Seguiment de transaccions de Bitcoin

Guillem Lázaro Ortega

Resum

El projecte tracta de dissenyar i implementar una eina que automàticament faci un seguiment de les transaccions de Bitcoin donada una adreça inicial per la qual volem començar. Aquesta eina ha de generar un arbre que representa el flux de transaccions, és a dir, el seguiment de les transaccions d'un pagament en la cadena de blocs de Bitcoin. A més, s'apliquen algunes tècniques que permeten agrupar adreces de Bitcoin que probablement pertanyen al mateix usuari, o bé que identifiquen patrons, anomenades tècniques de "clustering". Es fa un ús intensiu d'APIs específiques per extreure dades i representar-les visualment. A més, es té en compte alguns aspectes de seguretat i privacitat, complint amb el Reglament General de Protecció de Dades (RGPD) i altres normatives aplicables.

Paraules clau: Cadena de blocs, Bitcoin, Transaccions, Clustering, Eina de seguiment, Seguretat, Privacitat, API Blockchain Explorer, Python.

Abstract

The project is about designing and implementing a tool that automatically tracks Bitcoin transactions given an initial address that we want to start with. This tool must generate a tree that represents the flow of transactions, that is, the tracking of the transactions of a payment on the Bitcoin blockchain. In addition, some techniques are applied that allow you to group Bitcoin addresses that probably belong to the same user, or that identify patterns, called "clustering" techniques. Extensive use is made of specific APIs to extract data and represent it visually. In addition, some security and privacy aspects are taken into account, complying with the General Data Protection Regulation (GDPR) and other applicable regulations.

Index Terms: Blockchain, Bitcoin, Transactions, Clustering, Tracking Tool, Security, Privacy, API Blockchain Explorer, Python.

1 Introducció - Context del treball

La tecnologia blockchain ha transformat profundament el món financer, aportant una manera nova i innovadora de gestionar transaccions digitals d'una forma descentralitzada, segura i transparent. Dins d'aquest univers, Bitcoin s'ha consolidat com una de les criptomonedes més influents, gràcies al seu abast global i al seu ús com a actiu digital. Tot i això, la seva naturalesa pseudoanònima planteja desafiaments importants, especialment pel que fa al seguiment de transaccions i a la identificació de patrons o actors dins la xarxa.

Aquest treball té l'objectiu de desenvolupar una eina capaç de rastrejar el flux de transaccions de Bitcoin a partir d'una adreça inicial. Mitjançant aquesta eina, es genera un arbre visual que representa el seguiment de les transaccions al llarg de la blockchain.

Per fer-ho, s'han aplicat tècniques com el clustering d'adreces i heurístiques dissenyades per agrupar adreces que corresponen a un mateix usuari o entitat, cosa que permet simplificar i estructurar la informació.

A més, aquest projecte no només busca ser útil per al rastreig de possibles transaccions sospitoses o per auditories, sinó que també pren constància de temes rellevants com la seguretat, la privacitat i l'ètica en l'ús de les dades blockchain. També fa ús de tecnologies modernes, com APIs especialitzades, per obtenir les dades necessàries i aconseguir una visualització atractiva i funcional.

Aquest document està estructurat per oferir una visió clara del procés seguit. Comença amb una explicació del context i els objectius del treball, seguit de la metodologia, el desenvolupament tècnic i els resultats obtinguts. Finalment, es presenten les conclusions i propostes de millora.

Aquest projecte pretén ser una contribució pràctica i significativa al camp de l'anàlisi de la blockchain i la seva aplicació en la traçabilitat de transaccions.

2 Objectius

2.1 Objectiu general

L'objectiu principal del projecte és el de dissenyar una eina automatitzada que generi el seguiment de les adreces i transaccions en forma d'arbre. Tracta de generar una eina que faci el seguiment de manera automàtica de les

- *E-mail de contacte: 1562333@uab.cat*
- *Menció realitzada: Tecnologies de la Informació*
- *Treball tutoritzat per: Guillermo Navarro Arribas, Departament d'Enginyeria de la Informació i de les Comunicacions.*
- *Curs 2024/25*

transaccions i les mostri visualment en forma d'arbre. Es tenen en compte les certes tècniques de clustering i es fa un correcte anàlisi de dades.
D'aquest objectiu principal, sorgeixen els següents objectius més concrets a assolir durant la realització del treball. Estan ordenats per ordre de prioritat en la seva realització.

- **Entendre com funciona la blockchain de bitcoin i la seva infraestructura**

Per tal de realitzar el projecte amb coneixements sòlids sobre la cadena de blocs i temes relacionats amb el bitcoin és interessant conèixer aspectes tècnics i el seu funcionament. És per això que el primer objectiu a assolir és saber com es registren les transaccions dins de la blockchain i conèixer aspectes de seguretat i transparència per utilitzar correctament les dades obtingudes.

- **Definir estratègies pel seguiment de les transaccions**

En aquest objectiu el més important és analitzar diferents tècniques de seguiment de transaccions en la blockchain i definir les que són més adients. Aquest pas ha d'incloure la identificació de cada una de les adreces, permetent així la seva agrupació quan es detecta certa relació entre elles.

- **Investigar i estudiar tècniques per ajuntar adreces ("clustering")**

L'objectiu és investigar, estudiar i entendre algunes de les heurístiques existents per agrupar adreces que es creuen que pertanyen al mateix usuari.

- **Garantir aspectes a tenir en conte, com ara la seguretat i privacitat, durant la realització del treball**

S'han de tenir en compte alguns dels aspectes que són importants durant el desenvolupament del projecte i comentar en més o menys mesura. Alguns d'aquests podrien tenir relació amb la seguretat, la privacitat, aspectes legals o fins i tot una correcte presentació de l'"output" obtingut.
D'aquest objectiu se'n fa un estudi previ però s'aplica en tot el procés.

- **Implementar l'eina automatitzada en llenguatge Pyhton.**

És la part principal del projecte. Com a objectiu conté aplicar una de les tècniques d'agrupament d'adreces estudiat per a fer un seguiment més acurat i d'aquesta manera "estrènyer" l'arbre visual de l'eina de seguiment. Dins de l'objectiu, trobem les següents tasques:

- Guardar l'arbre generat

Una vegada l'arbre es genera correctament, s'ha de poder mostrar i guardar. El format triat per a guardar l'arbre generat és JSON.

- Comprovar si la generació de l'arbre és fiable mitjançant dades reals de la blockchain

És important, una vegada generat el fitxer, comprovar que el resultat obtingut és l'esperat i que la seva funcionalitat és la correcta i la esperada. Es pot provar i validar la eina generada amb dades reals de la cadena de blocs, per garantir-ne la seva efectivitat, fiabilitat i precisió.

2.2 Objectiu opcional

- **Millorar l'eina de seguiment**

Es poden estudiar costos i camins dels diferents seguiments de les transaccions per extreure'n dades d'interès. Aquest és un objectiu que ha sorgit cap a les últimes setmanes de treball i que no és prioritari. La prioritat és aplicar bé la funcionalitat de l'eina de seguiment.

