****

**TREBALL FINAL DE GRAU**

**TÍTOL:** Estructures de grafs amb equivalències d’arestes aplicades a l’anàlisi de dades relacionals

**AUTORS:** Rodríguez Navas, Laura

**DATA DE PRESENTACIÓ:** Febrer, 2017

**COGNOMS:** Rodríguez Navas **NOM:** Laura

**TITULACIÓ:** Grau en Enginyeria Informàtica

**PLA:** 2010

**DIRECTOR:** José Luis Balcázar Navarro

**DEPARTAMENT:** Ciències de la Computació

**QUALIFICACIÓ DEL TFG**

|  |  |  |
| --- | --- | --- |
|  | **TRIBUNAL** |  |
| **PRESIDENT** | **SECRETARI** | **VOCAL** |

J. M. Merenciano Saladrigues Bernardino Casas Fernández Sergio Sánchez Lopez

##### DATA DE LECTURA: 9 de febrer de 2017

###### Aquest Projecte té en compte aspectes mediambientals:  Sí  No

**RESUM**

En aquest treball es vol contribuir a la investigació de les 2-estructures mitjançant l’aportació d’una eina que faciliti el desenvolupament de treballs futurs que es fonamentin en la utilització de la teoria de les 2-estructures per a l’anàlisi de dades relacionals.

La teoria de les 2-estructures proporciona una infraestructura matemàtica per a la descomposició de grafs. Permet representar múltiples grafs en una sola estructura algebraica, una 2-estructura, que es deriva en una descomposició única de 2-estrcutures més simples.

Durant el treball, s’ha dut a terme un estudi amb dues finalitats:

* El disseny i la implementació d’un paquet de software que analitzi i representi visualment les principals estructures involucrades en la teoria de les 2-estructures.
* La investigació i el desenvolupament de possibles aplicacions de les 2-estructures en l’anàlisi de dades relacionals que formen part de les bases de dades relacionals.

**Paraules clau (màxim 10):**

|  |  |  |  |
| --- | --- | --- | --- |
| Python | Grafs | Ciències de la la Computació | Algorítmica |
| NetworkX | 2-estructures | Anàlisi de dades | Pydot |
| Graphviz | SQLite |  |  |

**ABSTRACT**

This project seeks to contribute to the research of the 2-structures by providing a tool to facilitate the development of future work which is based in the utilization of the theory of 2-structures for the relational data analysis.

The theory of 2-structures provides a mathematical infrastructure for the decomposition of graphs. Allows multiple graphs represented a single algebraic structure, a 2-structure, which results in a unique 2-structures simpler decomposition.

During the project, a study it has carried out with two purposes:

* Design and implementation of a software package to analyse and visualize the main structures representations involved in the theory of 2-structures.
* The research and development of possible applications of 2-structures in the relational data analysis that are part of the relational databases.

**Keywords (10 maximum):**

|  |  |  |  |
| --- | --- | --- | --- |
| Python | Graphs | Computer Science | Algorithmic |
| NetworkX | 2-structures | Data Analysis | Pydot |
| Graphviz | SQLite |  |  |

AGRAÏMENTS

Aquest treball representa el final d'una segona etapa de la meva vida. Que va començar per casualitat quan vaig decidir deixar l'esport d'elit. Durant aquesta segona etapa he tingut la gran sort de conviure amb un grup de companys excepcionals, a l’EPSEVG i també a la FIB. La seva companyia ha fet el camí a recórrer més fàcil i divertit.

No intentaré anomenar-los a tots, ja que són molts i no me’n vull deixar cap. Confio amb que ells ja saben qui són i els hi dono mil gràcies per la seva fantàstica companyia durant aquests anys. Tot i així, estic segura que comprendran que faci una excepció i mencioni de manera especial a la meva gran amiga, i companya de pràctiques durant gairebé tota la carrera, Noemí. Durant el temps que vaig passar a la FIB com el que he passat a l’EPSEVG ha estat un pilar tan en l’àmbit acadèmic com en el personal.

Per motius semblants anomenats anteriorment tampoc faré una llistat de professors, dels que en gran part guardo un bon record y amb els que he après molt. No obstant, he d’agrair la confiança, la llibertat, la motivació, la inspiració i el recolzament rebut per en Jose Luís Balcázar alhora de desenvolupar aquest projecte. També m’agradaria agrair, a en Bernardino Casas per ensenyar-me a programar i a la Neus Català per iniciar-me en el món de l’anàlisi de dades, aptituds bàsiques per a aquest projecte.

Per últim, el més important. No tinc paraules per expressar el meu agraïment per tot el que els hi dec als meus pares, Lluís i Montse, que han estat al meu costat en els bons moments però també en els dolents. Sense ells, l'imminent enginyera informàtica que ha escrit aquestes línies no seria res.

A tots ells els hi dedico aquest treball.

ÍNDEX

ÍNDEX DE FIGURES

ÍNDEX DE TAULES

ÍNDEX DE CODIS

PART I

INTRODUCCIÓ I PLANIFICACIÓ

1. INTRODUCCIÓ

En aquest capítol es descriuen els elements fonamentals en el que s’emmarca el projecte com poden ser l’abast, els objectius, l’estat de l’art, la metodologia emprada i l’avaluació tecnològica. A més s’indica de forma abreviada el contingut i distribució de la resta de capítols de la memòria.

* 1. Abast i objectius

Degut a que el temps disponible pel desenvolupament del projecte és limitat, l’abast del projecte és el d’analitzar i trobar la millor solució per a la representació visual de les 2-estructures que faciliti la comprensió de l’anàlisi de dades relacionals. A més, també s’incorpora la investigació de possibles noves funcionalitats que es podrien incorporar a l’anàlisi de dades relacionals a partir de les 2-estructures.

Per tant, segons l’abast, podem agrupar i resumir els objectius en la consecució de dues fites.

La primera fita del treball consisteix en el disseny i la implementació d’un paquet de software que compleix els requisits següents:

* Permet la creació i la consulta de propietats de la estructura principal que cobreix l’estudi fonamental sobre la teoria de les 2-estructures.
* Dibuixa gràficament la representació d’aquesta l’estructura principal anomenada anteriorment.
* Integra els components anteriors en un programa interactiu extern (no implementat sota l’entorn de desenvolupament del treball) que permet la visualització i l’emmagatzematge d’aquesta estructura principal.

La segona fita té com a propòsit la investigació i el desenvolupament de possibles aplicacions de les 2-estructures a l’anàlisi de les dades relacionals utilitzades en les bases de dades relacionals.

Per aquest motiu es pren com a punt de partida el següent objectiu:

* Establiment de conjectures i/o resultats que permetin enfortir l’anàlisi de dades relacionals mitjançant l’ús de les 2-estructures.

Algun dels objectius del treball inclouen la utilització d’una aplicació gràfica.

Aquesta aplicació, no està integrada en l’entorn de programació, és a dir, que s’hi accedeix des d’un programari interactiu extern. De manera que l’usuari ha de sortir de l’entorn de treball per a realitzar les visualitzacions gràfiques de les 2-estructures amb aquest programari interactiu extern.

A més, un dels objectius principals del projecte, tot i que l’usuari no hi interactua de manera directa, es el de implementar un algoritme que dibuixi correctament el model de visualització gràfica d’una 2-estructura. Es tracta d’un model basat en DOT (llenguatge descriptiu en text pla) que proporciona una manera simple de descriure gràfiques i que les fa més entenedores per a l’usuari.

Tots els models basats en DOT seran creats des de cero i l’algoritme encarregat de crear els models, generarà una disposició dels elements que facilita a l’usuari la tasca d’entendre el model ràpidament.

