Laboratorio di Web Scraping
CRAFTY:
Clustering and scRAping For biTcoin deanonYmization
Progetto di Fine Corso A.A. 2022/23

Lo scopo del progetto è l'implementazione di un insieme di analisi e tecniche di deanonimizzazione per la blockchain di **Bitcoin**. Viene fornito un DataSet di **Bitcoin** che contiene una selezione delle transazioni incluse nei blocchi compresi tra il blocco genesis, minato in data **03-01-2009, 17:15:05** e il blocco di altezza **214562**, minato in data **31-12-2012, 11:52:37.**
Il DataSet è stato ottenuto tramite una serie di trasformazioni effettuate sui dati pubblici reperiti dalla blockchain di **Bitcoin**. In particolare, per diminuire la dimensione del DataSet:

- sono state eliminate tutte le transazioni **Coinbase** il cui output non è stato speso alla data dell'ultimo blocco.
- gli hash delle transazioni, gli indirizzi contenuti negli output delle transazioni, e gli script sono stati sostituiti con identificatori univoci interi. La corrispondenza tra gli indirizzi della blockchain e gli identificatori univoci del DataSet è stata memorizzata in un ulteriore file di mapping.

1. Descrizione del DataSet

Il DataSet consiste di **4 files CSV**

- **transactions.csv**, che contiene una riga per ogni transazione del DataSet, con i campi:
  - **timestamp**: timestamp del blocco che contiene la transazione. Corrisponde al tempo **UNIX** del miner, e indica il momento in cui il blocco è stato minato
  - **blockId:** identificatore del blocco che contiene la transazione. Indica l'altezza di tale blocco, ovvero la sua distanza dal blocco genesis di **Bitcoin**
  - **txId**: identificatore unico della transazione corrispondente all'hash del contenuto della transazione
  - **isCoinbase:** indica se la transazione è una **Coinbase**, ovvero una transazione che trasferisce la ricompensa al miner che ha risolto la **PoW** (0 false, 1 true)
  - **fee:** eventuale commissione volontaria contenuta nella transazione, attribuita al miner che la inserisce in un blocco. Può essere zero.
- **inputs.csv**, che contiene una riga per ogni campo di input di ogni transazione del DataSet, con i campi:
  - **txId:** identificatore della transazione all'interno della quale si trova questo input
  - **prevTxId:** identificatore della transazione che ha creato l'output attualmente speso da questo input
  - **prevTxpos**: posizione dell'output attualmente speso come input, all'interno della transazione che lo ha creato (diversa da quella che contiene questo input)
- **outputs.csv**, che contiene una riga per ogni campo di output di ogni transazione del DataSet, con i campi:
  - **txId**: identificatore della transazione all'interno della quale si trova questo output
  - **position**: posizione di questo output all'interno della transazione che lo ha creato
  - **addressId:** indirizzo a cui viene inviato questo output, è un identificatore univoco che viene mappato sull'indirizzo reale tramite il file di mapping
  - **amount:** valore trasferito da questo output

- **`scripttype:`** codice che identifica lo script contenuto in questo output. Gli script possono essere di diversi tipi (la Tabella 1 mostra i tipi di script definiti da **`Bitcoin`** e il rispettivo codice contenuto nel DataSet). Tuttavia, dato che il DataSet contiene solo transazioni generate nei primi 4 anni di vita di **`Bitcoin`**, solo i primi 4 script della tabella sono significativi per questo DataSet. Se lo script è di tipo 0 significa che lo script non è standard e spesso non ha un address associato.

- **`mapping.csv`**, file di mapping degli indirizzi, campi:
  - **`hash:`** hash del corrispondente indirizzo contenuto nella blockchain di Bitcoin
  - **`addressId:`** identificatore unico di ogni indirizzo contenuto in almeno un output delle transazioni del DataSet.
    Nel caso di output con script di tipo 0 che non contengono address, nel file di mapping si trova un identificatore univoco rappresentato da una # seguita da un numero che rappresenta quell'output e solo quello, associato con l'identificatore utilizzato per quell'output nel DataSet. .

La struttura del DataSet è mostrata in Fig.1.

