Universitat de Girona
Escola Politècnica Superior

Treball de Fi de Grau

Estudi: Grau en Enginyeria Informàtica

Títol: Ús de tècniques d'Intel·ligència Artificial per a l'anàlisi de xarxes d'infraestructura urbana

Document: Memòria

Alumne: Joan Saló Grau, Roser Brugués i Pujolràs

Tutor: Eusebi Calle Ortega
Departament: Arquitectura i Tecnologia de Computadors (ATC)
Àrea: ATC (Grup de recerca: BCDS)

Convocatòria (mes/any): Juny 2021

Universitat de Girona
Escola Politècnica Superior

Treball de Fi de Grau

Grau en Enginyeria Informàtica

Ús de tècniques d'Intel·ligència Artificial per a l'anàlisi de xarxes d'infraestructura urbana

Autors:

Joan Saló Grau

Roser Brugués i Pujolràs

Tutor: Eusebi Calle Ortega

Departament d'Arquitectura de Computadors

Universitat de Girona

Girona, Juny 2021

Resum

Introducció

El contingut d'aquest TFG s'engloba dins del projectes **CLEaN-TOUR** i **VIRWASTE**, realitzats conjuntament per membres de l'ICRA (Institut Català de Recerca de l'Aigua) i del grup de recerca BCDS (*Broadband Communications and Distributed Systems*) de la Universitat de Girona. Les línies de recerca d'aquests dos projectes se centren en estudiar com utilitzar la infraestructura urbana de la xarxa de transport d'aigua per tenir un impacte positiu en la societat.

En concret, **CLEaN-TOUR** busca estudiar l'ús d'aigua regenerada en una ciutat turística. Nosaltres ens centrem en el disseny i la planificació del recorregut d'aquesta xarxa de manera que tingui el mínim cost possible. Aquest disseny es pot enfocar des d'una manera centralitzada (enviar l'aigua des de la depuradora a uns destins) o descentralitzada (es divideix tot el territori en clusters, i a cada cluster hi ha un dipòsit. Aleshores, l'aigua s'envia de la depuradora als dipòsits, els quals l'envien als destins corresponents.)

Pel que fa a **VIRWASTE**, aquest és un dels projectes seleccionats dins la convocatòria PANDÈMIES 2020 de la Generalitat de Catalunya. Dins d'aquest projecte, des de l'ICRA i el BCDS portem treballant des de l'estiu passat en el posicionament de sensors per detectar rastres de SARS-CoV-2 en aigües residuals, recerca que s'emmarca dins el context d'aquest projecte.

Objectius

El primer objectiu, que engloba tot el contingut del treball, ens permet integrar tant el projecte de CLEaN-TOUR com el de VIRWASTE en una eina comuna. Aquest es basa en integrar diferents bases de dades i repositoris externs (dades del cadastre, estadístiques de consum d'aigua, model d'elevacions del terreny...) de forma automatitzada perquè el procés sigui el més transparent possible per l'usuari. Aquesta integració permet aplicar els algorismes i mètodes descrits posteriorment.

Els objectius concrets del projecte CLEaN-TOUR són els d'estudiar, implementar i comparar algorismes d'encaminament pel disseny de xarxes d'aigua regenerada, algorismes de clusterització per fer xarxes descentralitzades, i d'optimització per maximitzar la distribució d'aigua a partir d'un pressupost fixat.

Quant als objectius del projecte VIRWASTE, aquests són els d'estudiar, implementar i comparar algorismes per la selecció de punts de mostreig en aigües residuals per la detecció de SARS-CoV-2.

Desenvolupament i implementació

Respecte la part d'encaminament, aquesta es redueix a resoldre el problema de l'arbre d'Steiner. Tot i que ja hi havien dos algorismes desenvolupats per membres del grup per resoldre el problema (algorismes de Kou i Takahashi), s'havia fet poc procés de recerca. Nosaltres hem implementat dos algorismes més: l'algorisme de Mehlhorn, basat en tècniques de teoria de grafs, el qual disminueix de forma molt clara el temps de còmput requerit, i un altre que resol el problema a partir d'aplicar *Ant Colony Optimization*, una tècnica d'Intel·ligència Artificial

En relació al problema d'optimització s'han aplicat tres algorismes per resoldre el problema: el primer consisteix en una cerca per força bruta, el segon es basa es aplicar la minimització de Goemans i Williamson, mentre que el tercer consisteix en aplicar una cerca voraç.

Pel que fa a la clusterització, s'han implementat 4 algorismes, seleccionats d'entre els molts que hi ha en base a un criteris recollits a la memòria, com ara que facin un agrupament estricte i no hi hagi solapament (*hard clustering* en anglès). Aquests algorismes són:

- Espectral - algorisme de connectivitat basat en teoria de grafs
- Agrupament jeràrquic aglomeratiu - algorisme d'aprenentatge automàtic, basat en la connectivitat dels nodes.
- Self Organizing Maps (SOMs) - algorisme d'aprenentatge profund no supervisat.
- Graph Neural Network (GNN) - algorisme d'aprenentatge profund

Quant a la part de posicionament de sensors per detecció de contaminants, s'han implementat dos algorismes per determinar, de forma dinàmica, els punts de mostreig en la xarxa de clavegueram quan es vol localitzar el pacient zero o una àrea de contagi. En aquest sentit, partíem d'un primer algorisme ja implementat per membres d'ICRA i del BCDS, el qual posiciona un determinat número de sensors estàtics a la xarxa, i d'una definició dels algorismes dinàmics publicada en un article de recerca de la revista PLOS ONE.

