基於 Apache Spark 建構串流 XML Veracity 真實度之模型

中文摘要

近幾年大數據的數據量飛快地成長，超過 TB 等級的數據已是隨處可見，而數據傳輸在大數據中是一個重點探討的問題，大數據在做資料交會的時候，由於資料量龐大，所以資料的真實度不易掌控，也就是大數據 5V 當中的 Veracity 是我們最關切的問題。

而 XML(延伸標記語言 eXtensible Markup Language) 作為現今通用的網路資料交換格式，隨著網際網路資料的增長，也已經同樣具有大數據 (Big Data) 的特徵，在近幾年來，產業界與學界都將大數據列為重要研究議題，並投入相當多的資源支持大數據的研究。

本研究提出使用 Apache Spark 建構串流 XML Veracity 真實度模型以及真實度模型應用程式開發介面 (Veracity Model Application Programming Interface, Veracity Model API)，來解決大數據在資料傳輸當中我們所關心的真實度 Veracity 的問題，並可以讓使用者知道文件有多少可信度，XML 真實度模型計分是一個給定一個被量測 XML 文件，讓模型進行被量測 XML 文件與標準 XML 文件的比較，並給定一個量化分數的系統，本研究建構一個利用 Apache Spark 的平行化處理，加速真實度模型整體的運算速遞及效能的模組，讓使用者進行串流的 XML 文件上傳，再將上傳文件放入由 Apache Spark 建構的真實度計分模型來進行真實度計分，XML 文件真實度有很多面向可以做詮釋，本研究設計一個基於物件導向程式設計的真實度模型，藉由物件導向的特性，設計抽象類別來建構一個真實度的基本框架 ($Dimension\ i, D_i$)，再讓使用者繼承此類別實作模型的功能，換言之就是讓使用者決定需要有什麼樣的屬性 ($Attribute\ j, P_{i,j}$) 才可以評價真實度，以及需要有什麼樣的量化和計分方式才可以將真實度呈現出來，如此這個模型的設計將具有彈性且有助於使用者在面對大量 XML 資料的時候能夠有一個真實度計分的量化標準，在未來開發大數據應用的時候，可以對於在傳輸資料的時候有一個依據來知道資料的可靠程度。

Contents

List of Figures

1 緒論

1.1 背景

近年來數據以飛快的速度成長，TB 或是 PB 等級的數據隨處可見，在這資料快速產生且數據快速交換的時代，大數據一詞也越常被提及，國際數據公司 (International Data Corporation, IDC) 有研究指出，2008 年全球生產的資料量為 0.49ZB，2009 年全球產生的資料量為 0.8ZB，2010 年增長為 1.2ZB，2011 年的資料量更是已經達到 1.82ZB，這相當於全球每人生產 200GB 的資料，這麼龐大的資料也成了產業界與學術界所需要探討的重要議題，而有這麼大量的資料也意味著會有大量的應用會產生，而這一些應用當中一定會需要資料的交換，而在教會資料的時候，大多數的應用會選擇 XML。

在大數據中，有所謂的 5V，所謂的 5V 是指 Volume, Value, Veracity, Vleocity, Variety，Volume 是指產生的資料量;Value 是指資料的價值;Veracity 是指資料的可信度或是真實度;Velocity 是指資料產生的速度;Variety 是指資料的多樣性。而大數據在做資料交換的時候我們所關心的是資料的可信度或是真實度，也就是上述所提到的 Veracity，這會影響到我們在做資料分析的結果可信度，假使進來的資料不具可信度的話，那麼所分析出來的結果自然也就不具有任何價值，本研究基於 Apache Spark 建構 XML Veacity 真實度模型，利用 Apache Spark 分散式大數據處理引擎，來計算並建構大型 XML 文件的 Veracity 模型，並計算串流或批次輸入文件的 Veracity 真實度評分。

1.2 XML

可延伸標記式語言 (Extensible Markup Language，簡稱 XML)，是一種作為資料交換的結構化資料格式，XML 是從標準通用標記式語言 (SGML) 中簡化修改出來的，XML 是從 1995 年開始有雛形，並向全球資訊網聯盟 (World Wide Web Consortium，簡稱 W3C) 提案，而在 1998 年 2 月發布為 W3C 的標準 (XML 1.0)。

