資料索引 期末專案報告

劉昌澈 405410075

林柏亨 405410047

1. 專案目標

為了完成後續專案的目標（一個可以根據使用者喜好推薦新聞的應用程式），我們需要一個強而有力的爬蟲。因此在此專案中，我們擴充原先爬蟲功能，效能與健壯性。

1. 開發環境
   1. OS： Ubuntu 20.04 LTS 64 bit
   2. Virtualization: KVM (Parallel Desktop)
   3. Memory: 3.8 GB
   4. Processor: Intel® Core™ i5-7360U CPU @ 2.30GHz × 3
   5. Graphics: Intel® Iris(TM) Plus Graphics 640
2. API套件
   1. X DevAPI: c++ 連接mysql api

<https://dev.mysql.com/doc/x-devapi-userguide/en/>

* 1. Md5: convert url string to md5

<https://github.com/JieweiWei/md5/tree/30774d0c9ecf82fb003fc15978b1f21bc85858f7>

* 1. Libcurl: fetch html page

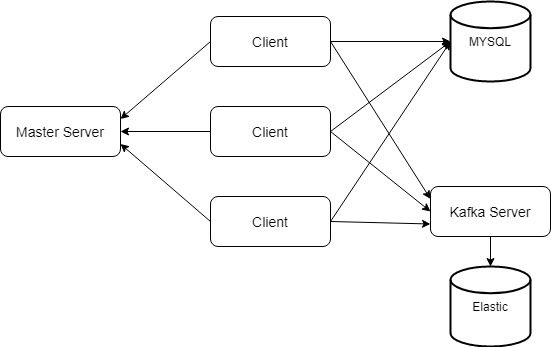
<https://curl.haxx.se/libcurl/>

* 1. ElasticSearch: Search Kernel  
     <https://www.elastic.co/>
  2. Qt5  
     <https://www.qt.io/>
  3. Youtube

<https://developers.google.com/youtube/v3>

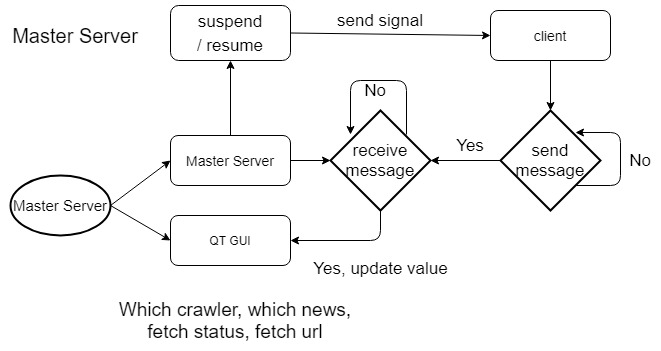
* 1. Apache Kafka & librdkafka  
     <https://kafka.apache.org/>  
     <https://github.com/edenhill/librdkafka>

1. 爬蟲架構

1. 主要架構:

以1個master server與多個client slaves進行溝通，而每個client都會傳送訊息給Mysql和kafka，最後maintext再由kafka傳送給Elasticsearch。

