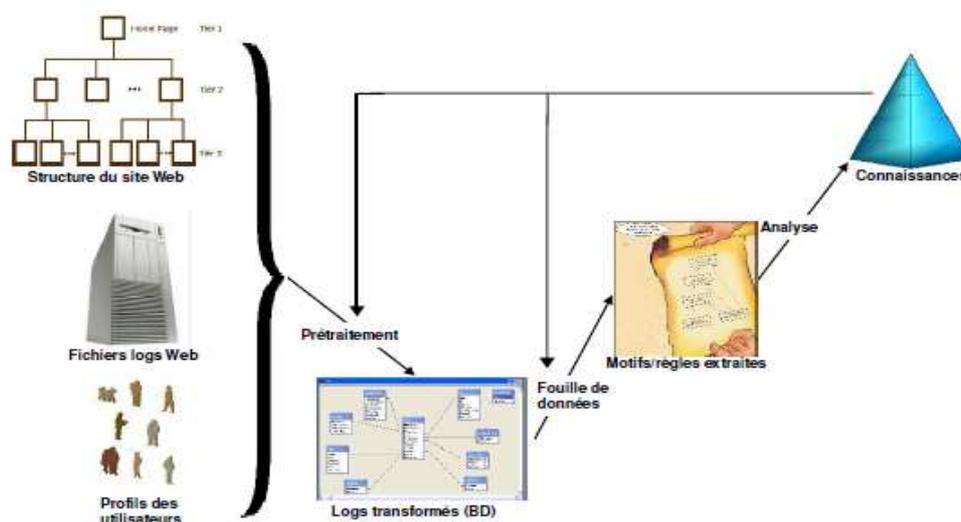


Figure II.1 Le schéma du processus du Web Usage Mining.

II.5 Les données du Web et leurs sources

Les données utilisées dans le Web Mining sont classifiées en quatre types :

- **Données relatives au contenu** : données contenues dans les pages Web (textes, graphiques).
- **Données relatives à la structure** : données décrivant l'organisation du contenu (structure de la page, structure inter-page).
- **Données relatives à l'usage**: données fournissant des informations sur l'usage telles que les adresses IP, la date et le temps des requêtes.
- **Données relatives au profil de l'utilisateur** : données fournissant des informations démographiques sur les utilisateurs du site Web.

Ces données sont généralement stockées dans un Data-Warehouse, appelé data-Webhouse, utilisé pour collecter et stocker des données propres à la fréquentation des sites Web afin d'analyser les comportements de navigation.

Les principales sources des données permettant d'alimenter les Data-Webhouses sont :

- **Les fichiers Logs du serveur Web**: journal des connexions, qui conserve les traces des requêtes et des opérations traitées par le serveur.
- **Les bases de données clients** : ce sont les sources des données des entreprises.
- **Les cookies (ou Témoins)** : ce sont des fichiers que le serveur d'un site Web glisse au sein du disque dur de l'internaute le plus souvent à son insu (fichiers temporaires ou dossier Cookies) afin de stocker de l'information et mémoriser ses visites. Il permet, par exemple de l'identifier lorsqu'il revient visiter un site régulièrement [25].

II.6 Quelques définitions

Avant de commencer on peut donner quelques définitions qui nous seront utiles à la suite de notre travail :

- **Un URL (Uniforme Resource Identifier)** : est une chaîne de caractères utilisée pour identifier une ressource abstraite ou physique.
- **Une ressource web**: est une ressource accessible par une version du protocole http ou un protocole similaire (par exemple http 1.1).
- **Serveur web** : un serveur qui donne accès à des ressources web.
- **Requête web** : une requête pour une ressource web, faite par un client (navigateur web) à un serveur.
- **Page web** : ensemble des informations, consistant en une (ou plusieurs) ressource(s) web identifiée(s) par un seul URI (Uniform Resource Identifier). Exemple : un fichier HTML, un fichier image et un applet java accessible par un seul URI constituent une page web.
- **Navigateur web (Browser)**: logiciel de type client chargé d'afficher des pages à l'utilisateur et de faire des requêtes http au serveur web.
- **Utilisateur (internaute)**: personne qui utilise un navigateur web.
- **Navigation(s)** : ensemble de clics utilisateurs sur un seul serveur web (ou plusieurs lorsqu'on a fusionné leurs fichiers log) pendant une session utilisateur, qu'on appelle aussi visite(s).

- **Session utilisateur:** un ensemble délimité des clics utilisateurs sur un (ou plusieurs) serveur(s) Web.
- **Visite(s) :** L'ensemble des clics utilisateur sur un seul serveur Web (ou sur plusieurs lorsque on a fusionné leurs fichiers logs) pendant une session utilisateur. Les clics de l'utilisateur peuvent être décomposés dans plusieurs visites en calculant la distance temporelle entre deux requêtes http consécutives et si cette distance excède un certain seuil, une nouvelle visite commence [26].

II.7 Axes de développement du Web Mining

Les trois axes de développement du Web Mining sont : le Web Content Mining, le Web Structure Mining et le Web Usage Mining.

- **Web Content Mining :** Le Web Content Mining (WCM) consiste en une analyse textuelle avancée intégrant l'étude des liens hypertextes et la structure sémantique des pages Web. Ainsi, les techniques de description, de classification et d'analyse de chaînes de caractères du Text Mining sont très utiles pour traiter la partie textuelle des pages. Le WCM s'intéresse également aux images. Il permet, par exemple, de quantifier les images et les zones de texte, pour chaque page. Ainsi par l'analyse conjointe de la fréquentation des pages, il est possible de déterminer si les pages contenant plus d'images sont plus visitées que les pages contenant plus de texte.
- **Web Structure Mining :** Il s'agit d'une analyse de la structure du Web i.e. de l'architecture et des liens qui existent entre les différents sites. L'analyse des chemins parcourus permet, par exemple, de déterminer combien de pages consultent les internautes en moyenne et ainsi d'adapter l'arborescence du site pour que les pages les plus recherchées soient dans les premières pages du site. De même, la recherche des associations entre les pages consultées permet d'améliorer l'ergonomie du site par création de nouveaux liens.
- **Web Usage Mining :** Cette dernière branche du Web Mining consiste à analyser le comportement de l'utilisateur à travers sa navigation, notamment l'ensemble des clics effectués sur le site (on parle d'analyse du clickstream). Cette approche permet de mesurer l'audience et la performance d'un site Web (combien de temps passé par page, combien de visites, à quel moment, qui est l'utilisateur, quelle est la fréquence de ses consultations,..). L'intérêt du WUM est d'enrichir les sources de données de l'entreprise (bases de données clients, bases marketing,...) par les données brutes du clickstream afin d'affiner les profils clients ainsi que les modèles comportementaux [25].

En 1997, Cooley en décrit une taxonomie (voire Figure II.2). Celle-ci instaure l'emploi des notions de Web Content Mining et de Web Usage Mining et illustre leurs définitions à l'aide de travaux existants. Cette structuration des publications doit permettre leur évaluation plus précise et leur comparaison plus rapide.

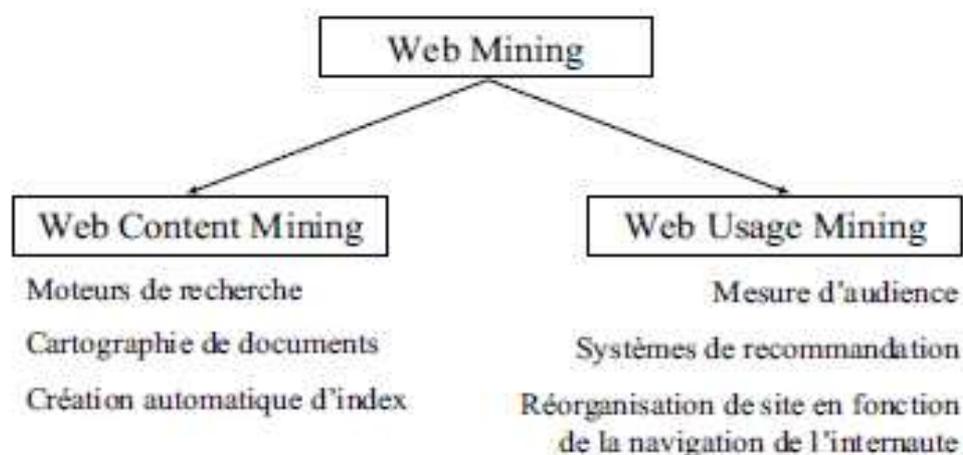


Figure II.2 Taxonomie du Web Mining proposées par Cooley en 1997.

Dans notre travail, nous nous intéressons au Web Usage Mining, pour cela nous précisons le fonctionnement des techniques du Web Usage Mining et leurs applications.

II.8 Web Usage Mining

Le Web Usage Mining (noté WUM par la suite) extrait des patrons de comportement à partir des fichiers logs et, éventuellement, des informations sur la structure du site (le plan des liens web) et sur les profils des utilisateurs. Le WUM consiste en "l'application des techniques de fouille de données pour découvrir des patrons d'utilisation à partir de données web dans le but de mieux comprendre et servir les besoins des applications web".

Le WUM comporte trois étapes principales : le prétraitement, l'extraction des motifs (ou de règles) et l'analyse (l'interprétation) des motifs.

Les principales données exploitées dans le WUM proviennent des fichiers Logs. Cependant, il existe d'autres sources d'informations qui pourraient être exploitées à savoir les connaissances sur la structure des sites Web et les connaissances sur les utilisateurs des sites Web.

II.8.1 Présentation des fichiers logs

Une des importantes sources d'information sur la navigation dans un site web est le fichier log, connu aussi sous les noms de "Accès log", "journal de connexion" ou "fichier de traces de navigation sur internet", enregistre sur un fichier texte les différentes actions effectuées par les visiteurs d'un serveur Web. La figure II.3, montre en schéma les étapes d'enregistrement des traces de navigation sur un internet dans le fichier log.

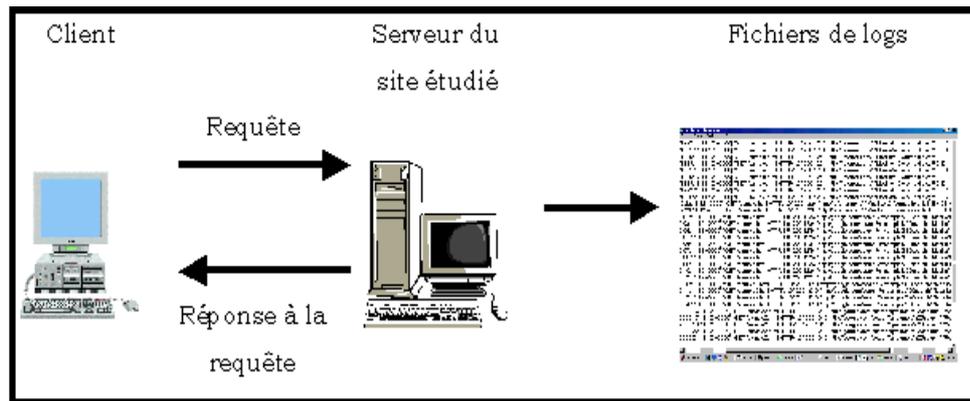


Figure II.3 Enregistrement des requêtes adressées au serveur gérant un site Web dans un fichier Log.

Selon [25], les fichiers logs peuvent se présenter sous deux formats principaux: CLF (Common Log Format), ECLF (Extended Common Log Format).

a) **Le format CLF (Common Log file Format)** : selon ce format six informations sont enregistrées:

- Le nom du domaine ou l'adresse de Protocole Internet (IP) de la machine appelante.
- Le nom et le login HTTP de l'utilisateur (en cas d'accès par mot de passe).
- La date et l'heure de la requête.
- La méthode utilisée dans la requête (GET, POST, etc.) et le nom de la ressource Web demandée (l'URL de la page demandée).
- Le code de statut attribué à la requête (prend la valeur 200 en cas de réussite).
- La taille de la page demandée en octets.

b) **Le format ECLF (Extended Common Log file Format)** : représente une version plus complète du format CLF. En effet, il indique en plus l'adresse de la page de référence (la page précédemment visitée ou le moteur de recherche utilisé pour rejoindre la page Web suivi des mots clés demandés), la configuration du client, c'est-à-dire, son navigateur Web (Firefox, Internet Explorer, etc.) et son système d'exploitation (Windows, Linux, Mac OS, etc.). Le format du fichier log a été standardisé par W3C. Le format ECLF:

[ip] [name] [date] [url] [statut] [taille] [refferer] [agent].

- Les Log d'erreur gardent la trace des erreurs survenues lors du téléchargement d'une page par le visiteur.
- Les Log référentiels indiquent d'où vient l'utilisateur quand il se connecte sur le site.
- Les Log d'agent renseignent sur le type de navigateur (Nescape, Explorer,...) utilisé par les visiteurs du site.

Pour illustrer le caractère massif de l'information collectée, nous allons représenter quelques lignes de la connexion d'un site. Le fichier résultant, Figure II.4, se présente comme une succession de lignes ou hits.

```

194.51.254.3 -- [01/Dec/1996:01:36:46 -0100] "GET /cgi-bin/Count.cgi?tr=N&dd=C|df=polar.dat HTTP/1.0" 200 907
194.51.254.3 -- [01/Dec/1996:01:37:55 -0100] "GET /entertainment/polar/polarweb/pwtete.htm HTTP/1.0" 200 2440
194.51.254.3 -- [01/Dec/1996:01:38:28 -0100] "GET /entertainment/polar/polarweb/album.htm HTTP/1.0" 200 2344
194.51.254.3 -- [01/Dec/1996:01:42:09 -0100] "GET /entertainment/polar/polarweb/pwcritik.htm HTTP/1.0" 200 9407
194.51.254.3 -- [01/Dec/1996:01:45:28 -0100] "GET /entertainment/polar/polarweb/pwlinks.htm HTTP/1.0" 200 14973
} Une Connexion

202.131.0.29 -- [01/Dec/1996:12:12:12 -0100] "GET /vl/vlis.html HTTP/1.0" 200 1876 ← Une page visualisée
202.131.0.29 -- [01/Dec/1996:12:12:20 -0100] "GET /vl/metrics.html HTTP/1.0" 200 9841

crm.univ-mrs.fr -- [01/Dec/1996:12:12:12 -0100] "GET /vl/vlis.html HTTP/1.0" 200 1876

```

Figure II.4 Exemple d'un fichier log.



Figure II.5 Schéma illustratif des champs d'une requête.

II.8. 2 La difficulté du décryptage de l'User-Agent

D'une part, les navigateurs utilisent des formats différents pour s'identifier (absence de format standard). En effet, Netscape présente généralement un User-Agent de la forme:

Produit [langue] (plateforme; niveau de sécurité; description du système d'exploitation)

Le niveau de sécurité prend la valeur U pour sécurité de haut niveau et I pour sécurité de bas niveau.

Exemple : *Mozilla / 4.04 (X11; I; SunOS 5.4 sun4m)*

Par contre, Internet Explorer présente un format différent : *Mozilla/4. 0 (compatible; MSIE 5. 23; Mac_ PowerPC).*

D'autre part, la version du navigateur n'est pas toujours explicite. Il faut dans certains cas la dégager à partir du «Build Number». Dans l'exemple : *Mozilla/5.0 (Macintosh; U; PPC Mac OS X; de-de) AppleWebKit/85. 7 (KHTML, like Gecko) Safari/85.7*, le «Build Number» vaut 85.7 et le navigateur est Safari dont la version est 1.0

Le tableau II.1 présente un exemple sur le décryptage de l'User-Agent :

OS	Navigateur	User-Agent
Windows XP	Mozilla Firefox 1.0	Mozilla/5.0 (Windows ; U ;Windows NT 5.1 ; en-US ; rv :1.8.1.20) Gecko/20081217 Firefox/1.0
Windows 7	Mozilla Firefox 2.0.0.20	Mozilla/5.0 (Windows ; U ;Windows NT 6.1 ; fr ; rv :1.8.1.20) Gecko/20081217 Firefox/2.0.0.20
Windows Vista	Mozilla Firefox 3.0.1	Mozilla/5.0 (Windows ; U ;Windows NT 6.0 ; fr ; rv :1.9.0.1) Gecko/200870208 Firefox/3.0.1
Linux	Mozilla Firefox 3.5.1	Mozilla/5.0 (X11 ; U ;Linux i686 ; fr ; rv :1.9.1.1) Gecko/20090715 Firefox/3.5.1
Windows XP	Google Chrome 0.2.149.27	Mozilla/5.0 (Windows ; U ;Windows NT 5.1 ; en-US) AppleWebKit/525.13 (KHTML, Like Gecko) chrome/0.2.149.27 Safari/525.13
Windows Vista	Google Chrome 0.2.149.27	Mozilla/5.0 (Windows ; U ;Windows NT 6.0 ; en-US) AppleWebKit/525.13 (KHTML, Like Gecko) chrome/0.2.149.27 Safari/525.13
Windows 95	Internet Explorer 1.0	Microsoft Internet Explorer/4.0b1 (Windows 95)
Windows NT	Internet Explorer 1.5	Mozilla/1.22 (compatible; MSIE 1.5; Windows NT)
Windows Vista	Internet Explorer 7.0	Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0)

Tableau II.1 Décryptage de l'User-Agent.

En outre, La difficulté d'identification du système d'exploitation est à prendre en considération. En effet, souvent le système d'exploitation Windows s'identifie en utilisant des noms de versions différentes de celles réelles. Le tableau suivant présente les versions mentionnées dans l'User-Agent et les versions réelles correspondantes.

Version mentionnée	Version réelle
Windows NT 5.1	Windows XP
Windows NT 6.1	Windows 7
Windows NT 6.0	Windows Vista
Windows NT 5.0	Windows 2000
Windows NT 5.2	Windows serveur 2003

Tableau II.2 Identification du système d'exploitation.

II.8.3 Problèmes spécifiques aux données des fichiers Logs

Bien que les données fournies par les fichiers Logs soient utiles, il importe de prendre en compte les limites inhérentes à ces données lors de leur analyse et de leur interprétation. Parmi les difficultés qui peuvent survenir:

- **Les requêtes inutiles** : Chaque fois qu'il reçoit une requête, le serveur enregistre une ligne dans le fichier Log. Ainsi, pour charger une page, il y'aura autant de lignes dans le fichier que d'objets contenus sur cette page (les éléments graphiques). Un prétraitement est donc indispensable pour supprimer les requêtes inutiles.
- **Les firewalls** : Ces protections d'accès à un réseau masquent l'adresse IP des utilisateurs. Toute requête de connexion provenant d'un serveur doté d'une telle protection aura la même adresse et ce, quel que soit l'utilisateur. Il est donc impossible, dans ce cas, d'identifier et de distinguer les visiteurs provenant de ce réseau.
- **Le Web caching**: Afin de faciliter le trafic sur le Web, une copie de certaines pages est sauvegardée au niveau du navigateur local de l'utilisateur ou au niveau du serveur proxy afin de ne pas les télécharger chaque fois qu'un utilisateur les demande. Dans ce cas, une page peut être consultée plusieurs fois sans qu'il y' ait autant d'accès au serveur. Il en résulte que les requêtes correspondantes ne sont pas enregistrées dans le fichier Log.
- **L'utilisation des robots** : Les annuaires du Web, connus sous le nom de moteurs de recherche, utilisent des robots qui parcourent tous les sites Web. Afin de mettre à jour leur index de recherche. Ce faisant, ils déclenchent des requêtes qui sont enregistrées dans tous les fichiers Logs des différents sites, faussant ainsi leurs statistiques.
- **L'identification des utilisateurs** : L'identification des utilisateurs à partir du fichier Log n'est pas une tâche simple. En effet, en employant le fichier Log, l'unique identifiant disponible est l'adresse IP et «l'agent» de l'utilisateur. Cet identifiant présente plusieurs limites [27].
- **Adresse IP unique / Plusieurs sessions serveurs**: La même adresse IP peut être attribuée à plusieurs utilisateurs accédant aux services du Web à travers un unique serveur Proxy.
- **Plusieurs adresses IP / Utilisateur unique**: Un utilisateur peut accéder au Web à partir de plusieurs machines.
- **Plusieurs agents / Utilisateur unique** : Un internaute qui utilise plus d'un navigateur, même si la machine est unique, est aperçu comme plusieurs utilisateurs.
- **L'identification des sessions** : Toutes les requêtes provenant d'un utilisateur identifié constituent sa session. Le début de la session est défini par le fait que l'URL de provenance de l'utilisateur est extérieure au site [28]. Par contre, aucun signal n'indique la déconnexion du site et par suite la fin de la session.

- **Le manque d'information** : Le fichier Log n'apporte rien sur le comportement de l'utilisateur entre deux requêtes : Que fait ce dernier? Est-il vraiment en train de lire la page affichée? De plus, le nombre de visites d'une page ne reflète pas nécessairement l'intérêt de celle-ci. En effet, un nombre élevé de visites peut simplement être attribué à l'organisation d'un site et au passage forcé d'un visiteur sur centaines.

