INSA LYON

TP C++ n° 4 :

Analyse de logs apache

DURAND Bastien FOURNIER Romain

9 février 2018

Table des matières

1 Cahier des charges et spécifications **2**

2 Conception **3**

3 Organisation des données **4**

A Manuel d'utilisation **5**

B Tests de validation **6**

Introduction

L'objectif de ce TP est de développer un outil d'analyse d'un fichier log apache, en utilisant autant que possible les outils de la STL. Il s'agit de simplifier le développement en évitant de recréer des outils déjà existants. Pour ce faire, nous avons d'abord dû nous familiariser avec la syntaxe de ce type de fichier, puis sélectionner la structure de donnée de la STL la plus adaptée à nos besoins, puis enfin implémenter l'outil. Nous avons aussi dû effectuer quelques recherches sur le language GraphViz et les outils associés nécessaires à la création d'un graphe représentant le fichier log.

1 Cahier des charges et spécifications

Nous avons créé un programme d'exploitation de fichiers log qui respecte les besoins furmulés dans l'énoncé :

- -par défaut, l'outil affiche dans la console sous forme textuelle la liste des 10 documents les plus populaires par ordre décroissant. L'outil est appelé en ligne de commande par la commande : ./analog [options] nomfichier.log,
- -l'outil peut prendre 3 options en ligne de commande :
  - [-g nomfichier.dot] produit un fichier au format Graphviz,
  - [-e] permet d'exclure les ressources qui ont une extension js, image ou css,
  - [-t] permet de ne prendre en compte que les hits dans un créneau horaire correspondant à l'intervalle [heure , heure+1].

L'expression initiale des besoins étant un peu floue, nous y avons ajouté quelques spécifications :

- **Par défaut**, l'outil affiche dans la console sous forme textuelle la liste des 10 documents les plus populaires par ordre décroissants :
  - Si il y a moins de 10 documents consultés, l'outil donnera la liste de tous les documents classés par popularité,
  - Si la recherche n'a retourné aucun documents, l'outil donnera une réponse prédéfinie,
  - Si le fichier .log est vide, inexistant, mal formé ou non spécifié, l'outil affiche l'aide dans le terminal,
  - **Amélioration possible :** ajouter une option permmettant le contrôle du nombre de ressources affichées.
- **[-g nomfichier.dot]** produit un fichier au format Graphviz :
  - Pour les ressources n'ayant pas de referer (accès direct à la ressource), un noeud " - " représente les accès direct dans le graphe,
  - Si le nom du fichier .dot n'est pas spécifié, l'outil affiche l'aide dans le terminal,
  - Si un fichier du même nom existe déjà, une confirmation est demandée à l'utilisateur avant d'écraser le fichier,
  - **Amélioration possible :** nettoyer les referers contenant des passages de paramètres pour un affichage plus clair du graphe.
- **[-e]** permet d'exclure les ressources qui ont une extension .png, .jpg, .bmp, .gif, .css, .js, .ico :
  - Si une ressource n'a pas d'extension, elle sera séléctionnée,
  - **Amélioration possible :** -e PATTERN permettant de choisir un ensemble d'extensions.
- **[-t heure]** permet de ne prendre en compte que les hits dans un créneau horaire correspondant à l'intervalle [heure, heure+1] :
  - L'outil devra aussi fonctionner si la plage horaire s'étend sur 2 jours,
  - Si l'heure n'est pas spécifiée ou au mauvais format, l'outil affiche l'aide dans le terminal,
  - **Amélioration possible :** plus de précision sur le choix du créneau horaire, voir même sur la date.

2 Conception

Notre application ne comporte qu'une seule interface et une seule réalisation associée. En effet, nous avons décider d'utiliser une structure de donnée de la STL pour gérer nos données. Nous n'avons donc pas eu à implémenter de classes supplémentaires. Nous avons séparé le fonctionnement de l'outils en différentes fonctions pour nous répartir le travail de manière efficace. Ainsi, nous avons établi l'architecture suivante :

- une fonction genererCatalogue qui crée et remplie la structure de données à partir du fichier log,
- une fonction genererGraphe qui crée le fichier .dot à partir de la structure de données,
- une fonction ajouterRessource (appelée par la fonction genererCatalogue). En effet, la fonction genererCatalogue telle que présentée plus haut étant la partie la plus compliquée, nous l'avons séparée en 2 méthodes : genererCatalogue qui recherche les informations dans le fichier log et ajouterRessource qui ajoute l'information dans la structure de donnée.