3 METODOLOGIA

Per garantir que s'aconsegueixen els objectius proposats dins del termini marcat en la planificació, he plantejat un desenvolupament incremental de les eines de seguiment, que asseguri la continuïtat de la mateixa: començar per una versió bàsica i senzilla que mostri resultats, i a posteriori anar implementant les tècniques de clustering que afegiran complexitat al projecte.

S'utilitza el llenguatge de programació Python, així com articles, tesis, informes tècnics, llibres, cursos, vídeos, eines d'intel·ligència artificial, tutorials, eines d'anàlisi, normatives, manuals i documentacions oficials per a la informació i documentació del treball, citant-ne sempre l'autor i afegint el material corresponent a la bibliografia.

Desenvolupar algoritmes de clustering i integrar-los amb les eines de seguiment, és part de la metodologia adoptada. S'agrupen adreces i s'identifiquen relacions entre les transaccions. Aquestes són validades posteriorment amb transaccions reals obtingudes de la cadena de blocs de Bitcoin, on aquesta és utilitzada en remot, via crides a la API.

3.1 Planificació

S'ha seguit la planificació en el format setmanes, és a dir, per a cada pas de la planificació s'han establert certes setmanes de treball, adjudicant "deadlines" a cada tasca. Quan es fa una tasca, s'actualitza l'estat de la mateixa de "per assolir" a "en procés", i una vegada finalitzada, s'actualitza l'estat a "assolit". Aquesta idea de planificació personal sorgeix de la metodologia àgil SCRUM, la qual he fet servir i en conec el procediment. L'he adaptat per orientar el meu projecte i assegurar l'assoliment correcte de tots els objectius en els terminis especificats.

En tot moment es manté una documentació continuada dels corresponents informes a lliurar, que a la vegada són punts per a guardar la feina i veure si estic en el camí correcte i tenir temps a rectificar, si és el cas. D'aquesta manera, s'assegura un correcte desenvolupament del projecte i una redacció progressiva amb el pas del temps.

La planificació del treball està pautada, però ha estat sotmesa a petites modificacions segons el volum de feina i/o treball.

4 ESTAT DE L'ART

En aquesta secció s'explica com funciona la cadena de blocs i el funcionament de les transaccions dins de la mateixa. És bàsic tenir aquests coneixements per a poder desenvolupar l'eina de seguiment.

4.1 Funcionament de la cadena de blocs

La cadena de blocs, també coneguda com a blockchain, per definició és una estructura de dades formada per blocs. Aquests, són ordenats cronològicament, i cada bloc conté un conjunt de transaccions verificades, així com informació sobre les transaccions.

Aquesta estructura de dades formada per blocs, és unida mitjançant un hash. Cada bloc s'enllaça amb l'anterior mitjançant un hash que apunta al bloc precedent. En concret, cada bloc és identificat per un hash criptogràfic, i conté informació a la seva capçalera. Aquesta capçalera inclou un hash del bloc anterior, que connecta cada bloc amb el seu predecessor, creant així una seqüència contínua de blocs que es remunta fins al bloc gènesi, que és el primer bloc creat [1,2].

Aleshores, quan es crea un nou bloc, el funcionament és que aquest es propaga per la blockchain per ser validat pels nodes. Els nodes que reben el bloc l'inclouen en la seva còpia local de la cadena de blocs només si compleix amb les regles de validació, com la verificació del hash del bloc anterior.

Aquest mecanisme, assegura que tots els blocs estan lligats de manera seqüencial fins a aquest bloc inicial, d'aquí a que estan ordenats cronològicament. A més a més, fa que qualsevol modificació a un bloc requereixi el canvi de tots els blocs posteriors, assegurant així la seguretat i la integritat de la cadena de blocs. Aquesta característica s'anomena immutabilitat i garanteix que la cadena de blocs sigui resistent a manipulacions.

4.2 Funcionament de les transaccions

En primer lloc, cal tenir clar que una transacció de bitcoin conté un o varis inputs, un o varis outputs i una o vàries signatures digitals. L'input és des d'on es transfereixen els bitcoins, i conté informació com ara el hash de la transacció, la signatura digital del propietari i l'índex de la sortida específica -en el cas que la transacció anterior tingui varis outputs. L'output, en canvi, defineix cap a on s'envien els bitcoins. Aquests contenen l'adreça del destinatari i la quantitat que transfereix cada un. Per últim, la signatura és necessària per demostrar que el remitent té propietat sobre aquella quantitat de bitcoins que vol transferir. Són imprescindibles per a la seguretat i la validació, i a més, garanteixen la integritat de la transacció.

Les transaccions de Bitcoin inclouen l'adreça o adreces d'origen, l'adreça o adreces de destí i la quantitat transferida. Quan es crea una transacció, és perquè el propietari d'una cartera ha seleccionat els inputs que sumen o superen l'import que vol transferir. Després es defineixen els outputs i es crea una signatura digital per a cada input.

En la figura 1, tenim un exemple de transacció que té 9 inputs i 2 outputs. Cada adreça d'origen i destí té associada la quantitat de bitcoins que transfereix, i es pot consultar de cada una els scripts que contenen la signatura. A la figura 2, veiem la signatura del segon input de la transacció que es mostra a la Figura 1. A més, observem com l'identificador de la transacció de la Figura 1 (hash) és el 36763bbc60718862997668086f32753890ddddbf1d80e2d5e49af2875d143250, així com el dia i la hora que es va realitzar.

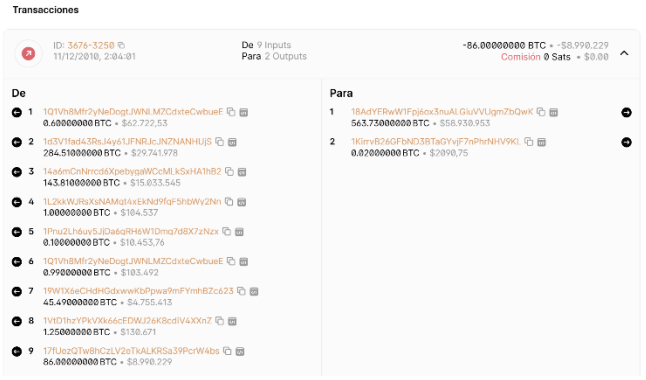

Figura 1. Transaccions de l'adreça 17fUezQTw8hCzLV2eTkALKRSa39PcrW4bs

Figura 2. "Scripts" de l'adreça 1d3V1fad43RsJ4y61JFNRJcJNZNANHUjS

Un cop la transacció s'envia, es difon a la xarxa Bitcoin a través dels nodes. Aquests verifiquen que les signatures són vàlides, i també que els inputs no s'han gastat prèviament. Quan els nodes han validat la transacció, aquesta s'ha de verificar i incloure en un bloc mitjançant un procés anomenat mineria. Aquí és on els miners competeixen per guanyar el dret a incloure un bloc, que implica resoldre un problema criptogràfic que requereix força computacional, i rebre bitcoins com a recompensa. A aquest procés se l'anomena prova de treball.