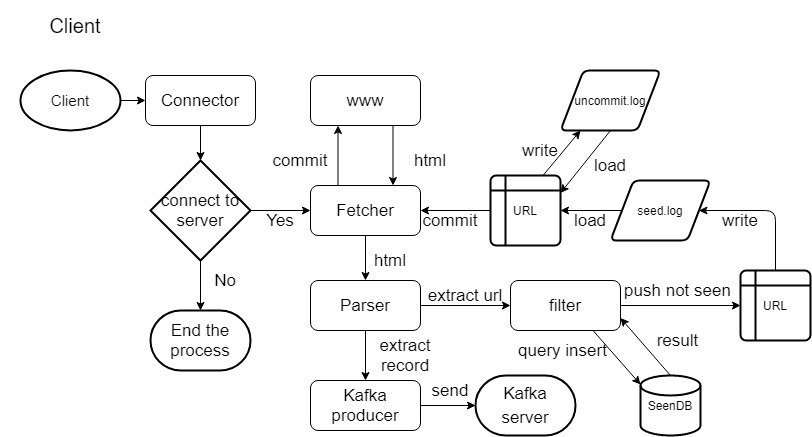
2. Server 與 Clients 架構



Clients 每爬完一筆url，會傳送該url的資料給server，包含fetch狀態、哪

個新聞網站以及url字串，server接收後經處理，最後update在Qt的GUI。

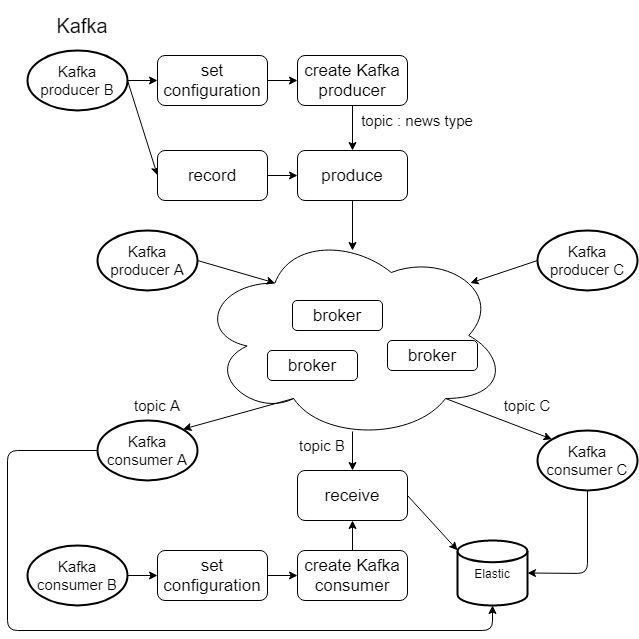
3. Client端的架構



利用connector與master server連線， 驅動爬蟲的啟動，爬蟲啟動後：

* 1. 從以分類好的seed.log，uncommit.log（例如：ettoday.net所分配到的log檔分別為ettoday\_seed.log，ettoday\_uncommit.log）讀取url至暫存url的vector中(commitQueue)
  2. Fetcher從commitQueue中獲取url並對該網址發出request下載完整的html檔至記憶體中，同時更新爬蟲介面的資訊
  3. 將完整的html檔傳入Parser中，此時Parser會擷取出外部連結，與資料（包含網址，標題與正文）
  4. 資料會傳入一vector中（recordQueue），待recordQueue達100KB上線後再將資料寫入相對應的rec檔，並利用Kafka producer將該資料送到Kafka server，並依正文進行分類。
  5. 外部連結會傳入filter，此時filter會向mysql database問此網址有無爬取過，如果以爬取過就拋棄此網址，如果沒有的話將此網址傳入vector(master\_filter\_queue)中，待master\_filter\_queue達100KB的上限後再將資料寫回對應到seed.log

4. Kafka與Elasticsearch溝通架構



每一個client都會有一個producer，專門負責傳送處理好的maintext，再利用Kafka特有的topic分類各個新聞媒體，傳送給負責該新聞的consumer放入Elasticsearch

1. 初版爬蟲問題
   1. 嚴重的記憶體流失
      1. 爬20分鐘後，監看程式記憶使用量已超過8G
   2. 不廣用的正文抽取
      1. 原先正文抽取依照ettoday.net中的新聞架構來爬取正文與標題，因此造成爬蟲只能爬取ettoday.net而不能爬取其他網站
   3. 網站爬取選擇單一
      1. 初版爬蟲原先只能爬取ettoday.net，使得資訊來源相當有限。
   4. 無效流量控制
      1. 時間計算有誤使得流量控制時間限制與程式設計師所指定時間不同
   5. 使用者不友善的應用程式
      1. 無完善的使用與安裝文件
      2. 需手動於特定log檔放置爬尋網址才能啟動爬蟲
      3. 簡陋的query介面
   6. 無後台紀錄
      1. 無法針對爬蟲效率或網站爬蟲做分析，無法有效率的校正並改善
2. 多線程版爬蟲
   1. master-slave架構
      1. 以一個master server與多個client slaves為主軸
      2. Server為集中數據處，client為crawler
      3. 進行有效率的多線程爬蟲
      4. Server可以對各個crawler進行控制(如暫停等)
      5. Server端可以監看各個client爬蟲進度
   2. General maintex extraction
      1. 閱讀 [1] 與 [2] 後，設計爬蟲正文抽取的功能
      2. 利用tidy [3] 將原先的html檔整理成xhtml，因為xhtml較有架構，方便後續生成DOM tree
      3. 使用C++ boost中的property tree解析xhtml檔
      4. 依照property tree 額外生成另一DOM tree，
         1. 此DOM tree含有tagname，文字濃度與子樹這些attribute
         2. 此DOM tree不含原xhtml所包含的css，javascript和comment
      5. 設置<body> tag的文字濃度為threshold
      6. 從子樹往下擷取正文，如果文字濃度大於threshold則擷取該節點的文字
   3. 流量控制