XML 的特色在於文件內的標籤名稱可以由使用者自行定義，而 XML 文件必須是結構完整的 (well-formed)，所謂的 well-formed 是指 XML 除了要符合每一個標籤都要有起始元素之巢狀結構之外，還要符合格式規範，在 XML 中我們為了確保文件的格式正確性，會使用 DTD(Document Type Definition) 或是 XML Scheme，DTD 是 XML 提供的文件驗證機制，這是用來定義文件合法區塊，也就是定義元素的架構，有使用 DTD 的 XML 都必須依照 DTD 所定義的格式來呈現，下面是一個使用 DTD 且 well-formed 的 XML 文件範例：

```
<?xml version = "1.0"?>
<!DOCTYPE note[
  <!ELEMENT note (to, from, heading, body)>
  <!ELEMENT to (#PCDATA)>
  <!ELEMENT from (#PCDATA)>
  <!ELEMENT heading (#PCDATA)>
  <!ELEMENT body (#PCDATA)>
]>
<note>
  <to> Tove </to>
  <from> jani </from>
  <heading> Reminder </heading>
  <body> Don't␣forget␣me␣this␣weekend!␣</body>
</note>
```

但由於 XML DTD 並不能完全滿足 XML 自動化處理的要求，也缺乏對於文件結構屬性和資料屬性的規範，所以 W3C 在 2001 年的時候提出 XML Schema，XML Schema 使用的語法與 XML 相同，且支援四十多種資料類型，下面是一個使用 XML Schema 且使用此 Schema 驗證的 XML 文件範例：

```
<?xml version = "1.0"?>
<card xmlns = "http://businesscard.org"
  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation = "business.xsd">

  <name> John Doe </name>
  <title> CEO, Widget Inc. </title>
  <email> john.doe@widget,com </email>
  <phone> (202) 456-1414 </phone>
  <logo url = "widget.gif"/>
</card>
```

```
<schema xmlns = "http://www.w3.org/2001/XMLSchmea"
    xmlns:b = "http://businesscard.org"
    targetNamespace = "http://businesscard.org">

  <element name = "card" type = "b:card_type" />
  <element name = "name" type = "string" />
  <element name = "title" type = "string" />
  <element name = "email" type = "string" />
  <element name = "phone" type = "string" />
  <element name = "logo" type "b:logo_type" />

  <complexType name = "card_type">
    <sequence>
      <element ref = "b:name" />
      <element ref = "b:title" />
      <element ref = "b:email" />
      <element ref = "b:phone" minOccurs = "0" />
      <element ref = "b:logo" minOccurs = "0" />
    </sequence>
  </complexType>

  <complexType name = "logo_type">
      <attribute name = "url" type = "anyURI" />
  </complexType>
</schema>
```

上面有提到大數據資料真實度的重要性，假設資料缺乏真實度或是可信度，將增加資料處理的負擔，所以判斷資料的真實度是十分重要的事情，而要對大數據資料或是大型 XML 做資料真實度判斷有以下兩點困難

1. 資料來源多元且格式各異，有些 XML 有使用 DTD，有些沒有，再來是 XML 標籤名稱是可以自行指定的，所以導致 XML 的資料格式不一。
2. 假設我們要求資料的一致性，那麼我們在資料進入系統的時候必須做資料清洗的動作，但是在大數據的資料或是大型 XML 的資料規模巨大，所以必須耗費大量的時間在資料清洗上面，在處理速度和資料真實度之間取捨，資料分析之後的結果要能夠呈現出資料的真實度以及可信度，所以真實度的重要性可想而知。

根據上述原因，XML 文件的 Veracity 真實度這個特性是會有不同的變化的，例如 XML 文件如果以是否符合 DTD 規範，well-formed 以及 vaild 的 XML 文件在真實度 Veracity 的特性上面就會有不同的變化，我們以兩份 well-formed XML 來說，上述有提到 XML 文件的標籤是可以讓使用者自行定義的，所以會造成兩份文件出現相同標籤名稱，意思不同，或是不同標籤名稱，意思相同的狀況，此時真實度應該很高，接著可以再比對這兩份 XML 文件是否使用相同的 DTD 或是 Scheme 規範，這樣真實度可以再提高。