Per a mes informació sobre com es registren les transaccions a la blockchain de Bitcoin, veure referències *Mastering Bitcoin* [1], *Bitcoin and Cryptocurrency Technologies* [2] i *Lecture 4: Transactions and the UTXO Model* del curs *Cryptocurrency Engineering and Design* [3].

També cal afegir un concepte interessant sobre les transaccions: els arbres de Merkle. S'utilitzen per resumir de

manera eficient les transaccions dins d'un bloc i verificarne les transaccions. Això permet comprovar si una transacció està present en un bloc sense necessitat de revisar tot el bloc, afavorint la verificació ràpida per nodes que no emmagatzemen la cadena completa.

Cal remarcar doncs, que les transaccions de la cadena de blocs són segures, i que poden ser rastrejades ja que estan connectades entre elles.

5 SEGUIMENT DE LES TRANSACCIONS I TÈCNIQUES DE CLUSTERING

Ara que tenim una visió més detallada de com funciona la blockchain i la seva infraestructura tècnica, anem a veure com es fa el seguiment de les transaccions i algunes de les tècniques d'agrupació d'adreces.

5.1 Seguiment de les transaccions a la blockchain de bitcoin

Una manera de fer el seguiment de transaccions és utilitzant eines d'exploració de la cadena de blocs, com ara *Blockchain Explorer* [4]. Aquestes permeten veure detalls públics sobre transaccions, adreces i blocs. Cada transacció i adreça de Bitcoin no està directament associada a la identitat real d'un usuari, però es pot rastrejar el flux de bitcoins entre adreces, que és l'objectiu del treball. Es poden identificar patrons, sabent l'origen i la destinació de cada transacció.

Un recurs que resulta útil per entendre com es fa el seguiment de les transaccions dins de la cadena de blocs és el vídeo *Lecture 7: OP_RETURN and Catena* del curs *Cryptocurrency Engineering and Design* [3].

La idea principal és desenvolupar una eina, que de manera automatitzada, faci el seguiment del flux de les transaccions en forma d'arbre o de graf per a veure per quines adreces passen els pagaments. Això es desenvolupa mitjançant la blockchain explorer, i es valdia amb dades reals. Més endavant es podria validar amb plataformes de "blockchain intelligence", com ara *Arkham Intelligence, Deanonymizing the Blockchain* [12].

5.2 Heurístiques per a l'agrupació d'adreces

Fer el seguiment de les transaccions també implica aplicar tècniques avançades de seguiment, com ara la necessitat d'incloure clustering d'adreces i heurístiques. Aquestes es basen en agrupar adreces que pertanyen al mateix usuari, comprovant els inputs i outputs de les transaccions i buscant patrons com la reutilització d'adreces. Per altra banda, les heurístiques defineixen adreces que tenen múltiples entrades i sovint pertanyen al mateix usuari, reduint així l'amplada de l'arbre que mostrarà el flux de bitcoins. A continuació, algunes de les heurístiques que he estudiat i que són aptes per aplicar.

5.2.1 Heurística del canvi

Aquesta heurística es pot aplicar quan l'emissor d'una transacció vol enviar un import inferior al saldo actual d'aquella adreça. Quan passa això, es crea una sortida addicional per enviar el canvi que ha de tornar. La clau és que aquesta adreça de canvi normalment pertany al mateix usuari que l'input. Per tant, pot ser identificada comparant-la amb el la resta de sortides de la transacció.

Com es mostra en la Taula 1, l'adreça X envia 1200 bitcoins. El destinatari té l'adreça Y, que ha de cobrar 1000 bitcoins, mentre que l'adreça Z mai havia estat utilitzada i segurament és una adreça de canvi controlada pel mateix usuari de l'adreça X. Aquesta adreça Z molt probablement retorna el canvi de 199 bitcoins.

Taula 1. Exemple de l'adreça del canvi

FROM	TO
Adreça X : S'envien 1200 BTC →	Adreça Y : Es reben 1000 BTC
	Adreça Z : Es reben 199 BTC (1 de fee, p.e.)

5.2.2 Heurística de múltiples entrades

Les transaccions que tenen múltiples entrades, es poden considerar que han estat creades per una sola entitat. Aquesta heurística va relacionada amb les claus privades que es fan servir per signar una transacció, i és que només un usuari que té accés a les claus privades de totes les adreces implicades és capaç de signar totes les entrades d'una mateixa transacció.

Vegeu un exemple a la Taula 2, on podem suposar que l'adreça X, Y i Z pertanyen al mateix usuari, ja que envien el contingut a una única sortida, l'adreça N.

Taula 2. Exemple de l'adreça de múltiples entrades

FROM	TO
Adreça X : S'envien 5 BTC →	Adreça N : Es reben 15 BTC
Adreça Y : S'envien 5 BTC →	
Adreça Z : S'envien 5 BTC →	

5.2.3 Heurística temporal i de comportament

Aquesta heurística es basa en el fet de mirar si existeixen patrons temporals en les transaccions, és a dir, que dues o més transaccions hagin estat enviades en un interval molt curt de temps. Se li afegeix credibilitat si, a més a més, les transaccions es fan de manera seqüencial. Seguint aquests patrons, es pot identificar si diferents transaccions amb diferents adreces pertanyen al mateix usuari o estan vinculades a un mateix emissor. En aquest cas, també ens podem trobar que s'utilitzi un servei automatitzat per a realitzar les transaccions, sobretot si es segueix un patró repetitiu seqüencial i les transaccions han estat fetes gairebé al mateix temps.

5.2.4 Heurística d'estructura de les sortides

També coneguda com a heurística de subrutines, es basa en reconèixer els patrons específics que fan servir algunes aplicacions o serveis de cartera. Aquests són predictibles per la forma en que es generen les transaccions.

Un exemple seria quan un servei de cartera (conegut com a "Exchange") fa una transacció enviant diners (adreça X) a múltiples adreces amb imports semblants (adreces Y, Z, N, M, P). Veient això, es fàcil imaginar que les adreces receptores són comptes individuals d'alguns dels clients de l'aplicació. Gràcies a que la aplicació centralitza els pagaments, es pot detectar quan els clients reben les transaccions. Vegeu Taula 3.

Taula 3. Exemple de l'estructura de sortida

FROM	TO
Adreça X : S'envien 50 BTC →	Adreça Y : Es reben 10 BTC
	Adreça Z : Es reben 10 BTC
	Adreça N : Es reben 10 BTC
	Adreça M : Es reben 10 BTC
	Adreça P : Es reben 10 BTC

5.2.5 Heurística de transaccions agregades

En aquest cas, la heurística ve donada perquè diversos usuaris fan varis pagaments, i aquests es combinen en una única transacció que té varies sortides. Quan es dona aquest cas, es pot suposar que hi ha un sol usuari (o més aviat entitat) que gestiona el procés de les transaccions. Aquesta heurística és similar a l'anterior: un exemple seria que un servei fa un pagament massiu a varis usuaris a la vegada, encara que les transaccions tinguin imports diferents. Vegeu Taula 4.