* 1. Estat de l’art
     1. *Teoria de grafs*

El precursor de la teoria de grafs va ser Leonhard Euler, que la va iniciar tot intentant resoldre el problema dels set ponts de Königsberg (Euler, 1736) [[R](https://en.wikipedia.org/wiki/Seven_Bridges_of_K%C3%B6nigsberg)].



Figura 1. Solució d’Euler.

La teoria de grafs és una branca de les matemàtiques i la informàtica que es dedica a l'estudi dels grafs, estructures matemàtiques utilitzades per a modelitzar relacions entre parelles d'objectes.

La teoria de grafs es troba estretament relacionada amb la teoria de les 2- estructures. Concretament, en l'etiquetatge de les 2-estructures, que es representa amb arestes de colors en grafs dirigits.

L’estudi previ sobre la teoria de grafs s’ha extret d’assignatures com Estructura de la Informació (ESIN), Projecte de Programació (PROP) i Recuperació de la Informació (REIN).

* + - 1. *Conceptes bàsics*

Els grafs s’utilitzen per a representar relacions (simples o d’ordre) entre objectes del mateix tipus. Els objectes reben el nom de nodes o vèrtex i les relacions entre ells s’anomenen arestes.

Un graf G es pot definir com la parella G = (V, A) on V és un conjunt de vèrtexs i A un conjunt d’arestes (subconjunt de V x V).

Existeixen grafs dirigits i grafs no dirigits depenen de si les arestes estan orientades o no ho estan.

**Exemple 1**. A la figura 2 veiem una representació gràfica d’un graf no dirigit. En un graf no dirigit les relacions són simètriques: A (u, v) = A (v, u).



Figura 2. Graf no dirigit.

**Exemple 2**. A la figura 3 veiem la representació gràfica de l’exemple 1 però en un graf dirigit. En un graf dirigit les relacions no són simètriques i tenen una direcció: A (u, v) va del vèrtex u al vèrtex v, denotat amb una fletxa u → v.



Figura 3. Graf dirigit.

Els grafs (dirigits o no) poden ser etiquetats o no etiquetats en funció de si les arestes tenen o no informació associada.

Donat un domini V de vèrtexs i un domini E d’etiquetes, es defineix un graf dirigit i etiquetat G com una funció que associa etiquetes a parells de vèrtexs.

*G ∈ {f: V x V = E}*

Per a un graf no etiquetat la definició és similar: G és una funció que associa un booleà a parells de vèrtexs.

**Exemple 3**. A la figura 4 veiem la representació gràfica de l’exemple 1 però en un graf ponderat. Un graf ponderat és un tipus particular de graf etiquetat on la informació associada a l’aresta és un nombre real positiu que normalment representa un cost.



Figura 4. Graf ponderat.

* + 1. *Teoria de les 2-estructures*

La teoria de les 2-estructures va ser introduïda per Ehrenfeucht i Rozenberg el 1990.

La teoria de les 2-estructures proporciona una infraestructura matemàtica per a la descomposició i la transformació de grafs. Es tracta d’un formalisme molt potent i robust que permet representar múltiples grafs en una sola estructura algebraica, una 2-estructura, i derivar d’ella una descomposició única en 2-estructures més simples.

Per a l’elaboració d’aquest treball s’ha treballat principalment amb l’estudi previ [R] on s’estableixen els fonaments de la teoria de les 2-estructures i que recopila i amplia el contingut sobre la teoria de les 2-estructures actuals fins a la data de la seva publicació.

La part 'estàtica' de la teoria de les 2-estructures (capítols 3 a 10) es refereix principalment a la descomposició i a la no-descomposició (o primitiva). En els capítols 3 a 7 es considera la 2-estructura 'no etiquetada’ que es defineix per una relació d'equivalència en les arestes. La noció clau aquí és que un clan d'una 2-estructura d’un graf G, que és un subconjunt X de nodes de tal manera que un element y no en X no pot fer una distinció entre els elements de X per la relació d'equivalència proporcionada per G. Per fer la classificació de les arestes més específiques s’afegeix una funció per a etiquetar les classes d'equivalència de les arestes, obtenint d'aquesta manera un 2-estructura etiquetada; aquestes estructures es mostren en els capítols 8 a 10.

Gran part de la motivació per a la recerca presentada en aquest llibre prové de la zona de les gramàtiques de grafs. Perquè la teoria de les 2-estructures enriqueix l'estudi de les gramàtiques de grafs posant a disposició les eines de la teoria de grups i la teoria de la commutació.

*1.2.2.1 Conceptes bàsics*

Una 2-estructura és una seqüència de valors agrupats, formada per un subconjunt finit D, anomenat domini, i una relació d’equivalència *R ⊆* *E2(D) x E2(D)* en el conjunt *E2(D) = {(u, v) | u ∈ D, v ∈ D, u ≠ v}* de les seves arestes. D’aquesta manera una 2-estructura es pot definir com:

*G = (D, R)*

Els elements del domini D són els nodes de la 2-estructura G.

**Exemple 4**. A la figura 5 veiem una representació gràfica d’una 2-estrcutura. El color de les arestes serveix per a distingir a quina classe d’equivalència pertany.

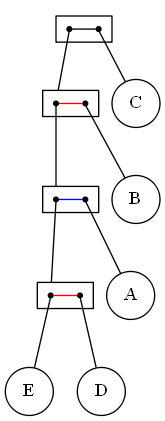


Figura 5. La forma de la 2-estructura de l’exemple 4.

Una 2-estructura és completa quan només té definida una classe d’equivalència. És lineal quan té dues classes d’equivalència i les arestes que conté, cadascuna estableix un ordre total sobre els elements d’un graf. I es primitiva quan no és ni completa ni lineal i a més no es pot des composar en altres 2-estructures.

1.3 Metodologia i avaluació tecnològica

*1.3.1 Metodologia*

Per a la organització del treball la metodologia triada, si bé és senzilla, ha resultat molt efectiva pel desenvolupament del treball. Com que era un projecte on l’equip de desenvolupament consistia en una única persona (en aquest cas l’estudiant que realitza el treball), es va decidir seguir una metodologia similar a Scrum, que és una metodologia àgil pensada sobretot pel desenvolupament de software.

Per tant, es va decidir realitzar les reunions pròpies amb el director del projecte constantment, amb l’objectiu de facilitar la proposta de solucions i la obtenció de feedback.

Els rols es van assignar de la manera següent:

* Project Master: és el que s’encarrega de fer el seguiment d’allò que realitza l’equip de desenvolupament, en aquest cas és el director del treball.
* Product Owner: és qui representa els interessats en el projecte, en aquest cas també és el director del treball.
* L’Equip de Desenvolupament: el que s’encarrega de desenvolupar el producte, en aquest cas és l’estudiant que realitza el projecte.

També es va decidir utilitzar el sistema Kanban per a la organització de les tasques del treball mitjançant l’aplicació web Wrike [[R](https://www.wrike.com/)].

* Wrike: és una eina en línia per a la gestió de projectes i col·laboració en el treball. Que permet als seus usuaris gestionar i realitzar un seguiment dels projectes, terminis, horaris i altres processos de flux de treball. També permet als usuaris col·laborar entre si. Wrike també es pot trobar en dispositius iOS i Android. Pràcticament totes les versions del programari compten amb un cicle de treball que actualitza els usuaris de qualsevol activitat realitzada per altres usuaris en els grups de treball. Les característiques socials també estan integrades en el programari. A més també s'integra amb un seguit d'altres productes com Google Apps, Microsoft Outlook, Microsoft Excel, Microsoft Project, Google Drive, Dropbox, Apple Mail Box, IBM Connections, i altres.