改用clock.h中的時間函式，對同一網站，計算前一次發出要求和當前發出要求的時間差，將時間差限制在0.005秒

* 1. 記憶體流失清除

使用valgrind對程式進行記憶體使用監測，發現了一個有趣的現象，再分配給getline(stdio.h中)自動分配記憶體大小的參數時（getline(null, 0, file pointer)），函式會在讀取到eof時仍自動分配記憶體，因此這個記憶體也是要清除的

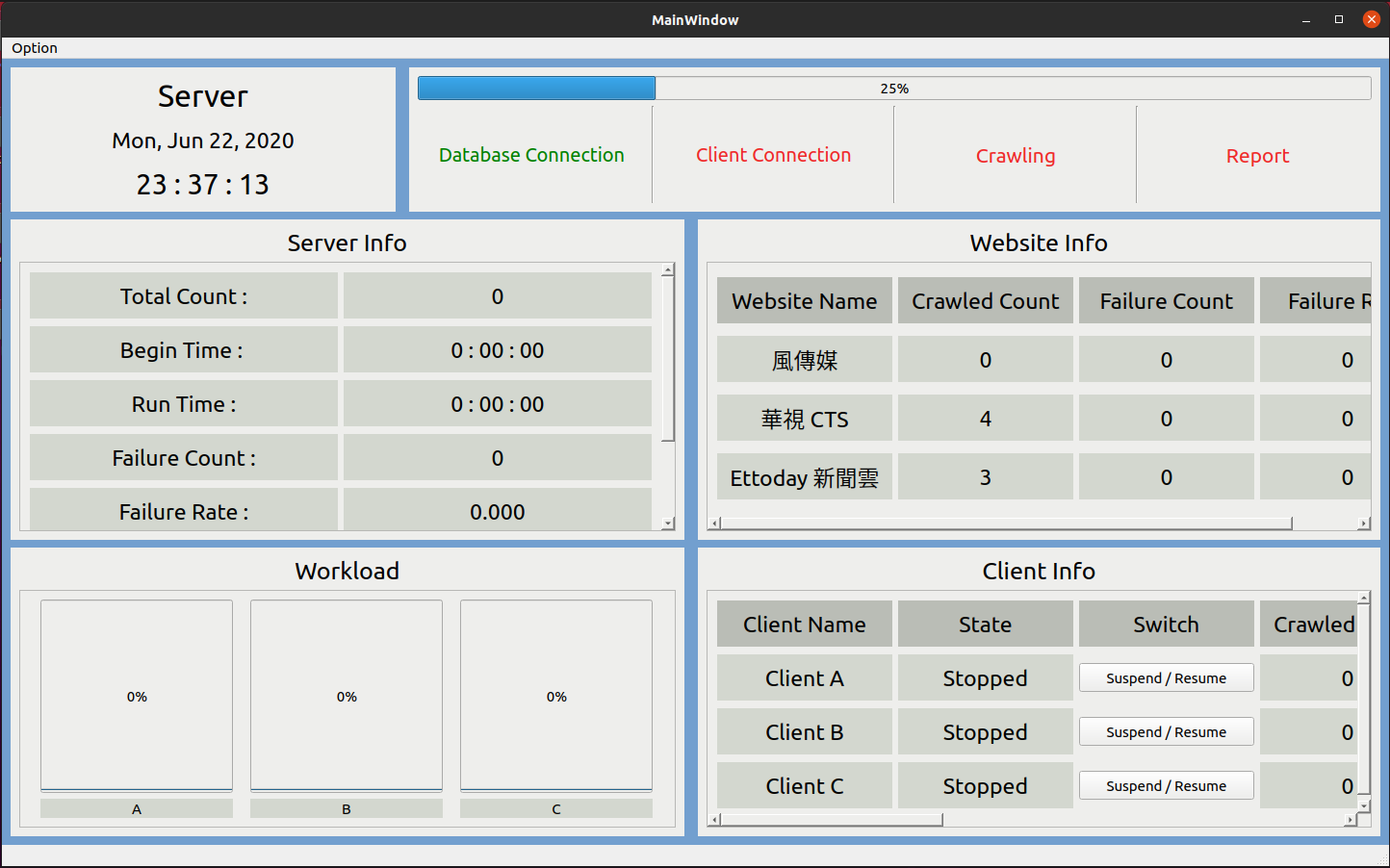
* 1. Apache Kafka
     1. 抽出正文後，藉由Apache的kafka進行傳輸資料
     2. producer可以針對不同新聞媒體(topic)，進行個別傳輸到各個consumer
     3. 特別有利於在不同新聞中，會有欄位的資料不同的需求  
        (e.g. A新聞有標題、時間、正文，B新聞有標題、時間、正文、圖片、關鍵字)
     4. 可以針對各種新聞媒體進行客製化的格式並輸入到Elasticseach
  2. Elasticsearch
     1. 強大的搜尋引擎系統
     2. GET, POST, PATCH, DELETE進行查詢與刪除
     3. 外接ik斷詞plugin進行查詢
     4. 擁有score評比系統，分數高的查詢相關度較高
     5. 繁體中文字典作為斷詞依據
  3. 豐富資訊的爬蟲觀測介面
     1. 因為爬蟲是用C++所撰寫，因此Qt可以做為最直接的GUI
     2. 可以快速進行update各種數據
     3. 圖形化介面可以讓人快速理解數據意義
  4. 多樣化的新聞平台
     1. 一般的新聞網頁，都是以文字圖片為主，但是其實許多新聞在youtube中也都有片段
     2. 可以同時查詢相關新聞的網頁以及youtube新聞片段
  5. 簡易安裝
     1. 撰寫了install.sh的shell script，幫助使用者安裝驅動此爬蟲所需之套件
  6. Github完整readme