1.3 Apache Spark

在以前的大數據計算解決方案大家通常採用的是 Hadoop [?]，而 Hadoop 是由兩個元件所構成的，分別是 MapReduce 和 HDFS(Hadoop Distributed File System)，MapReduce 是由 Map 和 Reduce 所組成的，Map 的動作是將大的計算任務切割成小任務來操作，Reduce 是將前面 Map 切出來並算完的小任務做合併，來取得最終的結果。HDFS 是 Hadoop 下的分散式檔案系統，其功能是將大檔案切割成小檔案作儲存與備援，因為大數據的檔案大小或是檔案數量都已經是 TB 或是 PB 等級了，這樣的檔案大小也已經不是單一台機器所能夠儲存的，所以就需要像這樣的分散式檔案系統，分散式檔案系統是將單一個檔案切割成數份，分別複製及儲存到不同機器當中，以達到儲存大型檔案的目的，這樣做的好處有三個：

- 可容錯性 (Fauit Tolerance)
- 可擴展性 (Data Scalability)
- 並行性 (High Concurrency)

可擴展性是當我們儲存空間不足的時候可以非常輕易的做擴充；容錯性是指當儲存節點有損壞的時候，我們可以從其他節點找到遺失的資料切片的備份來還原資料本身；並行性是指藉由分散是檔案系統可以讓資料並行處理。

Apache Spark [?] 為一個開源的大數據分散式引擎，最初是由加州大學柏克萊分校 AMPLab 所提出，為 In-memory 的計算，跟傳統的 Hadoop 相比，Apache Spark 的計算效率比起 Hadoop 來說有顯著的提升，其原因為 Apache Spark 在運算的時候，將運算中間產生的資料暫存在記憶體中，因此可以加快整體運行速度，而 Hadoop 則是在每一次計算完成或是產生中間資料的時候，都必須對硬碟動作，這個動作降低了 Hadoop 的執行效率，而 Apache Spark 的設計則能夠增加計算的效率。除了上述提到的效能問題之外，Hadoop 只支援 Map 和 Reduce 這兩種運算，對於現今要處理的資料來說，在撰寫程式的時候靈活度不夠，而且 MapReduce 在運行任務的時候，任務排程以及啟動開銷大，基於上述原因，Apache Spark 是目前較好的大數據處理引擎。

Apache Spark 主要的對資料的操作是使用叫做 RDD(Resilient Distributed Datasets，彈性分佈資料集) [?]，Fig 1. 是 RDD 的結構，在 Fig 1. 中黃色的區塊是 RDD 當中所謂的 Partition，是指資料的分片，而 RDD 大多數的情況都是儲存在記憶體當中，也就是說一個 RDD 裡面會有多個在不同機器上的 Partition，RDD 是一個可以並列操作且有容錯機制的資料集合。RDD 可以透過參照外部儲存系統的資料集建立，或者是透過現有的 RDD 轉換而創建，例如 map, join, reduce 等，而在 Apache Spark 對於 RDD 的操作中，有分為轉換 (Transformation) 和動作 (Action)，Apache Spark 在做資料操作的時候，是所謂的惰性求值，也就是說當 Apache Spark 在做 Transformation 的時候，並不是馬上會做資料的轉換，而是會先把要轉換的動作記錄下來，等到有呼叫 Action 的 API 的時候才會依照剛才做資料的操作並輸出結果，這樣可以使執行效能提高，例如一個資料經過 MapReduce 處理後會得到一個結果，而不是返回一個大的資料集。