Figura 6. Taulell Kanban a Wrike.

Per a la implementació i les proves del treball, a mesura que s’anava avançant i es van començar a plantejar els primers diagrames de classes, es va triar la metodologia orientada a objectes.

Les bases d’aquest tipus de metodologia són:

* Tot és un objecte, amb una identitat pròpia.
* Un programa és un conjunt d’objectes que interactuen entre ells.
* Cada objecte pertany a un tipus concret: una classe.
* Objectes del mateix tipus tenen un comportament idèntic.

El mètode d’avaluació proposat per a la implementació es divideix en:

1. Periòdicament es posa a prova la implementació mitjançant un conjunt de jocs de prova que es composen de casos senzills i aleatoris per assegurar-se de la pròpia correctesa.
2. Per comprovar la correcta visualització de la implementació mitjançant el programari interactiu extern es realitzen proves a nivell d’usuari (beta-testing) per assegurar-se de que és suficientment intuïtiu i es comprova el funcionament en tots els seus àmbits. Òbviament aquestes comprovacions es realitzen en fases avançades del treball.

*1.3.2 Avaluació tecnològica*

*1.3.2.1 Llenguatges de programació*

El llenguatge de programació triat pel desenvolupament del paquet de software és Python [[R](https://www.python.org/)]. Perquè és un llenguatge molt utilitzat en entorns de computació científica, ja que ofereix un entorn de programació obert i flexible, en el que és fàcil iniciar-se, gràcies a la seva filosofia de disseny, que busca llegibilitat en el codi, i la seva sintaxi, que permet expressar conceptes en menys línies de codi del que seria possible en altres llenguatges, com per exemple Java o C.



Figura 7. Logotip del llenguatge de programació Python.

Python està dissenyat per ser un llenguatge molt visual, i, com a característica principal utilitza el [sagnat](https://ca.wikipedia.org/wiki/Sagnat). Això és poc comú en [llenguatges de programació](https://ca.wikipedia.org/wiki/Llenguatge_de_programaci%C3%B3), ja que molts utilitzen delimitadors. A més, té una llibreria estàndard gran i aporta eines ja programades que poden crear una gran varietat de funcions.

Una de les característiques de la llibreria que s’ha utilitzat és la combinació amb altres tipus de llenguatge, C i C++, i mòduls per a connectar bases de dades relacionals.

El paquet de software implementat és compatible amb les versions de Python 2 i 3.

*1.3.2.2 Llibreries*

Pel tractament de les bases de dades SQLite es va utilitzar el mòdul sqlite3 [[R](https://docs.python.org/3/library/sqlite3.html)] que proporciona una interfície SQL compatible amb l'especificació DB-API 2.0.

* SQLite és una llibreria de C que proporciona una base de dades que no requereix un procés de servidor independent i permet l'accés a la base de dades utilitzant una variant no estàndard del llenguatge de consulta SQL. També és possible crear prototips d'una aplicació que utilitza SQLite i després exportar el codi a una base de dades més gran, com PostgreSQL o Oracle.

Per a la creació i la manipulació dels grafs es va utilitzar NetworkX [[R](https://networkx.github.io/)].

* NetworkX és un paquet de programari de llenguatge Python per a la creació, manipulació, i l'estudi de l'estructura, la dinàmica i les funcions de les xarxes complexes. L'audiència potencial de NetworkX inclou matemàtics, físics, biòlegs i informàtics.

Proporciona:

* Estructures de dades per a grafs, digrafs i multi-grafs.
* Molts algoritmes de grafs estàndards.
* Generadors de grafs simples, grafs aleatoris, etc.
* Llicència BSD de codi obert.
* Una interfície d’algoritmes numèrics existents i codificats en C, C++ i Fortran.
* La capacitat de tractar grans quantitats de dades.
* Fiabilitat amb més de 1800 proves d'unitat.

Per a generar tots els subconjunts possibles d’un graf, que té un alt cost de processament, i altres recorreguts, es va pensar en utilitzar la llibreria itertools de Python [[R](https://docs.python.org/3/library/itertools.html)]. Ja que proporciona funcions per a recórrer molt eficients. La llibreria s’inspira en construccions APL [[R](https://en.wikipedia.org/wiki/APL_(programming_language))], Haskell [[R](https://www.haskell.org/)], i SML [[R](https://en.wikipedia.org/wiki/Standard_ML)].

Una vegada creats els grafs i les 2-estructures es necessitava una eina de visualització per a observar els resultats. La llibreria triada va ser Pydot [[R](https://github.com/erocarrera/pydot)].

* Pydot és una interfície per a Graphviz [[R](https://pypi.python.org/pypi/graphviz)], que pot analitzar i bolcar en llenguatge DOT [[R](http://www.graphviz.org/doc/info/lang.html)], llenguatge que utilitza el programari Graphviz [[R](http://www.graphviz.org/)] i està escrit en Python pur. A més, NetworkX pot convertir els seus grafs en aquest format.

El paquet de software implementat és compatible amb diferents sistemes operatius (Windows, OS X i Ubuntu) i amb diferents versions de Python. Per fer-ho s’han utilitzat les diferents llibreries següents: os [[R](https://docs.python.org/3/library/os.html)], sys [[R](https://docs.python.org/3/library/sys.html?highlight=sys#module-sys)], subprocess [[R](https://docs.python.org/3/library/subprocess.html?highlight=subprocess#module-subprocess)] i six [[R](https://pythonhosted.org/six/)], amb les seves funcionalitats respectives:

* Mòdul os: llibreria que permet l’ús de funcionalitats pròpies d’un S.O [R].
* Mòdul sys: llibreria que permet l'accés a algunes variables utilitzades per l'intèrpret i funcions que interactuen amb l'intèrpret.
* Mòdul subprocess: llibreria que té la intenció de reemplaçar la llibreria os.
* Mòdul six: llibreria de compatibilitat entre Python 2 i 3.

*1.3.2.3 Programari*

Les particularitats de la representació de la descomposició d’un graf en una 2-estructura i la creació d’una 2-estructura va provocar la cerca d’un programa extern per a la visualització. Aquesta eina és Graphviz.

* Graphviz és un programari de codi obert de visualització gràfica que converteix els programes de disseny en format DOT a diagrames o imatges.



Figura 8. Logotip del programari Graphviz.

**Exemple 5.** A la figura 9 es pot veure el resultat de la conversió que efectua el programari Graphviz. A partir d’un fitxer dissenyat en format DOT (a l’esquerra de la imatge) es crea la visualització corresponent ( a la dreta de la imatge).



Figura 9. Exemple de conversió del programari Graphviz.

Durant el desenvolupament es va utilitzar l’entorn de programació integrat i multi plataforma Pycharm [[R](https://www.jetbrains.com/pycharm/)] i es va mantenir una copia actualitzada del paquet de software en un emmagatzematge remot amb GitHub [[R](https://github.com/)]; [(https://github.com/lrodrin/TFG.git]((https:/github.com/lrodrin/TFG.git)).

* Pycharm està desenvolupat per la companyia Jetbrains, basat en IntelliJ IDEA, l’ l’entorn de programació integrat de la mateixa companyia però enfocat cap a Java (que ja havia utilitzat anteriorment) i la base d’Android Studio. Principalment permet escriure bon codi (seguint les normes PEP-8 [[R](https://www.python.org/dev/peps/pep-0008/)]), ràpid i eficient, gràcies a les funcions que incorpora. Ofereix compilació de codi intel·ligent, correccions de codi, ressaltat d'error en temps real i completesa de codi, juntament amb la re factorització de codi automatitzat i capacitats de navegació. Proporciona suport de primera classe per a Python, JavaScript, CoffeScript, TypeScript, CSS, etc. També utilitza la cerca intel·ligent per saltar a qualsevol classe, arxiu o símbol, o fins i tot qualsevol finestra d'acció IDE o eina.