Remarque : notre structure de données étant déjà définie, nous avons commencé par ces deux fonctions que nous avons pu développer en même temps (en testant la fonctions genererGraphe avec un jeu de données défini dans notre programme)

3 Organisation des données

Les données tirées de l'exploitation des logs seront stockées dans plusieurs conteneurs de la stl de type "unordered_map". A la différence d'une "map" ce conteneur ne stocke pas ses clées triées. Ainsi ce n'est pas sous forme d'arbre binaire rouge et noir qu'une "unordered_map" est implémentée mais sous la forme d'une table de hachage. Le cout en insertion ou recherche est en O(1) en moyenne et O(n) dans le pire des cas.
Les données extraites des logs sont des ressources et des referers. Voici la manières dont nous les stockons :

- chaque ressource est une clé dans une "unordered_map" "générale",
- clé étant associée à une valeur de type "unordered_map",
- les referers pointant sur une ressource sont clés de l'"unordered_map" liée à cette ressource,
- clé étant associée à une valeur de type int,
- cet entier désigant le nombre de fois que la ressource a été demandée via ce referer.

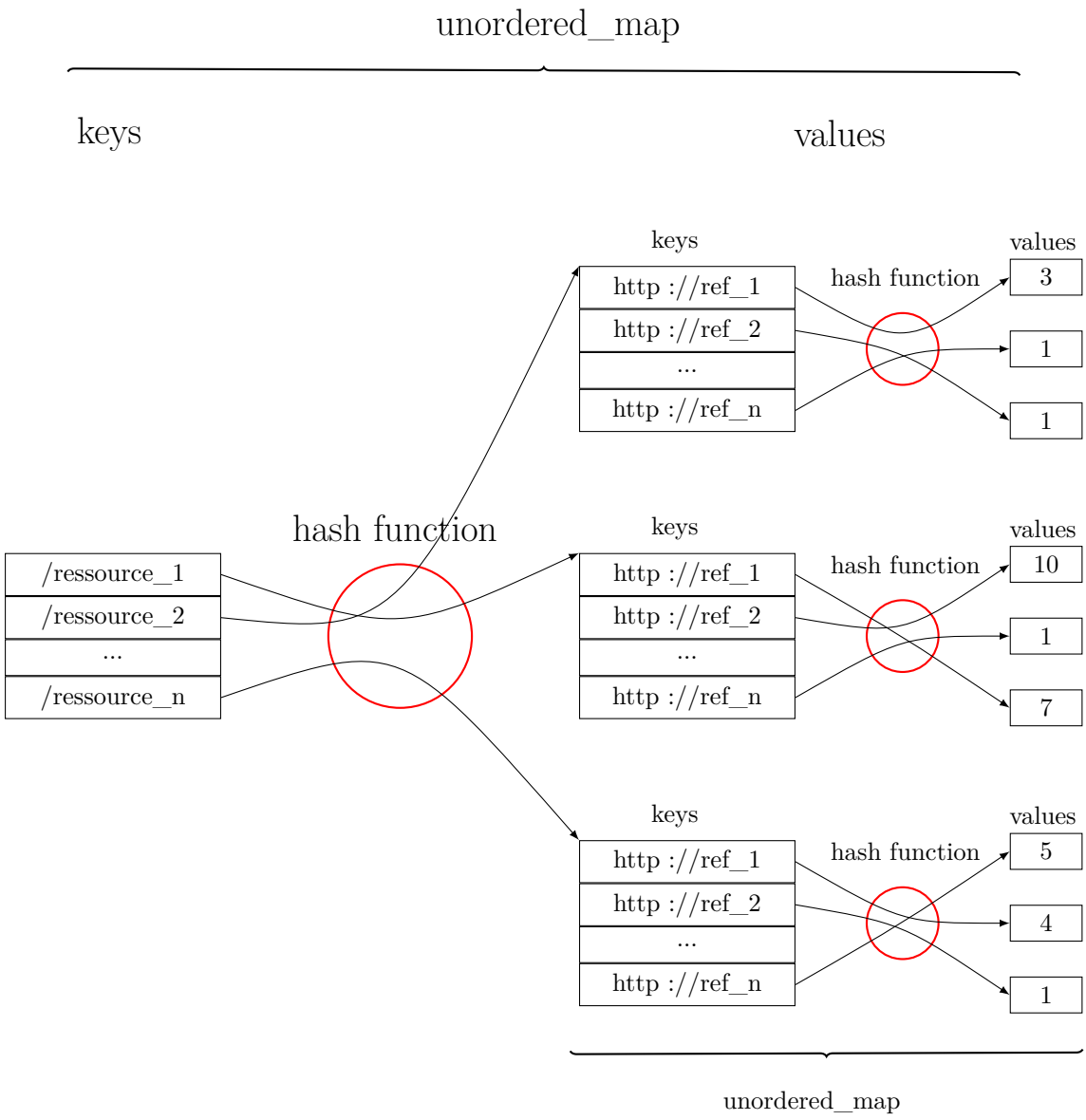

Figure 1 – Structure de données utilisée pour stocker et organiser les données extraites des logs

A Manuel d'utilisation

NOM

analog

SYNOPSIS

analog FILE [**-e**]... [**-t** HOUR]... [**-g** FILE]...

DESCRIPTION

analog permet d'analyser le fichier donné en entré comme un enregistrement de logs apache.
Par défault, sont donnés en sortie les 10 ressources les plus demandées sur le serveur. Seule les requêtes GET de code retour 200 sont analysées.

OPTIONS

-e

Ignore les ressources de type image : .png, .jpg, .bmp, .gif, .css, .js, .ico.

-t HOUR

Ignore les logs ne correspondant pas à l'intervalle [HOUR ; HOUR + 1].

-e FILE

Produit un fichier FILE qui est un graphe modélisant l'ensemble des ressources demandées via leur referer.

B Tests de validation

Afin de réaliser des tests tous en controlant les résultats fourni par le programme, un échantillon de logs a été crée. Le fichier "commmentedSamples" contient ces lignes de logs triés par caractéristiques (status, heure, image, même ressource...). De cette manière il est plus facile de controler la sortie du programme qui verra ces logs triés comme des ensembles avec ou sans intersections. Le fichier "samples" est équivalent au fichier "commentedsamples" dont on a retiré les commentaires. C'est ce fichier non commenté qui est donné en paramètre au programme.