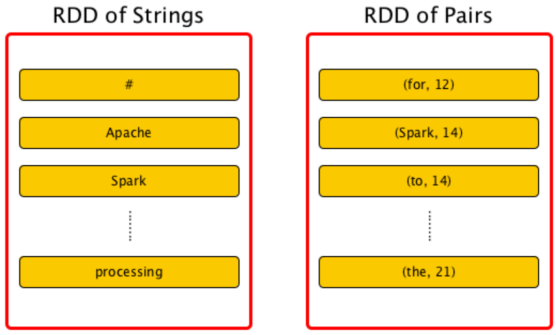

Figure 1: RDD 結構

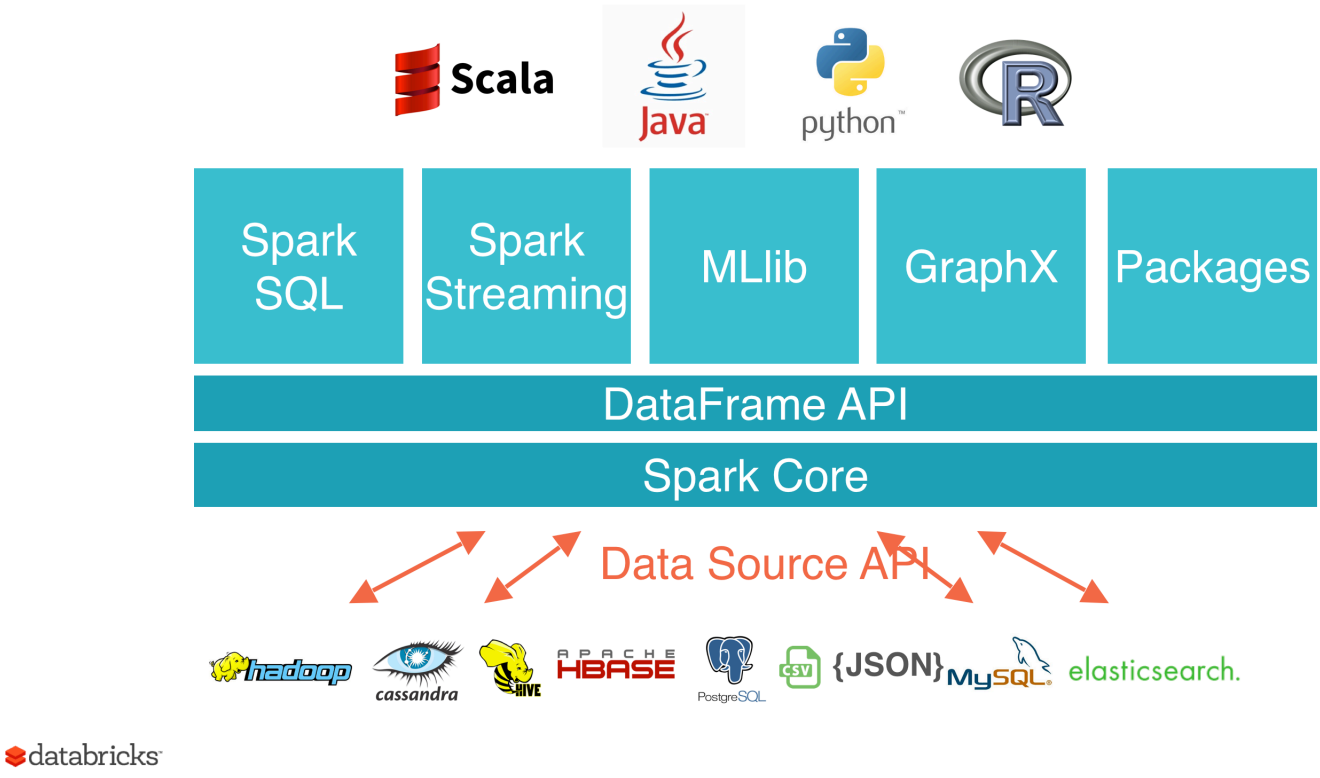

Figure 2: Apache Spark 核心架構

Apache Spark 有著豐富的函數庫，也針對 Python, Java, Scala, R 等程式語言提供一致的 API，核心架構如 Fig.2，下面是 Apache Spark 所提供的 API

- Spark Core
- Spark SQL
- Spark Streaming
- MLlib
- GraphX

Spark Core 是整個 Apache Spark 的基礎，提供了分散式任務調度、排程和基本 I/O ，而主要的程式抽象化結構 RDD 也是定義在這邊，RDD 抽象化是經由 Apache Spark 所支援的程式語言整合 API 呈現的，簡化了寫程式的複雜性。

Spark SQL 在 Spark Core 上定義了一個叫做 SchemaRDD 的資料抽象化概念，提供結構化和半結構化資料的相關支援，另外 Spark SQL 允許使用者使用 SQL 遠法做資料查詢，也允許程式開發人員將 SQL 查詢與其他 RDD 所支援的資料處理方式一起使用。

Spark Streaming 是在處理即時串流資料的元件，例如 web server 所產生的紀錄，或是服務狀態的改變，Spark Streaming 擷取了微批次 (micro batch) 的資料並對資料執行 RDD 的轉換。

MLlib 是 Apache Spark 所提供的機器學習 (Machine Learning) 函數庫，MLlib 可以使用許多常見的機器學習和統計學的演算法，大幅度簡化了機器學習的時間，其中有：