Figura 10. Logotip de Pycharm.

.

Pycharm pot integrar el sistema de control de versions Git [[R](https://git-scm.com/)], que utilitza GitHub. Aquest fet va afavorir la manipulació alhora d’utilitzar el control de versions.

* Git és un programari de sistema de control de versions dissenyat per Linus Torvalds, pensat en l'eficiència i confiabilitat de manteniment de versions d'aplicacions amb una enorme quantitat de fitxers de codi font. Proporciona una gestió eficient de projectes grans, donada la rapidesa de gestió de diferents arxius, entre altres millores d'optimització de velocitat d'execució.

Entre les característiques més rellevants es troben:

* Forta incidència en la no-linealitat dels canvis.
* Gestió distribuïda. Els canvis s'importen com a ramificacions, i poden ser barrejades en la manera en què ho fa una ramificació de l'emmagatzemament en local.
* Els magatzems d'informació poden publicar-se per HTTP [R], FTP [R], SSH [R], rsync [[R](https://rsync.samba.org/)] o mitjançant un protocol natiu, a part de ser possible emular CVS [[R](http://cvs.nongnu.org/)].



Figura 11. Logotip de Git.

Per accelerar el processament de les dades relacionals es va utilitzar el programari Apriori [[R](http://www.borgelt.net/apriori.html)].

* Apriori és un programa per trobar regles d'associació i el conjunt d'elements més freqüents amb l'algoritme apriori [[R](https://en.wikipedia.org/wiki/Apriori_algorithm)] que porta a terme una recerca en amplada sobre les dades. Aquesta aplicació és molt ràpida, ja que utilitza un arbre ja definit per a organitzar els comptadors per al conjunt d'elements.

*1.3.2.4 Sistemes Gestors de Base de Dades (SGDB)*

En la última fase del treball es va dur a terme la investigació de les possibles aplicacions de les 2-estructures en l’anàlisi de dades relacionals. I per l’anàlisi i gestió d’aquest tipus de dades es va triar les bases de dades SQLite.

* SQLite és una base de dades relacional continguda en una llibreria escrita en C. Incorpora un motor de bases de dades SQL independent. Que a diferència d’altres bases de dades relacionals, no és un sistema que funciona amb un paradigma client-servidor, sinó que s’integra dins d’altres programes. Les bases de dades SQLite s'emmagatzemen en un fitxer que conté tant la definició de l'estructura de les dades com les dades mateixes.



Figura 12. Logotip de SQLite.

Bàsicament es va triar aquest tipus de base de dades relacional perquè el codi per SQLite és de domini públic, multi plataforma i és la base de dades de major desplegament al món. Alhora de començar una línia d’investigació és important la senzillesa i la generalització que aporta.

1.4 Estructura del document

El treball s’estructura en dues parts: memòria i apèndix. A la memòria es sintetitza les aportacions realitzades per l’estudiant. A l’apèndix es dóna una guia d’instal·lació i ús de l’entorn d’anàlisi i visualització del paquet de software.

El contingut dels capítols de la memòria és el següent. En el capítol 2 es mostra la planificació que s’ha dut a terme durant el desenvolupament del treball. En el capítol 3 i 4 es presenta l’anàlisi econòmic i de sostenibilitat del treball. En el capítol 5 es documenta l’anàlisi i el disseny del paquet de software desenvolupat.

En els capítols 6, 7, 8 i 9 es documenta la implementació del paquet de software desenvolupat. Concretament, en el capítol 6 es descriuen les dades d’entrada que es necessiten. En el capítol 7 es descriu la generació de grafs. En el capítol 8 s’explica com funciona la descomposició dels grafs en 2-estrcutures. I finalment el capítol 9 s’associa explícitament a les 2-estructures.

En el capítol 10 es resumeix el treball realitzat, es dóna una valoració personal d’aquest i es relaciona amb les maneres possibles de continuar-lo.

1. PLANIFICACIÓ TEMPORAL

2.1 Descripció de les tasques

Inicialment el treball es va planificar fins al torn de presentació de febrer. No obstant, per la falta de temps, es va prorrogar la presentació fins al torn de maig. Per tant, la planificació del treball s’ha estès de la següent manera:

* Fase 1 (Fase inicial): Des del 11/07/2016 al 09/09/2016 al diagrama de Gantt. Fase d’entrar en contacte amb l’entorn i valorar les diferents opcions i algoritmes a implementar.
* Fase 2 (Fase d’implementació bàsica): Des del 12/09/2016 al 25/11/2016 al diagrama de Gantt. Fase on, un cop decidits els algoritmes, les llibreries de visualització i altres detalls, es comencen a implementar els algoritmes bàsics sobre unes dades mínimament realistes en un entorn de visualització senzill.
* Fase 3 (Fase d’implementació avançada): Des del 28/11/2016 al 27/01/2017 al diagrama de Gantt. Fase on, una vegada tenim la base, podem realitzar el procediment per acabar d’implementar els algoritmes definitius del treball i la integració d’aquests amb les bases de dades relacionals.
* Fase 4 (Fase d’adaptació): Des del 30/01/2017 al 28/02/2017 al diagrama de Gantt. Fase on, ja desenvolupats els algoritmes, els adaptem per a que siguin més específics i realistes de cara als aspectes de l’anàlisi de dades relacionals.
* Fase 5 (Fase d’optimitzacions i millores): Des del 01/03/2017 al 31/03/2017 al diagrama de Gantt. Acabat el programari principal, es busquen tot tipus d’optimitzacions per a millorar el temps i els resultats del programa.

2.2 Diagrama de Gantt

Figura 13. Diagrama de Gantt.

1. GESTIÓ ECONÒMICA I PRESSUPOST

3.1 Identificació de costos

Podem dividir els costos involucrats d’aquest treball en: costos de recursos humans, de maquinari i de programari.

3.2 Recursos humans

Pel desenvolupament d’aquest treball es necessita com a mínim un/a programador/a que tingui coneixement previ en ciències de la computació i tingui per tant el coneixement sobre la teoria de grafs i sobre la cerca i l’anàlisi d'informació massiva.

El sou brut anual d’un perfil d’aquest estil, amb una titulació superior en enginyeria informàtica, pot ser aproximadament uns 32.000 €. Sense tenir en compte els extres, aproximadament, uns 2.667 € al mes.

Per calcular els costos reals que suposa pel contractant hem d’afegir-hi les cotitzacions socials del 29,70% (23,60% de contingències comuns, 5,50% d'atur i 0,60% de formació professional).

|  |  |
| --- | --- |
| **Costos pel contractant** | **Preus** |
| Sou brut mensual | 2.667 € |
| +23,60% Cotitzacions de contingències comuns | 629,41 € |
| +5,50% Cotitzacions d'atur | 146,68 € |
| +0,60% Cotitzacions de formació professional | 16 € |
| **Total** | 3.459,09 € |

Taula 1. Costos que suposa el treballador pel contractant.

Pel que fa al/a treballador/a se li descompta del sou brut mensual: el 6,35% de cotitzacions socials (4,70% de contingències comuns, 1.55% d'atur, 0,10% de formació professional) i el 20% de IRPF.

|  |  |
| --- | --- |
| **Costos pel treballador** | **Preus** |
| Sou brut mensual | 2.667 € |
| -4,70% Cotitzacions de contingències comuns | -125,35 € |
| -1,55% Cotitzacions d'atur | -41,34 € |
| -0,10% Cotitzacions de formació professional | -2,67 € |
| -20% de IRPF | -533,40 € |
| **Total** | 1.964,24 € |

Taula 2. Sou net que rep el treballador.