II.9 Prétraitement de fichier Log

La première étape d'un processus du Web Usage Mining consiste en un prétraitement des fichiers Log. En effet, le format des fichiers log Web est impropre à une analyse directe par les diverses techniques de fouille de données. Leur nettoyage et leur structuration sont donc nécessaires avant toute analyse.

L'étape du Web Usage Mining se compose principalement de deux types de tâches:

1. Tâches classiques de prétraitement : fusion des fichiers logs web, nettoyage et structuration de données.
2. Tâches avancées de prétraitement : stockage des données structurées dans une base de données, généralisation et agrégation des données.

Le prétraitement des données se décompose en deux phases principales : une phase de nettoyage des données et une phase de transformation.

La méthode de prétraitement que nous présentons est un résumé des travaux de D.Tanasa présentés dans [29]. Elle contient huit étapes distinctes dont quatre de nettoyage et quatre de transformation de données. La succession chronologique de ces étapes est donnée dans le tableau II.3.

Unité de traitement	Requête			Session	Visite	
		N1	N2 (a, b)			N2c
Nettoyage des données						
Transformation des données	T1			T2	T3	T4

Tableau II.3 Les étapes du prétraitement dans le WUM.

II.10 Nettoyage des données

Le nettoyage des données pour les fichiers logs Web consiste à supprimer les requêtes pour les ressources Web qui ne font pas l'objet de l'analyse (étape N1) et les requêtes ou visites provenant des robots Web (étape N2).

Pour les sites Web très populaires la dimension de fichiers logs Web est comptée en gigabytes par heure. Par exemple, Yahoo, le plus populaire site Web, collecte presque 100 GB de données logs Web par heure. Même avec les systèmes et les logiciels de nos jours, manipuler des fichiers de telles dimensions devient très compliqué. Pour cette raison, bien nettoyer ces données avant toute analyse est crucial dans le WUM. Par le filtrage des données inutiles, non seulement on gagne de l'espace disque, mais, dans le même temps on rend plus efficaces les tâches qui suivent dans le processus de WUM.

II.10.1 Suppression des requêtes pour les ressources Web non-analysées (N1)

Etant donné l'objectif de notre analyse, étudier le comportement des utilisateurs sur un site Web, nous allons supprimer les requêtes auxiliaires comme celles pour les images ou les fichiers multimédia.

Donc, Le nettoyage des données vise à éliminer toutes les requêtes considérées comme inutiles de l'ensemble de logs de départ. En effet, une quantité non négligeable des enregistrements d'un serveur web ne reflète pas le comportement de l'internaute, nous expliquons donc les problèmes de ces enregistrements en détail et ce que nous apportons pour leur gestion.

Le serveur web permet de disposer des ressources de tout type : page web, élément multimédia, programme, donnée quelconque. Lors d'une requête correspondant à une page intégrant d'autres ressources (généralement des images, ou de petites animations), le client exécute effectivement plusieurs requêtes vers le serveur: une pour la page (le contenant), une pour les divers éléments (les contenus). Ainsi, pour une page demandée, plusieurs requêtes peuvent aboutir au serveur.

En se référant au but de notre travail, à savoir l'exploration des traces de navigation sur internet, il nous paraît judicieux de ne conserver que les pages web (dites contenant), sans les éléments incorporés. Nous simplifions les données des logs en enlevant tout ce qui ne correspond pas à une page web. Certaines expériences ont montré que la suppression des URL concernant des images réduisait la taille du fichier log original de 40 % à 85 %. Le nettoyage des images consiste à supposer qu'un utilisateur ne peut cliquer à la fois (au même instant) sur plusieurs images pour les visualiser.

Avant de concevoir notre corpus d'apprentissage nous commençons par nettoyer le fichier Log, qui consiste à supprimer les requêtes inutiles des fichiers Logs, à savoir :

a) Code Statut non valides:

Ce sont les requêtes dont le statut est inférieur à 200 ou supérieur à 399. En effet, le code d'état (statut), entier codé sur trois chiffres, a un sens propre dont la catégorie dépend du premier chiffre:

- 1xx indique uniquement un message informel,
- 2xx indique un succès,
- 3xx redirige le client sur un autre URL,
- 4xx indique une erreur côté client,
- 5xx indique une erreur côté serveur.

b) Requêtes aux images:

Cette étape de nettoyage consiste à supprimer les fichiers dont les extensions sont : .jpg, .gif, .png, etc, et les fichiers multimédia dont l'extension est : .wav, .wma, .wmv, etc.

Deux méthodes ont été utilisées pour supprimer les requêtes aux images. La première consiste à utiliser la carte du site afin d'identifier les URLs des images nécessitant de cliquer sur un lien pour être affichées. Les images incluses dans les fichiers HTML sont supprimées car elles ne reflètent pas le comportement de l'internaute. A titre d'exemple, la page dont l'URL est `www.cck.rnu.tn\arabe\ntic_tunisie\ntic_ar.Htm` comporte les images suivantes qui s'affichent sans avoir besoin de cliquer sur un lien :

```
www.taseek.com - [01/Jan/2010:01:11:03 +0100] "GET /images/taseek-1.jpg
www.taseek.com-[01/Jan/2010:07:12:48+0100]"GET/images/taseek_r6_c3.png
```

Figure II.6 Requêtes aux images.

Cependant, ce n'est pas toujours possible d'identifier toutes les images inintéressantes quand le site est volumineux. Dans ce cas, on propose une autre méthode dont l'application nécessite tout d'abord l'identification des sessions.

c) Requêtes dont la méthode est différente de « GET »:

Les méthodes généralement utilisées sont: GET, HEAD, PUT, POST, TRACE et OPTIONS:

- La méthode GET est une requête d'information. Le serveur traite la demande et renvoie le contenu de l'objet.
- La méthode HEAD est très similaire à la méthode GET. Cependant le serveur ne retourne que l'en-tête de la ressource demandée sans les données. Il n'y a donc pas de corps de message.
- La méthode PUT permet de télécharger un document, dont le nom est précisé dans l'URI, ou d'effacer un document, toujours si le serveur l'autorise.
- La méthode POST est utilisée pour envoyer des données au serveur.
- La méthode TRACE est employée pour le débogage. Le serveur renvoie, dans le corps de la réponse, le contenu exact qu'il a reçu du client. Ceci permet de comprendre, en particulier, ce qui se passe lorsque la requête transite par plusieurs serveurs intermédiaires.
- La méthode OPTIONS permet de demander au serveur les méthodes autorisées pour le document référencé.

Vu que le WUM s'intéresse à l'étude du comportement de l'internaute sur le Web et par conséquent aux ressources qu'il demande, il faut garder seulement les requêtes dont la méthode utilisée est GET.

d) Scripts:

Généralement, le téléchargement d'une page demandée par un utilisateur est accompagné automatiquement par le téléchargement des scripts tels que les scripts Java (fichiers .js), des feuilles de style (fichiers .css), des animations flash (fichier .swf), etc. Ces éléments doivent être supprimés du fichier Log étant donné que leur apparition ne reflète pas le comportement de l'internaute.

e) Requêtes spécifiques à l'activité sur le site:

Ce sont les requêtes relatives au trafic sur le site objet de l'analyse. Cette étape montre que la méthodologie d'analyse du comportement des internautes sur le Web n'est pas unique et qu'elle dépend de plusieurs facteurs, en particulier du site analysé. Cette étape consiste à supprimer:

- Les requêtes pour les pages « proxy.pac ».
- Les requêtes aux pages:

Exemple :

“http://www.cck.rnu.tn/haut.htm”

“http://www.cck.rnu.tn/haut.asp”

Car ces pages s'affichent automatiquement avec la page d'accueil du site et servent d'entête (frame) pour toutes les autres pages.

- Les requêtes pour les annonces (les popups). En effet, les annonces apparaissent toutes seules dès que l'utilisateur se connecte sur un site. De ce fait, les requêtes correspondantes ne reflètent pas son comportement. Pour éliminer ces requêtes, il faut identifier les URLs correspondantes de la forme: “www.cck.rnu.tn/popup/pop.htm”.

II.10. 2 Suppression des requêtes et visites provenant des robots Web (N2a, b, c)

Les robots Web (WRs) sont des logiciels utilisés pour balayer un site Web, afin d'extraire son contenu. Ils suivent automatiquement tous les liens d'une page Web. Les moteurs de recherche, comme Google, envoient régulièrement leurs robots pour extraire toutes les pages d'un site Web afin de mettre à jour leurs index de recherche. Le nombre de demandes d'un WR est en général supérieur au nombre de demandes d'un utilisateur normal. En mode paradoxal, si le site Web n'attire pas beaucoup de visiteurs, les demandes faites par tous les WR qui l'ont visité peuvent être supérieures aux demandes faites par des humains.

La suppression des entrées dans le fichier log produites par WRs simplifie la tâche de fouille de données qui suivra et permet également de supprimer les sessions non-intéressantes, en particulier en cas de reconception de site. Habituellement, un WR s'identifie en employant le champ « User Agent » dans les fichiers logs. Cependant, aujourd'hui, il est presque impossible de connaître tous les agents qui représentent un WR car chaque jour apparaissent des nouveaux WRs et ceci rend cette tâche très difficile.

Nous avons utilisé cinq heuristiques pour identifier les requêtes (N2a, b) ou visites (N2c) issues des WRs:

- Identifier les adresses IP et les « User-Agents » connus comme étant des robots Web. Ces informations sont fournies généralement par les moteurs de recherche.
- Identifier les IP ayant fait une requête à la page « \robots.txt ».
- Identifier les « User-Agents » comportant l'un des mots clés suivants: « crawler », « spider » ou encore « bot ».
- Identifier les requêtes effectuées par des aspirateurs de sites Web (HTTrack par exemple), ou par des modules de certains navigateurs permettant la consultation de pages hors ligne tels que DigExt d'Internet Explorer. L'identité de ces aspirateurs ou de ces modules est trahie par la mention de leurs noms au niveau de leurs User-Agents. Pour les aspirateurs qui cachent leurs User-Agents, leur identification est effectuée ultérieurement en se basant sur la durée de leurs requêtes généralement nulle.
- Utiliser un seuil pour la vitesse de navigation BS « Browsing Speed » égale au nombre de pages visitées par seconde. Le calcul du Browsing Speed n'est possible qu'après détermination des sessions et des visites, c.-à-d. cette étape doit être exécutée après l'étape T4 car, pour calculer BS, il faut que les requêtes soient groupées en visites. Une fois que toutes les requêtes/visites venant des WRs ont été identifiées, nous pouvons procéder à leur suppression.

II.11 Transformation des données

Dans l'étape de transformation des données, nous avons fusionné des fichiers logs Web (T1), groupé les requêtes par sessions (même IP, même User Agent) (T3). Ensuite, les sessions ont été divisées en visites en choisissant un $\Delta t = 30min$ (T4).

II.11.1 Fusionnement des fichiers logs ensemble (T1)

Avant même de commencer le processus de nettoyage, nous allons fusionner les différents fichiers logs. Les requêtes de tous les fichiers logs ont été mises ensemble dans un seul fichier.

II.11.2 Rendre anonymes les IP des utilisateurs (T2)

Pour des raisons de confidentialité, nous avons remplacé le nom original ou l'adresse IP de la machine appelante avec un identificateur. Toutefois dans le codage de l'identificateur, nous gardons l'information sur l'extension du domaine (pays ou type d'organisation: .com, .org, .edu, etc.).

II.11.3 Identification de l'utilisateur (T3)

L'identification des utilisateurs à partir du fichier log n'est pas une tâche simple en raison de plusieurs facteurs comme: les serveurs proxy, les adresses dynamiques, le cas d'utilisateurs utilisant le même ordinateur (dans une bibliothèque, club Internet, etc.) ou celui d'un même utilisateur utilisant plus d'un navigateur Web ou plus d'un ordinateur. En effet, en employant le fichier log, nous connaissons seulement l'adresse de l'ordinateur (IP) et (l'agent) de l'utilisateur. Il existe d'autres méthodes qui fournissent plus d'informations. Les plus utilisées sont: les "cookies", les pages dynamiques Web (avec un identifiant de session dans l'adresse URL), les utilisateurs enregistrés, les navigateurs modifiés etc.

Nous allons utiliser le couple (IP, User Agent) pour l'identification de l'utilisateur. Pour obtenir la session de chaque utilisateur, nous allons ordonner le fichier log par le couple (IP, User Agent) et ensuite par le temps. Cette session contient toutes les requêtes de l'utilisateur dans la période analysée. Ensuite, nous allons diviser la session en visites.

a) Identification des internautes et des sessions

En ce qui concerne l'identification de l'utilisateur, pour les sites Web, il est indispensable d'identifier clairement chaque utilisateur. Si le serveur ne peut différencier les requêtes qui lui parviennent, toute solution proposée n'est pas optimale.

b) Adresse IP

Sur Internet, les ordinateurs communiquent entre eux grâce au protocole TCP/IP « Transmission Control Protocol ». Chaque ordinateur appartenant au réseau est identifié par une séquence unique de 32 bits (l'adresse IP) écrite à l'aide de quatre nombres compris entre 0 et 255.

Les adresses IP ont l'avantage d'être toujours disponibles et de ne nécessiter aucun traitement préalable. En revanche, elles possèdent principalement deux limites :

- Premièrement, une adresse IP peut n'identifier qu'un groupe d'ordinateurs « cachés » derrière le serveur proxy d'un fournisseur d'accès à Internet ou d'un réseau local. Rappelons qu'un serveur proxy a une double fonction : il permet aux ordinateurs d'un réseau utilisant des adresses IP privées d'accéder à Internet par son intermédiaire. Il peut également servir de cache, c'est-à-dire qu'il peut garder en mémoire les pages les plus souvent visitées pour les fournir plus rapidement. Ainsi, tous les internautes utilisant un serveur proxy seront identifiés par l'unique adresse IP de ce serveur. Le site visité ne peut alors déceler s'il a à faire à un ou plusieurs visiteurs.
- Le deuxième inconvénient de l'utilisation des adresses IP comme identifiants vient de son inadéquation à la rétribution dynamique. La majorité des internautes se voient en effet prêter une adresse IP par leur fournisseur d'accès le temps d'une connexion à Internet. Cet inconvénient est particulièrement influant sur les sites ayant de nombreux visiteurs, les adresses IP attribuées dynamiquement pouvant être réutilisées immédiatement par d'autres utilisateurs. Par ailleurs, l'attribution dynamique ne permet une identification valable que pour une seule session ininterrompue : si l'internaute interrompt sa visite en se déconnectant un bref instant, sa session sera toujours en cours, mais son adresse IP aura changé, l'identification ne sera donc plus possible.

c) Définition d'une session

Une session est composée de l'ensemble de pages visitées par le même utilisateur durant la période d'analyse, cependant, la combinaison des champs IP (adresse) et User Agent (le navigateur Web) d'un fichier log Web identifie correctement l'utilisateur dans 92.02 % des cas et seul un nombre limité de ces combinaisons (1.32 %) sont utilisés par plus de trois utilisateurs. Chaque session est caractérisée par le nombre de requêtes effectuées par l'utilisateur durant cette session, le nombre de pages consultées (URLs différentes) et la durée de la session. Nous pouvons considérer la combinaison adresse IP plus navigatrice comme

étant un critère acceptable pour l'identification d'un utilisateur dans le cadre d'une activité ponctuelle.

d) Algorithme d'identification

Afin de mieux identifier les sessions, nous adoptons l'algorithme proposé ci-dessous.

```

Tant qu'il y'a dans enregistrements dans la base faire
  Lire l'enregistrement i
  Récupérer l'adresse  $IP_i$  et le User Agent  $UA_i$ 
  Si le couple  $(IP_i, UA_i) = (IP_{(i-1)}, UA_{(i-1)})$ 
    Alors ajouter l'enregistrement i a la session  $S_{(i-1)}$ 
    Sinon recommencer une nouvelle session  $S_i$ 
  Fin Si
Fin Tant Que

```

Algorithme II.1 Algorithme d'identification des utilisateurs.

II.11.4 Identification des visites (T4)

Une fois l'utilisateur (approximativement) identifié, le problème consiste alors à détecter toutes les requêtes en provenance de cet utilisateur. Concernant cette problématique, la méthode la plus simple pour le groupement des requêtes en sous-ensembles de requêtes appartenant au même utilisateur est d'utiliser le temps de latence maximum entre deux requêtes successives. Selon [30], ce temps est estimé de manière empirique à 25,5 minutes. La majorité des méthodes d'analyse de l'usage du Web ont donc adopté le temps de latence de 30 minutes. Dans [31], La navigation définit comme l'ensemble de requêtes (clics) en provenance d'un même utilisateur séparées au-delà de 30 minutes, et session comme l'ensemble de navigations d'un même utilisateur.

Cette stratégie serait capable d'identifier les requêtes en provenance d'un même utilisateur dans le contexte d'une seule navigation. Cependant, on ne peut pas généraliser cette combinaison pour l'identification de plusieurs navigations appartenant à un même utilisateur, puisque nous n'avons aucune garantie que l'utilisateur d'avant aura les mêmes valeurs pour le couple adresse (IP + navigateur) lors d'une prochaine visite sur le site Web.

Une fois les visites identifiées, la durée de consultation de la dernière page de chaque visite (la dernière page de chaque session et les pages dont la durée de consultation a dépassé 30 minutes celles qui ont permis la construction des visites) est obtenue à partir de la moyenne des durées de consultation des pages précédentes appartenant à la même visite.

En premier temps, on doit collecter des informations sur l'identification des visites d'utilisateurs. Soit les variables suivantes :

- R_i = Requête i
- $V [R_i]$ = Visite à laquelle appartient la requête i
- $S [R_i]$ = Session à laquelle appartient la requête i

- $T [R_i]$ = Temps de déclenchement de la requête i .
 - $Durée [R_i]$ =Durée de la requête i .
 - $Durée$ = Somme des durées de requêtes de chaque visite.
 - NV = Nombre de requêtes dans chaque visite.
1. Ordonner les requêtes suivant la variable $S [R_i]$.
 2. Détermination de la durée de chaque requête.
 3. Construction des visites.
 4. Détermination de la durée de la dernière requête de chaque visite.

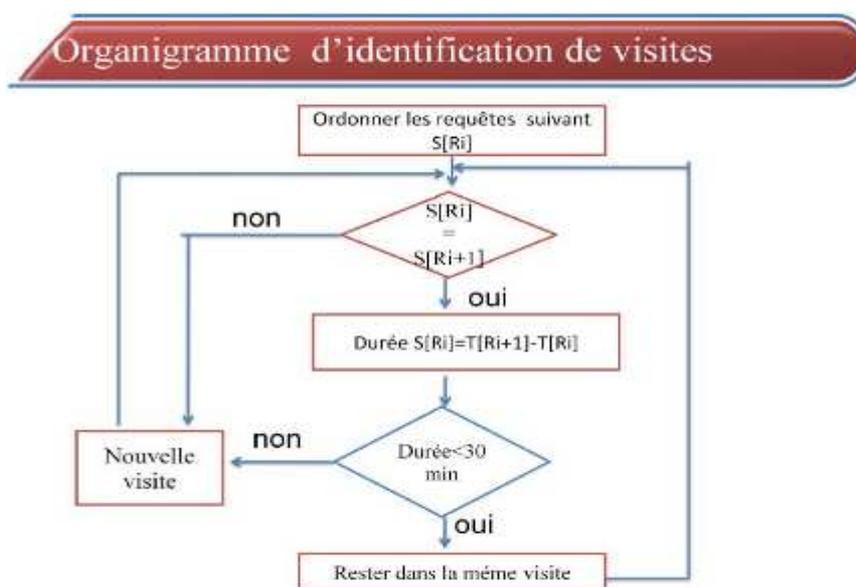


Figure II.7 Organigramme d'identification des visites d'utilisateur.

Pour l'identification des visites d'utilisateur, on utilise les mêmes variables citées dans l'organigramme, pour cela l'algorithme sera écrit comme suite :

- Ordonner les requêtes suivant la variable S [Ri] puis la variable T [Ri].
- Détermination de la durée de chaque requête: $i=1$;

Pour i de 1 à N-1

Si S [Ri] =S [Ri+1] **Alors**

Durée [Ri] = T [Ri+1] - T [Ri] ;

Flag [Ri] = 1;

Sinon

Flag [Ri] = 0 ; (Ri est la dernière requête de la session)

FinSi

$i = i+1$;

FinPour

Flag [RN] = 0; (RN est la dernière requête dans la base)

1. Construction des visites:

$V [R1] = 1$;

Pour i de 1 à N

Si Durée [Ri] > 30 minutes ou Flag [Ri] = 0 **Alors**

$V [Ri+1] = V [Ri] +1$;

Sinon

$V [Ri+1] = V [Ri]$;

FinSi

$i = i+1$;

FinPour

2. Détermination de la durée de la dernière requête de chaque visite:

$i=1$;

Pour i de 1 à N-1

Si Flag [Ri] = 0 ou Durée [Ri] > 30 minutes

Durée [Rk]) / NV [Ri] -1 Tel que $V [Rk] = V [Ri]$

FinSi

$i = i+1$;

FinPour

Algorithme II.2 Algorithme d'identification des visites des utilisateurs.

En résumé, le schéma caractérisant l'identification des sessions et des visites prend la topologie suivante indiquée dans la figure II.8.