1. 結果討論與分析
   1. 爬蟲結果
      1. 爬取資料比數: 大約50萬筆資料
      2. 爬取時間：9個小時
      3. 爬取網站
         1. Ettoday：起初，處理html request非常快速，然後到後期速度有減緩的跡象
         2. 華視新聞網：相當強大的網站，處理html request非常快速
         3. 風傳媒：爬取解果較差，每個request會隔一段時間才回傳
   2. 正文抽取

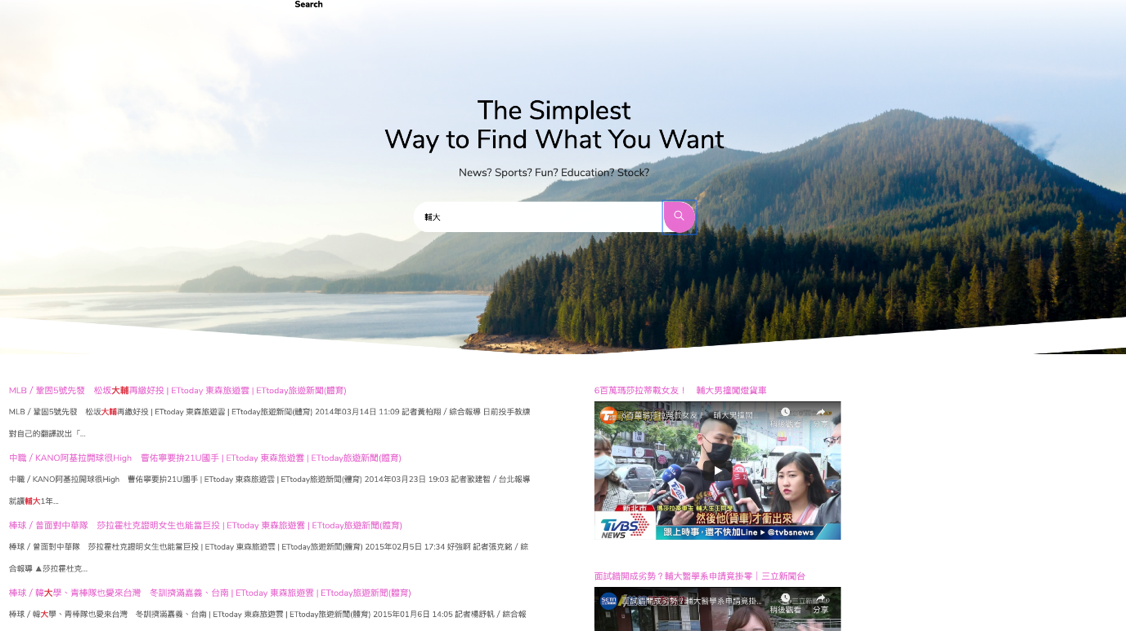
將此專案爬蟲的正文擷取與[2]論文中正文擷取方法做比較，計算precision，recall和F1 socre，從CETD此資料集中（600個html檔與正文）隨機選取22個html檔，擷取其中10筆結果：







(Qt GUI介面，可以一目了然整個爬蟲數據)



(縮小後的搜尋介面，左邊為新聞網頁搜尋結果，右邊為新聞影音片斷搜尋結果)

Reference

1. Fei Sun “DOM Based Content Extraction via Text Density”
2. 洪鸿辉 “基于文本及符号密度的网页正文提取方法“
3. <http://tidy.sourceforge.net/>

分工

* 劉昌澈
  + 正文抽取
  + 記憶體清除
  + Master slave 架構
* 林柏亨
  + Crawler qt介面
  + Query 介面
  + Elasticsearch