3.3 Recursos de maquinari

Els recursos de maquinari necessaris per a aquest treball són molt limitats. Bàsicament es necessita un únic ordinador de gama alta per a la realització de les proves dels algoritmes, el control de versions extern, la gestió i l’administració d’una base de dades i per a l’execució del programari corresponent.

No obstant, com que ja es disposava d’un ordinador de gama alta, els costos de maquinari són mínims i es resumeixen de la següent manera:

|  |  |
| --- | --- |
| **Maquinari** | **Preus** |
| Ordinador de gama alta | (1.200 €) 0 € |
| **Total** | 0 € |

Taula 3. Preu del maquinari.

3.4 Recursos de programari

Tots els recursos de programari que s’utilitzen no comporten cap despesa. Fins i tot, el programari que normalment és de pagament, s’ha pogut demanar la llicència estudiant durant un any i s’ha utilitzat gratuïtament. També es tenen en compte les dades relacionals que s’utilitzen en aquest treball, però que tampoc tenen cap cost degut a que són proporcionades pel director del treball.

|  |  |
| --- | --- |
| **Programari** | **Preus** |
| Llibreries (NetworkX, Pydot, etc.) | 0 € |
| Programari Graphviz | 0 € |
| Programari Pycharm Professional | (199 €/any) 0 € |
| Aplicació Web Wrike Professional | (588 €/any) 0 € |
| Programari Astah Professional | (57 €/any) 0 € |
| Control de versions GitHub | 0 € |
| Data Sources | 0 € |
| **Total** | 0 € |

Taula 4. Preu del programari.

3.5 Resum de costos

Finalment, podem resumir els diferents costos mensuals de recursos humans, de maquinari i de programari de la següent manera:

|  |  |
| --- | --- |
| **Concepte** | **Preus** |
| Salaris | 3.459,09 € |
| Amortitzacions de maquinari | 0 € |
| Manteniment de llicències de programari | 0 € |
| **Total** | 3.459,09 € |

Taula 5. Resum de costos mensuals.

I el cost total d'aquest treball, si tenim en compte que es realitza durant 11 mesos, aproximadament, es resumiria de la següent manera:

|  |  |
| --- | --- |
| **Concepte** | **Preus** |
| Salaris | 38.049,99 € |
| Amortitzacions de maquinari | 0 € |
| Manteniment de llicències de programari | 0 € |
| **Total** | 38.049,99 € |

Taula 6. Resum de costos totals.

S'ha de tenir en compte que el cost real d'aquest projecte, al estar realitzat dins de l’àmbit d'un Treball Final de Grau de modalitat A, seria nul en el concepte de salaris.

1. SOSTENIBILITAT I COMPROMÍS SOCIAL

4.1 Valoració de la sostenibilitat del treball

La viabilitat econòmica d’aquest treball és alta, principalment perquè els recursos a invertir són mínims i si el treball acaba sent utilitzat per alguna empresa, la devolució de la inversió resultant seria elevada. A més els costos de materials són mínims, ja que és un treball purament de programació, i el material que s’ha utilitzat ja s’havia adquirit anteriorment. Per aquest motiu, pel fet de reutilitzar material existent, la valoració ambiental és molt alta. I una vegada s’acabi la vida útil del material d’aquest projecte es pot seguir reutilitzant, per exemple donant-l’hi a la associació de la UPC, Tecnologia per a Tothom (TxT) [[R](https://txt.upc.es/)], que permet reutilitzar ordinadors antics per a donar-los a gent més necessitada.

La viabilitat social d’aquest treball també és bona, ja que amb la seva aplicació es podrà millorar l’eficiència de l’anàlisi de dades relacionals, i per tant suposarà la millora tant com per a les empreses que perden molt de temps amb un munt de dades per a analitzar com pels treballadors que les analitzen.

En els següents apartats es detallen els aspectes de sostenibilitat econòmics, socials i ambientals responent un conjunt de preguntes.

4.2 Aspectes econòmics

* *El cost del treball ho faria viable si hagués de ser competitiu?*

Sí. Perquè només es necessita un sol programador/a amb coneixement de ciències de la computació. I el maquinari i el programari són limitats o directament gratuïts com s’ha comentat a l’anàlisi econòmic. No obstant, s’hauria de comptar amb la col·laboració d’alguna empresa del sector Big Data que tingui accés a més dades. Cosa que faria augmentar el cost del treball. Però, tot i així seria viable.

* *Es podria realitzar un treball similar en molt menys temps o amb molts menys recursos, i per tant menor cost?*

Es segur que amb més recursos, com per exemple amb més programadors i els seus propis ordinadors es podria realitzar aquest treball més ràpidament. Però és difícil reduir el cost en general, ja que, com s’ha comentat anteriorment els costos del treball són mínims.

* *El temps dedicat a cada tasca és proporcional a la seva importància (s'ha dedicat molt de temps a desenvolupar parts del treball que podien haver estat reutilitzades amb tecnologies / projectes / coneixements existents)?*

Totes les tasques desenvolupades en aquests treball no tenen una base ja existent, s’ha portat a terme un treball d’investigació. I s’ha dedicat més temps del normal en general.

4.3 Aspectes socials

* *Hi ha una necessitat real del producte / servei?*

L’anàlisi de les dades relacionals per part de les empreses “no és un treball futur”, sinó que és ja essencial per a la seva estratègia de negoci i cap empresa, independentment de la seva mida, n’hauria de quedar fora, donat que s’ha demostrat que és un element clau que permet prendre millors decisions, tant estratègiques com en el dia a dia, fins al punt de anticipar-se al mercat. Amb aquest treball es vol millorar l’anàlisi d’aquestes dades i l’adaptació a la situació actual de forma molt més eficient.

* *Satisfer aquesta necessitat, millora la qualitat de vida dels consumidors?*

Una major quantitat de dades implica més vulnerabilitat per als consumidors ja que n’augmenta el perill d’ús maliciós. I protegir el consumidor d’això és fonamental.

A més, l’ús de l’anàlisi de dades en camps com en la fixació de preus, una tendència a l’alça, pot servir per equilibrar el preu de venda en temps real en funció de l’oferta i la demanda. Com a conseqüència tindríem preus més adients pels consumidors.

4.4 Aspectes ambientals

* Quins recursos es poden reaprofitar d'altres treballs?

La majoria del maquinari que s’utilitza en aquest projecte no s’ha adquirit per a la realització del mateix.

* S’ha tingut en compte el desmantellament del procés/producte al medi ambient? Si és un producte, s’han tingut en compte en el seu disseny criteris de facilitació de les seves posterior reciclatge?

Una vegada acabi la vida útil dels ordinadors utilitzats durant el treball es poden donar a l‘associació per a la reutilització dels ordinadors de la UPC anomenada TxT (Tecnologia per a Tothom) que readaptaran els ordinadors per a poder distribuir-los a persones que no en tenen.

**PART II**

**IMPLEMENTACIÓ**

1. ANÀLISI I DISSENY DE LES 2-ESTRUCUTRES

L’anàlisi i el disseny realitzat per a la implementació es comenta breument a continuació. Concretament, el model de dades de les 2-estructures1 i les principals decisions de disseny adoptades.

A la figura 14 es mostra el diagrama de classes amb les representacions de les principals estructures i les seves relacions2.



Figura 14. Diagrama de classes amb el modelat i el disseny de les 2-estructures.

La classe Data gestiona i adequa les dades d’entrada per a crear el Graph.