Conclusions

De la part de CLEaN-TOUR, es conclou que l'opció centralitzada sempre és més econòmica que la descentralitzada, ja que no calen dipòsits per emmagatzemar l'aigua. Però aquesta opció no sempre és viable en ciutats grans, on les distàncies són molt llargues i les pressions de bombeig són massa elevades. Pel que fa als algorismes de clusterització, no n'hi ha clarament un que ofereixi millors resultats, quant a costos, que la resta, i caldria provar amb més ciutats de topologies diferents per veure si es pot arribar a una conclusió en aquest sentit.

Del projecte VIRWASTE, en comparació a altres aproximacions de l'algorisme dinàmic existents, la nostra minimitza lleugerament el número de punts de mostreig necessaris pel fet d'utilitzar dades poblacionals i d'utilitzar el resultat d'un algorisme per posicionar sensors estàtics com a punt de partida.

Agraïments

Per començar vull agrair molt especialment als meus pares, al meu germà i a l'Aleix, pel suport incondicional rebut quan vaig prendre la decisió de deixar-ho tot i tornar a estudiar, així com al llarg d'aquests 4 anys, que no han sigut fàcils. Sense vosaltres, no ho hauria aconseguit.

També a tots els membres del BCDS i d'ICRA que, d'alguna manera o altra, han col·laborat en aquest projecte i m'han ajudat quan ho he necessitat al llarg del seu desenvolupament, especialment en Miquel i en David.

Al meu tutor, l'Eusebi, per haver-me guiat com és degut i estar sempre disponible a ajudar al llarg de tot el procés del treball.

Per acabar, a en Joan, amb qui he compartit molt la major part del grau. No m'imagino millor company de pràctiques o de TFG que ell. Part dels meus resultats acadèmics són gràcies també a ell i al seu compromís i nivell d'exigència en tot el que fa.

Continguts

Índex de figures

Índex de taules

1. Introducció

El contingut d'aquest TFG s'engloba dins del projectes **CLEaN-TOUR** i **VIRWASTE**, realitzats conjuntament per membres de l'ICRA (Institut Català de Recerca de l'Aigua) i del grup de recerca BCDS (*Broadband Communications and Distributed Systems*) de la Universitat de Girona. En el marc d'aquests dos projectes, s'obren dues línies de recerca, les quals se centren en estudiar com utilitzar la infraestructura urbana de la xarxa de transport d'aigua per tenir un impacte positiu en la societat:

i Disseny de xarxes d'aigua regenerada

ii Posicionament de sensors per detectar contaminants en aigües residuals (*sensor placement*)

CLEaN-TOUR és un projecte que busca estudiar l'ús d'aigua regenerada en una ciutat turística. És per això que el seu objectiu és dissenyar de forma automàtica una xarxa de reutilització d'aigua: a partir de les aigües residuals que arriben a la depuradora, s'aplica un procés de neteja i filtratge d'aquesta aigua per després tornar-la a utilitzar no com a aigua potable, sinó per a determinats usos com ara el reg, per a la cisterna del vàter, etc. Per tal de fer aquesta redistribució, s'ha de construir una xarxa nova de canonades des del punt de filtratge fins als destins. Nosaltres ens centrem en el disseny i la planificació del recorregut d'aquesta xarxa de manera que tingui el mínim cost possible. Des del grup de recerca i ICRA s'està treballant en una futura publicació.
El projecte ha comptat amb la col·laboració i supervisió d'ABM, empresa de consultoria i serveis d'enginyeria.

VIRWASTE és un dels projectes seleccionats dins la convocatòria PANDÈMIES 2020 de la Generalitat de Catalunya per analitzar l'impacte de la Covid-19 amb l'objectiu de proposar mesures, models i línies d'actuació per ajudar a superar les conseqüències de la pandèmia. Si bé la resolució de la convocatòria és força recent, des de l'ICRA i el BCDS portem treballant des de l'estiu passat en el posicionament de sensors per detectar rastres de SARS-CoV-2 en aigües residuals, i hi ha un article enviat a la revista *Environment International*, en procés de revisió.

1.1 Motivacions

Tenint en compte que entre el 1960 i el 2010 el consum d'aigua s'ha duplicat, juntament amb el canvi climàtic, el creixement de població i el creixement econòmic; es preveu que l'any 2050 el 70% dels comtats dels Estats Units tinguin escassetat d'aigua potable. Per tant, reutilitzar l'aigua no només ajudaria a reduir la demanda sobre aquesta, sinó que ajudaria a disminuir l'impacte ambiental i eliminaria la necessitat de transportar-ne tanta, entre d'altres beneficis.

Altrament, la pandèmia de la Covid-19 ha posat de manifest per enèsima vegada el problema del canvi climàtic. L'economia global, els sistemes de producció i els nostres patrons de consum no només amenacen el planeta, sinó també la nostra salut. Estudis recents apunten que hi ha un vincle molt estret entre la desforestació i l'aparició de virus d'origen animal en reservoris de fauna silvestre. L'emergència sanitària del coronavirus ens ha de fer reflexionar com a societat i hem d'aprofitar l'oportunitat per analitzar què ens ha portat a aquesta situació per idear un futur més sostenible i evitar futures pandèmies, o afrontar-les més ben preparats.