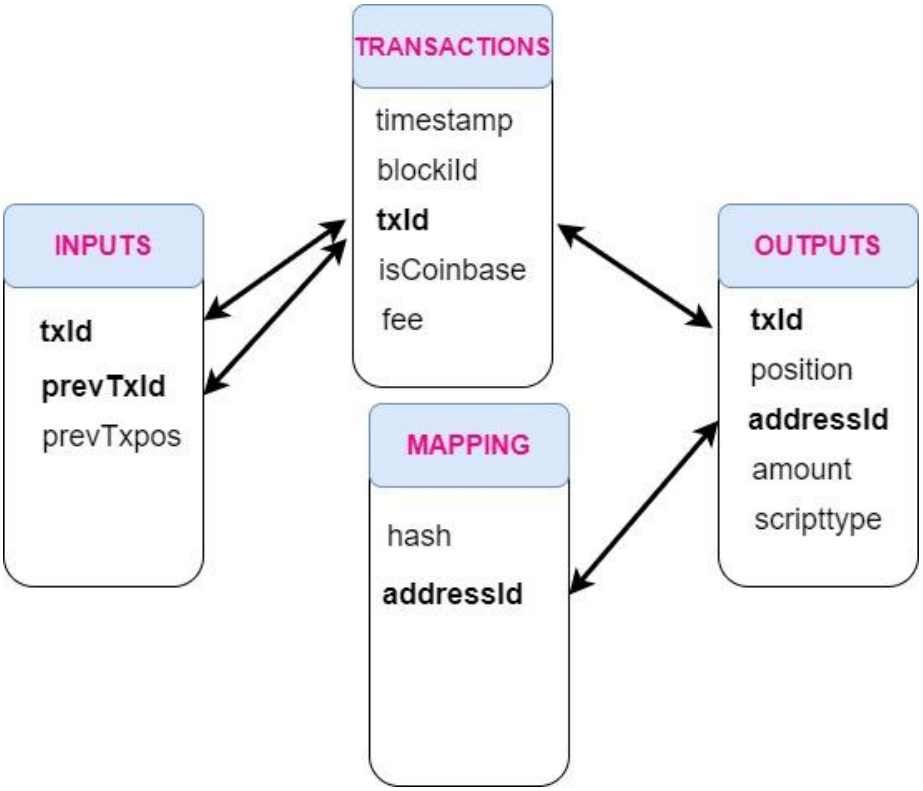

Figura 1: Struttura del DataSet

Script Code	0	1	2	3	4	5	6	7
Script Type	Unknown	P2PK	P2KH	P2SH	RETURN	EMPTY	P2WPKH	P2WSH

Tabella 1: codifica degli script

Il DataSet è disponibile su Drive al link:

https://drive.google.com/file/d/1q-nmPc3VGdiP4e3Qa83ShMJ7qlpF78Fy/view

2 Analisi richieste

Il progetto richiede l' implementazione delle seguenti analisi

2.1 Analisi generali dei dati della blockchain

E' richiesto di implementare le seguenti analisi sul segmento iniziale di transazioni contenuto nel **`DataSet`**

- distribuzione del numero di transazioni per blocco (occupazione del blocco), nell'intero periodo temporale considerato
- evoluzione dell'occupazione dei blocchi nel tempo, considerando intervalli temporali di due mesi. In questo caso produrre un grafico che riporti il numero di transazioni medie per ogni periodo considerato
- ammontare totale degli **`UTXO`** al momento dell'ultima transazione registrata nella blockchain considerata
- distribuzione degli intervalli di tempo che intercorrono tra la transazione che genera un valore in output (**`UTXO`**) e quella che lo consuma, per gli output spesi nel periodo considerato.
- una ulteriore analisi proposta dallo studente

Per ogni analisi scegliere il meccanismo di plotting di **`Matplotlib`** più adeguato.

2.2 Clusterizzazione degli indirizzi di Bitcoin: euristica multi-input

In questa parte del progetto viene richiesto di implementare un insieme di tecniche di *deanonimizzazione* il cui scopo è associare un''identità ad alcuni indirizzi di Bitcoin. La prima tecnica dell'*address clustering* è la base per la procedura di deanonimizzazione mostrata nel paragrafo 2.3 e consiste nel clusterizzare insiemi di indirizzi che sono probabilmente controllati dallo stesso utente **`Bitcoin`**, basandosi sull'analisi delle transazioni. L'address clustering utilizza un'euristica descritta nella Sezione IV-A di [1], e consiste nel considerare le transazioni che hanno un numero di input maggiore di 1, e nell'associare ad uno stesso utente tutti gli indirizzi utilizzati negli input della stessa transazione. L'idea alla base dell'euristica si basa sul fatto che, per rendere valida una transazione, occorre fornire le firme utilizzando le chiavi private corrispondenti ad ogni chiave pubblica/indirizzo utilizzato in input dalla transazione. Lo scenario più probabile in cui questo avviene è quello in cui chi genera la transazione controlla tutte le chiavi private utilizzate per firmare gli input e quindi è lo stesso utente. Si noti che l'algoritmo di clustering deve permettere la fusione nello stesso cluster di indirizzi provenienti da transazioni diverse. Questo può avvenire quando due transazioni diverse hanno un indirizzo in comune, possibile perché **`Bitcoin`** permette a un utente di riutilizzare lo stesso indirizzo in transazioni diverse.