La classe Graph crea un graf amb les dades que genera la classe Data per a la construcció d’una 2-estructura. El graf pot ser de tipus pla, pla amb llindar, lineal o exponencial. El tipus de graf amb el que es crea una 2-estructura indica el tipus de la 2-estructura.

Des del primer moment es va pensar en representar les classes d’equivalència [R] pintant les arestes del graf de diferents colors. Perquè el resultat final que es vol sigui el més visual i intuïtiu possible. Cada classe d’equivalència està representada per un únic color.

La classe Subset genera tots els subconjunts possibles que té el Graph.

Per a cada parella <Graph, Subset> existeix un únic Clan, el qual constitueix la descomposició del graf que està formada per un número finit de subconjunts d’aquest segons si tenen la mateixa classe d’equivalència o no. L’ús dels clans millora l’eficiència de les consultes sobre les propietats de les 2-estructures.

A la figura 14 també podem observar que un Clan pot ser Trivial o Primal i per

tant es generalitza amb aquestes dues subclasses. La classe Trivial representa els clans de longitud u i els clans que contenen tots els elements del graf. I la classe Primal representa els clans que no experimenten overlapping [R]. A més, un clan trivial sempre es considera que també és primer. Es va prendre aquesta decisió perquè una 2-estructura bàsicament està formada de clans primers.

La classe Structure crea la 2-estructura que està basada en la descomposició del graf en clans. Com s’ha comentat anteriorment, el graf i la 2-estrcutura són del mateix tipus. I per tant, la 2-estructura pot ser de tipus pla, pla amb llindar, lineal o exponencial.

Finalment la classe Interface genera la visualització del graf i de la 2-estructura amb l’ajuda del programari interactiu Graphviz.

A continuació es mostra el diagrama de classes complet amb les representacions de les principals estructures, les seves relacions i operacions:



Figura 15. Diagrama de classes complet amb el modelat i el disseny de les 2-estructures.

Durant la fase d’optimització i millores es va veure que la unitat central de processament no podia dur a terme l’execució del paquet de software per a grans grups de dades. Per aquest motiu es va canviar, una mica, el disseny anteriorment esmentat. I enlloc de generar tots els subconjunts possibles del graf es generen els subconjunts més freqüents possibles del graf.

A la figura 16 es mostra el nou diagrama de classes amb les representacions de les principals estructures i les seves relacions2.



Figura 16. Nou diagrama de classes amb el modelat i el disseny de les 2-estructures.

La diferència amb el primer disseny és que la classe Data, ara gestiona i adequa les dades d’entrada per a crear els subconjunts possibles del graf mitjançant l’algoritme apriori. El funcionament de l’algoritme apriori s’explica més endavant. Així, la classe Subset generà tots els subconjunts més freqüents possibles que té el Graph.

1Veure l’estat de l’art per a una introducció a les 2-estructures.

2S'han omès els atributs i les operacions de les classes per claredat.

A continuació es mostra el nou diagrama de classes complet amb les representacions de les principals estructures, les seves relacions i operacions:



Figura 15. Diagrama de classes complet amb el modelat i el disseny de les 2-estructures.

1. Dades d’entrada

Alhora de treballar amb el paquet de software podem distingir tres tipus de fitxers de dades d’entrada. O sigui que només es permet la càrrega de dades relacionals dels tipus de fitxers següents:

* Fitxers de tipus ARFF [[R]](http://www.cs.waikato.ac.nz/ml/weka/arff.html)
* Fitxers de tipus TXT
* Fitxers de tipus DB [[R](https://www.sqlite.org/fileformat.html)]

Els fitxers de dades contenen les dades relacionals necessàries per a la creació de les taules de les bases de dades relacionals, que s’utilitzen per a la creació dels grafs i també per a la generació dels subconjunts més freqüents de cada graf. I conseqüentment per a cada 2-estructura.

Es van triar els fitxers de tipus ARFF i BD perquè són dos formats molt utilitzats en el camp de la mineria de dades. Els fitxers de tipus TXT també s’utilitzen però la seva elecció va lligada a la simplicitat i els pocs errors que això provoca.

* 1. Fitxers ARFF

Els fitxers ARFF van ser desenvolupats pel Projecte d'aprenentatge automàtic en el Departament de Ciències de la Computació de la Universitat de Waikato, Nova Zelanda, per al seu ús amb el programari d'aprenentatge de màquina Weka [[R](http://www.cs.waikato.ac.nz/~ml/)]. És un tipus de fitxer molt utilitzat en la mineria de dades [[R](https://en.wikipedia.org/wiki/Data_mining)].

Un arxiu ARFF (Atribut-Relació File Format) és un arxiu de text ASCII que descriu una llista d'instàncies que comparteixen un conjunt d'atributs. L'estructura del fitxer és molt senzilla i es divideix en 3 seccions, @relation, @attribute i @data:

* @relation <relation-name>

Tot fitxer arff ha de començar amb aquesta declaració a la primera línia ja que no es poden deixar línies en blanc al principi. *<relation-name>* és una cadena de caràcters amb el nom que se li vol donar a *@relation*.

* *@attribute <attribute-name> <datatype>*

En aquesta secció s’inclou una línia per a cada atribut (o columna) que es vulgui incloure en el conjunt de dades, indicant el nom i el tipus de dada. Amb *<attribute-name>* es proporciona el nom de l'atribut, que ha de començar per una lletra. I amb *<datatype>* es representa el tipus de dada per a aquest atribut (o columna).

El tipus de dada de *<datatype>* pot ser:

* Numèric. Nombres reals o enters.
* string (text).
* date [*<date-format>]* (data). Format de la data, que és del tipus "yyyy-MM-dd'T'HH: mm: ss" (ISO 8601 [[R](https://tools.ietf.org/html/rfc3339)]).
* *<nominal-specification>*. Llista de possibles valors: *{<nominal-name1>, <nominal-name2>, <nominal-name3>, ...}.*

Els tipus de dades numèric, text i data són sensibles a majúscules.

El tipus de dada text és molt útil en aplicacions de mineria de dades ja que permet crear conjunts de dades a partir de diferents cadenes d’atributs.

* *@data*

En l’ultima secció s’inclouen les dades pròpiament dites. Les columnes es troben separades per comes i totes les files (cada línia és una fila) han de tenir el mateix nombre de columnes, el número total de declaracions *@attribute* de la secció anterior.

Si no es disposa d'alguna dada, es col·loca un signe d'interrogació (?) en el seu lloc. El separador de decimals ha de ser obligatòriament un punt i les cadenes de tipus string han d'estar entre cometes simples.

**Exemple 6.** A la figura 16 es pot veure el format d’un fitxer ARFF apte per l’entorn de proves del paquet de software.