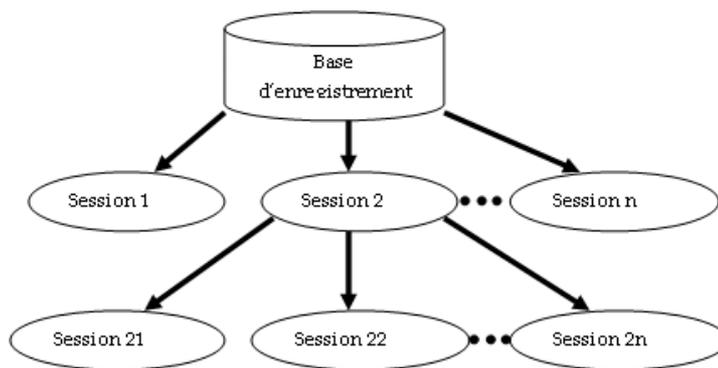


Figure II.8 Identification des sessions et des visites.

II.12 Mesure de similarité entre sessions web

La définition de la similarité entre sessions peut se faire selon différents critères : par exemple on peut considérer que deux sessions sont proches si elles ont accédé le même nombre de fois aux pages du site, ou pendant la même durée ou si elles forment les mêmes séquences de pages. Pour cela, on peut considérer qu'une session correspond à un ensemble de pages, sans prendre en compte l'ordre dans lequel elles ont été visitées, et de définir que deux sessions sont similaires si elles accèdent à des pages similaires.

La similarité entre pages est définie en fonction de la similarité du chemin de leurs urls comme dans [46] et non à partir du contenu des documents qui nécessite une phase coûteuse d'extraction et d'indexation. Cette approche présuppose que la structure des répertoires du site reflète son contenu. Plus précisément, la similarité $S_{url}(u_1, u_2)$ entre 2 urls u_1 et u_2 est calculée comme la somme pondérée des éléments identiques des chemins des 2 urls (les répertoires et les fichiers accédés) en donnant un poids plus important aux éléments les plus proches de la racine du site.

La similarité normalisée entre deux sessions s_1 and s_2 est ensuite calculée comme :

$$S(s_1, s_2) = \frac{\tilde{S}(s_1, s_2)}{\sqrt{\tilde{S}(s_1, s_1)\tilde{S}(s_2, s_2)}} \quad \text{où} \quad \tilde{S}(s_1, s_2) = \sum_{u_1 \in s_1} \sum_{u_2 \in s_2} S_{url}(u_1, u_2)$$

Donc :

$$S(s_1, s_2) = \begin{cases} 1 & \text{si } s_1 = s_2 \\ 0 & \text{sinon} \end{cases}$$

II.13 Conclusion

Sur internet, l'utilisateur fournit des éléments par une simple navigation représentés sous forme de traces et stockés sous forme de fichier. Dans ce chapitre, nous avons présenté une étude de ces traces utilisées dans notre travail.

Nous avons aussi présenté les problèmes de données inutiles ou incohérentes dans ces derniers apparaissent comme rédhibitoires à l'utilisation dans le processus de fouille d'usage du Web, très sensible aux données bruitées. Nous avons présenté une méthode de prétraitement des données avec une mise en place d'heuristiques qui permettent de transformer l'ensemble de requêtes enregistrées dans les fichiers Logs à des données structurées, nettoyées et exploitables.

Chapitre III :

Classification et Clustering par les Fourmis Artificielles

III.1 Introduction

La classification de données ou "clustering" en anglais vise à partir d'un ensemble de données (individus ou objets), de regrouper ces données en sous ensembles (groupes, classes ou clusters) d'une manière pertinente [32]. Pertinence émanant du jugement d'un expert du domaine ou encore selon le degré d'homogénéité des groupes du point de vue des caractéristiques servant à décrire les données qui les composent. D'autres définissent le clustering comme un processus de classification d'objets en groupes ou amas dont les membres sont d'une certaine manière similaires. Un cluster ou un groupe est donc un ensemble d'objets qui sont « similaires » entre eux et « différents » des objets appartenant aux autres groupes.

III.2 Taxonomie des méthodes de classification ou clustering

Il existe plusieurs familles de méthodes de classification, certaines sont dites exclusives dans lesquelles la donnée n'appartient qu'à une et une seule classe et d'autres sont non exclusives où une donnée peut appartenir à plusieurs classes simultanément avec des degrés d'appartenance. On parle alors de recouvrement ou "overlapping clustering method".

La méthode de classification peut être non supervisée, ce qui revient à créer une partition d'un ensemble de données tel que les éléments d'une même classe, possèdent des caractéristiques similaires, quant à l'apprentissage supervisé, il vise le classement de nouveaux éléments à partir d'une partition existante. Enfin, on distingue les deux grandes familles d'approches, l'approche par partitionnement qui ne construit qu'une seule partition des données, alors que l'approche hiérarchique ne se contente pas d'une seule partition, mais crée une hiérarchie de parties qui constituent un arbre. Une classification hiérarchique n'est autre qu'une suite de séquence de partitionnement [33].

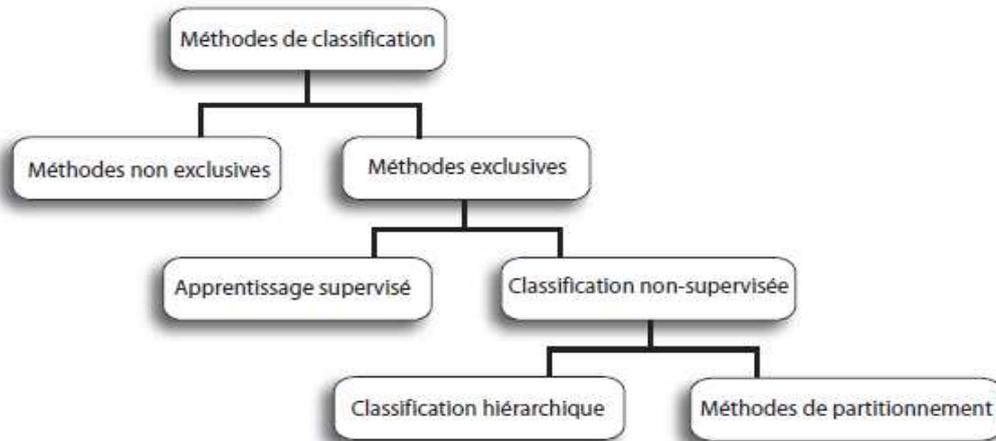


Figure III.1 Hiérarchie des méthodes de classification.

Il existe de nombreux algorithmes qui permettent la classification tels que les algorithmes des colonies de fourmis qui représentent notre méthode de classification des traces de navigation sur internet. Ces algorithmes de classification sont utilisés dans des domaines divers et variés, tels que la biologie, le marketing, la bibliothèque, les assurances, l'urbanisme, etc.

III.3 Algorithme des fourmis artificielles

Il existe deux types d'algorithmes des fourmis artificielles :

III.3.1 Algorithmes de fourmis artificielles avec l'utilisation de phéromones

Les fourmis utilisent les phéromones (mélange d'hydrocarbures) pour construire des chemins entre leur nid et une source de nourriture. Ces phéromones servent alors à mémoriser le chemin mais également à recruter d'autres ouvrières pour exploiter la source de nourriture. Ces substances chimiques s'évaporent avec le temps et le chemin disparaît s'il n'est pas renforcé par le passage des fourmis et le dépôt de nouvelles phéromones. Les phéromones déposées sur le sol constituent donc une forme de mémoire collective et permet aux fourmis d'apprendre collectivement, et sans communication directe, tout en ayant la possibilité d'oublier des chemins qui auraient perdu de leur intérêt.

La première démonstration en informatique de la capacité et de la performance de la communication chimique par phéromone observée chez les fourmis réelles a été réalisée par Dorigo en 1992 au travers de l'heuristique Ant Colony Optimization (ACO).

1) Brève introduction à ACO

Les recherches sur le comportement des colonies de fourmis ont montré que leur communication est essentiellement basée sur la production et la détection d'agents chimiques appelés phéromones. Parmi les différents types de phéromones, certaines sont déposées sur le sol. Un chemin marqué par de telles phéromones, par exemple d'une source de nourriture au nid, est alors suivi par les autres fourmis avec des fluctuations qui sont aléatoires ou dues à la manière dont la fourmi perçoit son environnement et y réagit (sa "visibilité locale"). Ces phéromones sont également volatiles. Leur évaporation progressive permet de diversifier les chemins empruntés ou d'en explorer de totalement nouveaux. Des expérimentations biologiques et des résultats théoriques ont montré que les fourmis utilisent ces phéromones pour trouver le chemin le plus court vers une source de nourriture, et donc de ce fait résolvent un problème d'optimisation. Déterminer le chemin le plus court est en effet une tâche similaire à de nombreux problèmes combinatoires tels que le problème du voyageur de commerce. Ceci a conduit les chercheurs à s'inspirer du comportement des fourmis pour élaborer de nouveaux algorithmes de résolution. L'approche fut ensuite étendue à une méta-heuristique pour les problèmes d'optimisation discrets connue sous le nom de « Ant Colony Optimization », [34].

2) Méta-heuristique ACO et algorithmes

Marco Dorigo propose une méta-heuristique ACO et un algorithme (Ant System) pour les problèmes d'optimisation discrets combinatoires. Cet algorithme a initialement été appliqué à la résolution du problème du voyageur de commerce (PVC) ou Traveling Salesman Problem.

Plusieurs extensions à l'algorithme AS (Ant System) ont vu le jour afin d'améliorer la répartition des phéromones sur le graphe ainsi que d'ajuster plus finement la probabilité de choix d'une ville par les fourmis. Nous ne détaillerons pas ces mises à jour dans ce document mais nous pouvons cependant citer les travaux suivants :

- Dans [35], Gambardella et Dorigo introduisent un système d'apprentissage par renforcement,
- Dans [36], Stützle et Hoos proposent le système MAX -MIN AS introduisant des seuils minimum et maximum à la trace de phéromone,
- Dans [37], Dorigo et Gambardella donnent l'algorithme Ant Colony System (ACS). Ce dernier introduit principalement plus de choix probabiliste dans la décision d'exploration d'une ville et correspond à l'heuristique à base de fourmis artificielles apportant les meilleurs résultats pour le PVC.

III.3.2 Algorithmes de fourmis artificielles sans l'utilisation de phéromones

Des chercheurs se sont intéressés à la manière dont les fourmis classent leurs proies et ont proposé de modéliser des stratégies de prédation sans phéromones pour résoudre des problèmes d'optimisation.

Nous décrivons ici une nouvelle stratégie de fourrage observé chez les fourmis de la famille *Pachycondyla apicalis*. Ces fourmis vivent en Amérique du Sud et plus précisément au Mexique.

Elles peuvent être considérées comme assez primitives, n'utilisent pas de communication poussée à base de phéromone lors de la recherche de nourriture et leur comportement de chasse reste relativement simple. En effet, les fourmis chassent individuellement en tentant de couvrir un espace particulier autour du nid de la colonie. Pour cela, elles sélectionnent plusieurs sites de chasse et effectuent des explorations locales autour de ces sites. Il s'agit à la fois d'une stratégie d'optimisation globale et locale qui s'est révélée assez efficace pour maintenir jusqu'à nos jours l'existence de ces fourmis.

D'un point de vue biologique, ces fourmis vivent dans de petites colonies comportant rarement plus d'une centaine d'individus et établissent leur territoire d'exploration autour de leur nid en le partitionnant en sites de chasse. Chaque fourmi s'attribue des sites particuliers qu'elle va explorer aléatoirement en mémorisant toutefois préférentiellement ceux sur lesquels elle a pu rencontrer un certain succès. Au fur et à mesure de leur sortie, les ouvrières chargées de récolter de la nourriture s'éloignent de plus en plus du nid couvrant petit à petit une grande partie de leur espace de recherche. Plus généralement, trois règles décrivent le comportement de ces fourmis:

- La découverte d'une proie entraîne toujours le retour sur le même site de chasse lors de la sortie suivante,
- La découverte d'une proie pèse sur la décision de sortie des fourrageuses en réduisant l'intervalle de temps passé au nid,
- Les fourrageuses semblent apprendre progressivement une association entre une direction de recherche opposée au nid et l'augmentation de la probabilité de succès.

D'autres contraintes pouvant avoir un impact dans un algorithme d'optimisation peuvent être mentionnées. Notamment, du fait de la fragilité du nid, des déménagements fréquents et réguliers ont lieu. Les sites choisis dépendent de la qualité des terrains découverts par des fourmis éclaireuses. On peut remarquer qu'un système de repérage par phéromone pourrait nuire à cette stratégie en trompant les fourmis incapables de distinguer des traces laissées menant à un nid maintenant abandonné. Afin d'éviter ce genre d'incident, les fourmis utilisent une mémoire visuelle qui les aide à oublier toutes les connaissances qu'elles ont accumulées à chaque déplacement du nid et leur permet de reconstruire un réseau de nouveaux sites de chasse autour du nouveau nid.

Dans [38], l'auteur a modélisé le comportement des fourmis *Pachycondyla apicalis* et a repris ces principes pour résoudre des problèmes d'optimisation numériques (par exemple recherche d'un minimum global) et combinatoire (problème du PVC). Ce modèle a aussi été utilisé dans [39] par F.Picarougne pour la recherche de document sur internet.

III.3.3 Le rassemblement d'objets

Dans la réalité les fourmis ont à résoudre des problèmes de classification. L'exemple du tri du couvain est le premier à avoir inspiré les chercheurs dans ce domaine.

Quand leur nid est dérangé les fourmis trient spontanément leurs œufs par stades d'évolution : elles font de petits tas d'œufs, de larves et de nymphes. Elles sont

capables de distinguer les différents types d'œufs grâce aux phéromones. En effet un œuf de fourmi émet des phéromones différentes suivant son stade d'évolution. Par contre, la "myopie" des fourmis leur interdit d'avoir une vue d'ensemble du couvain. On peut alors supposer qu'une fourmi dispose d'une mémoire à court terme des œufs qu'elle a rencontré durant ces dernières secondes de déplacement.

Ce modèle biologique a donné naissance à plusieurs applications nécessitant un algorithme de classification. Dans [40], E. D. Lumer et B. Faieta sont les premiers à avoir adapté ce modèle à un problème classique de classification : les données sont initialement réparties aléatoirement sur une grille 2D. Chaque fourmi est située dans une case de cette grille et ne perçoit que les données situées dans son voisinage. Une donnée sur la grille est ramassée avec une probabilité d'autant plus grande qu'elle est peu similaire aux données voisines. De la même manière, une donnée portée par une fourmi est plus facilement déposée dans une région comportant des données qui lui sont similaires.

III.4 Classification par colonies de fourmis

Le problème du regroupement d'objets ou le clustering est très présent dans la nature et de nombreuses espèces ont développé des comportements souvent sociaux pour le résoudre comme c'est le cas du tri du couvain chez les fourmis qui sont capables d'organiser divers éléments du couvain tels que les œufs, les larves et les nymphes. Le modèle de règles sur lequel elles se basent est simple :

- Lorsqu'une fourmi rencontre un élément du couvain, la probabilité qu'elle s'en empare est d'autant plus grande que cet élément est isolé ;
- Lorsqu'une fourmi transporte un élément du couvain, elle le dépose avec une probabilité d'autant plus grande que la densité d'éléments du même type dans le voisinage, est grande.

Plusieurs algorithmes de clustering basés sur le modèle des fourmis, ont été développés tels que AntClass AntClust et AntTree.

III.4.1 Algorithme AntClass

Cet algorithme basé sur le modèle des fourmis, s'inspire de l'aptitude des fourmis à effectuer des tris collectifs. Elles sont en effet capables d'organiser divers éléments à l'intérieur du couvain. Son principe est le suivant :

Les fourmis se déplacent sur une grille à deux dimensions et peuvent transporter des objets pour faire des tas d'objets sur une case et ainsi former des clusters. Au départ, tous les objets sont éparpillés aléatoirement sur la grille et chaque fourmi est positionnée de manière aléatoire et uniforme. Le déplacement de chaque fourmi n'est pas complètement aléatoire. En effet, après un déplacement dans une direction, elle a une probabilité P de continuer dans cette direction sinon elle en choisit une autre au hasard. Quand une fourmi ne transporte aucun objet, elle cherche à en ramasser un. Lorsqu'un tas d'objets a été trouvé, elle décide en fonction du nombre d'objets du tas, des dissimilarités entre les objets et de probabilités, si elle prend ou non, un objet et lequel. Quand une fourmi transporte un objet, elle décide

Classification et Clustering par les Fourmis Artificielles

en fonction de la taille du tas, de la similarité entre son objet et les objets du tas, si elle dépose ou non, son objet sur la case. De plus, chaque fourmi a une mémoire dans le but d'accélérer la classification. L'algorithme s'arrête quand le nombre d'itérations a été atteint. Généralement cet algorithme est combiné à l'algorithme de type K-means pour accroître ses performances par la correction des erreurs de classification.

Le déploiement de cet algorithme s'effectue comme suit :

Pour chaque itération
Faire Pour chaque fourmi F
 Faire déplacer F
 Si (F ne porte pas d'objet)
 Alors Scruter les cases adjacentes et faire, si possible, une des actions suivantes :
 1) Prendre un objet seul
 2) Prendre un des objets d'un tas de 2
 3) Prendre l'objet le plus dissimilaire d'un tas de plus de 2 objets
 Sinon /* F porte un objet O */
 Scruter les cases adjacentes et faire, si possible, une des actions suivantes :
 4) Poser O sur une case vide
 5) Poser O sur un objet seul
 6) Poser O sur un tas
FinPour
FinPour

Algorithme III.1 Algorithme AntClass.

Une des applications de cet algorithme est la formation de communautés d'utilisateurs basée sur le rapprochement de leurs profils relativement à un critère particulier. Le critère peut être basé sur le contenu ou sur l'évaluation des utilisateurs des contenus. L'algorithme AntClass est appliqué pour construire ce genre de communautés. Il répartit à cet effet, une population à classer sur un plan en reflétant la proximité entre les individus qu'ils la composent.

III.4.2 Algorithme AntClust

Le principe de cet algorithme s'inspire de la capacité de chaque individu à reconnaître l'appartenance ou la non appartenance d'un autre individu au nid de fourmis. Chaque fourmi dispose ainsi d'une carte d'identité olfactive qui lui permet d'identifier sa colonie et c'est ainsi qu'elle protège son nid de toute intrusion ennemie. AntClust est un algorithme relativement simple à mettre en place et contrairement à l'algorithme du type K-Means, il ne nécessite aucune initialisation.

Une des applications intéressante de cet algorithme est l'évaluation de la manière dont un site Web est visité (fréquence, type d'utilisateur). Cette évaluation

permet de prévoir les pages les plus susceptibles d'être visitées dans un proche avenir et donc de les charger en mémoire dans le serveur Web pour augmenter la rapidité de connexion. Pour cela une « session Web » reflétant l'activité de l'utilisateur sur le site pendant un laps de temps, est analysée et évaluée. Cette évaluation permet par application de l'algorithme de classification AntClust, de faire émerger des groupes d'utilisateurs ayant des habitudes de navigation similaires.

III.4.3 Algorithme AntTree

Cet algorithme s'inspire de la capacité qu'ont les fourmis de former des structures vivantes en s'auto-assemblant et dont la finalité est de découvrir une nouvelle façon de trouver une classification arborescente non-supervisée des données d'un ensemble [33].

Notre travail est basé sur cet algorithme qui va être expliqué dans les chapitres suivants.

III.5 Domaines d'application

Il existe d'autres applications des colonies de fourmis telles que les algorithmes d'optimisation proposés par Marco Dorigo, pour la résolution des problèmes d'optimisation combinatoire telle que le TSP (Travelling Salesman Problem). Ces algorithmes s'inspirent du comportement des fourmis réelles qui sont des insectes vivants dans les colonies et sont capables de trouver le plus court chemin entre la fourmière et la source de nourriture. En se déplaçant, elles déposent une substance chimique appelée le « phéromone » avec lequel elles forment une piste que les fourmis de la colonie peuvent sentir et suivre selon que la piste en question est la plus fréquentée ou non. La piste de phéromone une fois empruntée et tracée par la colonie de fourmis donne le plus court chemin, quand plusieurs chemins sont disponibles entre la fourmière et la source de nourriture.

Les algorithmes de colonie de fourmi ont été appliqués dans de nombreux domaines cités comme suite [41] :

- Applications au problème symétrique et asymétrique de voyageur de commerce.
- Applications au problème d'ordonnancement séquentiel.
- Applications aux problèmes d'affectation quadratique.
- Applications aux problèmes de tournées des véhicules.
- Applications aux problèmes d'établissement d'horaires.
- Applications aux problèmes de coloration de graphe.
- Applications aux problèmes de partitionnement.
- Applications aux réseaux de télécommunications.
- Implémentations parallèles.
- D'autres applications.