Taula 4. Exemple de transaccions agregades

FROM	TO (Al mateix temps)
Adreça X : S'envien 50 BTC →	Adreça Y : Es reben 5 BTC
	Adreça Z : Es reben 7 BTC
	Adreça N : Es reben 10 BTC
	Adreça M : Es reben 13 BTC
	Adreça P : Es reben 15 BTC

5.3 Limitacions

Tot i que les heurístiques ens ajuden a fer el seguiment de les transaccions, també cal tenir en compte que tenen certes limitacions: els usuaris poden utilitzar tècniques per protegir la seva privadesa. Alguns exemples són la fragmentació de transaccions (fer una gran transacció en varies de petites), la barreja de monedes (enviar monedes a diferents adreces en diferents moments), o l'ús de carteres deterministes (generen moltes adreces a partir d'una única clau). L'objectiu d'aquestes tècniques és dificultar el seguiment i l'agrupació d'adreces, ja que afegeixen confusió a l'hora d'analitzar les transaccions.

6 DESENVOLUPAMENT

6.1 Definició de l'entorn de treball

He triat Python com a llenguatge de programació ja que disposa de vàries llibreries que proporcionen eines útils per al projecte com veurem a continuació.

Primer de tot, he hagut de configurar l'entorn. He fet servir PyCharm, i m'he instal·lat llibreries necessàries per a la correcta execució del projecte. Entre d'altres, a continuació es detallen les que he fet servir i algunes altres que són útils:

- Per a la gestió de paquets i entorns: *pip* [18]
- Blockchain: web3, *bit* [19]
- Anàlisi: *numpy* [20], pandes.
- Per visualització de les dades: *matplotlib* [21], plotly
- Per generar l'estructura del graf i creació d'arbres: *networkx* [22]
- Comunicació amb la API: *requests* [23]

API Blockchain

Per a fer el seguiment de les transaccions he decidit utilitzar una API Blockchain. En el meu cas, he triat utilitzar *Blockchain Explorer API* [26], per accedir a dades de la cadena de blocs i guardar-les, i poder-les mostrar posteriorment ja sigui de manera visual en forma de graf o simplement al terminal, depenent del temps.

He triat aquesta API perquè no requereix d'una API Key per accedir a la majoria de les funcionalitats bàsiques, i perquè ha estat recomanada pel meu tutor. Per utilitzar-la, m'he hagut de registrar a blockchain.com/Exchange.

6.2 Primer disseny de l'eina

6.2.1 Introducció al codi

Principalment, el codi que genera l'eina de seguiment té quatre fitxers python:

- arbre_transaccions.py

És el fitxer que conté les funcions que construeixen l'arbre de seguiment de les transaccions. La funció principal construeix l'arbre de manera recursiva, fent cas a les sortides rellevants (de més quantiat de bitcoins) fins a un màxim de nivells determinat.

- blockchain_api.py

Aquest fitxer conté les funcions necessàries per interactuar amb l'API Blockchain Explorer, i obté dades de les adreces, transaccions i els blocs.

- main.py

Bàsicament, executa el flux del programa: definint una adreça inicial per al seguiment, obtenint informació de l'adreça i les transaccions associades, generant l'arbre, guardant-lo en format JSON i mostrant el graf resultant.

- output.py

Conté les funcions necessàries per a construir i visualitzar l'arbre de transaccions en forma de graf visualment.

6.2.2 Funcionalitats de l'API Blockcahin Explorer

Accedint a l'apartat de desenvolupadors de la pàgina *Blockchain Explorer API* [16], podem consultar la documentació de l'API que ens permet identificar algunes de les principals funcionalitats de les que gaudeix. Aquestes són les següents:

- Buscar transaccions per hash.
- Obtenir informació sobre adreces.
- Accedir a la llista de transaccions en un bloc.
- (...)

Cada una d'aquestes funcionalitats, està programada dins del arxiu blockchain_api.py, i és cridada des de main.py per a fer la nostra consulta.

6.2.3 Recopilació de dades

Una vegada tenim clares les funcionalitats que ens permet fer la API de Blockchain Explorer, l'objectiu és veure com gestionar aquestes dades per tal d'obtenir el resultat esperat, que és generar l'arbre de transaccions a partir de una adreça inicial.

El primer pas és definir com s'obté la informació de les transaccions (input/output) utilitzant l'API. Es crea una funció que recull totes les transaccions associades a una adreça inicial, mitjançant la API de Blockchain.com.

Una vegada fet això, es defineix quin serà el primer node, o sigui l'adreça inicial de l'arbre. I com s'implementarà. També s'ha de definir quin volem que sigui el límit per al qual es traçarà el flux de transaccions. Per fer-ho es pot crear una funció que, mitjançant l'adreça inicial, comenci a crear l'arbre de transaccions. Aquesta funció se li pot passar un paràmetre que defineixi també el màxim de nivells fins on es vol explorar l'arbre.

Per aconseguir les dades necessàries, és necessari fer consultes a la API de blockchain. Algunes de les dades que són necessàries són el hash de transacció, les adreces d'entrada/sortida o els imports transferits.

6.2.4 Disseny visual

Mitjançant les dades, es crea un graf de transaccions on els nodes representen adreces, i les arestes transaccions. Aquest inicialment es visualitza amb JSON que guarda les dades estructurades de la blockchain.

Les opcions que he explorat són *networkx* [22] per crear les connexions entre adreces com si fos un graf. Aquestes poden representar les relacions entre inputs i outputs: cada node de l'arbre pot representar una adreça o transacció, i els enllaços mostren el flux de bitcoins. Per a la visualització d'aquestes dades, es pot utilitzar *Matplotlib* [21], que és una biblioteca de visualització de dades.

6.3 Funcionalitats inicials

6.3.1 Obtenció de les dades de la API

Per a fer les consultes a l'API de Blockchain.com, he fet servir les següents funcions, que s'inclouen dins del fitxer blockchain_api.py, utilitzant la biblioteca "requests" per fer les peticions HTTP.
Per obtenir les dades sobre una adreça específica:

- https://blockchain.info/rawaddr/{adreca}

Per obtenir les dades sobre una transacció específica:

- https://blockchain.info/rawtx/{hash_transaccio}

Per obtenir les dades sobre el bloc més recent:

- https://blockchain.info/latestblock

6.3.2 Definició del primer node

Al fitxer arbre_transaccions.py, he creat la funció "construir_arbre" que utilitza una adreça inicial que podem triar ("adreca_inicial") passada com a paràmetre per iniciar la construcció de l'arbre de transaccions.
Exemple: adreca_inicial=
"bc1qawvd8fyxl5d5y90d7ntcj52q78z5600sx65z3q"
Al main.py fem la crida a la funció:
arbre = construir_arbre(adreca_inicial, max_nivells=2)
On passem l'adreça inicial i el número màxim de nivells que volem que tingui l'arbre.

6.3.3 Obtenció de les transaccions de l'adreça inicial

Al fitxer blockchain_api.py, he creat la funció anomenada "obtenir_informacio_adreca(adreca)" que obté totes les transaccions associades a una adreça inicial mitjançant la API de Blockchain.com.

6.3.4 Processament de les transaccions

Extreu les sortides (outputs) per cada transacció obtinguda, amb la finalitat d'dentificar les adreces a les que s'envien els bitcoins. Per fer-ho, al fitxer blockchain_api.py, he creat la funció "obtenir_informacio_transaccio(hash_transaccio)", i el que fa és consultar la informació específica d'una transacció, retornant les seves dades.