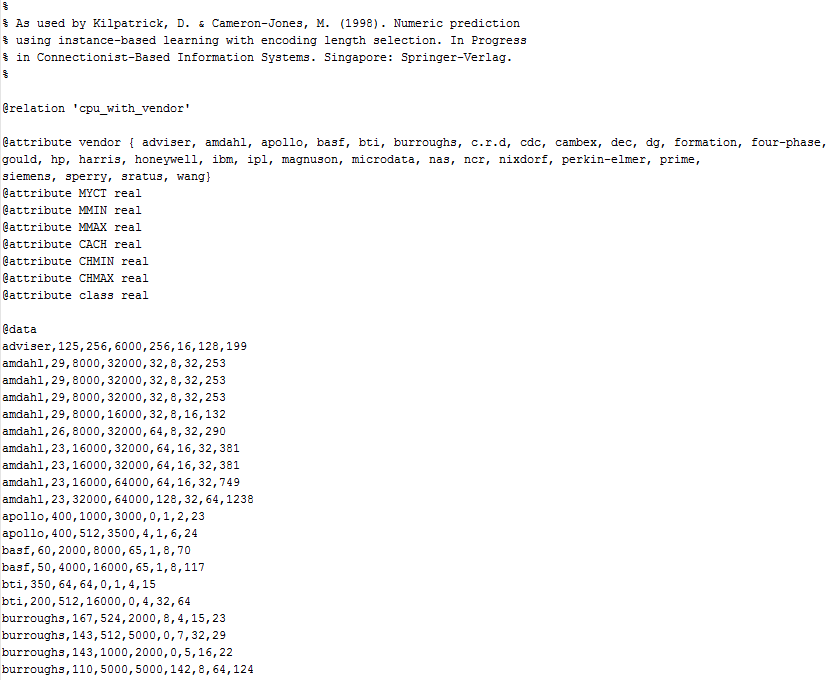


Figura 16. Fitxer ARFF.

Les línies que comencen amb el caràcter % són comentaris. Les declaracions @relation, @attribute i @data són sensibles a majúscules.

6.2 Fitxers TXT

Un fitxer TXT és un document de text estàndard que conté text sense format. Això fa que sigui reconegut per a qualsevol programa de processament d'edició de text.

A causa de la seva simplicitat, els fitxers de text s'utilitzen molt sovint per a l'emmagatzematge d'informació. Ja que eviten alguns dels problemes trobats en altres formats de fitxer, com ara l’ordre de bytes (*endianness* [[R](https://www.google.es/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&cad=rja&uact=8&ved=0ahUKEwjcw6v1i7vTAhULVxQKHcq-AfMQFggmMAE&url=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FEndianness&usg=AFQjCNGSFyerytQ4VYCGNO0RELDc6-lqbg)]), els bytes de farciment, o diferències en el nombre de bytes en una paraula màquina.

Un desavantatge dels arxius de text és que en general tenen una entropia baixa, el que significa que la informació ocupa més espai d'emmagatzematge del necessari.

**Exemple 7.** A la figura 17 es pot veure el format d’un fitxer TXT apte per l’entorn de proves del paquet de software.



Figura 17. Fitxer TXT.

La primera línia del fitxer representa les columnes. I a la resta de línies hi ha les dades de cada fila. Cada línia està formada per parells de dades <columna:dada>.

6.3 Fitxers DB

Els fitxers DB són un format de fitxer de bases de dades genèriques. La informació que emmagatzema un fitxer d’aquest tipus normalment és una sèrie de taules, camps de taules i valors de dades per a aquests camps. Després la informació és organitzada d'acord amb el model de dades, el model d'estructura corresponen al treball és el model relacional [[R](https://en.wikipedia.org/wiki/Relational_model)].

Els fitxers de tipus DB es poden exportar a CSV [[R](http://fileformats.archiveteam.org/wiki/CSV)].

1. GRAFS

Tot graf implementat en el paquet de software és complet. Un graf és complet quan entre tot parell de vèrtexs existeix una aresta.

* |A| = n · (n-1) si el graf és dirigit.
* |A| = n · (n-1)/2 si el graf no és dirigit.

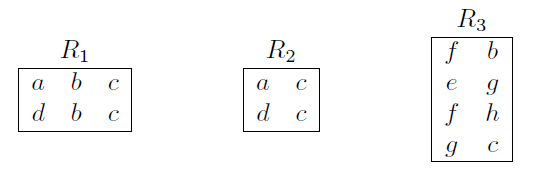
**Exemple 8.** A la figura 18 es pot veure un graf complet de 5 nodes. L’etiquetatge de les arestes és de 0 a n-1.



Figura 18. Graf complet.

A més, tots els grafs del paquet de software implementat compleixen la localitat de Gaifman [R].

**Exemple 9**. Considerant una base de dades que conté les taules R1, R2 i R3 següents:



El graf de Gaifman o graf primer per a aquesta base de dades és la següent graf no dirigit:

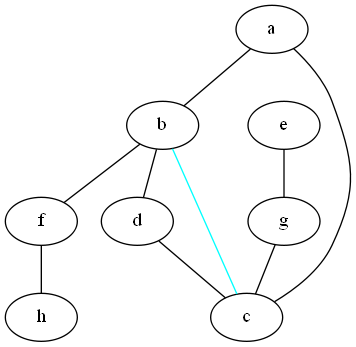


Figura 19. Graf primer.

Per a cada parell de valors hi ha una aresta entre ells si es troben junts en una fila. Per exemple, a i b es troben junts a la primera fila de R1, i per tant existeix una aresta entre ells en el graf de Gaifman. D'altra banda, no hi ha cap fila que contingui a i g, i per tant no hi ha cap aresta entre ells en el graf de Gaifman.

L’aresta b-c es dibuixa d’un color diferent perquè el valor b-c es troba repetit a la primera taula R1.

7.2 creació dels grafs

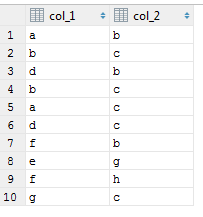
Com s’ha comentat anteriorment, un graf és complet i primer.

A partir de la llibreria NetworkX es crea un graf buit sense nodes i arestes. A NetworkX, els nodes poden ser qualsevol objecte, per exemple, una taula hash (estructura de dades que associa claus o claus amb valors), una cadena de text, una imatge, un objecte XML, un altre graf, etc.

*7.2.1 Inicialització dels grafs*

D’una taula SQLite, ja anteriorment creada a partir d’algun fitxer que conté les dades d’entrada, es seleccionen les dades per a afegir els nodes i les arestes del graf. Cada columna de la taula representa un node del graf i per a cada parell de valors de les files es crea una aresta entre ells.

**Exemple 10.** Considerant la taula SQLite següent:



El graf per a és la següent graf no dirigit:

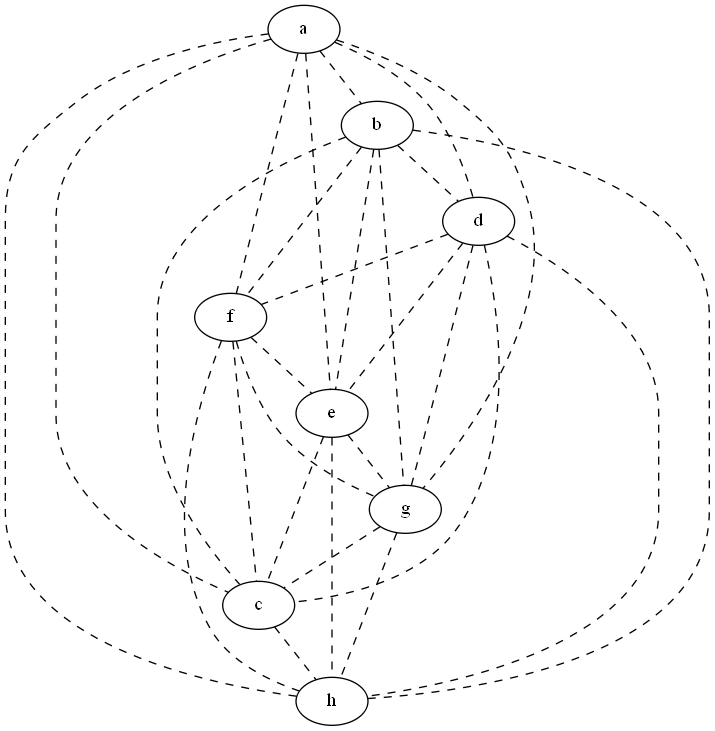


Figura 19. Resultat de la inicialització d’un graf.