III.6 Principes d'auto-assemblage chez les insectes sociaux

On observe chez plusieurs espèces d'insectes sociaux un comportement consistant à accrocher des individus entre eux de manière à construire des structures vivantes. Dans cette optique est défini l'auto-assemblage comme l'assemblage de plus de deux éléments semblables par des dispositifs d'accrochage qui leurs sont propre. On peut ainsi définir l'auto-assemblage en établissant plusieurs règles :

- Existence de contacts physiques entre individus, soit transitoires, soit permanents,
- Ces contacts sont soumis à des contraintes physiques qui peuvent trouver leurs origines dans la physiologie des animaux et/ou dans le milieu,
- La structure mise en place par l'auto-assemblage des individus est fortement dépendante des individus qui la composent, c'est à dire que la suppression d'un ou de quelques individus, conduit à la disparition de cette structure.

Les auteurs regroupent plusieurs phénomènes d'auto-assemblage que l'on peut observer chez les animaux, en particulier les insectes sociaux. Nous en citons quelques-uns ci-dessous.

La formation de chaînes et de ponts chez les fourmis fileuses *Oecophylla* permettant le franchissement d'obstacles naturels (voir Figure III.2 a)) ou servant à la construction de nids (voir Figure III.2 b)) et la formation de grappes d'ouvrières suspendues dans le vide par les fourmis de type *Linepithema humile* (voir Figure III.2 c) et d)).



Figure III.2 Réaliser dans [42]. a) construction de chaînes et b) construction d'un nid par les fourmis *Oecophylla*, c) et d) construction de grappes de fourmis par les *Linepithema humile*.

Contrairement aux autres espèces de fourmis, les fourmis soldats ne vivent pas dans des terriers ou des nids mais dans des structures temporaires formées par leurs corps appelées «bivouac». En s'accrochant les unes aux autres elles forment

Classification et Clustering par les Fourmis Artificielles

ainsi plusieurs couches de fourmis. Le même comportement est observé chez les fourmis légionnaires *Eciton* n'ayant également pas de domicile fixe. La colonie se déplace quelques jours, établit un nid temporaire (un bivouac), y reste un certain temps et repart ensuite. Ce cycle migratoire se répète inlassablement et est lié aux activités de reproduction de la reine.

Pour établir un bivouac, les fourmis choisissent une cavité naturelle sous un vieux tronc ou entre les racines d'un arbre puis les ouvrières s'accrochent les unes aux autres et finissent par former une véritable masse vivante au centre de laquelle se cache la reine (voir Figure III.3). La reine peut alors se mettre à pondre en toute sécurité. L'arrivée de toutes ces nouvelles recrues est le signal que la colonie doit déclencher un nouveau départ. Les ouvrières tombent alors par grappes de la masse du bivouac et les fourmis se dispersent alors dans toutes les directions.



Figure III.3 Accrochage chez les fourmis du genre *Eciton*.

Lorsque leur nid est inondé, certaines espèces de fourmis construisent des radeaux en s'accrochant les unes aux autres, des amas d'ouvrières s'organisent ainsi en radeaux permettant le sauvetage d'une partie du couvain (voir Figure III.4 a)). Ce comportement apparaît comme une stratégie de survie contre les inondations durant les saisons pluvieuses.

Ce phénomène d'auto-assemblage trouve aussi des applications à la stratégie de prédation chez les abeilles japonaises *Apis cerana japonica* (voir Figure III.4 b)). Lorsqu'elles veulent attaquer un frelon venu violer leur nid, celles-ci forment une boule autour de la proie (le frelon), ce qui a pour conséquence d'augmenter la température interne de la boule. Les frelons ayant une température létale inférieure à celle des abeilles, cette action entraîne une mort certaine du frelon en épargnant les abeilles qui se trouvent à l'intérieur de la boule. La boule formée par les abeilles a eu l'effet d'un four à faire cuire le frelon.

Ce type de formation en essaim (voir Figure III.4 c)) est utilisé chez de nombreuses colonies d'abeilles (par exemple *Apis mellifera*) où l'accrochage est impliqué notamment dans la thermorégulation. Pour produire plus de chaleur les abeilles s'assemblent en formant un manteau dense d'abeilles immobiles et inversement la distance entre elles augmente pour laisser circuler l'air plus facilement de manière à rafraîchir le nid.

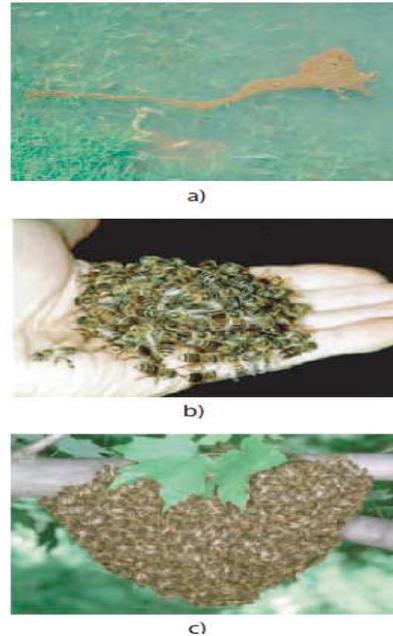


Figure III.4 a) colonie de fourmis construisant un radeau, b) rassemblement d'abeilles japonaises *Apis cerana japonica* en forme de boule, c) exemple d'essaim d'abeilles.

Toutes ces techniques impliquent la contribution de plusieurs individus à la réalisation d'une tâche complexe. Un individu unique est en effet incapable d'effectuer ces tâches en ne prenant en compte que ses capacités individuelles. Il paraît par conséquent très intéressant de s'inspirer de ces modèles de comportement collectifs pour résoudre des problèmes complexes difficilement résolubles par une approche globale.

III.7 Modèle d'auto-assemblage chez les fourmis réelles

Nous présentons dans cette section le modèle réel d'auto-assemblage qui nous a servi d'inspiration pour développer nos algorithmes de fourmis. Nous nous sommes ainsi basés sur l'étude réalisée sur les fourmis *Linepithema humile* et les *Oecophylla* par C.Sauwens et A.Lioni dans [43] et [44] durant leur thèse de doctorat.

- **Les fourmis *Linepithema humile*** : Originaire d'Amérique du sud, ces fourmis de petite taille (moins de 3mm pour les ouvrières) sont envahissantes et très opportunistes. Elles ont la capacité à former des grappes d'ouvrières suspendues dans le vide mais ce comportement reste associé à une fonctionnalité méconnue jusqu'à ce jour. Cette capacité n'a été mise en évidence que récemment à partir de dispositifs expérimentaux: l'accrochage interindividuel y est réalisé au moyen des tarse. De plus, des amas (gouttes) de fourmis peuvent se décrocher de cette grappe et chuter (voir Figure III.2 c) et d)).
- **Les fourmis *Oecophylla longinoda*** : C'est une colonie de fourmis très ancienne. On distingue deux espèces, les unes habitant les forêts de l'Afrique tropicale : les *Oecophylla longinoda*, et les autres se trouvant sur une partie de

Classification et Clustering par les Fourmis Artificielles

l'océan et du continent asiatique : les *Oecophylla smaragdina*. Elles élaborent deux types de chaînes qui diffèrent selon la fonctionnalité et le mode d'accrochage. Il y a d'une part les chaînes dites de passage permettant le franchissement d'un espace vide. L'accrochage est alors réalisé par leurs tarsi. D'autre part, les chaînes de construction du nid permettent le rapprochement de feuilles : l'accrochage entre individus est ici réalisé en refermant les mandibules sur le pétiole de l'individu suivant. Pour les deux structures, on observe après un certain temps la résorption de la chaîne.

Une série d'expériences effectuées sur les espèces *Oecophylla* et *Linepithema humile* ont été réalisées par un groupe de chercheurs dirigé par Guy Théraulaz au laboratoire d'éthologie et de modélisation des comportements collectifs à Toulouse. À l'issue de ces expériences, quatre phases communes aux deux types de fourmis ont été observées durant la construction de grappe et de chaînes :

- Une phase d'exploration de la structure (temps de latence),
- Une phase d'accrochage interindividuel de la fourmi "croissance de la structure" : on observe une augmentation du flux d'entrée des fourmis, début de construction d'une chaîne par intégration successive de fourmis et croissance de la grappe par accrochages interindividuels,
- Une période relativement stable, peu de fourmis se déplacent sur la chaîne ou sur la grappe,
- Une phase où les fourmis quittent la structure "le décrochage" : on observe le phénomène de résorption de la chaîne pour les *Oecophylla* et un début de chute de goutte de fourmis chez les *Linepithema humile*. Selon les biologistes la pesanteur est l'élément principal qui intervient dans le phénomène de décrochage des gouttes de fourmis : l'effet du poids entraînant la rupture des liens interindividuels existant entre les fourmis.

Le phénomène de croissance des chaînes chez les fourmis *Oecophylla* et la construction de grappes chez les *Linepithema humile* fait appel à des comportements de base tout à fait similaires : des ouvrières gagnent la structure collective (chaîne ou grappe), et y séjournent un certain temps au cours duquel elles se déplacent, éventuellement s'immobilisent avant de quitter cette structure.

À partir de ces principes nous avons développé un modèle de règles comportementales pour des fourmis artificielles que nous appliquons dans ce travail à la classification de données. Ce modèle respecte les propriétés suivantes :

- Les fourmis construisent ce type de structure à partir d'un point de départ sur un support fixe (tige, feuille, ...),
- Des fourmis peuvent se déplacer sur la structure en cours de construction, comme s'il s'agissait d'une structure fixe ; elles sont en déplacement,
- Les fourmis peuvent se fixer a priori n'importe où sur la structure puisque elles peuvent atteindre tous les points qui la composent. Néanmoins, pour la formation de chaînes par exemple, les fourmis vont se fixer prioritairement sur l'extrémité de la chaîne, car elles peuvent être notamment contraintes par la pesanteur mais aussi attirées par l'objet à atteindre,

Classification et Clustering par les Fourmis Artificielles

- La majorité des fourmis qui composent la structure peuvent être bloquées sans aucune possibilité de déplacement. Par exemple, dans le cas d'une chaîne de fourmis, cela correspond aux fourmis placées en milieu de chaîne,
- Un certain nombre de fourmis (généralement beaucoup plus réduit que celui du point précédent) sont reliées à la structure mais par un lien qu'elles maintiennent par elles-mêmes, et peuvent donc se détacher de la structure comme elles le veulent. A nouveau pour les chaînes de fourmis, cela correspond aux fourmis placées aux extrémités de la chaîne,
- On observe des phénomènes de croissance mais aussi de décroissance de la structure (des fourmis qui se décrochent de la structure).

On peut donc dégager un certain nombre de remarques de cette méthode de construction de structure :

- Il n'y a pas de contrôle centralisé, c'est-à-dire un plan de construction global qui indiquerait à chaque fourmi où aller se placer et à quel moment le faire,
- Les fourmis ne communiquent pas directement entre elles pour construire la structure,
- Les fourmis communiquent indirectement entre elles par la forme prise par la structure construite qui influence leur comportement (notion de stigmergie),
- Vraisemblablement, les fourmis ne sont sensibles qu'à la forme de la structure perçue localement, et n'ont pas de vue d'ensemble de la structure.

III.8 Conclusion

Les techniques de classification sont appliquées dans presque tous les domaines où il existe des données à regrouper ou à classer, toutefois selon la spécificité de ces données et du domaine sur lequel elles vont être appliquées, une technique s'adapte mieux que les autres et délivre par conséquent une qualité de classification meilleure. Appliquées à notre étude, ces techniques permettent de regrouper et de classer les utilisateurs d'un site précis selon la similarité des pages visitées et du parcours suivi par chaque internaute.

Chapitre IV :

L'algorithme AntTree_{Stoch} et son extension AntTree_{Sans-Seuil}

IV.1 Introduction

Nous proposons dans ce chapitre les algorithmes de classification s'inspirant du comportement des fourmis réelles et plus généralement des systèmes biologiques utilisés dans [32]. Il s'agit de modéliser la manière dont les fourmis forment des structures vivantes et d'utiliser ce comportement pour organiser les données selon un arbre qui se construit de manière distribuée.

Le principe utilisé est le suivant : intuitivement, chaque fourmi/donnée est située au départ sur un support fixe (racine de l'arbre qui va être construit). Le comportement des fourmis consiste alors soit à se déplacer soit à s'accrocher à la structure pour la prolonger et permettre à d'autres fourmis de venir et de s'accrocher à leur tour. Ce comportement est déterminé notamment par la similarité entre les données et la structure locale de l'arbre.

Ce chapitre se rapporte donc à la présentation de la première version de l'algorithme de classification hiérarchique non supervisée AntTree_{Stoch}. Nous détaillerons l'algorithme de construction d'arbres et les règles locales de comportement des fourmis artificielles à partir de l'étude qui a été faite sur le comportement d'auto-assemblage chez les fourmis. Ensuite nous présenterons son extension appelé AntTree_{Sans-Seuil} qui n'utilise aucun seuil et plus généralement aucun paramètre. Le seuil nécessaire va dépendre uniquement de la structure locale de l'arbre et il utilise aussi le principe de décrochage observé chez les fourmis réelles.

IV.2 L'algorithme AntTree_{Stoch}

Le principe de cet algorithme est de construire un arbre dont les nœuds sont connus a priori et représentent les données, et les arcs restent à déterminer. Notons que l'arbre construit contiendra des données à tous les nœuds. A la fin de la construction de l'arbre, nous obtenons une classification des données, tel que l'arbre sera représentées en plusieurs sous-arbres, chaque sous-arbre est une classe ou une catégorie de données similaires entre eux.

Chaque nœud représente une donnée à classer. Nous considérons que nous disposons d'une mesure de similarité $\text{Sim}(i, j)$ qui prend comme paramètre un couple de nœuds i et j , et qui retourne une valeur comprise entre 0 (i et j sont totalement différents) et 1 (i et j sont identiques). Nous ne faisons pas d'autres hypothèses sur la représentation des données qui peuvent être de tous types de représentation (numériques, symboliques ou encore textuelles), il suffit que l'on puisse définir une mesure de similarité.

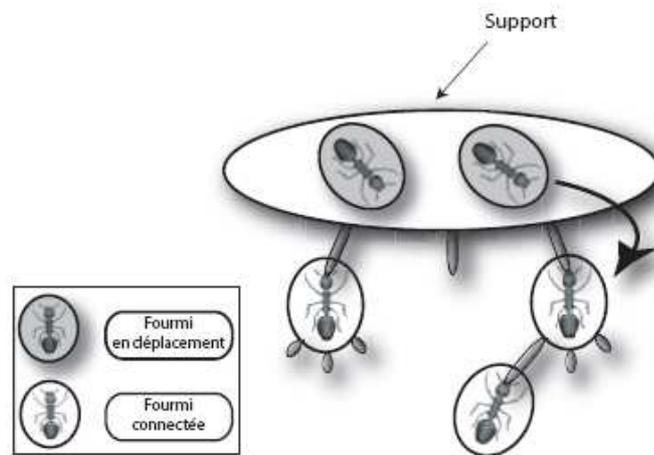


Figure IV.1 Construction de l'arbre par des fourmis : principe général. Les fourmis qui sont en déplacement sont représentées en gris et les fourmis connectées en blanc.

Le principe de l'algorithme AntTree_{Stoch} est le suivant :

Chaque fourmi f_i , $i \in [1, N]$ (N est le nombre de données initiales) représente un nœud de l'arbre à assembler, c'est-à-dire une donnée à classer et l'arbre représente la structure que les fourmis vont construire.

Partant d'un point de départ matérialisé par un nœud fictif f_0 qui représente le support sur lequel va être construit le graphe, les fourmis vont progressivement se fixer sur ce point initial, puis successivement sur les fourmis fixées à ce point initial, et ainsi de suite jusqu'à ce que toutes les fourmis soient rattachées à la structure (voir figure IV.1).

Tous ces déplacements et ces accrochages dépendent de la valeur retournée par la fonction de similarité $\text{Sim}(i, j)$ entre les données, et du voisinage local de la fourmi en déplacement. Pour chaque fourmi f_i nous allons donc définir les notions suivantes :

- Un lien sortant est un lien que f_i peut maintenir avec ses tarse, mandibules, etc, vers une autre fourmi. Pour construire un arbre, un seul lien sortant est autorisé,
- Les liens entrants de f_i sont les liens que les autres fourmis maintiennent vers f_i , ces liens peuvent être les pattes de la fourmi. Nous les avons limités à un nombre maximum appelé L_{\max} pour forcer l'algorithme à créer plusieurs niveaux hiérarchiques et pour limiter le nombre maximum de sous-catégories pouvant apparaître sous chaque nœud,
- Une information d_i représentée par chaque fourmi et qui représente la donnée à classer (i -ème donnée de la base),
- Un seuil de similarité $S_{\text{Sim}(f_i)}$ et un seuil de dissimilarité $S_{\text{Dissim}(f_i)}$ local à chaque fourmi f_i (et mis à jour par f_i) permettant de définir si la donnée d_i est suffisamment similaire

ou bien suffisamment dissimilaire à une autre donnée représentée par une autre fourmi,

- Considérons également que le voisinage autour d'une fourmi f_k connectée est défini de la manière suivante : f_k et f_j sont voisines s'il existe un lien de f_k vers f_j ou un lien de f_j vers f_k .

Lors de la construction de la structure (figure IV.1 et IV.2), chacune des fourmis sera soit :

- En déplacement sur l'arbre. Dans ce cas, nous considérons que chaque fourmi f_i est positionnée sur le support f_0 ou sur une autre fourmi f_{pos} mais qu'elle n'est pas assemblée/connectée à la structure. f_i est donc totalement libre de se déplacer sur le support (sa position initiale) ou bien vers une autre fourmi. Les déplacements auront lieu aléatoirement vers les voisins du nœud (f_{pos}) sur lequel la fourmi se trouve (voir figure IV.2),
- Assemblée à une extrémité ou au milieu de l'arbre. Elle ne pourra plus se dégager de la structure.

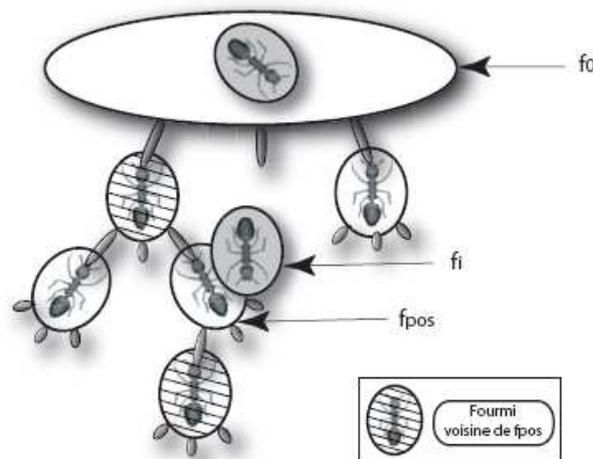


Figure IV.2 Calcul du voisinage d'une fourmi. Les voisines d'une fourmi f_i en déplacement sont hachurées.

IV.2.1 Description de l'algorithme

L'algorithme $AntTree_{Stoch}$ fonctionne de la manière suivante (algorithme IV.1) : À chaque itération, une fourmi f_i extraite de la liste des fourmis triées va se connecter ou bien se déplacer au mieux en fonction de la similarité avec son voisinage. Tant qu'il existe une fourmi f_i en déplacement, on simule une action pour f_i en fonction de sa position, sur le support (f_0) ou sur une autre fourmi (f_{pos}).

-
- (1) Initialement toutes les fourmis sont sur le support et leurs seuils de similarité et de dissimilarité sont respectivement initialisés à 1 et à 0 (voir autres méthodes d'initialisation dans la section 3.3.1)
 - (2) **tantque** il existe une fourmi f_i non connectée **faire**
 - (3) **si** sa position est le support **alors**
 - (4) *Cas support*
 - (5) **sinon**
 - (6) /* f_i est sur une autre fourmi */
 - (7) *Cas fourmi*
 - (8) **finsi**
 - (9) **fantantque**
-

Algorithme IV.1 Algorithme principal de construction d'arbres par des fourmis artificielles.