6.3.5 Construcció de l'arbre

L'objectiu és construir un arbre que segueixi les transaccions de bitcoin donada una adreça inicial. Primer s'obté la informació de l'adreça inicial, i després es processen les transaccions. A continuació, s'ha d'iterar per les adreces de sortida, és a dir, fer una construcció recursiva de subarbres (s'ha de repetir el procés per a cada adreça sortida). Finalment, es mostren les entrades i sortides de les transaccions agrupades per adreces i guarda cada transacció processada de manera ordenada.

Per fer-ho, he creat la funció "construir_arbre" dins de arbre_transaccions.py.

6.3.6 Guardar l'arbre en format JSON

A través d'aquest es poden visualitzar les dades de manera senzilla i en forma d'arbre.

Com que el resultat de generar l'arbre és un diccionari python, es fa servir "json.dump(arbre, fitxer, indent=4)" per escriure l'arbre en format JSON i assegurar que sigui llegible [24, 25].

6.4 Comprovació que guardem l'arbre de forma correcta

Una vegada s'ha aconseguit tractar les dades de les transaccions i les adreces, he comprovat que s'imprimeix el missatge: "Arbre de transaccions generat i desat a 'arbre_transaccions.json'.", que és el que he posat com a control d'error per a la creació del fitxer JSON. A més d'això, he comprovat que el fitxer es crea al directori d'execució i que es pot obrir i llegir correctament.

Per veure resultats, consultar Apèndix A1.

6.5 Integració de tècniques de clustering

La tècnica que s'aplica té a veure amb l'heurística del canvi. Es fa una selecció de sortides rellevants, és a dir, es limita el nombre de camins explorats des de cada transacció, fent que només es centri en les sortides més significatives (probablement les altres són de canvi). Per a fer-ho, es prioritzen les sortides segons el seu valor, ordenant-les de major a menor, i es selecciona la primera adreça de destinació que no coincideix amb l'adreça d'origen. Això permet reduir l'amplitud de l'arbre generat i selecciona només una sortida per a cada transacció, agafant la que té el valor més significatiu.

A part, també es fa un control de les adreces i transaccions ja processades, evitant cicles i redundàncies en la construcció de l'arbre i permetent que s'evitin duplicar transaccions que ja han estat analitzades anteriorment.

6.6 Seguretat, privacitat.

S'ha evitat guardar dades importants innecessàries, per lliurar-nos de responsabilitats, així com evitar registrar informació dels usuaris. També s'ha evitat vincular adreces amb identitats reals.

7 RESULTATS

El resultat principal del projecte ha estat la generació d'un arbre de transaccions que permet visualitzar el flux de bitcoins a partir d'una adreça inicial. Aquest arbre estructura les dades de manera que cada node representa una adreça o una transacció, mentre que les connexions representen el flux de valors entre elles. A més, s'ha fet una comprovació del temps d'execució de cada arbre generat. Ho veiem a l'apartat 7.2.

Aquests resultats mostren que l'eina desenvolupada és capaç de construir un arbre de transaccions de la blockchain de Bitcoin a partir d'una adreça inicial. Aquest procés no només revela el flux de bitcoins entre adreces, sinó que també permet identificar patrons que poden tenir implicacions més àmplies en àmbits com la seguretat, el rastreig de transaccions sospitoses i l'anàlisi de xarxes financeres.

A més d'això, mitjançant crides a la API es poden validar funcions essencials, com ara l'obtenció d'informació d'adreces i transaccions, el processament de sortides, i la construcció recursiva de l'arbre.

El format JSON generat permet un anàlisi detallat i la integració amb altres eines per a futures investigacions.

7.1 Validació amb dades reals

7.1.1 Exemple 1

Font:
https://www.blockchain.com/explorer/addresses/BTC/1DhWP7GnKpodPcioHSUi8zPJgqKjx7ypvj

Adreça origen:
1DhWP7GnKpodPcioHSUi8zPJgqKjx7ypvj

En la Figura 3, obtinguda de la pàgina de l'adreça d'origen, observem d'on ha rebut els bitcoins (part inferior) i cap a on ha enviat els 3.27 bitcoins que conté (part superior). En aquest cas veiem que dels 3.27, n'ha enviat 3.00 a l'adreça 15vScfMHNrXN4QvWe54q5hwfVoYwG79CS1i 0.27 a l'adreça 1JHJYGshG8Ds9XXHbXuTrDkf8XAXzNhi5c. Per tant, el que ens interessa és seguir l'adreça que conté la major part dels bitcoins per fer-ne un seguiment real, imaginant que l'altra transacció retorna els diners a l'adreça original (heurística del canvi).

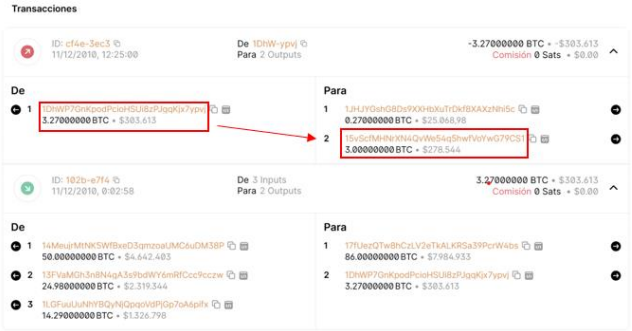

Figura 3. Transaccions de l'adreça origen

En la Figura 4, observem a la part inferior el que hem vist anteriorment, que és d'on provenen els 3.00 bitcoins que actualment té l'adreça en la qual estem. A la part superior, el mateix cas que abans, veiem que 2.99 bitcoins van a l'adreça 1Am9UTGfdnxabvcywYG2hvzr6qK8T3oUZT i 0.01 a l'adreça 1H8ANdafjpqYntniT3Ddxh4xPBMCSz33pj. Ens interessa fer el seguiment de l'adreça que conté la major part, que és la que conté 2.99 bitoins.

Figura 4. Transaccions de l'adreça 15vScfMHNrXN4QvWe54q5hwfVoYwG79CS1

En la Figura 5, veiem la informació de la pàgina de l'adreça que hem esmentat anteriorment. Ara, a la part de dalt, observem que aquesta transacció conté múltiples entrades (heurística de l'adreça de múltiples entrades). Això és indiferent, perquè els bitcoins de l'adreça inicial que hem triat estan dins de l'output, encara que aquest contingui altres bitcoins procedents d'altres adreces (que segurament pertanyen a la mateixa entitat). Veure que a la Figura 6, la sortida d'aquesta adreça és de 10 BTC.

Figura 5. Transaccions de l'adreça 1Am9UTGfdnxabvcywYG2hvzr6qK8T3oUZT

Si anem fent el seguiment anterior, al final el resultat que esperem és un arbre que es podria representar de la manera en que ho veiem a la Figura 6.