1.2 Objectius i abast

Tot i que ja hem mencionat els objectius de forma general, els enumerem de forma més detallada:

1. Automatització del procés d'obtenció i integració de dades de diferents fonts i capes per poder utilitzar-ho en la implementació dels algorismes dels projectes CLEaN-TOUR i VIRWASTE.
2. Estudi, implementació i comparativa de diferents algorismes d'encaminament pel disseny de xarxes d'aigua regenerada pel projecte CLEaN-TOUR.
3. Estudi, implementació i comparativa de diferents algorismes d'optimització per maximitzar la distribució d'aigua dins d'un pressupost donat pel projecte CLEaN-TOUR.
4. Estudi, implementació i comparativa de diferents algorismes de clusterització per la sectorització de ciutats pel projecte CLEaN-TOUR.
5. Estudi, implementació i comparació d'algorismes per la selecció de punts de mostreig en xarxes d'aigua residuals per la detecció de contaminants, SARS-CoV-2 en el cas del projecte VIRWASTE.
6. Documentar i recollir tota la informació, resultats i conclusions del projecte en una memòria.

1.2.1 Atomització de tasques

Un cop enumerats els objectius principals del nostre projecte, a continuació els desglossem en forma de tasques per tal d'acomplir-los.

1. Obtenció i integració de dades
    - Descàrrega del graf de carrers de la ciutat objecte d'estudi a través de l'API de OpenStreetMaps.[1]
    - Verificació i reconciliació de la xarxa de clavegueram de la ciutat de Girona facilitada per Aigües de Girona.[1]
    - Descàrrega del model digital d'elevacions (DEM) a través de l'API de l'Instituto Geográfico Nacional (IGN).
    - Descàrrega de dades referents a l'ús i superfície de les parcel·les del cadastre.[1]
    - Obtenció d'indicadors demogràfics i de consums de cada parcel·la.[1]
    - Generar el graf que contingui i integri tota la informació necessària per cadascun dels dos projectes.[2]
2. Encaminament
    - Cerca bibliogràfica per identificar referències i articles relacionats
    - Implementació de l'arbre d'Steiner amb Ant Colony Optimization
    - Implementació de l'arbre d'Steiner amb l'algorisme de Mehlhorn
    - Anàlisi i comparació de resultats
3. Optimització
    - Cerca bibliogràfica per identificar referències i articles relacionats
    - Implementació del problema del pressupost a partir d'una cerca de força bruta
    - Implementació del problema del pressupost a partir de la minimització GW
    - Implementació del problema del pressupost amb implementació voraç
    - Anàlisi i comparació de resultats
4. Clusterització
    - Cerca bibliogràfica per identificar referències i articles relacionats
    - Implementació de clusterització espectral

[1] Tasca en la qual no hem participat.
[2] Tasca en la qual hem participat juntament amb altres persones.

- Implementació de l'agrupament jeràrquic
- Implementació d'un *Self Organizing Map* (SOM)
- Implementació d'una *Graph Neural Network* (GNN)
- Anàlisi i comparació de resultats

5. Detecció contaminants

- Cerca bibliogràfica per identificar referències i articles relacionats
- Implementació de l'algorisme dinàmic de selecció de punts de mostreig
- Comparació de la nostra proposta amb d'altres ja existents

1.2.2 Abast del projecte

Per la part de treball enfocada en el projecte **CLEaN-TOUR** s'han fet servir les ciutats de Lloret de Mar i Girona com a casos d'estudi. Segons l'Institut Català d'Estadística (dades de 2020), la superfície de Lloret és de 48.71 km^2 de superfície i té 39089 habitants, mentre que Girona té 39.12 km^2 de superfície i una població de 103369 habitants. El disseny de la xarxa s'ha fet des de dues perspectives: i) model centralitzat, ii) model descentralitzat. Encara que en ambdós casos l'aigua residual arriba primer a la depuradora, on és filtrada, en el primer model es reparteix directament als punts de destí; mentre que en l'opció descentralitzada l'aigua es trasllada primer a uns dipòsits intermedis (1 per cada clúster), des d'on després es distribueix als destins.
Per comprovar la correctesa dels resultats, la xarxa dissenyada s'hauria de verificar amb EPANET, un programari de domini públic de modelització de xarxes de distribució d'aigua capaç de dur a terme simulacions de llarg temps del comportament hidràulic i de la qualitat de l'aigua en xarxes de canonades pressuritzades. Aquest pas, però, queda fora de l'abast del nostre projecte.

En quant a la part de treball de posicionament de sensors i detecció de contaminants, emmarcat dins del projecte **VIRWASTE**, s'ha fet servir la ciutat de Girona com a cas d'estudi. Partim del posicionament de K sensors estàtics (fixes) a la xarxa de sanejament a partir d'una funció d'avaluació. Llavors, quan un d'aquests sensors detecta la presència del virus, s'agafa la part de la xarxa coberta per aquest sensor i es fa córrer l'algorisme dinàmic que hem implementat per determinar en quins punts cal extreure mostres per acabar trobant el focus d'infecció/contagi.
En aquest sentit, seria bo poder fer una modelització de l'arrossegament i acumulació de contaminants (traces de SARS-CoV-2) amb SWMM, que és un programari de codi obert capaç de simular el moviment de l'aigua de precipitació (esdeveniments pluviomètrics) i dels contaminants a través de les xarxes d'aigües residuals i col·lectors. Aquesta part no és objecte del treball.