Initialement, toutes les fourmis sont en déplacement sur le support. Leurs seuils de similarité et de dissimilarité sont initialisés respectivement à 1 et à 0. La première fourmi est connectée directement au support. Ensuite pour chaque fourmi f_i , on distingue 2 cas :

- f_i est sur le support (figure 4.3) : si f_i est suffisamment similaire à f_+ (suivant son seuil de similarité), où f_+ est la fourmi connectée au support la plus similaire à f_i , on déplace f_i vers celle-ci de manière à la placer dans la même branche (classe) que f_+ . Sinon (f_i n'est pas assez similaire à f_+), si f_i est suffisamment dissimilaire à f_+ et qu'il existe un lien entrant libre sur le support (le nombre de liens entrants du support est inférieur à L_{\max}) on connecte f_i au support, ce qui signifie que l'on vient de construire une nouvelle classe. En effet dans notre algorithme on considérera que chaque sous-arbre constitue une classe trouvée. S'il n'existe pas de lien entrant libre sur le support, f_i est repositionnée sur f_+ en diminuant son seuil de similarité pour rendre f_i plus tolérante étant donné qu'initialement elle était dissimilaire à f_+ . Enfin, si f_i n'est ni suffisamment similaire, ni suffisamment dissimilaire, on met à jour ses seuils en diminuant le seuil de similarité et en augmentant le seuil de dissimilarité pour la rendre plus tolérante et augmenter ses chances de se connecter à la prochaine itération la concernant. De plus, entretemps d'autres fourmis auront eu la possibilité de modifier l'arbre.
- f_i est sur une autre fourmi notée f_{pos} (figure IV.4) : s'il existe un lien entrant libre sur f_{pos} (le nombre de liens entrants de f_{pos} est inférieur à L_{\max}) et que f_i est suffisamment similaire à f_{pos} et suffisamment dissimilaire aux voisines de f_{pos} , on connecte f_i à f_{pos} . Dans ce cas, f_i appartient à la même classe que f_{pos} et sa dissimilarité avec les "filles" de f_{pos} lui permettra de représenter une sous-classe la plus dissimilaire possible aux autres sous-classes de f_{pos} . S'il n'existe pas de lien entrant libre sur f_{pos} le choix qui nous reste est de positionner f_i aléatoirement sur une des voisines de f_{pos} , dans ce cas on ne met pas à jour les seuils, ces derniers ont permis de bien localiser f_i dans l'arbre et en les modifiant on risquerait de mal classer f_i . Maintenant, si f_i est suffisamment similaire à f_{pos} mais pas suffisamment dissimilaire aux voisines de f_{pos} , on positionne f_i aléatoirement sur une des voisines de f_{pos} et on met à jour les seuils (de la même façon que dans le cas support) pour que f_i soit plus tolérante à la prochaine itération

la concernant. f_i se déplace dans ce cas dans l'arbre et pourra trouver un meilleur endroit pour se fixer. Enfin si f_i n'est pas suffisamment similaire à f_{pos} , on change de chemin en positionnant f_i aléatoirement sur une des voisines de f_{pos} . L'algorithme se termine lorsque toutes les fourmis sont connectées.

```

(1)  si aucune fourmi n'est connectée au support  $f_0$  alors
(2)    connecter  $f_i$  à  $f_0$  /* une nouvelle classe */
(3)  sinon
(4)    /*  $f_+$  la fourmi connectée au support la plus similaire à  $f_i$  */
(5)    si  $Sim(f_i, f_+) \geq S_{Sim}(f_i)$  alors
(6)      /*  $f_i$  est suffisamment similaire à  $f_+$  */
(7)      déplacer  $f_i$  vers  $f_+$ 
(8)    sinon
(9)      si  $Sim(f_i, f_+) < S_{Dissim}(f_i)$  alors
(10)       /*  $f_i$  est suffisamment dissimilaire à  $f_+$  */
(11)       si place libre sur le support alors
(12)         /* nombre de liens entrants  $< L_{max}$  */
(13)         connecter  $f_i$  au support  $f_0$  /* une nouvelle classe */
(14)       sinon
(15)         diminuer  $S_{Sim}(f_i)$  et déplacer  $f_i$  vers  $f_+$ 
(16)       finsi
(17)     sinon
(18)       diminuer  $S_{Sim}(f_i)$  et augmenter  $S_{Dissim}(f_i)$   $f_i$  /* rendre  $f_i$  plus
(19)       tolérante */
(19)     finsi
(20)   finsi
(21) finsi

```

Algorithme IV.2 Cas support : algorithme simulant le comportement d'une fourmi f_i en déplacement sur le support f_0 .

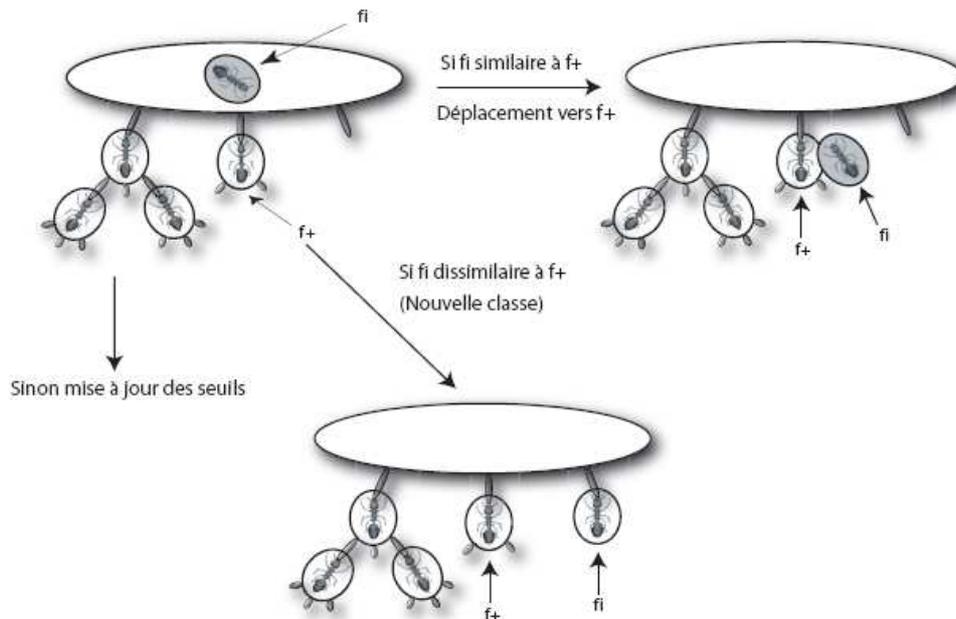


Figure IV.3 Règles de comportement de f_i si elle est positionnée sur le support (voir algorithme IV.2).

-
- (1) /* f_{pos} la fourmi sur laquelle f_i est en déplacement */
 - (2) **si** $Sim(f_i, f_{pos}) \geq S_{Sim}(f_i)$ **alors**
 - (3) **si** $Sim(f_i, f_+) < S_{Dissim}(f_i)$ **alors**
 - (4) /* f_+ la fourmi voisine de f_{pos} la plus similaire à f_i */
 - (5) **si** place libre sur f_{pos} **alors**
 - (6) /* nombre de liens entrants $< L_{max}$ */ connecter f_i à f_{pos}
 - (7) **sinon**
 - (8) déplacer f_i aléatoirement vers f_k /* f_k $k=1..n$, tous les éléments (fourmis ou support) connectés à f_{pos} */
 - (9) **finsi**
 - (10) **sinon**
 - (11) diminuer $S_{Dissim}(f_i)$, augmenter $S_{Sim}(f_i)$ et déplacer f_i aléatoirement vers f_k
 - (12) **finsi**
 - (13) **sinon**
 - (14) déplacer f_i aléatoirement vers f_k
 - (15) **finsi**
-

Algorithme IV.3 Cas fourmis : algorithme simulant le comportement d'une fourmi f_i en déplacement sur une autre fourmi f_{pos} connectée à la structure.

Remarques :

Dans l'algorithme on peut remarquer que, dans le cas du support, la fourmi f_i ne se déplace jamais de manière aléatoire. Si f_i n'arrive pas à se connecter on la positionne sur f_+ (la fourmi connectée au support la plus similaire à f_i). En utilisant ce comportement on augmente la probabilité qu'une fourmi f_i se déplace dès sa première simulation vers le sous-arbre qui contient le plus de fourmis similaires à elle.

Ainsi f_i pourra se déplacer à l'intérieur de ce sous-arbre suivant les règles stochastiques définies précédemment afin de trouver sa meilleure position.

On peut également remarquer que la mise à jour des seuils de similarité et de dissimilarité est locale à chaque fourmi. L'adaptation de ces seuils aurait pu être globale pour toutes les fourmis (à chaque fois qu'une fourmi n'arrive pas à se connecter elle met à jour les seuils). Mais une méthode locale a l'avantage de permettre à chaque fourmi d'ajuster ses seuils en fonction du résultat de ses actions et de la donnée qu'elle représente. Ainsi une fourmi peu similaire aux autres (réciproquement très similaire) aura un seuil adapté à sa situation locale.

La terminaison de l'algorithme est assurée par le fait que le seuil de similarité diminue et le seuil de dissimilarité augmente, jusqu'à atteindre une valeur telle que toutes les fourmis pourront se connecter. Mais pour assurer cette convergence il est nécessaire d'introduire une notion supplémentaire. En effet, soit le cas illustré par la figure IV.5. Si f_i est suffisamment similaire et suffisamment dissimilaire à f_{pos} et que f_{pos} n'a pas assez de liens pour connecter f_i ($L_{Max} = 3$) alors on déplace f_i aléatoirement vers les voisines de f_{pos} (f_1, f_2, f_3 ou le support) sans modifier ses seuils. Supposons maintenant que f_i ne soit pas assez similaire au voisinage de f_{pos} (par exemple pas assez similaire aux fourmis f_1, f_2, f_3 pour se déplacer vers elles et pas suffisamment dissimilaire aux fourmis connectées au support pour créer une nouvelle classe), f_i se déplacera en boucle entre f_{pos} et les fourmis (f_1, f_2, f_3) sans jamais pouvoir se connecter puisqu'elle ne modifie pas ses seuils.

On a cherché, par conséquent, à détecter ce bouclage en vérifiant qu'une fourmi ne passe pas un trop grand nombre de fois sur la même position sans mettre à jour ses seuils. À défaut, on impose à la fourmi concernée de modifier ses seuils, ce qui assurera la terminaison de l'algorithme.

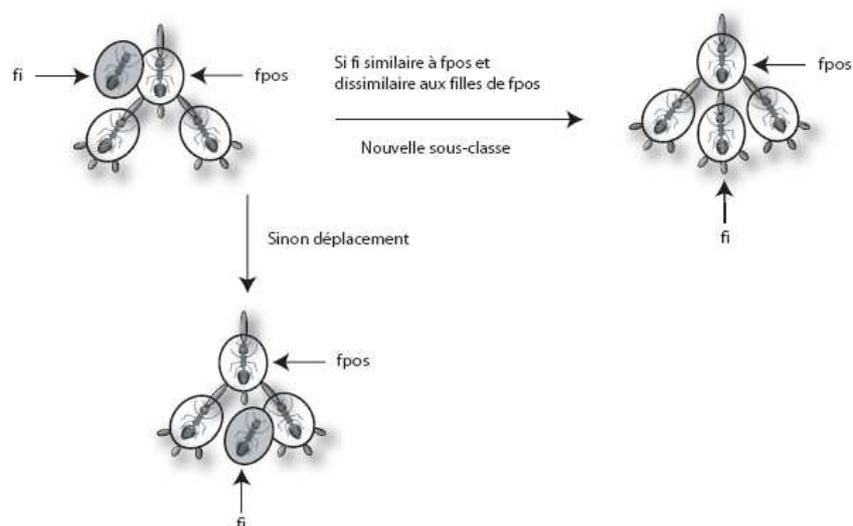


Figure IV.4 Règles de comportement de f_i si elle est positionnée sur une fourmi f_{pos} (voir algorithme IV.3).

IV.2.2 Propriétés de l'algorithme

La conception d'AntTree_{Stoch} permet de définir plusieurs propriétés. Tout d'abord, comme cela est représenté sur la figure IV.6, les sous-arbres placés directement sous le support (f_0) constituent la classification "plane" trouvée par AntTree_{Stoch}, chaque sous arbre correspondant à une classe constituée de toutes les données présentes dans ce sous arbre.

De plus, AntTree_{Stoch} répond aux propriétés visées pour une bonne classification des visiteurs dans un site web, c.-à-d.: chaque sous-arbre A représente les pages visitées par un seul utilisateur, tel que les pages représentent toutes les fourmis de A. Soit f_i la fourmi qui est à la racine d'un sous-arbre A.

Nous souhaitons que

1) f_i soit représentative de cette catégorie, c.-à-d. la première page visitée par cette utilisateur (les fourmis placées dans A sont les plus similaires possible à f_i), donc on peut aller de cette première page aux autres pages fille.

2) les fourmis filles de f_i qui représentent des sous-pages soient les plus dissimilaires possibles entre elles. C.-à-d., on ne peut pas passer d'une fourmi fille qui représente une page à une autre fourmi fille qui représente une autre page.

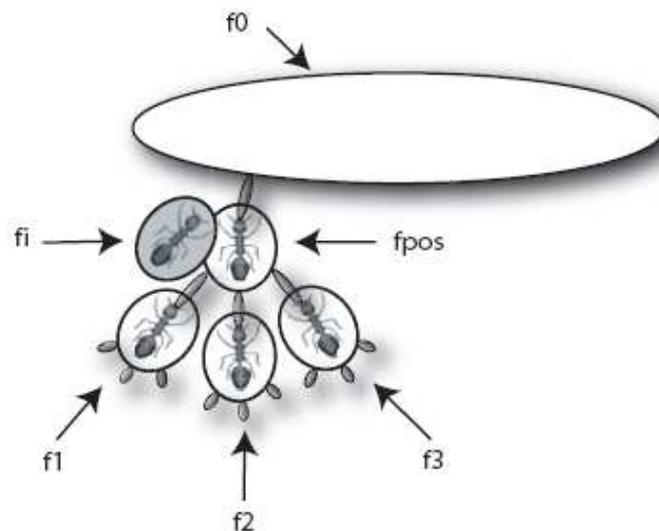


Figure IV.5 Effet de boucle dans AntTree_{Stoch}.

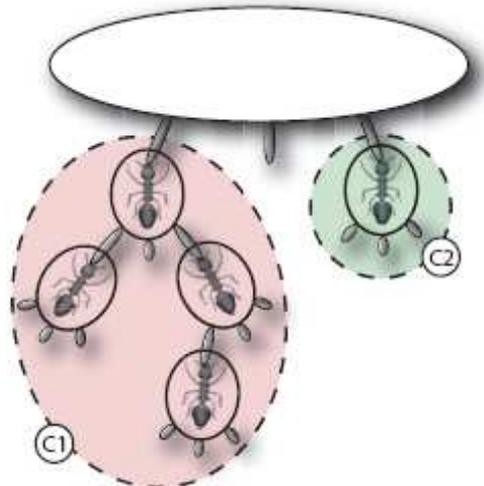


Figure IV.6 Transformation de l'arbre en classes (partitionnement), ici deux classes sont générées C1 et C2.

IV.2.3 Tri initial des données

La conception de l'algorithme fait que les résultats peuvent être influencés par la manière dont les données sont triées au départ. Les fourmis étant simulées les unes après les autres, l'ordre dans lequel elles apparaissent peut avoir une conséquence sur les résultats.

En effet, les fourmis sont initialement positionnées sur le support. Si l'une d'entre elles n'arrive pas à se connecter au support, elle se déplacera vers les fourmis qui se sont préalablement connectées au support. C'est pour cette raison que le choix de l'ordre de simulation des fourmis est important. En quelque sorte, dans notre algorithme, les premières fourmis connectées guident les autres pour trouver la bonne direction (le chemin le plus similaire).

On peut citer plusieurs stratégies de tri :

- Ordre aléatoire : les données sont triées dans un ordre aléatoire.
- Ordre croissant des similarités : on mesure pour chaque donnée sa similarité moyenne avec les autres données. On classe les données suivant les valeurs croissantes de cette similarité moyenne. On place donc en premier les fourmis les plus dissimilaires. Intuitivement, ces fourmis placées aux extrema dans l'espace des données seront peut être de bons points de départ pour la construction des classes, ou au contraire des données bruitées qui feront baisser les performances,
- Ordre décroissant des similarités : il s'agit de l'ordre inverse de celui donné précédemment.

Cette fois, les données les plus similaires aux autres seront placées en premier. Peut être que ces centres seront utiles pour démarrer une classe.

IV.3 Algorithme AntTree_{Sans-Seuil} avec décrochage de fourmis

Trouver le bon ajustement des paramètres est une tâche difficile pour la majorité des algorithmes et en particulier pour ceux issus de modèles inspirés de la biologie qui introduisent fréquemment de nombreux paramètres, [45].

Nous proposons dans cette section un nouvel algorithme AntTree_{Sans-Seuil} qui n'utilise aucun seuil et plus généralement aucun paramètre. Le seuil nécessaire va dépendre uniquement de la structure locale de l'arbre. AntTree_{Sans-Seuil} utilise aussi le principe de décrochage observé chez les fourmis réelles. Ce décrochage se traduit par le phénomène de résorption de chaînes observé chez les *Oecophylla longinoda* et la chute de gouttes de fourmis observé chez les *Linepithema humiles*.

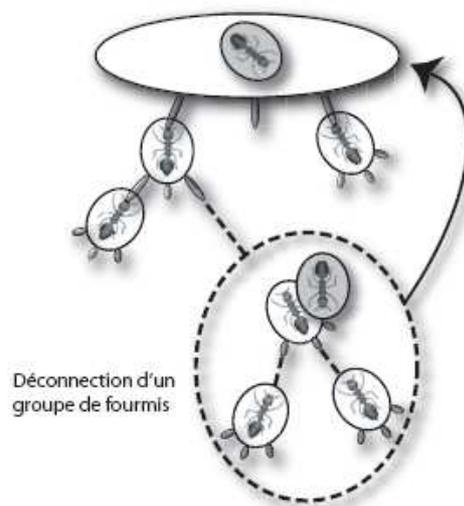


Figure IV.7 Décrochage de fourmis dans AntTree_{Sans-Seuil}.

Le fonctionnement de cet algorithme est globalement similaire à celui décrit pour AntTree_{Stoch}. La différence avec la version précédente se situe dans les déplacements déterministes, le décrochage, et le seuil calculé en fonction du voisinage local.

Initialement toutes les fourmis sont positionnées sur le support f_0 . A chaque itération, une fourmi f_i est choisie dans la liste des fourmis triée au départ. Cette fourmi cherchera alors à se connecter sur sa position courante (f_0 ou f_{pos}). Cette opération ne peut aboutir que dans le cas où elle est suffisamment dissimilaire à f_+ (la fourmi connectée à f_0 ou f_{pos} la plus similaire à f_i). Dans le cas contraire, f_i se déplacera de manière déterministe dans l'arbre suivant le chemin le plus similaire indiqué par f_+ . Le seuil permettant de prendre ces décisions ne va dépendre que du voisinage local.

Pour cette version de l'algorithme les fourmis utilisent les mêmes règles comportementales pour le cas support et le cas fourmi. Ainsi l'algorithme IV.4 regroupe les deux cas : support et fourmi (on considère que $f_{pos} = f_0$).

Notons $TDissim(f_0)$ (et respectivement $TDissim(f_{pos})$) la valeur de la similarité minimum observée entre chaque couple de fourmis connectées au support f_0 (respectivement à f_{pos}). Dans cette version le seuil $TDissim(f_i)$ dépend des valeurs de

similarité du voisinage V_{pos} perçu par f_i et est automatiquement déterminé par la structure de l'arbre lui-même. La valeur de $TDissim(f_i)$ est calculée par formule IV.1.

$$T_{Dissim} = Min \left(Sim_{j, k \in [1, n]}(f_j, f_k) \right) \dots \dots \dots (IV.1)$$

Où $f_1 \dots f_n$ représente l'ensemble de fourmis connectées à f_i .

-
- (1) **si** aucune ou une seule fourmi est connectée à f_{pos} **alors**
 - (2) connecter f_i à f_{pos} /* une nouvelle classe */
 - (3) **sinon**
 - (4) **si** 2 fourmis sont connectées à f_{pos} (uniquement le premier passage) **alors**
 - (5) **si** $Sim(f_i, f_+)$ < $T_{Dissim}(f_{pos})$ **alors**
 - (6) /* f_+ la fourmi connectée à f_{pos} la plus similaire à f_i */
 - (7) /* $T_{Dissim}(f_{pos}) = Sim(f_1, f_2)$ où f_1, f_2 représente les deux fourmis connectées à f_{pos} et $f_+ = f_1$ ou f_2 */
 - (8) - déconnecter f_+ de f_{pos} (et récursivement toutes les fourmis connectées à f_+)
 - (9) - replacer toutes ces fourmis déconnectées sur le support f_{pos}
 - (10) - connecter f_i à f_{pos}
 - (11) **sinon**
 - (12) déplacer f_i vers f_+
 - (13) **finsi**
 - (14) **sinon**
 - (15) /* plus de deux fourmis sont connectées au support ou deux fourmis sont connectées au support et on est dans le deuxième passage */
 - (16) - soit $T_{Dissim}(f_{pos})$ la valeur de la similarité minimum observée entre chaque couple de fourmis connectées au support f_{pos}
 - (17) **si** $Sim(f_i, f_+)$ < $T_{Dissim}(f_{pos})$ **alors**
 - (18) connecter f_i à f_{pos} /* f_i dissimilaire à f_+ */
 - (19) **sinon**
 - (20) déplacer f_i vers f_+
 - (21) **finsi**
 - (22) **finsi**
 - (23) **finsi**
-

Algorithme IV.4 AntTree(Sans-Seuil) (cas support et fourmis sont confondus).