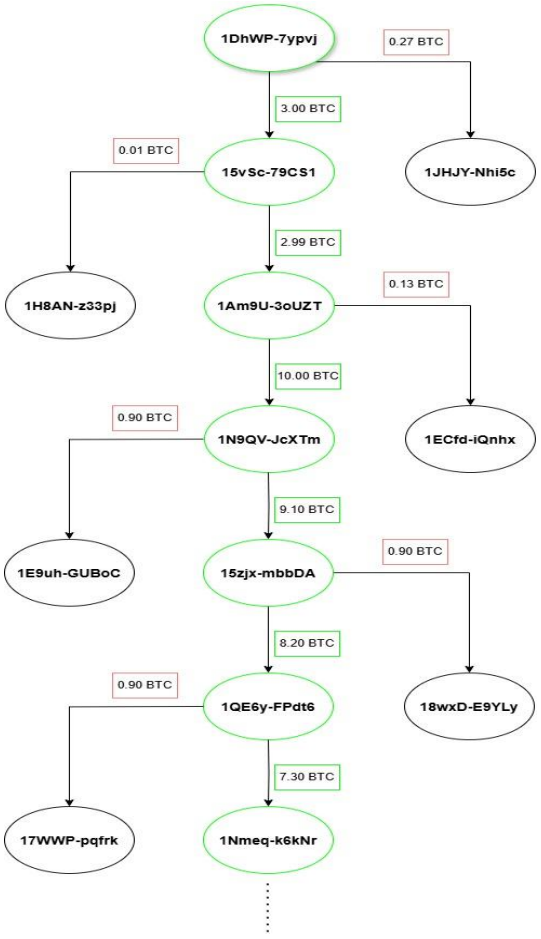

Figura 6. Arbre esperat amb adreça origen 1DhWP7GnKpodPcioHSUi8zPJgqKjx7ypvj

Com es mostra a l'Apèndix A1, veiem el resultat de la generació del seguiment de les transaccions per part de la meva eina, en format JSON, per aquest exemple amb dades reals de la cadena de blocs de Bitcoin.

També, tal com es mostra a l'Apèndix A2, veiem el graf resultant que s'ha creat automàticament a través de les dades extretes de les transaccions. És la representació directa de la sortida.

7.1.2 Exemple 2

Font:
https://www.blockchain.com/es/explorer/addresses/BTC/bc1q9dd8k6jtdsc7yrefk0ythmhpensn22m26ge7rl

Adreça origen:
bc1q9dd8k6jtdsc7yrefk0ythmhpensn22m26ge7rl

Si fem el seguiment d'aquesta adreça d'origen, al final el resultat que esperem és un arbre que es podria representar de la manera en que ho veiem a la Figura 7.

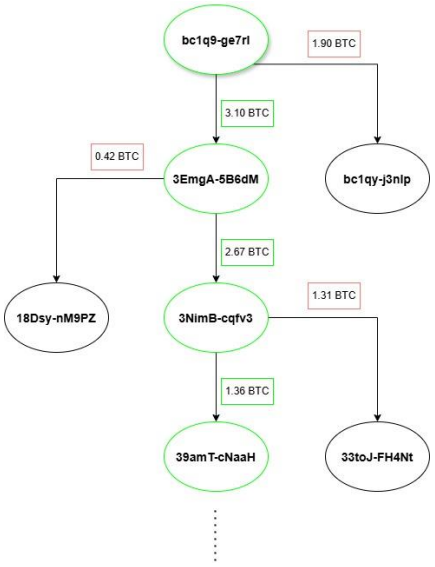

Figura 7. Arbre esperat amb adreça origen bc1q9dd8k6jtdsc7yrefk0ythmhpensn22m26ge7rl

7.2 Temps d'execució

Per a cada exemple, s'ha anotat el temps d'execució que ha tardat l'arbre en generar-se. Vegeu Taula 5.

Taula 5. Comparativa del temps d'execució

#Exemple	Nivells	Temps d'execució
Exemple 1	3	1.066697835922241 segons
Exemple 1	7	3.481755495071411 segons
Exemple 2	3	5.699449300765991 segons

La primera comparativa que hem fet és executar l'adreça de l'exemple 1 com a adreça d'inici, amb número màxim de nivells 3 i 7. Observem que al tenir només 3 nivells, tenim una millora de 3,5x respecte a si en tenim 7.

El segon que podem observar, és que per a l'exemple 1 i 2,

amb el mateix número de nivells (3) tenim que l'execució de l'exemple 2 és 5 vegades més lent. Això és degut a que en la última adreça que l'arbre explora hi ha moltes més transaccions associades, i això fa disparar el temps d'execució, i en conseqüència el programa tarda més en generar totes les relacions entre els nodes. Per tant, podem concloure que el temps d'execució dependrà del número de transaccions i adreces que tingui relacionades una adreça.

Podem veure gràficament aquest fenomen a l'Apèndix A3.

8 Problemes

Un dels problemes que m'he trobat era que la funció que construeix l'arbre seguia un flux molt llarg, i impedia que l'arbre es construís correctament. Algunes transaccions tenien sortides que apuntaven a la mateixa adreça inicial o a altres adreces ja processades en nivells anteriors. Aquest apartat em va obligar gairebé a aplicar l'heurística comentada, que era fer un control de les adreces de retorn i les que ja han estat processades, i que per tant, segurament representen el canvi en una transacció de bitcoin. Per a solucionar-ho, vaig afegir la variable *processades* per registrar les adreces ja processades, que permet evitar que les adreces ja processades es tornin a explorar. D'aquesta manera, evitem cicles infinits. També vaig aplicar l'heurística del canvi que elimina les transaccions de canvi que no ens interessen.

Un altre problema trobat, era que al executar el codi tardava molt i no sabia perquè, o fins i tot no em donava cap resultat. Això era degut a que tenia una recursivitat infinita. Per a solucionar-ho, vaig haver de limitar el número màxim de nivells de l'arbre, amb la variable *max_nivells*. Gràcies a aquesta implementació, ara es pot ajustar el paràmetre i garantir un límit en la profunditat, permetent així a jugar amb la configuració.

Per solucionar molts d'aquests problemes, he tingut que depurar el codi i fer un control d'errors amb la funció print() per veure què era el que realment fallava i buscar possibles solucions al problema.

També vull afegir que he tingut certes limitacions per part de l'API. Algunes funcionalitats no eren diferents, i és que si es fan gaires crides en poc temps et diu que hi ha hagut un error en consultar l'adreça degut a que s'han fet masses peticions en un cert període de temps.

9 Conclusió

Puc afirmar que el projecte ha assolit l'objectiu principal de dissenyar i implementar una eina automatitzada capaç de fer un seguiment detallat del flux de transaccions de Bitcoin. Alguns dels punts importants a destacar d'aquest projecte és que el desenvolupament implementa l'heurística de canvi, i això millora l'estructura i la interpretació de l'arbre. Els resultats obtinguts són satisfactoris i demostren que l'eina té resultats correctes. Aquesta eina és robusta i escalable, permetent que en un futur es pugui fer servir més enllà de l'àmbit acadèmic i serveixi per a auditar, per exemple, o per detectar fraus (com ara adreces de ransomware).