1.3 Especificació del punt de partida

Com ja hem comentat, aquest treball s'ha desenvolupat dins del grup de recerca BCDS, i hem partit d'una base i feina ja realitzada en el cas dels dos projectes que engloba el treball. Així, pel que fa al projecte **CLEaN-TOUR**, quan vam començar el treball ja disposàvem d'una sèrie d'elements, que són els següents:

- Graf de carrers de la ciutat objecte d'estudi, format pels pels carrers i les interseccions entre aquests. Assumim que la xarxa de reutilització d'aigua seguirà la distribució de la xarxa de carrers.
- Per cada parcel·la on hem de fer arribar aigua regenerada, a través de les dades disponibles del cadastre, tenim informació dels usos d'aquesta i del nombre d'habitants vinculats. D'aquesta manera, a partir dels consums mitjans per cada ús (tal com mostra la taula 1.1), s'ha estimat un consum aproximat per cada parcel·la. Tota aquesta informació, que ha estat obtinguda dins del marc del projecte CLEaN-TOUR, està recollida i guardada en forma d'atributs al seu punt (node) més proper del graf de carrers que hem mencionat en el punt anterior. També disposem d'un programa (*script*) per llegir la informació dels usos i canviar el seu format per poder treballar-hi més fàcilment.

Taula 1.1: Consums d'aigua per destí, segons usos

Ús	**Ús de l'aigua**	**Unitat**
Descàrrega de l'aigua de vàter a casa	0.1575	m^3/dia per llar
Descàrrega de l'aigua de vàter en hotel	0.0036	m^3/dia per m^2 d'hotel
Descàrrega de l'aigua de vàter en restaurant	0.0006	m^3/dia per m^2 de restaurant
Descàrrega de l'aigua de vàter en oficina	0.00204	m^3/dia per m^2 d'oficina
Reg de jardins domèstics	0.006	m^3/dia per m^2 de jardí
Reg de jardins privats	0.000154	m^3/dia per m^2 de jardí
Centre comercial	0.0012	m^3/dia per m^2 de centre comercial
Centre esportiu	0.0012	m^3/dia per m^2 de centre esportiu
Camp de futbol	0.006	m^3/dia per m^2 de camp de futbol
Càmping	0.00087	m^3/dia per m^2 de càmping
Horts	0.004	m^3/dia per m^2 d'hort

- Mètodes per calcular el cost d'una xarxa de reutilització d'aigua (no només el material, sinó també la seva instal·lació) donat el seu recorregut. El desglossament dels costos es pot veure en la taula 1.2. Aquesta informació ha estat proporcionada per ABM, a excepció dels costos per diàmetres entre 710-1400mm (*). Sabíem que aquests diàmetres són d'ús comercial, però no disposàvem de dades reals. Llavors els hem predit a partir de la regressió lineal dels costos coneguts i aplicant una extrapolació. Es pot veure la representació gràfica de tots els costos en la figura 1.1.

Taula 1.2: Cost de construcció per metre lineal de canonada

Canonada (mm)	Cost de material (€)	Cost de mà d'obra (€)	Cost total (€)	Canonada (mm)	Cost de material (€)	Cost de mà d'obra (€)	Cost total (€)
63	1.71	69.13	74.38	315	39.83	167.00	217.17
75	2.43	71.33	77.45	400	64.07	194.49	271.49
90	3.49	72.97	80.28	450	112.95	205.77	334.66
110	4.94	74.62	83.54	560	173.57	230.55	424.33
125	6.29	76.82	87.27	630	223.13	242.95	489.38
140	7.92	79.02	91.29	710	-	-	525.28 *
160	10.31	81.77	96.68	800	-	-	592.28 *
180	13.05	98.27	116.89	900	-	-	666.72 *
200	16.11	112.01	134.53	1000	-	-	741.15 *
225	20.43	125.76	153.50	1200	-	-	890.03 *
250	25.04	139.50	172.77	1400	-	-	1038.91 *

Cost de material (€) – inclou el preu de la canonada, Cost de mà d'obra (€) – inclou els preus dels treballadors i el d'obrir i tancar la rasa (57.04€), Cost total (€) – especifica el cost total de construcció per metre incloent els costos de material, de mà d'obra i un 5% addicional per costos indirectes.

Vàlvula (mm)	**Cost (€)**	**Vàlvula (mm)**	**Cost (€)**	**Vàlvula (mm)**	**Cost (€)**
40	89.29	150	334.97	450	3095.43
50	100.46	200	650.00	500	4058.26
65	125.77	250	865.55	600	8026.65
80	169.88	300	1116.81	700	9014.04
100	210.88	350	1812.51	800	12426.54
125	278.35	400	2388.50	900	14978.90

Dipòsit (m^3)	**Cost (€)**
450	240000
2500	350000
5000	440000
10000	560000
20000	760000

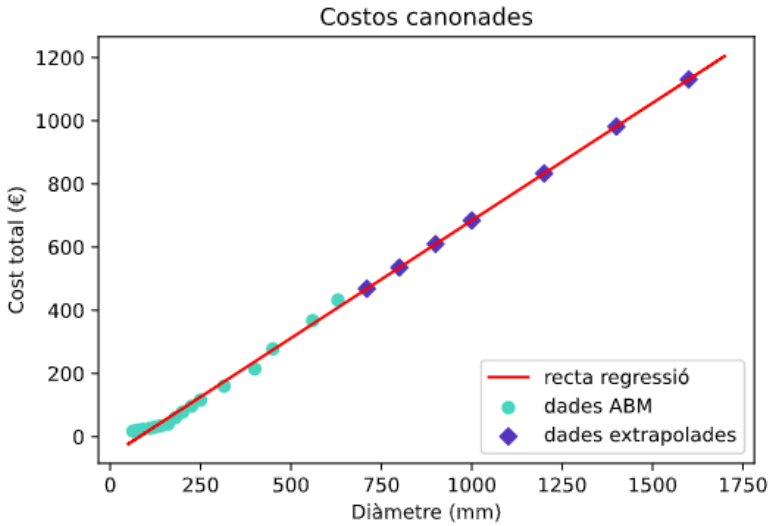


Figura 1.1: Extrapolació de costos

Pel que fa al projecte **VIRWASTE**, ja disposàvem de la xarxa de clavegueram verificada i reconciliada en forma de graf, on les canonades estaven representades per les arestes, i els pous de registre, que eren els nodes. Cada node tenia vinculat el nombre d'habitants que hi estaven connectats. També estava implementat l'algorisme estàtic, que donada aquesta xarxa i un nombre concret de sensors, avaluava la millor posició per situar aquests sensors i quins nodes servia.