Ainsi une fourmi f_i se connectera à f_{pos} (respectivement à f_0) si et seulement si cette action diminue la valeur de $TDissim(f_0)$ (respectivement de $TDissim(f_{pos})$). En d'autres termes, ceci consiste à comparer f_i à f_+ , dans le cas où les deux fourmis sont suffisamment dissimilaires entre elles ($Sim(f_i, f_+) < TDissim(f_0)$ ou $Sim(f_i, f_+) < TDissim(f_{pos})$) on connectera f_i à f_{pos} (respectivement à f_0) et dans le cas contraire, on déplacera f_i vers f_+ . Dans notre algorithme on peut voir que pour chaque nœud de l'arbre la

valeur de TDissim diminue. En effet à chaque fois qu'une fourmi arrive à se connecter elle baisse par définition la valeur du TDissim correspondant.

Il est évident que pour calculer la valeur de TDissim(f_0) (respectivement TDissim(f_{pos})) il faut avoir au minimum deux fourmis connectées à f_0 ou à f_{pos} (suivant la position de f_i). Par conséquent dans *AntTree*_{Sans-Seuil} les deux premières fourmis vont automatiquement se connecter à leur position sans aucun test de validité. Ceci entraîne une connection "abusive" de la seconde fourmi (sans test). De ce fait nous avons décidé qu'une fois une troisième fourmi connectée (pour cette troisième fourmi nous sommes sûre que le test de dissimilarité a bien réussi), nous déconnecterons une de ces deux premières fourmis. Plus précisément, sera décrochée celle qui est la plus similaire à la nouvelle fourmi connectée (voir point (7) dans l'algorithme IV.4).

Lorsqu'une fourmi f_i est déconnectée, toutes les fourmis connectées à elle se décrochent également. Toutes ces fourmis seront repositionnées sur le support (voir figure IV.6) pour de nouvelles simulations. Ensuite, elles vont chercher à se déplacer pour se connecter en utilisant les mêmes règles comportementales définies dans l'algorithme IV.4.

IV.4 Conclusion

Dans ce chapitre nous nous sommes inspirées de l'approche décrite dans [32] fondée sur les principes d'auto-assemblage observés chez une colonie de fourmis et son application au problème de classification hiérarchique non supervisée.

Au prochain chapitre, on appliquera le principe de la colonie de fourmis à la traçabilité de navigation sur un sens réel.

Chapitre V : Résultats du prétraitement des traces de navigation sur internet

V.1 Étude de cas

Dans cette étude de cas sur des données réelles, nous présentons la méthodologie que nous avons adoptée pour le prétraitement ainsi que les résultats de son application sur les fichiers Log du site web «www.taseek.com», un site de recrutement algérien. La figure suivante exhibe un aperçu de la page d'accueil du site en question.



Figure V.1 Aperçu de la page d'accueil du site web www.taseek.com.

Résultats du prétraitement des traces de navigation sur internet

Notre fichier log est constitué de l'ensemble de requêtes adressées au site du taseek pendant la période allant du 1 au 10 décembre 2010. Le fichier est composé de 60125 requêtes enregistrées suivant la norme ECLF (Extended Common Log Format), l'équivalent de 12837 ko. Pour chaque requête, nous disposons les champs suivants:

- la date de réalisation de la requête (date),
- l'heure à laquelle elle s'est produite (time),
- le nom de l'utilisateur authentifié ayant accédé au serveur (username),
- l'adresse IP,
- la méthode que tentait de réaliser le client (method),
- la requête que le client a essayé d'effectuer (url),
- la réponse du serveur (status),
- le nombre d'octets reçus par le serveur (bytes),
- l'agent utilisé par l'utilisateur (User- Agent),

Les variables qui prennent une valeur unique sont éliminées, à savoir username (tous les utilisateurs sont anonymes).

Le site analysé est accessible à l'adresse suivante : <http://www.taseek.com>

V.2 Implémentation

V.2.1 Fusionnement des fichiers log (T1)

Le tableau suivant indique la taille des différents fichiers log récupérés sur la période étudiée.

Période	Taille
01 au 02 décembre 2010	1313 Ko
02 au 05 décembre 2010	4715 Ko
05 au 07 décembre 2010	2930 Ko
07 au 10 décembre 2010	3879 Ko
Totale	12837 Ko

Tableau V.1 La taille des fichiers log a analysés.

V.2.2 Chargement du fichier Log et transformation en une table d'une BDD

Le fichier log est un fichier texte, appelé aussi journal des connexions. Généralement il est de la forme suivante :

Résultats du prétraitement des traces de navigation sur internet



Figure V.2 Fichier log a l'état brut.

Le fichier Log se transforme en une table composée de plusieurs colonnes. Chaque colonne correspond à un champ spécifique du fichier Log :

ip	DH	code	methode	url	referrer	ua
88.164.184.147	01/Dec/2010:0	200	GET	/	-	Mozilla/5.0 (Window
88.164.184.147	01/Dec/2010:0	200	GET	/images/intro_	-	Mozilla/5.0 (Window
88.164.184.147	01/Dec/2010:0	200	GET	/images/intro_	-	Mozilla/5.0 (Window
88.164.184.147	01/Dec/2010:0	200	GET	/images/intro_	-	Mozilla/5.0 (Window
88.164.184.147	01/Dec/2010:0	200	GET	/images/intro_	-	Mozilla/5.0 (Window
88.164.184.147	01/Dec/2010:0	200	GET	/images/go.sw	-	Mozilla/5.0 (Window
88.164.184.147	01/Dec/2010:0	200	GET	/images/intro_	-	Mozilla/5.0 (Window
88.164.184.147	01/Dec/2010:0	200	GET	/images/intro_	-	Mozilla/5.0 (Window
88.164.184.147	01/Dec/2010:0	200	GET	/images/intro_	-	Mozilla/5.0 (Window
88.164.184.147	01/Dec/2010:0	200	GET	/images/intro_	-	Mozilla/5.0 (Window
88.164.184.147	01/Dec/2010:0	200	GET	/images/intro_	-	Mozilla/5.0 (Window
88.164.184.147	01/Dec/2010:0	200	GET	/images/favicc	-	Mozilla/5.0 (Window
88.164.184.147	01/Dec/2010:0	200	GET	/accueil.html	-	Mozilla/5.0 (Window
88.164.184.147	01/Dec/2010:0	200	GET	/style.css	-	Mozilla/5.0 (Window
88.164.184.147	01/Dec/2010:0	200	GET	/style2.css	-	Mozilla/5.0 (Window
88.164.184.147	01/Dec/2010:0	200	GET	/script.js	-	Mozilla/5.0 (Window

Figure V.3 Base de données après import.

V.3 Nettoyage de données

Une fois le fichier log importé dans les espaces de stockage, les données concernant les pages possédant des requêtes non valides, des images ou de fenêtres publicitaires etc., n'apportent rien à l'analyse, elles seront donc filtrées et éliminées.

Résultats du prétraitement des traces de navigation sur internet

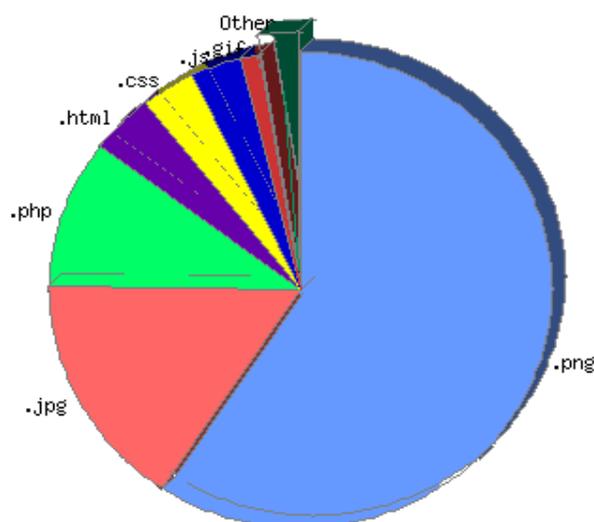


Figure V.4 Aperçu des différents contenus des pages de notre site.

Résultats du nettoyage

Le tableau suivant présente les résultats de suppression des requêtes inutile (N1) collectées pendant la période 1 au 10 Décembre 2011.

Types de requêtes		Nombres de requêtes	Pourcentage
Totale de requêtes		60123	100%
Code statu <>'200'		11199	18,62%
Méthode <>'Get '		5731	9,53%
Requêtes aux Images	.jpg	7709	12,82%
	.gif	905	01,50%
	.png	25657	42,67%
	.ico	414	0,68%
Scripts	.js	1479	2,45%
	.css	1552	2,58%
	.swf	495	0,82%
Total		55141	91,71%
Totale de requêtes après nettoyage		4982	8,28%

Tableau V.2 Résultat de suppression des requêtes inutiles (N1).

Résultats du prétraitement des traces de navigation sur internet

La nouvelle taille de la base (8,28% de la taille initiale) montre bien l'importance de l'étape du nettoyage. Cette étape a abouti à des fichiers nettoyés et structurés, prêts à l'analyse.

V.4 Identification des utilisateurs (sessions) et des visites (T3)

Sur les 4982 requêtes, nous avons supprimés quelques pages dynamiques dont le nombre de requêtes chute à 1714. Le nombre de sessions reconstruites est de 556 dont 358 sont uniques (i.e. proviennent d'une adresse IP qui n'apparaît qu'une fois dans le fichier log), donc nous obtenons 198 sessions. En appliquant l'algorithme d'identification des visites, le nombre de séquences de visites destinées à l'analyse est de 223 en utilisant un temps de latence égale à 30 minutes. Les séquences extraites reflètent les comportements fréquents des internautes connectés sur le site.

IP	UA	COUNT
41.102.160.63	"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.1; Win64; x64; Trident/4.0; NET CLR 2.0.50727; SLCC2; NET CLR 3.5.30729; NET CLR 3.0.30729; Media Center PC 6.0; NET4.0C; AskTbPF/5.9.1.14019)""	7
10.3.67.55	"Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)""	37
10.1.67.225	"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; NET CLR 1.1.4322; NET CLR 2.0.50727; InfoPath.2)""	19
41.221.25.124	"Mozilla/5.0 (Windows; U; Windows NT 6.0; fr; rv:1.9.2.13) Gecko/20101203 Firefox/3.6.13; ShopperReports""	2
10.3.65.11	"Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.9.2.13) Gecko/20101203 Firefox/3.6.13""	25
10.2.67.127	"Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; GTB6.6; .NET CLR 2.0.50727; NET CLR 1.1.4322; InfoPath.2)""	2
41.200.226.7	"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; NET CLR 2.0.50727)""	3
10.2.65.4	"Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.9.1.16) Gecko/20101130 Firefox/3.5.16""	21
41.102.138.32	"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.1; Trident/4.0; SLCC2; NET CLR 2.0.50727; NET CLR 3.5.30729; NET CLR 3.0.30729; Media Center PC 6.0)""	6
41.102.182.32	"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; GTB6.6; InfoPath.1)""	6
41.102.239.89	"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; GTB6.6; InfoPath.1)""	30
41.102.181.196	"Mozilla/5.0 (Windows; U; Windows NT 6.1; fr; rv:1.9.2.13) Gecko/20101203 Firefox/3.6.13""	48
41.102.143.211	"Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; GTB6.6; Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1); NET CLR 2.0.50727; InfoPath.2)""	4
41.102.168.190	"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; GTB6.6; InfoPath.1)""	15
41.102.138.32	"Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; GTB6.6)""	16
41.201.77.171	"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0; GTB6.6; SLCC1; NET CLR 2.0.50727; Media Center PC 5.0; NET CLR 3.5.30729; NET CLR 3.0.30616; InfoPath.2; NET4.0C; OfficeLiveConnector.1.3; OfficeLivePatch.0.0)""	5
10.1.19.177	"Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.9.2.13) Gecko/20101203 Firefox/3.6.13""	5
207.46.13.88	"Mozilla/5.0 (compatible; bingbot/2.0; +http://www.bing.com/bingbot.htm)""	2
41.102.200.208	"Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; Trident/4.0; GTB6.6; NET CLR 1.1.4322; InfoPath.2; NET CLR 3.0.4506.2152; NET CLR 3.5.30729; NET CLR 2.0.50727)""	3
10.1.67.122	"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident/4.0; NET CLR 2.0.50727; NET CLR 3.0.4506.648; NET CLR 3.5.21022; NET CLR 1.1.4322; NET CLR 3.0.4506.2152; NET CLR 3.5.30729; InfoPath.2)""	17
38.105.173.198	Lynx/2.8.4rel.1	7

Figure V.5 Table d'identification des sessions.

Parmi les 198 sessions, on présente la session du couple (ip, ua)=(41.97.166.26, Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.9.1.13) Gecko/20100914 Firefox/3.5.13) représenté dans la figure V.6 qui caractérise les pages visitées (urls différents) ainsi que la durée de la session d'un utilisateur, trouvée par la requete SQL suivante :

```
Select ip, ua,url,dh from tassek4 where ip like '41.97.166.26'
```

Résultats du prétraitement des traces de navigation sur internet

IP	UA	URL	DH
41.97.166.26	"Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.9.1.13) Gecko/20100914 Firefox/3.5.13"	/	06/Dec/2010:11:32
41.97.166.26	"Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.9.1.13) Gecko/20100914 Firefox/3.5.13"	/accueil.html	06/Dec/2010:11:33
41.97.166.26	"Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.9.1.13) Gecko/20100914 Firefox/3.5.13"	/theflogos.php	06/Dec/2010:11:33
41.97.166.26	"Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.9.1.13) Gecko/20100914 Firefox/3.5.13"	/theflogos.php	06/Dec/2010:11:33
41.97.166.26	"Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.9.1.13) Gecko/20100914 Firefox/3.5.13"	/offresemploi.html	06/Dec/2010:11:37
41.97.166.26	"Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.9.1.13) Gecko/20100914 Firefox/3.5.13"	/candidat_inscription.htm	06/Dec/2010:11:40
41.97.166.26	"Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.9.1.13) Gecko/20100914 Firefox/3.5.13"	/offresemploi.html	06/Dec/2010:11:45
41.97.166.26	"Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.9.1.13) Gecko/20100914 Firefox/3.5.13"	/accueil.html	06/Dec/2010:11:46
41.97.166.26	"Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.9.1.13) Gecko/20100914 Firefox/3.5.13"	/theflogos.php	06/Dec/2010:11:46
41.97.166.26	"Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.9.1.13) Gecko/20100914 Firefox/3.5.13"	/theflogos.php	06/Dec/2010:11:46

Figure V.6 Session d'un utilisateur.

V.5 Pages populaires et impopulaires

L'idée est de donner toutes les pages populaires et impopulaires selon le degré de popularité en se basant sur le nombre de fois qu'une page a été visitée dans une période spécifiée. La figure V.7 a été réalisée en utilisant la requête SQL suivante :

```
Select url, count(*) from log group by (url) having count(*)>1 ;
```

URL	COUNT(*)
/admin_candidat.php	3
/admin	5
/candidat_logon.htm	83
/mail_ok.html	7
/sendtofriend.php	3
/theflogos.php	374
/offresemploi.html	224
/candidat.php	20
/cv_etatcivil.php	20
/taseek_vision.html	8
/taseek_engagements.html	7
/mail_fail.html	5
/recruteur_logon.htm	43

Figure V.7 Nombre de visites des pages url.

Résultats du prétraitement des traces de navigation sur internet

V.6 Information sur les Internautes

Cette rubrique renseigne sur la fidélité des visiteurs, chaque adresse IP est équivalente à un utilisateur ou Internaute. La figure suivante a été trouvée en utilisant la requête SQL suivante :

```
Select ip,count(*) from log group by (ip) having count(*)>1 ;
```

IP	COUNT(*)
41.105.21.17	9
41.100.177.119	8
41.104.55.228	4
209.85.228.81	3
41.104.104.61	4
81.52.168.186	9
41.107.58.148	3
196.29.40.33	4
41.102.217.196	3
213.132.255.223	4
174.94.104.166	4
41.104.77.205	4
196.20.73.60	5

Figure V.8 Nombre de visiteurs du site.

Après avoir structuré l'ensemble des visites, nous avons dégagé des comportements similaires en fonction du chemin des URLs visités. Le nombre de visites a diminué de 223 à 75 séquences de visites. L'étude de nombre de séquences de visites trouvées met en évidence trois classes (Candidats, Recruteurs et Administrateurs) selon le comportement des utilisateurs sur le site.

V.7 Classe Candidats

V.7.1 Regroupement des visites de la classe candidat

La figure V.9 se présente sous la forme d'un histogramme cumulé de 32 barres qui représentent le nombre de groupes que contient la classe candidat, chacune des barres représente le chemin parcouru par chaque groupe ainsi que le nombre d'utilisateurs qui ont suivi le même parcours.

Résultats du prétraitement des traces de navigation sur internet

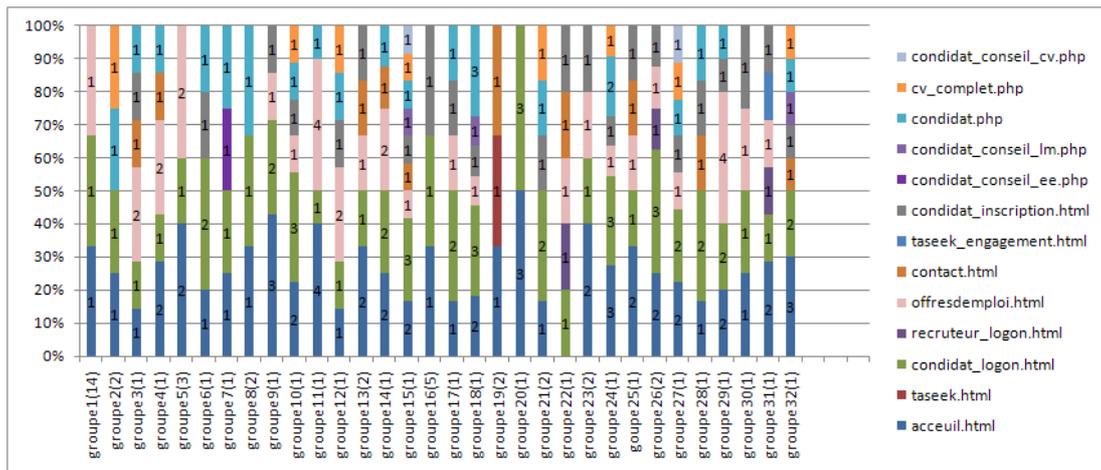


Figure V.9 Regroupement des visites de la classe candidat.

V.7. 2 Liens de provenance

La figure V.10 montre les pays de provenance des utilisateurs. C'est une information très importante pour savoir la popularité du site et sa puissance en ligne. Nous remarquons que la plupart des visiteurs du site durant la période d'analyse proviennent du pays l'Algérie, à savoir le pourcentage des utilisateurs à Alger, Oran et Constantine est de 48.39%, 16.9% et 12.53%. Il existe d'autres pays qui ont visité notre site comme la France avec 6.99%, et Londres avec 1.16%.

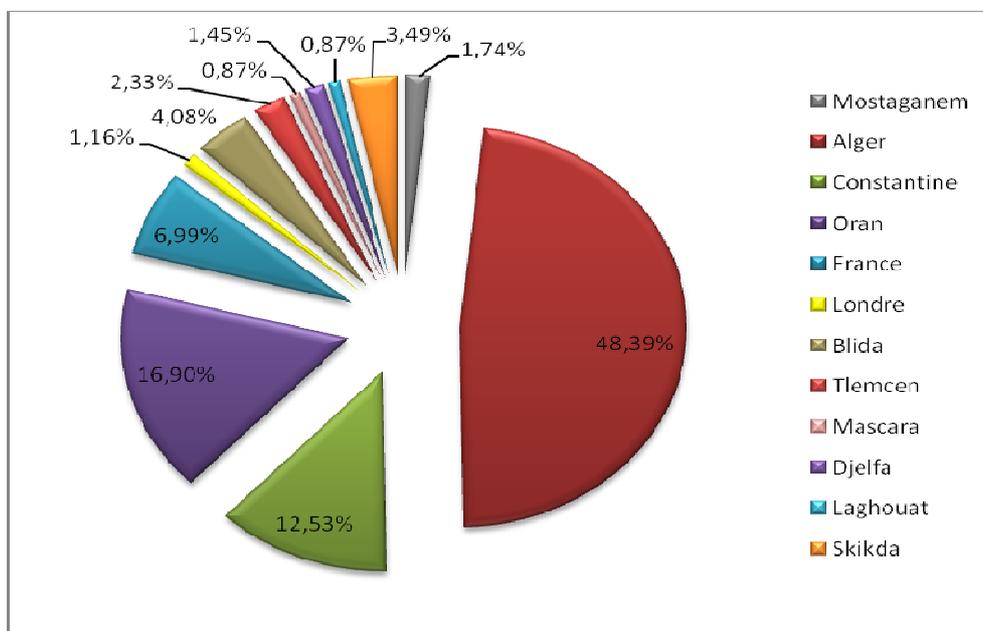


Figure V.10 Liens de provenance de la classe Candidat.

Résultats du prétraitement des traces de navigation sur internet**V.7. 3 Navigateurs**

La figure V.11 informe l'administrateur du site sur le type de navigateur le plus utilisé par ces visiteurs. Cette information pourra lui être utile au moment de la conception de son site. On remarque que la classe candidat a utilisé deux types de navigateurs, Mozilla/5.0 avec 67.26% et Mozilla/4.0 avec 32.73%.