Tests	**Options**	**Justifications**
1	aucune	Ce test appelle le programme sans options ni arguments. Il permet de valider l'affichage ou non du manuel d'utilisation.
2	aucune	Ce test appelle le programme sans options avec le fichier "samples" en paramètre. Nous controllons sur la sortie standard l'affichage des 10 ressources les plus demandées sans distinction des extensions ni des heures.
3	-e	Dans ce cas le programme est appelé avec l'option de filtrage des images "-e" avec le fichier "samples" en paramètre. Nous controllons sur la sortie standard l'affichage des 10 ressources les plus demandées et dont l'extension ne correspond pas à une image, sans distinction des heures.
4	-t 15	L'option de sélection d'un créneau horraire "-t" est testée avec ce test qui appelle le programme avec cette option et l'heure 15 sur le fichier "samples". Nous controllons sur la sortie standard l'affichage des 10 ressources les plus demandées et dont l'heure du log est comprise sur le créneau [15 ;16] sans distinction de l'extension.
5	-t 15 -e	Ce test appelle le programme avec deux options combinées "-t" et "-e" sur le fichier "samples". Nous controllons sur la sortie standard l'affichage des 10 ressources les plus demandées avec filtrage des images et dont l'heure du log est comprise sur le créneau [15 ;16].
6	-g testdot	Ici le programme est appelé avec l'option "-g" et la chaine "testdot" sur le fichier "samples". Avec cette option le programme doit générer un fichier de type "dot" et dont le nom est celui de la chaine donnée "testdot". Le fichier généré décrit un graphe modélisant l'ensemble des liens entre ressources et referers contenus dans les logs. Nous controllons sur la sortie standard l'affichage des 10 ressources les plus demandées et la création du fichier "dot" sans distinction des extensions et des heures.

Remarque : lors de l'éxecution du script test.sh la redirection de l'entrée standard sur le fichier std.in ne fonctionne pas tandis que l'appel au programme de la manière suivante analog < std.in prend bien en compte la redirection de l'entrée standard. Pour cette raison le test d'affichage du manuel (affichage nécessitant la saisie au clavier d'une lettre) ne fonctionne pas réellement.