1.4 Estructura de la documentació

Després d'aquest primer capítol introductori i d'exposició dels objectius del treball, la resta de memòria està estructurada en 9 capítols:

2. **Metodologia i planificació** - S'explica quina metodologia de treball hem seguit durant el desenvolupament del nostre TFG i la planificació temporal de les tasques involucrades.
3. **Marc de treball i conceptes previs** - S'introdueixen conceptes fora del camp de la informàtica necessaris per entendre el projecte, així com s'aprofundeix en d'altres del nostre àmbit que s'han implementat i són importants conèixer.
4. **Anàlisi i requisits de sistema** - Conté els requeriments, a nivell de dades, per a la implementació tant de la part del projecte CLEaN-TOUR com del VIRWASTE.
5. **Estudis i decisions** - Fa esment al llenguatge i entorn de programació utilitzat, així com les biblioteques més rellevants per la implementació del projecte.
6. **Implementació i proves** - S'explica tot el procés seguit en la implementació del codi, així com les proves que hem realitzat i com s'han fet.
7. **Resultats** - Es presenten i comenten els resultats obtinguts.
8. **Conclusions** - S'expliquen les conclusions a les quals hem arribat i s'enumeren les dificultats amb les que ens hem afrontat.
9. **Treball futur** - Recull un seguit de possibles tasques que es poden realitzar fruit del que s'ha fet a partir d'aquest TFG.
10. **Manual d'usuari** - S'enumeren els passos a seguir per instal·lar les biblioteques de python utilitzades en un entorn i executar els algorismes i scripts que hem implementat.

2. Metodologia i planificació

A continuació s'explica la metodologia que hem seguit per acabar desenvolupant aquest treball. També desglossem la feina a nivell de tasques en un diagrama de temps, de manera que quedi reflectida en quina part hem estat treballant cadascú de nosaltres.

2.1 Metodologia

La metodologia de desenvolupament de software juga un paper important en el desenvolupament d'un programari. Aquestes metodologies no impliquen aspectes tècnics, sinó que requereixen d'una planificació adequada, ja que proveeixen un marc de treball utilitzat per estructurar i controlar el procés de desenvolupament.

En el nostre cas, com que fèiem reunions setmanals amb el tutor, l'equip de programadors i ICRA, rebíem informació constant sobre com millorar o orientar el projecte. És per això que vam decidir seguir la metodologia RAD (Rapid application development), la qual prioritza la creació de prototips de forma ràpida, per anar interactuant amb el client i anar fent canvis constants. Així, a diferència d'altres metodologies, aquesta no requereix d'una forta planificació inicial, sinó que permet certs canvis durant el transcurs del projecte.

Per tal de mantenir el codi ordenat i compartir-lo entre nosaltres hem utilitzat la plataforma Git, que és una eina de programari lliure per tal de fer un control de versions.

Aquesta memòria l'hem redactat a través d'Overleaf, que és un editor LaTeX basat en núvol que s'utilitza per escriure, editar i publicar documents científics.

2.2 Planificació temporal

Si tenim en compte els 15 crèdits que té l'assignatura del TFG i que 1 crèdit ECTS representen 25 hores de feina, a aquest projecte li hauríem d'haver dedicat al voltant de 375 hores. Tenint en compte això, i segons el diagrama de Gantt de planificació temporal (veure figura 2.1), podem desglossar la nostra dedicació de la següent manera:

- Tasques de recerca i reunions: 240 hores
- Tasques de programació: 320 hores
- Tasques d'analista: 100 hores
- Redacció de la memòria: 120 hores

Considerant els honoraris de programador i analista a 40€/hora i els d'investigador a 30€/hora, l'import de contractació per dur a terme aquest projecte/treball ascendeix a 27.600€. Els impostos estan inclosos en el pressupost, així com totes les despeses i costos indirectes de contractació.