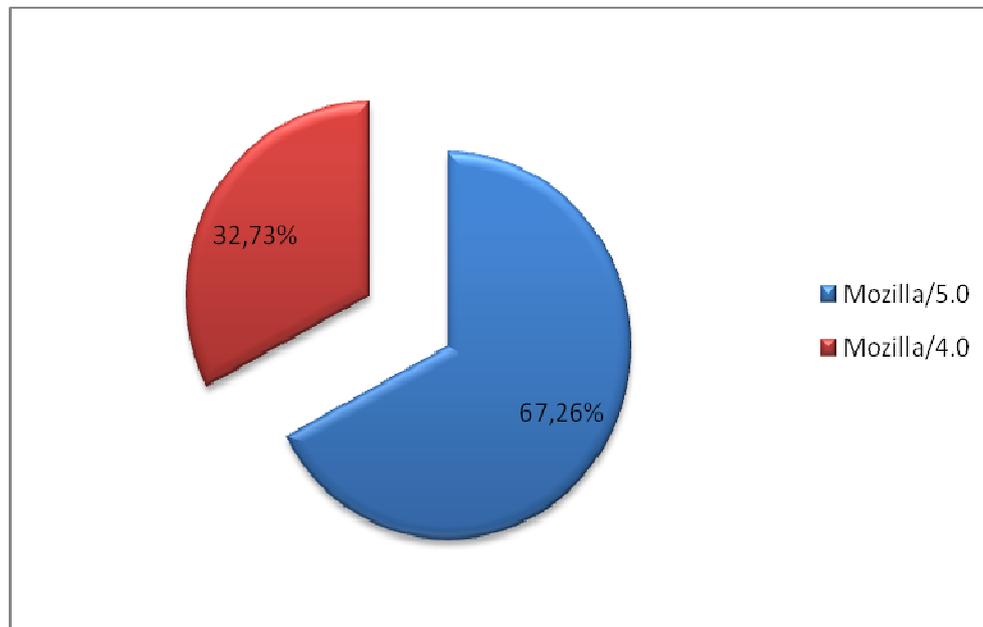


Figure V.11 Pourcentage des types de Navigateurs de la classe Candidat.

V.7. 4 Système d'exploitation

La Figure V.12 informe sur les types de systèmes d'exploitation utilisés par la classe candidat. On remarque que les types de systèmes d'exploitation les plus utilisés par la classe candidat est Windows XP et Windows 7 avec 73.8% et 12.24%, ensuite on a Windows vista avec un faible pourcentage.

Résultats du prétraitement des traces de navigation sur internet

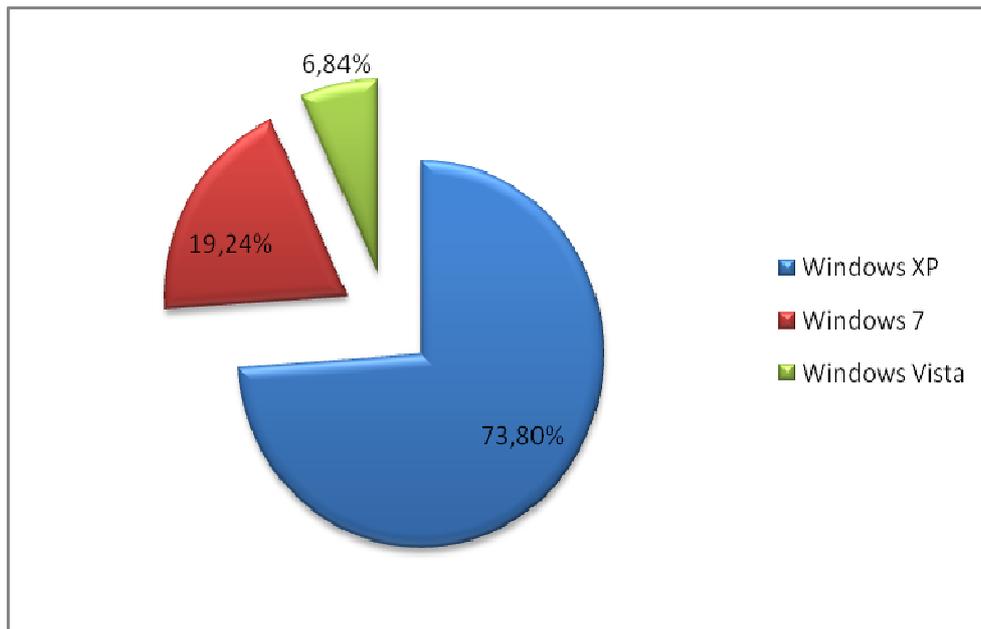


Figure V.12 Pourcentage des Systèmes d'Exploitation de la classe Candidat.

V.8 Classe Recruteur

V.8.1 Regroupement des visites de la classe Recruteur

La classe recruteur est représentée par 19 groupes, chaque groupe est identifié selon le nombre d'utilisateurs identiques qui ont visités les même pages URLS représentées dans la figure V.12. Prenons comme exemple la première barre cumulée qui représente le premier groupe, il contient 4 utilisateurs qui ont parcourus une seule fois les pages suivantes : accueil.html, recruteur_logon.html et offresemploi.html.

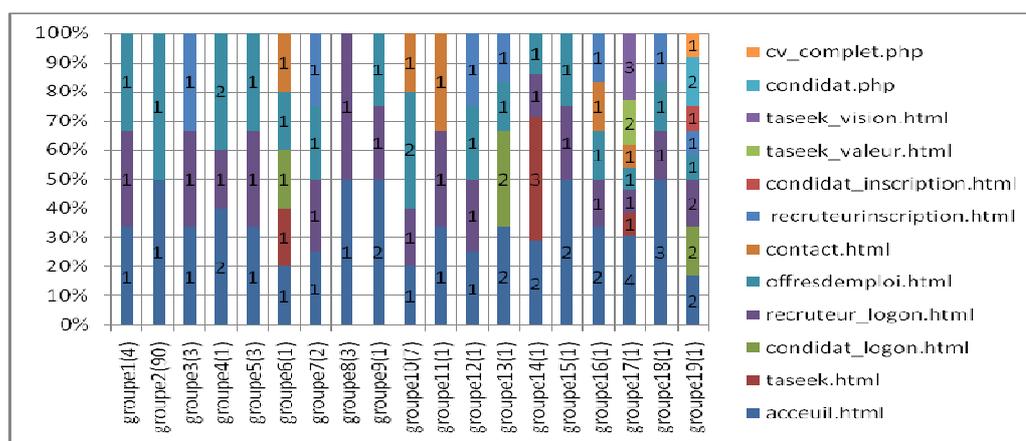


Figure V.13 Regroupement des visites de la classe recruteur.

Résultats du prétraitement des traces de navigation sur internet

V.8.2 Liens de provenance

En étudiant les pays de provenance, nous remarquons que la plus part des recruteurs viennent d'Alger avec 57.43% ensuite de Oran avec 18.2%, il existe d'autres régions comme Annaba, Jjel ..., mais avec de faible pourcentage.

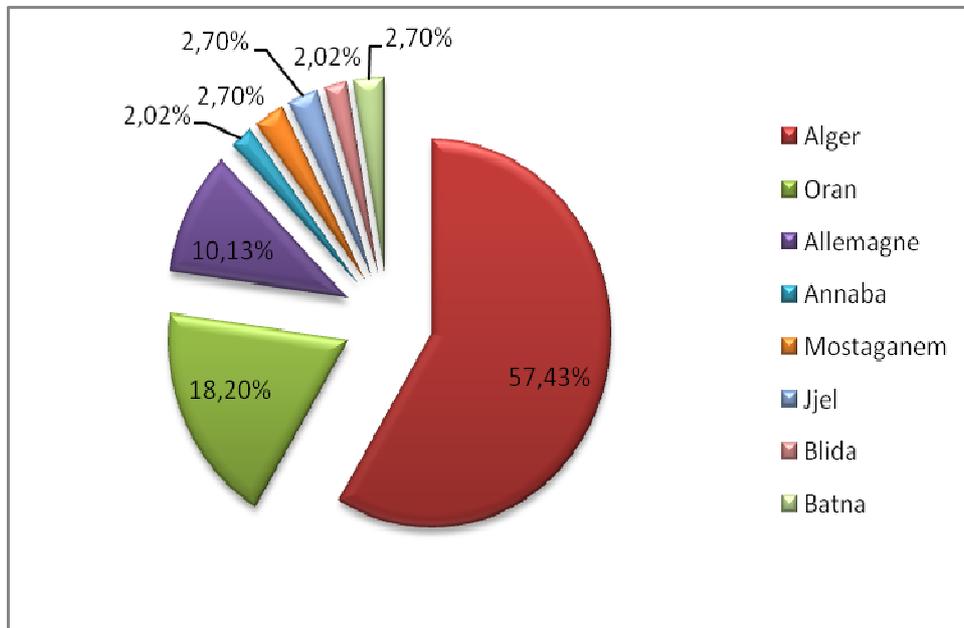


Figure V.14 Liens de provenance de la classe Recruteur.

V.8.3 Navigateurs

Les types de navigateurs les plus utilisés sont Mozilla/4.0 avec 57.14% et Mozilla/5.0 avec 42.85%, représentés dans la figure V.14.

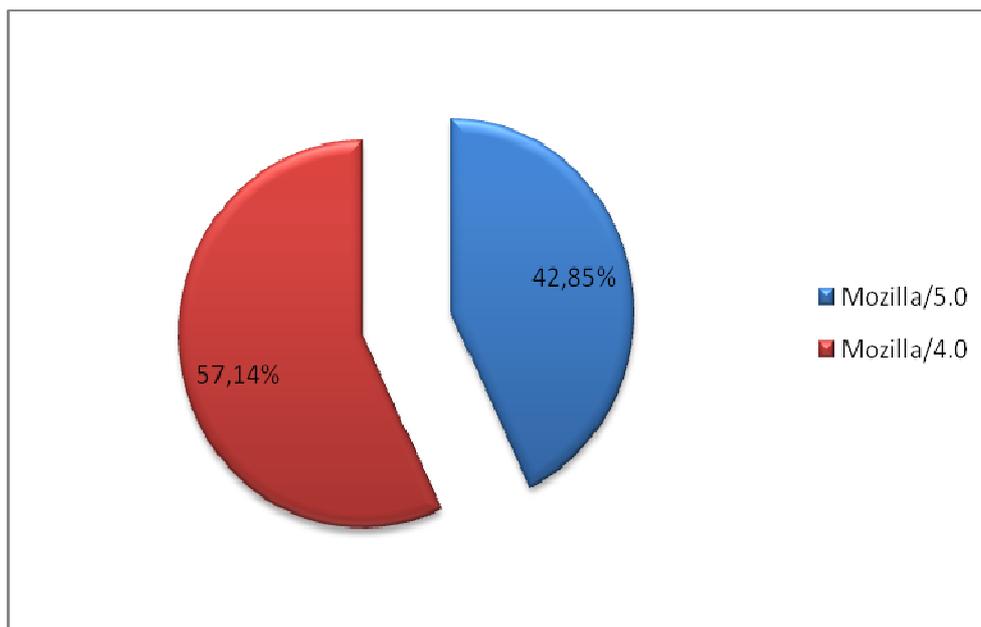


Figure V.15 Pourcentage des types de Navigateurs de la classe Recruteur.

Résultats du prétraitement des traces de navigation sur internet**V.8.4 Système d'exploitation**

On remarque que le type de système d'exploitation le plus utilisé est Windows XP avec 86.3%, il existe d'autres systèmes d'exploitation comme Windows 7 et windows Vista mais avec un faible pourcentage.

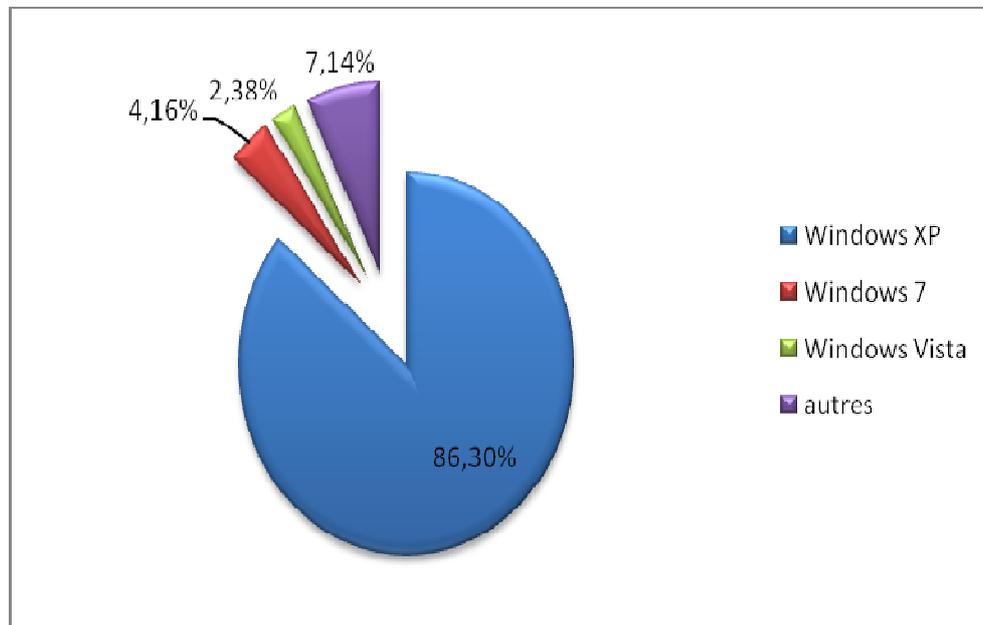


Figure V.16 Pourcentage des Systèmes d'Exploitation de la classe Recruteur.

V.9 Classe Administrateur**V.9.1 Regroupement des visites de la classe Administrateur**

La classe recruteur est représentée par 12 groupes, chaque groupe est identifié selon le nombre d'utilisateurs identique qui ont visités les même pages URLs représentées dans la figure V.16. Prenons comme exemple le groupe sept, il contient un seul utilisateur qui a parcouru une seul foi les pages suivantes : offredemploi.html, admin.php et admin_login.php.

Résultats du prétraitement des traces de navigation sur internet

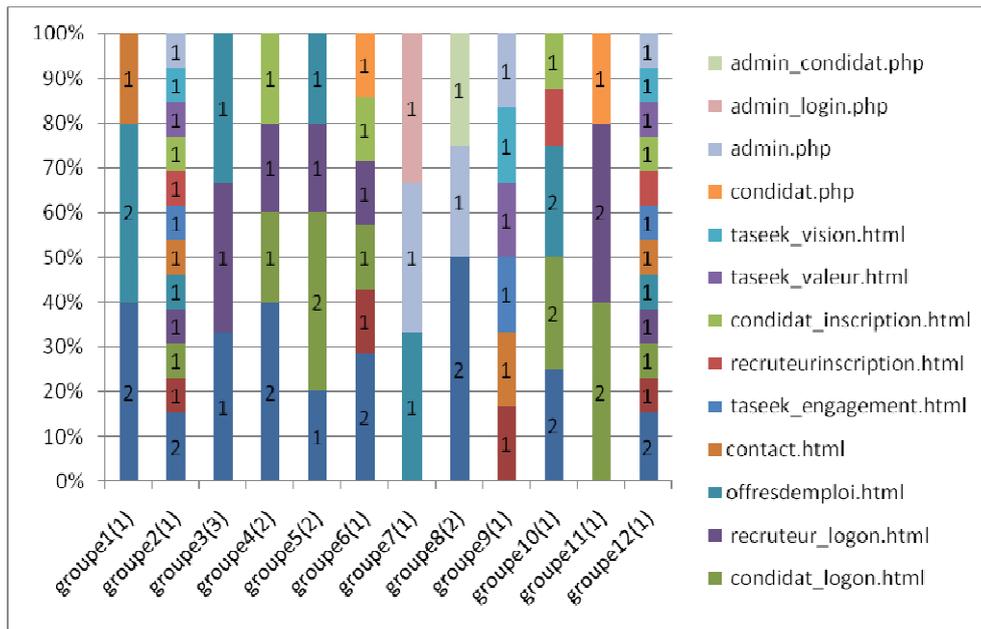


Figure V.17 Regroupement des visites de la classe administrateur.

V.9.2 Liens de provenance

Les pays qui ont le plus de visiteurs est Alger avec 51.37%, il existe d'autres pays mais avec un faible pourcentage, comme, la France avec 4.58%... .

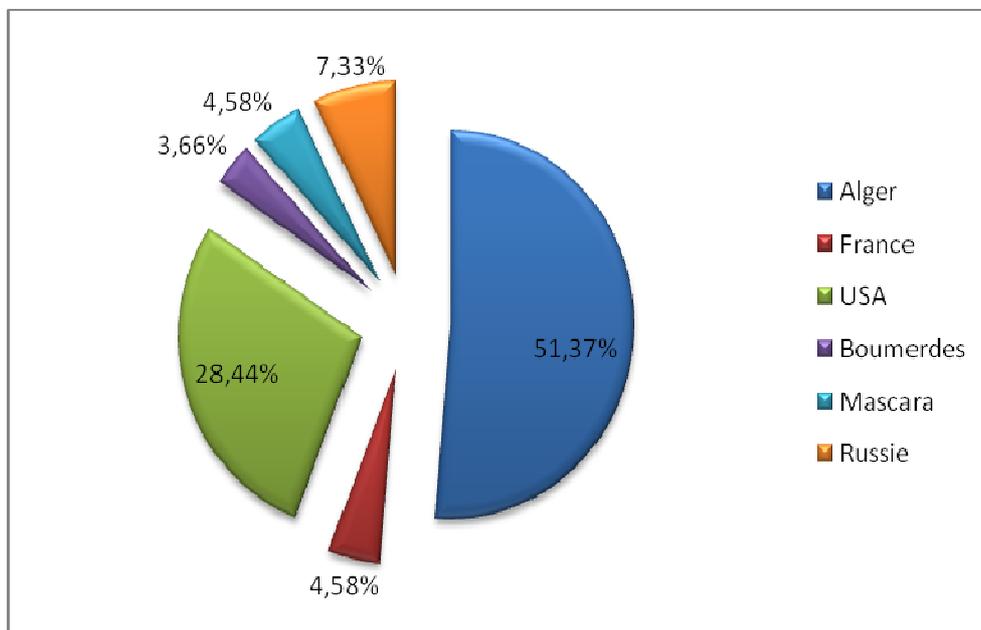


Figure V.18 Liens de provenance de la classe Administrateur.

Résultats du prétraitement des traces de navigation sur internet**V.9.3 Navigateurs**

Les types de navigateurs les plus utilisés sont Mozilla/4.0 avec 36.36% et Mozilla/5.0 avec 30.30% représentés dans La figure V.18

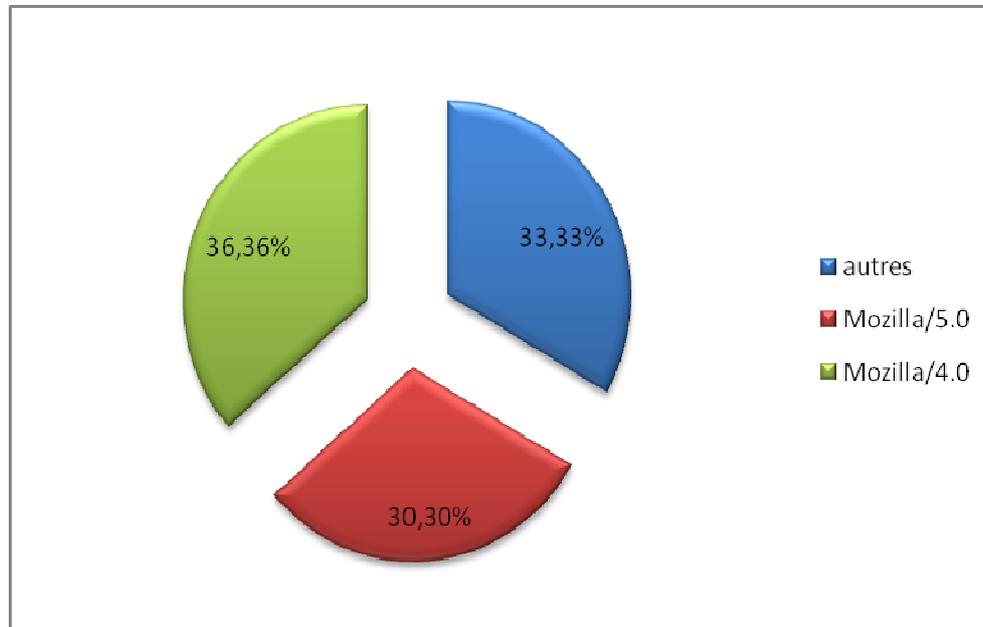


Figure V.19 Pourcentages des types de Navigateurs de la classe Administrateur.

V.9.4 Système d'exploitation

On remarque que le type de système d'exploitation le plus utilisé est Windows XP avec 81.77%, il existe d'autres systèmes d'exploitation comme Windows 7 et Windows serveur 2003 et Linux mais avec un faible pourcentage.