També comentar que s'han detectat reptes com ara la recursivitat infinita, bé el processament de grans volums de dades relacionat amb la limitació de l'API. Aquests han estat resolts amb satisfacció i ha sigut clau fer-ho per al correcte desenvolupament del projecte.

Fer aquest treball m'ha servit molt per conèixer a fons la cadena de blocs de bitcoin, per aprendre a fer un seguiment de les seves transaccions, i per veure que realment té una utilitat.

9.1 Aspectes legals

El *RGPD* [15] a Europa regula la normativa vigent en protecció de dades i transparència, per tant, s'ha de tenir en compte. He tingut clar que aquest és un projecte acadèmic i que, per tant, no es pot utilitzar amb finalitats comercials (a alguns països el clustering i seguiment de transaccions vulnera la privacitat i és castigat).

10 Treball futur i continuació del treball

Com a possibles extensions del treball, es podria millorar encara més l'exactitud i utilitat dels resultats, i també es podrien aplicar tècniques més avançades de visualització, de manera que la presentació dels resultats sigui més interactiva i dinàmica, a la vegada que intuïtiva.

A més, he conegut aspectes avançats de seguretat i eficiència de les signatures digitals, comentats al vídeo *Lecture 17: Anonymity, Coinjoin, and Signature Aggregation* del curs *Cryptocurrency Engineering and Design* [3]. Aquests són, per exemple, les "aggregate signatures" o el "Wagner's Attack". La primera és una tècnica criptogràfica que permet combinar múltiples signatures digitals en una sola signatura, mentre que el segon és un atac on una signatura està composta per una combinació de claus o fragments, i l'atacant pot provar combinant les claus en un patró que coincideixi amb un valor desitjat.

Altres aspectes també importants, que apareixen al vídeo *Lecture 18: Confidential Transactions* del curs *Cryptocurrency Engineering and Design* [3], són CoinJoin, l'encriptació, els "homomorphic commitments" i el problema dels negatius.

CoinJoin és una solució d'anonimat, que permet millorar la privacitat barrejant les transaccions dels usuaris en una sola operació. L'objectiu d'aquesta és dificultar saber quins inputs corresponen a quins outputs. Pel que fa a l'encriptació, el que fa és protegir la informació sensible (claus privades i carteres) convertint els elements en format codificat. Per altra banda, els compromisos homomòrfics permeten realitzar certes operacions amb dades encriptades sense la necessitat de desencriptar-les (un exemple útil seria per comprovar el saldo d'un wallet). Per últim, tenim el conegut com a "negative problem", on se'ns planteja la dificultat d'assegurar que certs valors en les transaccions no

són negatius: això pot suposar un problema a l'hora d'evitar possibles errors o fraus.

El fet d'ampliar el treball amb aquests coneixements, a part de millorar la solidesa del projecte i la seva ètica, també evita problemes que poden aparèixer si s'utilitza amb mala intenció.

AGRAÏMENTS

Voldria agrair, en primer lloc, al meu tutor Guillermo Navarro Arribas pel suport i per haver-me guiat constantment al llarg del projecte. També vull agrair a la meva família per les seves opinions i pel seu recolzament.

BIBLIOGRAFIA

[1] A. M. Antonopoulos, *Mastering Bitcoin*. [En línia]. Disponible: https://criptoinforme.com/wp-content/uploads/2024/01/Mastering-Bitcoin-Andreas-M.-Antonopoulos.pdf. [Consultat: Oct. 2024].
[2] A. Narayanan, J. Bonneau, E. Felten, A. Miller, and S. Goldfeder, *Bitcoin and Cryptocurrency Technologies*. Princeton, NJ, USA: Princeton University Press, 2015. [En línia]. Disponible: http://vis.usal.es/rodrigo/documentos/sisdis/papers/Narayanan2015.pdf,https://bitcoinbook.cs.princeton.edu/. [Consultat: Oct. 2024].
[3] Massachusetts Institute of Technology, *Cryptocurrency Engineering and Design*, 2018. [En línia]. Disopnible: https://ocw.mit.edu/courses/mas-s62-cryptocurrency-engineering-and-design-spring-2018/video_galleries/lecture-videos/. [Consultat: Oct. 2024].
[4] Blockchain.com, "Blockchain Explorer - Bitcoin Tracker & More." [En línia].
Disponible a: https://www.blockchain.com/explorer. [Consultat: Oct. 2024].
[5] Bitcoin.org, "Bitcoin Core". [En línia]. Disponible: https://bitcoin.org/en/download. [Consultat: Oct. 2024].
[6] Elliptic, "Blockchain Analytics & Crypto Compliance Solutions". [En línia]. Disponible : https://www.elliptic.co/. [Consultat: Oct. 2024].
[7] E. Androulaki, G. Karame, M. Roeschlin, T. Scherer, and S. Capkun, "Automatic Bitcoin Address Clustering," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS '15)*, Denver, CO, USA, 2015, pp. 149-160. [Consultat: Oct. 2024].
[8] Universidad Internacional de Valencia (VIU), "Clustering: concepto, métodos y aplicaciones en las distintas áreas," [En línia]. Disponible: https://www.universidadviu.com/es/actualidad/nuestros-expertos/claustering-que-es-y-que-aplicaciones-tiene. [Consultat: Oct. 2024].
[9] J. J. L. Marín García, *Seguridad y privacidad en redes de criptomonedas*, Ph.D. dissertation, Univ. Granada, Granada, Spain, 2018. [En línia]. Disponible a: 88489.pdf (ugr.es). [Consultat: Oct. 2024].
[10] OpenAI, "ChatGPT". [En línia]. Disponible: https://chat.openai.com/. [Consultat: Oct. 2024].
[11] Microsoft, "Microsoft Copilot". [En línia]. Disponible: https://www.microsoft.com/bing. [Consultat: Oct. 2024].
[12] Arkham Intelligence, "Deanonymizing the Blockchain". [En línia]. Disponible: https://www.arkhamintelligence.com/. [Consultat: Oct. 2024].
[13] GraphSense, "Cryptoasset Analytics Platform" [En línia]. Disponible: https://graphsense.org/, https://graphsense.info/. [Consultat: Oct. 2024].
[14] BlockSci. "Blockchain Analysis Tool". [En línia].
Disponible: https://citp.github.io/BlockSci/. [Consultat: Oct. 2024].
[15] Comissió Europea, *Reglament General de Protecció de Dades* (RGPD), 2016. [En línia]. Disponible: https://www.boe.es/doue/2016/119/L00001-00088.pdf.
[Consultat: Oct. 2024].
[16] W3Schools, "Python Tutorials." [En línia]. Disponible: https://www.w3schools.com/python/. [Consultat: Nov. 2024].
[17] HablemosPython.dev, "Cursos y Libros sobre Python." [En línia]. Disponible: https://hablemospython.dev/cursosylibros.html. [Consultat: Nov. 2024].
[18] Python.org, "Documentació oficial de Pip." [En línia]. Disponible: https://docs.python.org/es/3/tutorial/. [Consultat: Des. 2024].
[19] PyPI, "Bit: Bitcoin Library." [En línia]. Disponible: https://pypi.org/project/bit/. [Consultat: Des. 2024].
[20] Numpy.org, "Documentació oficial de NumPy." [En línia]. Disponible: https://numpy.org/doc/. [Consultat: Des. 2024].
[21] Matplotlib.org, "Documentació oficial de Matplotlib." [En línia]. Disponible: https://matplotlib.org/. [Consultat: Des. 2024].
Matplotlib.org, "Guia d'instal·lació de Matplotlib." [En línia]. Disponible a: https://matplotlib.org/stable/install/index. [Consultat: Des. 2024].
[22] NetworkX, "Tutorial de NetworkX." [En línia]. Disponible a: https://networkx.org/documentation/stable/tutorial.html. [Consultat: Des. 2024].
[23] Python-Requests.org, "Documentació oficial de Requests." [En línia]. Disponible: https://docs.python-requests.org/en/latest/index.html. [Consultat: Des. 2024].
[24] Python.org, "Mòdul JSON." [En línia]. Disponible: https://docs.python.org/3/library/json.html. [Consultat: Des. 2024].
[25] Python.org, "Entrada i Sortida: Llegir i Escriure Fitxers." [En línia]. Disponible: https://docs.python.org/3/tutorial/inputoutput.html#reading-and-writing-files. [Consultat: Des. 2024].
[26] Blockchain.com, "Blockchain Data API". [En línia]. Disponible a: https://www.blockchain.com/es/explorer/api/blockchain_api. [Consultat: Des. 2024].
[27] Draw.io, "Draw.io," [En línia]. Disponible: https://www.drawio.com [Consultat: Gen. 2025].