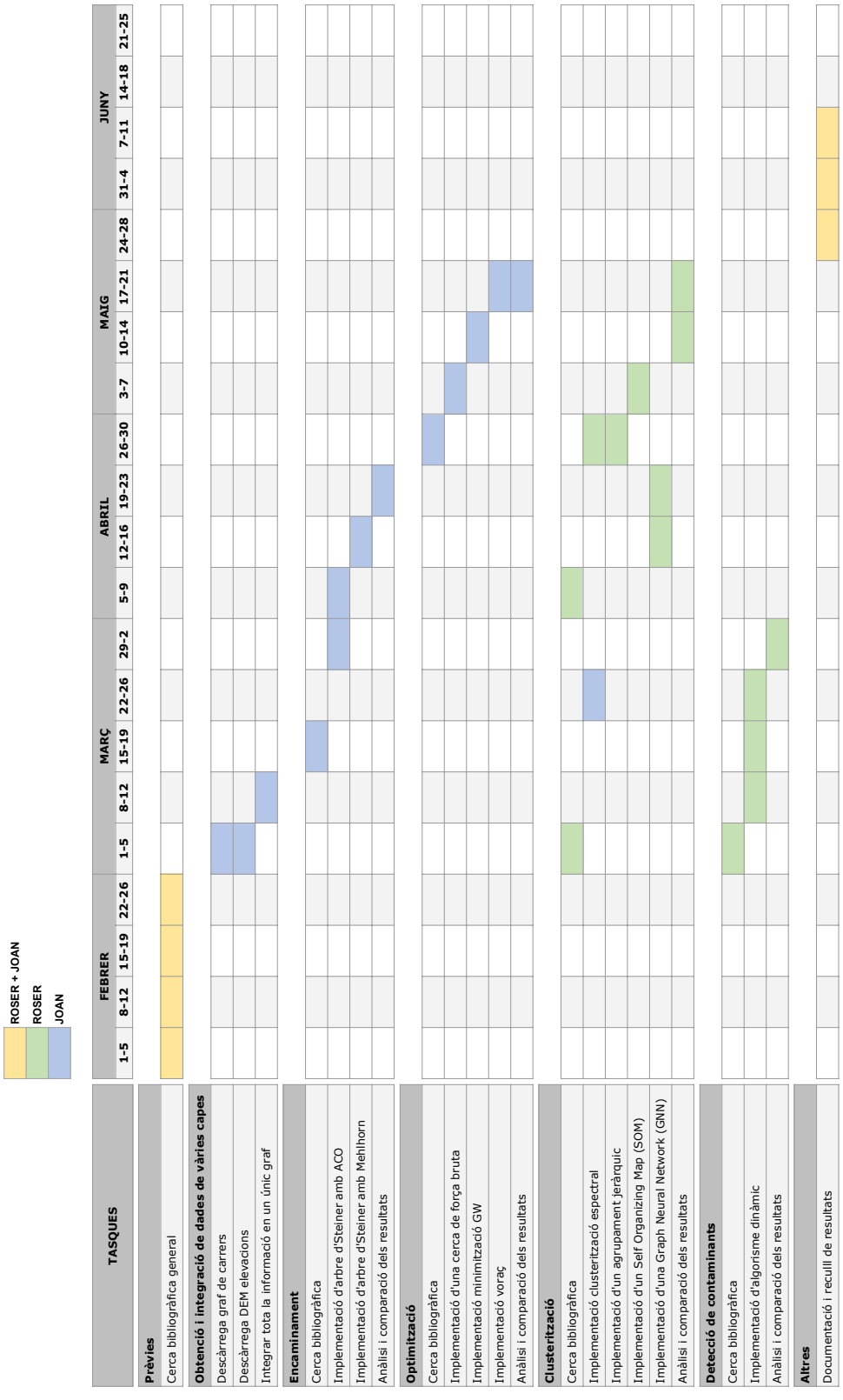


Figura 2.1: Diagrama de Gantt de la planificació temporal del TFG

3. Marc de treball i conceptes previs

Com ja hem mencionat a la introducció, el nostre treball s'engloba en el marc de dos projectes que han rebut finançament públic i que actualment s'estan desenvolupant conjuntament per investigadors de l'ICRA i pel grup de recerca BCDS de la Universitat de Girona.

A continuació s'expliquen conceptes que són importants conèixer per tal d'entendre el projecte i fer-ne un bon seguiment.

3.1 Infraestructura urbana

S'anomena infraestructura al conjunt d'elements o serveis que estan considerats necessaris per tal que una organització pugui funcionar o per a què una activitat es pugui desenvolupar i dur a terme. Si ens referim a una ciutat com a organització, llavors parlem d'infraestructura urbana.

La **infraestructura urbana** està conformada per totes les xarxes i serveis que permeten el normal funcionament de la vida ciutadana i que serveix de suport per el desenvolupament d'altres activitats.
Existeixen diferents branques dins d'una infraestructura urbana:

- Transport - Encarregada del desplegament i optimització de xarxes i vies de transport, ja sigui terrestre (qualsevol tipus de vies, com ara carreteres, autopistes, etc), marítim (ports i canals) o aeri (aeroports).
- Energia - Encarregada de l'aprovisionament de calor, combustibles (oleoductes, gasoductes) i electricitat (xarxes d'alta, mitja i baixa tensió, etc) a les llars, indústries i comerços.
- Aigua - Inclou les xarxes d'aigua potable pel consum humà, sistemes de clavegueram, xarxes d'aigua regenerada i estacions depuradores entre d'altres.
- Telecomunicació - Inclou les xarxes de telefonia (fixa i mòbil) i transmissió de dades, fibra òptica, televisió, repetidors, etc.

Pel nostre TFG ens interessa en concret la branca d'infraestructura hídrica, que comprèn les xarxes que intervenen en el recorregut realitzat per l'aigua des que es capta a la natura fins que s'aboca al medi o es tracta per reutilitzar-la després del seu ús. Això és el que es coneix com a cicle integral de l'aigua (Fig 3.1), i engloba les xarxes d'abastament (captació, potabilització i distribució), sanejament i reutilització (d'aigua regenerada).