Résultats du prétraitement des traces de navigation sur internet

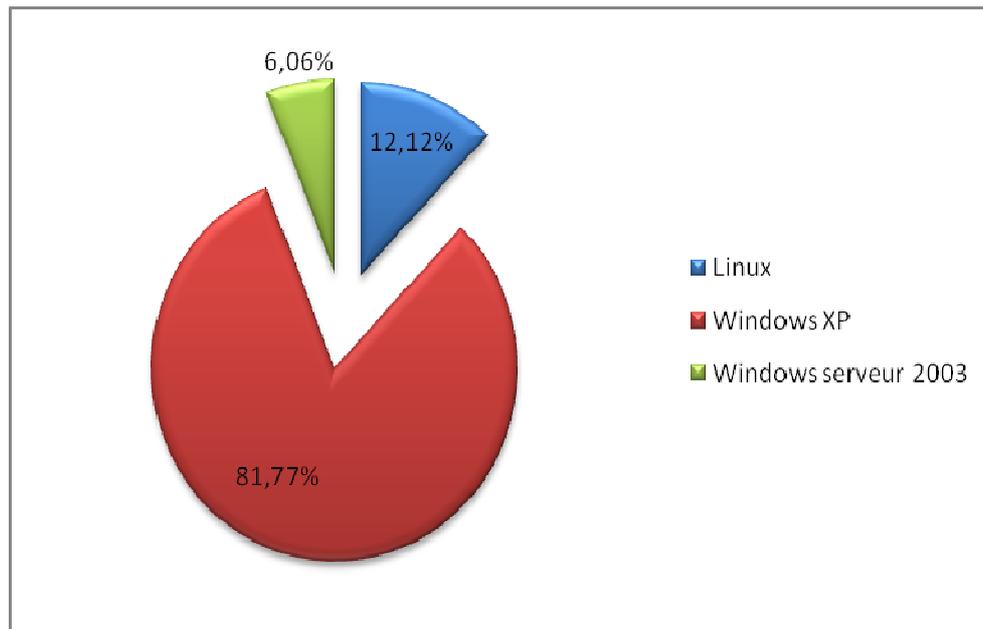


Figure V.20 Pourcentage des Systèmes d'Exploitation de la classe Administrateur.

V.10 Résultats finale

Après une étude du site, nous avons distingués quatres groupes :

- 1) Le premier groupe est déjà défini lors du pré-traitement des fichiers logs dont les requetes inutiles ont été supprimées.
- 2) Le dexieme groupe est celui des condidats du site, qui se sont inscrits, dont l'objectif de la navigation sur le site, la mise en ligne de leurs CV et la recherche d'offres d'emplois.
- 3) Le troisiéme groupe est celui des recruteurs (les entreprises...etc) qui propose à chaque fois des postes de travails et qui s'intéressent aux CV postulés.
- 4) Le quatriéme groupe et celui de l'équipe administrative du site, qui a pour rôle le bon déroulement des activités sur le site (les inscriptions,...) ainsi que la maintenance et le contrôle du contenu.

V.11 Conclusion

La classification des utilisateurs du site du taseek se résume en la découverte de motifs de navigation et des caractéristiques de chaque individu.

On a débuté notre travail avec une phase de nettoyage du fichier log du site web (www.taseek.com) tel que la nouvelle taille de la base (8.28 % de la taille initiale) montre bien l'importance de cette phase, ensuite on a passé au prétraitement qui a abouti a trouver 556 sessions, chaque session est représentée par les pages visitées d'un seul utilisateur. Parmi les 556 sessions on a supprimé 358 sessions qui visitent une seul page url, donc nous obtenons 198 sessions. En appliquant l'algorithme d'identification des visites on a trouvés 223 séquences de visites, tel que le temps

Résultats du prétraitement des traces de navigation sur internet

entre deux visites ne dépasse pas 30 minutes. Parmi ces 223 visites on a trouvé des visites qui sont identiques (suivent le même cheminement des pages du site étudié), en regroupant ces sessions le nombre a diminué à 75 visites. En étudiant le comportement de ces 75 visites, on a abouti à trois groupes d'utilisateurs à savoir, le groupe candidats (des clients ayant pour centre d'intérêt la recherche d'emplois), le groupe recruteur (des visiteurs qui sont gérés d'offrir les postes de travail aux clients) et le groupe administrateur (pour le maintien du site).

Conclusion générale

Dans ce travail, nous avons développé une méthodologie pour l'étude de comportement des utilisateurs lors d'une navigation dans un site web de grande taille, Cette méthodologie est composée de deux étapes :

- 1- La première étape concerne le prétraitement des fichiers Logs permettant en premier lieu de les nettoyer en supprimant les requêtes inutiles qui ne donne aucune information sur l'utilisateur, ensuite transformer l'ensemble de requêtes enregistrées dans les fichiers Logs du site à des données structurées et exploitables sous formes de sessions qui représentent les pages visitées de chaque utilisateur dans un laps de temps. Les sessions sont trouvées en regroupant tous les couples (ip, user agent) identique et enfin identifier les visites tel que chaque visite ne dépasse pas un laps de temps égale à 30 minute.
- 2- La deuxième étape est l'étape de classification utilisant les algorithmes de fourmis artificielles AntTree (sans Seuil), ce types d'algorithme utilise le principe d'auto assemblage des fourmis réelles, a partir de ces algorithmes nous avons réussi de classifier les fichiers log en groupes homogènes. Donner pour chaque groupe, des connaissances sur les visiteurs (détenir le pourcentage des visiteurs durant la période d'analyse, avoir une visibilité internationale :d'où proviennent les visiteurs ?), des connaissances sur les pages (reconnaître les pages web les plus et les moins consultées, connaître les combinaisons des pages consultées) et des connaissances sur les navigateurs et les User-Agent (connaître le pourcentage des navigateurs les plus utilisées, connaître également le pourcentage des systèmes d'exploitation les plus utilisés).

Perspectives

Le but de notre travail a été de regrouper les visiteurs d'un site web par similitude, afin de comprendre le comportement des utilisateurs et de détecter leurs parcours sur le site.

La visualisation du chemin parcouru sous forme graphique semble être intéressante, car elle fait passer beaucoup d'informations très rapidement a l'utilisateur, parmi les systèmes de visualisation on cite le système Tree maps, les arbres coniques et la visualisation en 3D avec VRMiner.

Finalement, la classification par les Fourmis Artificielles ne sont pas les seuls méthodes utilisées dans cet axe de recherche et on peut citer d'autres méthodes de classification comme les Cartes Auto-Organisées de Kohonen, afin de construire une carte qui définit un paquet de classes structurées en topologie prédéfinie, ensuite visualiser les différents transactions du fichier log en les projetant sur cette cartographie.

Références

- [1] N. Bousbia, « Analyse des traces de navigation des apprenants dans un environnement de formation dans une perspective de détection automatique des styles d'apprentissage », thèse de doctorat, Université Pierre et Marie Curie (France) et Ecole Nationale Supérieure d'Informatique (Algerie), pp : 50-51, 10 janvier 2011.
- [2] D. Cram, D. Jouvin, et A. Mille , « Visualisation interactive de traces et réflexivité : application à l'EIAH collaboratif synchrone eMédiathèque ». STICEF, volume 14, 2007.
- [3] P. A. Champin, Y. Prié, et A. Mille, « MUsETTE : a framework for Knowledge from Experience ». EGC'04, RNTI-E-2, Cepadues Edition, Clermont-Ferrand. France, pp: 129-134, 2004.
- [4] P. Jermann, A. Soller, et M. Muehlenbrock, « From Mirroring to Guiding: A Review State of the Art Technology for supporting Collaborative Learning ». Proceedings of the First European Conference on Computer-Supported Collaborative Learning, 2001.
- [5] J. P. Pernin, « un modèle de traitement de traces », CLIPS-IMAG, CSE, 2005.
- [6] L. Settouti, Y. Prié, A. Mille, et J.C. Marty, « Système à base de traces pour l'apprentissage humain ». Colloque international TICE 2006, Technologies de l'Information et de la Communication dans l'Enseignement Supérieur et l'Entreprise, 2006.
- [7] F. Fisher, « L'utilisation des traces numériques dans l'enseignement à distance ». Rapport de recherche bibliographique, 2005.
- [8] S. Iksal, et C. Choquet, « An open architecture for usage analysis in a e-learning context ». The 5th IEEE International Conference on Advanced Learning Technologies, Kaohsiung, Taiwan, pp: 177-181, 2005.
- [9] G. Stermsek, M. Strembeck, et G. Neumann, « A User Profile Derivation Approach based on Log-File Analysis ». International Conference on Information and Knowledge Engineering. Las Vegas, Etas-Unis, 2007.
- [10] K. Fansler, et R. Riegel, « A Model of Online Instructional Design Analytics ». 20th Annual Conference on Distance Teaching and Learning, Madison, Etats-Unis, 2004.
- [11] T. Beauvisage, « Sémantique des parcours des utilisateurs sur le Web ». thèse de doctorat, Université de Paris X, pp : 361, 2004.
- [12] T. Zhu, R. Greiner, et G. Haeubl, « Learning a model of a web user's interests ». The 9th International Conference on User Modeling (UM'03), 2003.
- [13] J. Blanchard, B. Petitjean, T. Artières, et P. Gallinari, « Un système d'aide à la navigation dans des hypermédias ». Actes de la conférence Extraction et Gestion des Connaissances, EGC'05, Paris, France, 2005.
- [14] G. C. Loghin, « Observer un Environnement Numérique de Travail pour réguler les activités qui s'y déroulent ». Thèse de Doctorat. Cotutelle entre l'Université de Savoie et l'Université Technique de Cluj-Napoca, 2008.
- [15] C. Michel, Y. Prié, L. Le Graet, « Construction d'une base de connaissance pour l'évaluation de l'usage d'un environnement STIC », 17eme Conférence Internationale Francophone sur l'Interaction Homme- Machine, Toulouse, France, pp: 199-202, 2005.
- [16] M.A. Sakka, Forum ATENA : « Traçabilité et monitoring », (Doctorant en thèse CIFRE Novapost/Samovar), pp : 2, 23 février 2009.
<http://www.forumatena.org/docu/12fev09/CRsakka.pdf>
- [17] K. Benabdeslem, « Approches connexionnistes pour la visualisation et la classification des séquences évolutives : Application aux données issues d'usages d'Internet », thèse de doctorat, université paris 13, pp : 19, 17 décembre 2003.
- [18] W.Gao, O. Sheng: « Mining characteristic Patterns to Identify Users », Proceedings of Workshop on Information Technology and Systems, vol. 13, 2004.
- [19] K. Benabdeslem, Y. Bennani: « Classification et visualisation des données d'usage d'Internet », Atelier Fouille du Web - Extraction et Gestion des Connaissances (EGC'06), Lille, France, 2006.
- [20] M. Charrad, Y. Lechevallier: « Extraction de connaissances à partir de fichiers Logs », Atelier Fouille du Web - Extraction et Gestion des Connaissances (EGC'06), Lille, France, 2006.

- [21] A.Büchner, M.Baumgarten, S.Anand, M.Mulvenna et J. Hughes: « *User-Driven Navigation Pattern Discovery from Internet Data* », Lecturer Notes in Computer Science, vol. 1836, p: 74-91, 1999.
- [22] R.Iváncsy, I. Vajk: « *Frequent Pattern Mining in Web log data* », Journal of Applied Science at Budapest Tech Hungary, Special Issue on Computational Intelligence, vol. 3, n° 1, pp: 77-90, 2006.
- [23] Q.Yang Q., H.Zhang, T.Li: « *Mining Web Logs for Prediction Models in WWW Caching and Prefetching* », 7th International Conference on Knowledge Discovery and Data Mining (KDD'01), pp: 473-478, 2001.
- [24] A. Pauchet, M. El Abed2, T. Merabti, É. Prieur, T. Lecroq, S.J. Darmoni, « *Identification de répétitions dans les navigations au sein d'un catalogue de santé* », article pour la revue: RIA - Numéro Spécial "Intelligence Artificielle et Web Intelligence", 21 octobre 2008.
<http://www-igm.univ-mlv.fr/~lecroq/articles/ria2009.pdf>
- [25] M. Charrad, « *Techniques d'extraction des connaissances appliquées aux données du Web*». Mémoire de Mastère présenté en vue de l'obtention du diplôme de Mastère en Informatique, Ecole Nationale des Sciences de l'Informatique de Tunis, Laboratoire RIADI (2005).
- [26] A. El Golli, « *Extraction de données symboliques et cartes topologiques : Application aux données ayant une structure complexe* », Université Paris IX Dauphine, thèse de doctorat, pp : 123-124, juin 2004.
- [27] J. Srivastava, R. Cooley, M. Deshpande, et P. Tan, « *Web Usage Mining: discovery and applications of usage patterns from Web data* » In SIGKDD Explorations, Volume1, Issue 2 editions ACM, pp: 12-23, Jan 2000.
- [28] Y. Lechevallier, D. Tonasa, B. Trousse, et R. Verde, « *Classification automatique: Applications au Web Mining* » In Yadolah Dodge and Giuseppe Melfi, editor, Méthodes et Perspectives en Classification, Presse Académiques Neuchâtel, Suisse, pp. 157-160, 10-12, September 2003.
- [29] D.Tanasa, « *Le prétraitement des fichiers logs Web dans le Web Usage Mining multi-sites* », Equipe Axis, INRIA Sophia Antipolis, 2004.
- [30] L. D. Catledge et J. E. Pitkow, « *Characterizing browsing strategies in the world-wide web* », Computer Networks and ISDN Systems, vol. 27, no. 6, pp: 1065-1073, 1995.
- [31] D. Tanasa, « *Web usage mining: Contributions to intersites logs preprocessing and sequential pattern extraction with low support* », Thèse de doctorat, University of Nice Sophia Antipolis, 3 June 2005.
- [32] H. Azzag, « *Classification hiérarchique par des fourmis artificielles : applications à la fouille de données et de textes pour le Web* », thèse pour obtenir le grade docteur de l'université de tours, pp : 24-42 et 45-48, 6 décembre 2005.
- [33] S. Boulkrinat, « *Modelisation hybride du profil utilisateur pour un système de filtrage d'informations sur le web* », mémoire de magister en informatique, Alger, 2007.
- [34] P. A. L. Henocque et M. Kleiner, « *Optimisation par colonie de fourmis pour la configuration* », Laboratoire LSIS ILOG S.A, Université de Saint-Jérôme 9 rue de Verdun, 2008.
- [35] L.M. Gambardella, M. Dorigo. « *Ant-Q : A reinforcement learning approach to the Travelling Salesman Problem* ». In A. Frieditis et S. Russell, editors, Proceedings of the Twelfth International Conference on Machine Learning, Morgan Kaufmann, San Mateo, California, pp: 252-260, 1995.
- [36] T. Stützle, H. Hoos. « *MAX -MIN Ant System and local search for the Traveling Salesman Problem* ». In Proceedings of the fourth International Conference on Evolutionary Computation, IEEE Press , pp: 308-313, 1997.
- [37] M. Dorigo, L.M. Gambardella, « *Ant colonies for the Traveling Salesman Problem* », BioSystems, pp: 43 et 73-81, 1997.
- [38] N. Monmarché, G. Venturini, et M. Slimane : « *On how Pachycondyla apicalis ants suggest a new search algorithm* », Future Generation Computer Systems, 168:937-946, 2000.
- [39] F. Picarougne : « *Recherche d'information sur Internet par algorithmes évolutionnaires* », Thèse de doctorat, Université de Tours, 19 novembre 2004.
- [40] E.D. Lumer et B. Faieta : « *Diversity and adaptation in populations of clustering ants* » In Proceedings of the Third International Conference on Simulation of Adaptive Behaviour, pp: 501-508, 1994.

- [41] V. C. Preda et D. Tali-Maamar, « *Optimisation Par Colonies de Fourmis* », pp : 11-23, 2004.
- [42] C. Anderson, G. Theraulaz, et J.L. Deneubourg. : « Selfassemblages in insect societies ». pp: 49 et 99-110, 2002.
- [43] C. Sauwens, « *Étude de la dynamique d'auto-assemblage chez plusieurs espèces de fourmis* », Thèse de doctorat, Université libre de bruxelles, 2000.
- [44] A. Lioni, « *Auto-assemblage et transport collectif chez oecophylla* », Thèse de doctorat, Université libre de bruxelles, Université Paul Sabatier, 2000.
- [45] Á. E. Eiben, R. Hinterding et Z. Michalewicz, « *Parameter control in evolutionary algorithms*», IEEE Trans. on Evolutionary Computation, pp: 124-141, 1999.
- [46] O.Nasraoui, A.Joshi et R.Krishnapuram, « *Relational Clustering Based on a New Robust Estimator with Application to Web Mining* », *Proc. of NAFIPS'99*, pp: 705-709, 1999.

Sites consulté:

- [47] <http://assiste.com.free.fr/p/abc/a/traces_internes_vos_traces.html>, 14 mars 2010.

Abstract

The Internet is a search space that contains many points represented as a graph with neighborhood relations, or a possible navigation between web pages is represented by an arc and nodes are represented by web pages.

During browsing on the Internet, we leave traces which can be internal or external, which anyone can read and can be used by people with access to our computer. In this study, we look at the types of signs bearing the name of log file, a text file that stores the traces of queries to level the server. This log file is represented by several lines, each line contains the IP address, date and time of the connection method, the access code, the page visited (url), the referrer (the page that we prompted a visit) and the user-agent. Our work was performed on log files of the website "www.taseek.com" is a recruitment site of Algeria in the period of 1 to 10 December 2010. From this log file, there was a study on the behavior of users when they visit the website. For this we have gone through several stages. First, we merged all the log files into one file. The log files are in the rough, they should be cleaned files into, why they will be pretreated through a cleaning step of deleting all the unnecessary requests provide no information on the behavior of users a website. After the cleaning phase it was found that this step is very important, since it reduces the size of the log file to 3.84% of original size. Then we will break the log file in multiple sessions, each session represents the pages visited by a single user. Found 364 sessions knowing that there are 68 sessions that contain only one page url so they are not considered. Then we apply the method of clustering AntTree (No-Threshold), the artificial ants algorithm inspired by the behavior of real ants using the principle of self-assembly. It is a behavior of clustering between individuals in order to build living structures. This classification leads us to find popular and unpopular pages, countries that have visited the website, browsers and operating systems most used by using the similarity measures.

Key words

Browsing on the Internet, Log file, Method of clustering, AntTree (No-Threshold), Similarity measures.

Résumé

L'internet est un espace de recherche qui contient beaucoup de points représentés comme un graphe avec des relations de voisinage, où une navigation possible entre deux pages web est représentée par un arc et les nœuds sont représentés par des pages web.

Lors d'une navigation sur internet, on laisse des traces qui peuvent être internes ou externes, dont n'importe qui peut prendre connaissance et qui peuvent être utilisables par les personnes ayant l'accès à notre ordinateur. Dans cette étude, on s'intéresse aux types de traces qui portent le nom de fichier log, un fichier texte qui enregistre les traces de requêtes au niveau du serveur. Ce fichier log est représenté par plusieurs lignes, chaque ligne contient l'adresse IP, la date et l'heure de la connexion, la méthode, le code d'accès, la page visitée (url), le referré (la page qui nous a amené à visiter le site) et l'user-agent.

Notre travail a été réalisé sur les fichiers log du site web "www.taseek.com" qui est un site de recrutement algérien dans la période du 1 au 10 décembre 2010. A partir de ce fichier log, on a pu réaliser une étude sur le comportement des internautes lors de leurs visites du site web. Pour cela on est passé par plusieurs étapes. Tout d'abord, on a fusionné tous les fichiers logs dans un seul fichier. Les fichiers logs sont à l'état brut, on doit les transformer en fichiers épurés, pour cela ils vont être prétraités en passant par une étape de nettoyage qui consiste à supprimer toutes les requêtes inutiles qui ne donnent aucune information sur le comportement des utilisateurs sur un site web. Après la phase de nettoyage, on a constaté que cette étape est très importante, puisqu'elle a réduit la taille de notre fichier log à 8.28% de la taille initiale. Ensuite on passe à l'étape de transformation où on commence par l'identification des sessions, chaque session représente les pages visitées par un seul utilisateur. On a trouvé 556 sessions sachant qu'il ya 358 sessions qui ne contiennent qu'une seule page url donc elles ne sont pas prise en considération. On appliquant l'identification des visites, tel que le temps entre deux visites ne dépasse pas 30 secondes, on a identifié 223 séquences de visites. Parmi ces 223 visites, il y-on-a qui sont identique (suivant le même cheminement des pages du site étudié). Ensuite on applique la méthode de classification non supervisée AntTree(Sans-Seuil), l'algorithme de fourmis artificielles qui s'inspire du comportement des fourmis réelles en utilisant le principe d'auto-assemblage. C'est un comportement consistant à regrouper des individus entre eux de manière à construire des structures vivantes. Cette classification nous amène à trouver les pages populaires et impopulaires, les pays qui ont visité le site web, les navigateurs et les systèmes d'exploitation les plus utilisés en faisant appel aux mesures de similarité.

Mots clé :

Navigation sur internet, Fichier log, Classification non supervisée, AntTree(sans-seuil), Mesure de similarité.