APÈNDIX

A1. JSON resultant de la simulació

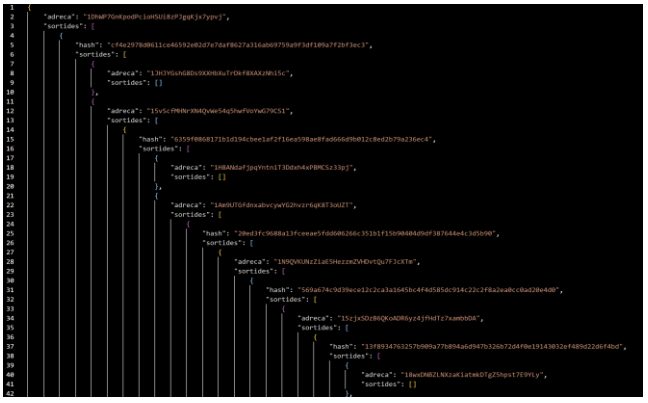

Figura A.1.1 JSON.

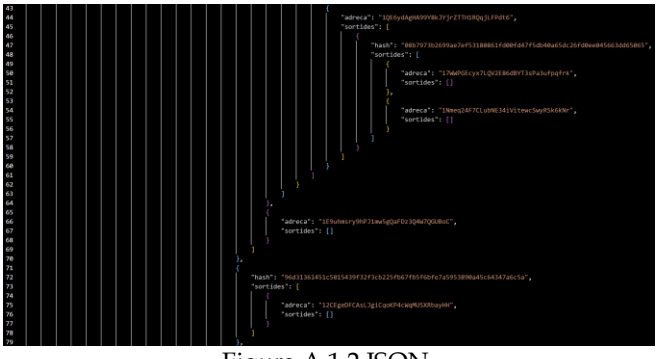

Figura A.1.2 JSON.

Figura A.1.3. JSON.

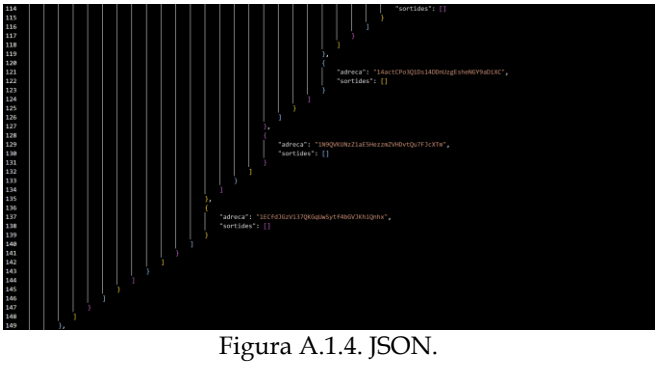

Figura A.1.4. JSON.

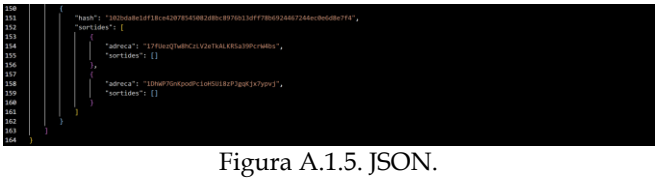

Figura A.1.5. JSON.

A2. Graf resultant de la simulació (Exemple 1)

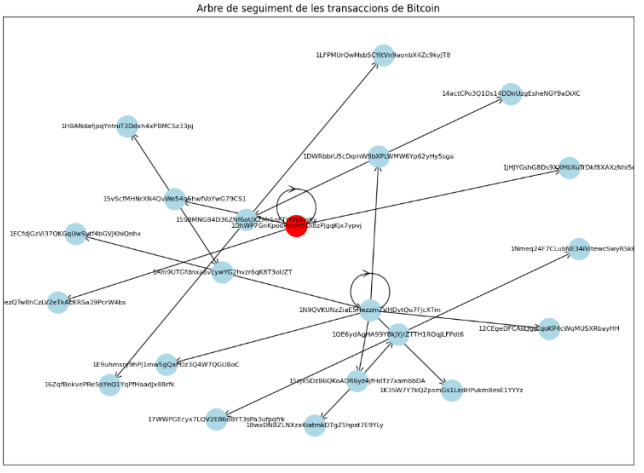

Figura A.2 Graf resultant de l'exemple 1.

A3. Graf resultant de la simulació (Exemple 2)

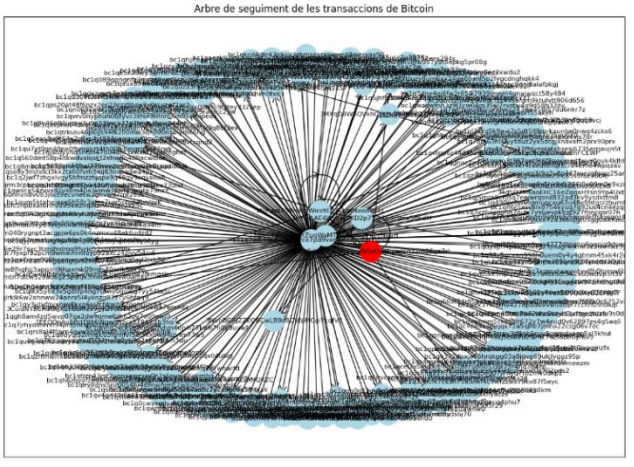

Figura A.3 Graf resultant de l'exemple 2.

Si comparem el número de nodes que tenen relació amb l'últim node explorat d'aquesta simulació, veiem que és molt més gran que el de l'exemple 1. Per això el temps d'execució d'aquest exemple és 5 vegades més que el de l'exemple 1.