- 匯總統計：相關性、分層抽樣、假設檢定、亂數據產生

- 分類與回歸：支援向量機 (Support Vector Machine，簡稱 SVM)、迴歸、線性迴歸、邏輯迴歸、決策樹 (Decision Tree)、樸素貝葉斯 (Naive Bayes)
- 協同過濾：ALS
- 分群：K 平均演算法 (K-means)
- 維度縮減：奇異值分解 (Singular Value Decomposition，簡稱 SVD)，主成份分析 (Principai Components Analysis，簡稱 PCA)
- 最佳化：隨機梯度下降 (Stochastic Gradient Descent，簡稱 SGD)，L-BFGS

GraphX 是 Apache Spark 上的分散式圖形處理框架，此框架可用於表達圖形計算並可以類比 Pregel 抽象化，GraphX 提供了許多處理圖像的操作，例如 subgraph 和 mapVertices 以及常見的圖形演算法。

1.4 大數據

所謂的大數據 [**?**] 是指無法以人工管理或是處理的資料集，也可以定義為多來源的結構化或非結構化的資料，而在大數據當中，我們常常用”5V” 來描述大數據的特性，這 5V 分別是規模性 (Volume)、價值 (Value)、真實性 (Veracity)、時效性 (Velocity) 及多樣性 (Variety)。

5V 當中，Volume 所代表的是大數據的規模，也就是大數據以傳統的儲存方式已經無法負荷此龐大的資料量或是傳統的資料處理方式已無法對其做運算了；Value 是指大數據透過處理後所得到的資料產生的價值；Veracity 是指資料的品質，可信度，資料是否可靠，或是結構的完整性；Velocity 是指大數據資料產生的速度；Variety 是指大數據的資料多樣性，在設計有關大數據的應用的時候，藉由上面五個的特性有助於檢視大數據的特徵。

而在這 5V 當中，本研究所要探討的是真實度 Veracity 在 XML 文件的建模與應用，在陳宇威學者的論文 [**?**] 當中有提到，大數據可以由兩個面向來討論，一個是資料理解性，一個是大數據的評價基準，而本研究所針對資料理解性面向進行建模，假設有 n 份基準 XML 文件以及 m 份被測量 XML 文件，使用者可以從多個角度對於 XML 文件進行真實度計分，XML 文件真實度包含的面向極廣，對於真實度的定義每一個人不盡相同，所以本研究建構一個在 Apache Spark 叢集上的 Veracity 真實度模型，使用物件導向程式設計 (Object-Oriented Programming，簡稱 OOP) 的觀念，設計一個維度、屬性以及評分的抽象類別，將基本架構定義完整，做成 Veracity Model API 且是跑在 Apache Spark 叢集上，將實作細節交給使用者來決定，前面有提到使用者可以從不同的角度來評價 XML 的真實度評分，以及決定真實度要有多少維度以及多少屬性。並且本研究的模型應用在串流 XML 上，有別於以往傳統批次文件的處理方式，串流的資料處理方式更符合現代的資料處理架構及場景，針對一個大數據串流 XML 真實度評價模型是一個除了評價資料真實度之外，也可以篩選真實度較高之資料的解決方案。

2 相關研究

本研究建構一個在 Apache Spark 叢集上的 Veracity 真實度模型，且應用於串流 XML 的真實度計算，而 XML 具有樹狀結構的特性，所以在分散式系統處理 XML 的時候，如何切割文件，但依然保有 XML 文件樹狀結構和父子節點的關係，以及在 Hadoop 或是 Apache Spark 的分散式架構下做 XML 的 Query，還有 OOP 的一些實作探討，本章節將就有關這些議題的參考文獻來做討論。

2.1 XML 文件樹狀結構的關係

2.2 大型 XML 文件的平行化

3 模型設計

4 Veracity 模型 API

本研究使用 OOP 的觀念與特性在 Apache Spark 上面實做真實度模型，在前面有提到對於資料的真實度評價可以從很多面向來做探討，換句話說，可以讓使用者決定所謂的真實度需要包含哪一些屬性，還有真實度的量化方法、量化公式等，藉由 OOP 的方式，可以讓 Veracity 真實度模型的詮釋方式更有彈性，量化方式可以隨著不同的應用做變化，搭配 Apache Spark 可以加速整個對於大型 XML 文件處理的效能。

4.1 物件導向程式設計

物件導向程式設計 (Object-Oriented Programming，簡稱 OOP) 是一種抽象化的程式開發方法，

4.2 模型建立

Veracity 真實度模型的建立，其中最重要的就是彈性，也就是用物件到像程式設計的特性來達到

4.3 模型實作