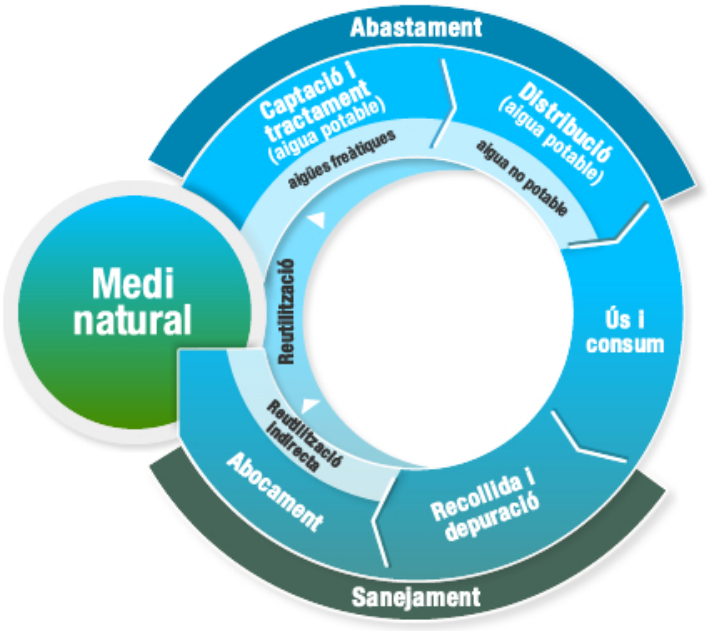


Figura 3.1: Esquema del cicle integral de l'aigua

3.1.1 Xarxa de clavegueram o sanejament

S'anomena clavegueram o xarxa de sanejament al sistema de canonades i construccions que s'utilitzen per la recollida i transport de les aigües residuals (urbanes i industrials) i pluvials d'una població o ciutat des del lloc on es generen fins on s'aboquen o es tracten per no contaminar (depuradores). Podem distingir entre dos tipus de xarxa, en funció del líquid que transporten:

- Xarxes unitàries - recullen en un mateix conducte les aigües residuals i pluvials.
- Xarxes separatives - consten de dos conductes per recollir de forma separada les aigües residuals de les pluvials.

Inicialment i per motius econòmics, a les ciutats es projectaven xarxes unitàries que abocaven directament al medi natural. Però amb l'arribada a mitjans del s.XX dels primers sistemes de depuració i tractament, això va canviar i es van començar a construir xarxes separatives, les quals

abocaven al medi natural l'aigua recollida de la pluja, i a depuradores les residuals. D'aquesta manera s'aconsegueix:

- Reducció dels costos de depuració i simplificació en els processos, ja que el cabal transportat a sistemes de depuració és menor i de flux continu. Això es deu al fet que les depuradores que tracten aigües de sistemes unitaris no treballen correctament quan hi ha excés de volum (degut a pluges) i alteren els paràmetres de l'aigua residual.
- Reducció de la càrrega contaminant final abocada al medi perquè es pot fer un millor tractament i depuració pel fet d'anar separades.

Actualment, però, cal tenir en compte que la xarxa d'aigües pluvials no és una xarxa d'aigua completament neta, ja que està altament pol·luïda per la pròpia contaminació ambiental de les ciutats, així com per productes químics utilitzats en la neteja de carrers o com a fertilitzants i plaguicides en hortes i camps. És per això que també cal donar-li un tractament adequat abans d'abocar-la al medi i reduir l'impacte ambiental. [4, 8]

3.1.2 Xarxa d'aigua potable - regenerada

L'aigua potable és l'aigua que reuneix les característiques que la fan apta per al consum humà o animal sense risc de contraure malalties. L'aigua regenerada és tota aigua que s'obté a partir d'un tractament més avançat a l'aigua ja depurada. Aquest tractament se l'anomena terciari o estació regenerada d'aigua. La utilització d'aquesta aigua pot servir per a determinats usos que no requereixin d'aigua potable:

- ambientals - recàrrega d'aqüífers
- agrícoles - reg de camps i hortes
- lúdics - reg de camps de golf i instal·lacions esportives
- municipals - neteja i baldeig de carrers
- industrials, ...

Aquesta pràctica possibilita disposar de més recursos hídrics i reduir el malbaratament d'aigua apta per al consum. Tanmateix, Catalunya disposa de normativa vigent per tal de garantir que l'ús d'aigua regenerada no comporti un risc per a la salut de les persones, al mateix temps que s'estableixen uns criteris de qualitat i gestió correctes.

Tant l'aigua potable com l'aigua regenerada es distribueixen en la quantitat necessària perquè l'usuari la rebi en qualsevol moment i dia. El transport es realitza mitjançant una extensa xarxa

de distribució i centrals de bombament. Per a l'emmagatzematge es fan servir grans dipòsits situats en punts elevats repartits per l'àrea a servir.

3.1.2.1 Cost d'una xarxa d'aigua

El disseny de la xarxa de subministrament obeeix a regles precises pel que fa al cabal i pressió de l'aigua que circula dins les canonades. És per això que aquestes han de tenir un diàmetre específic, segons el volum d'aigua que distribueixin. Com és lògic, com més gran sigui el diàmetre, més elevat serà el cost. D'aquest diàmetre també depèn la mida de les vàlvules, les quals es col·loquen en les bifurcacions de les canonades i es fan servir per regular el pas de l'aigua.

En el cas que la xarxa utilitzi dipòsits per emmagatzemar aigua, aquests han de tenir capacitat suficient per, com a mínim, assegurar el subministrament als destins als quals serveix durant dos dies.