S’obvien les repeticions de nodes i arestes entre diferents parells de valors.

*7.2.2 Classes d’equivalència*

Algoritme de pintar classes d’equivalències – labeled edges

Divisió del problema en 4 casos.

* 1. tipus de grafs
* Graf planar o pla

En teoria de grafs, un graf pla o planar és un graf que pot ser dibuixat en el pla sense que cap de les arestes talli a una altra en un punt que no sigui un vèrtex. Els grafs K5 i el K3,3 són grafs no plans minimals, la qual cosa permeten caracteritzar la resta dels grafs no plans.

**Exemple 11.** Anomenant u1, u2 i u3 als vèrtexs superiors i v1, v2, v3 als inferiors i representant com a uivj l'aresta que uneix el vèrtex ui amb el vèrtex vj obtenim el graf K3,3:

K3,3 = {uivj / i, j = 1, 2, 3}

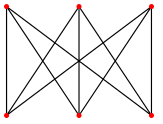


Figura 20. Graf K3,3.

K5 és un graf que té 5 vèrtexs i que té com a arestes totes les línies que connecten cada vèrtex amb tots els altres, com es pot veure a la figura següent:

K5 = {uivj / i = 1, 2, 3, 4, 5}

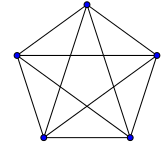


Figura 21. Graf K5.

Aquest graf també s’anomena graf complet de 5 vèrtexs.

També es diu que un graf és pla o planar si i només si no conté un [subgraf](https://ca.wikipedia.org/w/index.php?title=Subgraf&action=edit&redlink=1) (graf parcial d’un graf) que és una [subdivisió elemental](https://ca.wikipedia.org/w/index.php?title=Subdivisi%C3%B3_elemental&action=edit&redlink=1) de K5 o K 3,3 (Teorema de Kuratowski). És a dir, ni K5 ni K 3,3 són grafs plans (ja que cada un d'ells es conté a si mateix com a subgraf). O el que és el mateix, no es poden dibuixar en un paper amb la condició que cap aresta talli a una altra en un punt que no sigui des del principi d’un vèrtex.

A la pràctica, és difícil fer servir el teorema de Kuratowski per decidir ràpidament si un graf és pla. Per això hi ha altres teoremes per a determinar si un graf és pla.

Donat un graf de n vèrtexs i e el nombre d'arestes:

* + Teorema 1. Si n ≥ 3 llavors e ≤ 3 n – 6
  + Teorema 2. Si n > 3 i no hi ha cicles de longitud 3, llavors e ≤ 2 n - 4
* Graf planar o pla amb umbral
* Lineal
* Exponencial

Explicar què fa cada algoritme :D

7.3 visualització del grafs

Export DOT

Com és el fitxer DOT, una mica de palla xD

Exemples

1. DESCOMPOSICIÓ DEL GRAFS

La descomposició del clan de 2-estructura, també conegut com la descomposició modular o la descomposició de substitució, és un exemple d'una descomposició tal - que està estretament relacionades amb la descomposició per quocients en àlgebra.

La idea amb la que es basa la descomposició consisteix en trobar subconjunts d’un graf, anomenats clans, en els quals els elements continguts es relacionen de la mateixa manera amb tots aquells elements fora del subconjunt.

Exemple fàcil

Notar que a l’exemple 4, l’element C es relaciona de la mateixa manera amb els elements A, B, D i E. Això vol dir, que aquests elements pertanyen a la mateixa classe d’equivalència. També podem observar que l’element B es relaciona de la mateixa manera amb els elements A, D i E. O que l’element A o fa amb els elements E i D. I tots aquest elements es troben a les fulles de l’arbre que representa la 2-estructura.

Una propietat que es compleix en el procés de descomposició és que les 2-estructures amb les que es factoritza una 2-estructura original han de pertànyer com a mínim a alguna de les tres subclasses especials. Aquestes subclasses són les 2-estructures completes, lineals i primitives.

Donada una aresta (x, y) es diu que és simètrica si i només sí l’aresta (y ,x) pertany a la mateixa classe d’equivalència. En cas contrari es diu que és antisimètrica. Una 2-estructura és simètrica (antisimètrica) si totes les seves arestes són simètriques (antisimètriques). Per exemple, la 2-estructura que es mostra a la figura 1 és simètrica.

Primer es divideix el graf en clans i es miren si aquests clans són primers. Perquè només ens interessen els clans primers.

Definició del procés de descomposició

Algoritme de descomposició

Algoritme apriori

* 1. clans

Definició clan

* 1. clans trivials
  2. clans primers

dos clans són primers si un està dins de l'altre i són disjunts (disjunts?) explicar

Exemples everywhere

1. 2-ESTRUCTURES

Si be l'algoritme a implementar es basara en l'algoritme NAMOA\* descrit en l'anterior

apartat, la naturalesa del problema fa que s'hagi d'adaptar en alguns aspectes

que descriurem a continuacio.

Utilitzem el nom genèric de '2-estructura' per tal d'emfatitzar que investiguem els sistemes de relacions binàries en un entorn abstracte.

En particular investiguem representacions jeràrquiques de 2-estructures a través d'arbres on la relació "local" entre descendents directes d'un node interior d'un arbre es dóna a través d'una estructura de 2. El resultat principal d'aquest treball és que (per exemple definició jeràrquica) és suficient considerar arbres on aquestes relacions properes es donen ja sigui per un 2-estructura primitiva o una 2-estructura completa o una 2-estructura lineal. Per tant, cada 2-estructura pot ser construïda a partir de (descompost en) 2-estructures primitives, completes i lineals.

La noció d'una estructura 2 és més general que la noció d'un gràfic i menys general que la noció d'una estructura relacional. Aquest treball es desenvolupa la teoria de la 2-estructures, i en particular es demostra que cada 2-estructura pot ser construïda a partir de (descompost en) tres classes de 2-estructures "bàsics". Aquest resultat s'obté a través d'una representació (jeràrquica) de 2-estructures d'arbres.

* 1. creació de les 2-estructures
  2. tipus de 2-estructures

Primitives.

Planar o pla i exemple

Amb Umbral i exemple

Lineal i exemple

Exponencial i exemple

8.3 visualització de les 2-estructures

*8.3.1 Controls gràfics per a la visualització*

1. CONCLUSIONS

En aquest capítol es presenta un resum del conjunt del treball realitzat i es descriuen les possibilitats de continuació. També s’ofereix una valoració personal sobre el desenvolupament del treball.

* 1. Resultats del treball
  2. treball futur
  3. valoració personal

La realització del treball final de grau ha estat una experiència gratificant i enriquidora.

Sobretot pel repte que suposa realitzar un treball d’investigació. I per la doble naturalesa del treball, que es composa d’una part més teòrica i una part més pràctica.

A la part pràctica s’ha desenvolupat el paquet de software, on he pogut aplicar molts dels coneixements de programació adquirits durant la carrera. I a la part teoria es troba l’estudi de les 2-estructures, tema totalment desconegut, que gràcies a la seva exploració m’ha resultat molt interessant.

Aquest treball també s’ha convertit en la oportunitat d’estudiar alguns dels temes inclosos en la mineria de dades, que per problemes d’horaris no vaig poder cursar a l’assignatura de Mineria de Dades (MIDA).

Un altre dels motius a valorar molt positivament és la llibertat que he tingut i el suport que he rebut per a desenvolupar el treball.

**BIBLIOGRAFIA**

**GLOSSARI DE SIGNES, SÍMBOLS, ABREVIATURES, ACRÒNIMS I TERMES**

APÈNDIXS