Si richiede di:

- implementare un algoritmo di clustering che realizzi l'euristica descritta. Viste le dimensioni del DataSet è necessario limitare la complessità dell'algoritmo, in particolare si dovrà

implementare un algoritmo di complessità lineare rispetto al numero di indirizzi diversi contenuti nel **DataSet.** A tale scopo si può prendere spunto dalle slide presentate a lezione in data **5 Aprile 2023** e pubblicate sul sito del corso

- produrre alcune statistiche descrittive del clustering ottenuto (dimensione media, minima e massima dei cluster, distribuzione delle loro dimensioni, etc.)

2.3 Deanonimizzazione degli indirizzi

Una volta ottenuti i cluster, se si riesce a deanonimizzare anche solo un indirizzo appartenente al cluster, allora tutti gli indirizzi del cluster possono essere associati alla stessa entità.
L'ultima parte del progetto consiste quindi nell'implementare un **web scraper** il cui scopo è provare a deanonimizzare alcuni degli indirizzi contenuti nei cluster ottenuti al passo precedente. E' richiesto di

- considerare i 10 cluster di dimensione maggiore ottenuti al passo precedente
- prendere in considerazione gli indirizzi contenuti in questi cluster e provare a deadonimizzarli considerando i seguenti siti web:
    - *WalletExplorer:* si tratta di un servizio che collega un insieme di indirizzi **Bitcoin** a servizi noti (ad esempio piattaforme di exchange, servizi di gambling, marketplaces, etc.). Gli indirizzi sono stati individuati con una procedura simile a quella illustrata in questo progetto (clustering+ scraping), Link: https://www.walletexplorer.com/
    - *Bitcoininfocharts* Link: https://bitinfocharts.com/.
      Esempio: le informazioni relative al wallet di Binance si possono trovare alla URL https://bitinfocharts.com/bitcoin/wallet/Binance-coldwallet, con il comando "**Show addresses in Binance-coldwallet**" viene fornita una lista di indirizzi dell'exchanger **Binance**.
      Se invece si conosce un indirizzo **Bitcoin**, ad esempio **1NDyJtNTjmwk5xPNhjgAMu4HDHigtobu1s,** è possibile costruire una URL come la seguente
      https://bitinfocharts.com/bitcoin/address/1NDyJtNTjmwk5xPNhjgAMu4HDHigtobu1
      che punta al wallet di **Binance**
      https://bitinfocharts.com/bitcoin/wallet/Binance-wallet

Opzionale: per la deanonimizzazione, è possibile anche analizzare forum, come https://bitcointalk.org/, o subreddit come **Bitcoin, the currency of the Internet**, https://www.reddit.com/r/Bitcoin/, in cui gli utenti di **Bitcoin** talvolta pubblicano i loro indirizzi su cui ricevere pagamenti o per scambiarsi informazioni sui servizi collegati a **Bitcoin**.

3. Modalità di svolgimento e di consegna del progetto

Il progetto deve essere eseguito individualmente.
E' possibile scaricare il DataSet di riferimento da Drive,
Link: https://drive.google.com/file/d/1q-nmPc3VGdiP4e3Qa83ShMJ7qlpF78Fy/view. Il riferimento è a **Google Drive** fornito da Unipi, per cui l'accesso dovrebbe essere consentito con credenziali Unipi. In caso di difficioltà nell'accesso, inviare una mail a **laura.ricci@unipi.it.**
Il materiale da consegnare comprende:

- codice dell'applicazione (Notebook **.ipynb** e/o script **Python**).
- una breve relazione, eventualmente contenuta nel notebook **.ipynb.**
- il codice deve essere sviluppato in **Python 3.0** e per la parte di scraping si devono utilizzare le librerie **BeautifulSoup** e (eventualmente) **Selenium**

Relazione e codice sorgente devono essere consegnati su Moodle in un unico archivio compresso in formato zip.

Nel caso le dimensioni del DataSet si rivelassero troppo elevate per le risorse computazionali che lo studente ha a propria disposizione, potete mandate una mail a **laura.ricci@unipi.it**, per ricevere un DataSet ulteriormente ridotto.

Riferimenti

[1] *"Heuristics-Based Address CLustering in Bitcoin"*, Yuhang Zhang, Jun Wang, Jie Luo, **IEEE Access, Volume 8, 2020.**