3.2 Sistema d'informació geogràfica

Un sistema d'informació geogràfica (o GIS pel seu acrònim en anglès) és un sistema informàtic per tal d'operar amb qualsevol tipus d'informació geoespacial (informació geogràficament referenciada), de manera que es puguin manipular dades procedents del món real que estan vinculades a una referència espacial, facilitant així la incorporació d'aspectes socials, culturals, econòmics i ambientals.
L'Open Geospatial Consortium (OGC) és una organització internacional la missió de la qual és aprovar i mantenir estàndards perquè els mapes i el contingut geogràfic relacionat (la qual cosa inclou els GIS) estiguin disponibles i siguin accessibles des d'Internet.

Per tal d'obtenir l'elevació del terreny treballarem amb el servei WCS (Web Coverage Service), una de les interfícies definides per l'OGC per obtenir dades geoespacials en forma de cobertura d'informació geogràfica digital, concepte que fa referència a una representació de fenòmens de variació espacial i que típicament s'obté a partir d'imatges satèl·lit o fotos aèries. Aquesta cobertura està en format de dades ràster, que es fan servir quan es vol mostrar informació que és contínua a través d'una àrea i no es pot dividir fàcilment. A grans trets, un dataset ràster està compost de files i columnes de píxels, on cada píxel representa una regió geogràfica i el valor del píxel representa una característica d'aquesta regió. Com més gran sigui la regió representada per un píxel, de més baixa resolució serà la imatge (però òbviament, les dades ràster de més resolució requereixen de més capacitat d'emmagatzematge).

3.2.1 Sistemes de referència de coordenades

Per fer una petició en qualsevol plataforma GIS, és necessari saber en quin sistema de referències de coordenades (CRS) treballa aquesta. És a dir, un CRS defineix, mitjançant aquestes coordenades, com un mapa projectat en dues dimensions es relaciona amb ubicacions reals de l'espai. [15]

Per mostrar la superfície esfèrica de la Terra en dues dimensions s'han desenvolupat una sèrie de projeccions cartogràfiques. Aquestes, però, mai representen amb absoluta precisió la Terra, i com a resultat del procés, tots els mapes mostren distorsions (o bé no mantenen les propietats angulars, o bé hi ha distorsions de distàncies o àrees). És per això que hi ha diferents CRS, cadascun dels quals optimitza els elements de forma diferent, i quin és més adient depèn del tipus de projecte i anàlisi cartogràfic que se n'hagi de fer.

Cada CRS té un codi que els identifica de forma unívoca anomenat SRID (Spatial Reference System Identifier). Tot i que hi ha diverses organitzacions que defineixen els identificadors de forma diferent, la codificació més habitual és la definida per l'EPSG (European Petroleum Survey Group), la qual manté una base de dades per efectuar transformacions entre diferents CRS. [29]

A Espanya, l'organisme de referència per obtenir dades d'informació geogràfica és l'Instituto Geográfico Nacional (IGN), mentre que l'homòleg a nivell català és l'Institut Cartogràfic i Geològic de Catalunya (ICGC).

Citant a [20], alguns dels CRS més utilitzats són:

- **EPSG:4326** - Sistema de coordenades geogràfiques sense projectar, utilitzant el dàtum WGS84 (World Geodetic System 1984). És el sistema de referència espacial d'àmbit global més utilitzat. S'utilitza per defecte en els receptors de GPS i també en servidors de mapes d'àmbit global com GoogleEarth o Google Maps.
- **EPSG:4258** - Sistema de coordenades geogràfiques sense projectar, utilitzant el dàtum europeu ETRS89 (European Terrestrial Reference System 1989). És el sistema de referència espacial oficial per a Europa.
- **EPSG:25830** - Sistema de coordenades UTM (basat en la projecció del mateix nom) per a la zona 30 nord, utilitzant el dàtum ETRS89. És el sistema de referència espacial oficial a l'Estat espanyol des de juliol de 2007. Correspon al fus central dels tres fusos de la projecció UTM que cobreixen la península ibèrica. Pròpiament, només és vàlid per als territoris compresos dins del fus 30, però és el sistema utilitzat per a cobrir tota la península ibèrica amb coordenades UTM de forma contínua, convertint les coordenades UTM dels fusos adjacents, 29 i 31, a coordenades UTM del fus 30.

3.3 Grafs

Un graf és un conjunt d'objectes anomenats nodes (o vèrtexs) i arestes que permeten representar les interrelacions entre unitats que interactuen entre elles. Poden ser dirigits si l'aresta (a, b) (aresta que va del node a al node b) és diferent de l'aresta (b, a), o no dirigits altrament. Es diu que un graf és ponderat si les arestes poden tenir un pes associat (un valor que pot representar un cost, una longitud...). Normalment, quan parlem de grafs ens referirem a grafs simples. A diferència d'aquests, els multi-grafs són grafs que permeten multiarestes, és a dir, que es permet la presència de més d'una aresta incident als mateixos dos vèrtexs, i també arcs o bucles (arestes que connecten un vèrtexs amb si mateix).

Un concepte important en la teoria de grafs és el de components connexos, que són els subgrafs connexos màxim d'un graf no dirigit.

3.3.1 Representació de grafs en forma de matrius

Els grafs es poden representar en forma de matrius d'adjacència, que són matrius (taules rectangulars de nombres) on els índexs de les files i les columnes representen els nodes. La cel·la $[i, j]$ de la matriu valdrà el pes de l'aresta (i, j), o bé infinit si aquesta no existeix.

Una segona representació dels grafs no dirigits és la matriu de graus. En un graf no dirigit, el grau d'un node és el nombre d'arestes connectades a aquest. Així, la matriu de graus és una matriu diagonal (matriu amb el mateix nombre de files que de columnes, les entrades de la qual són nul·les a excepció de les de la diagonal principal, que poden ser nul·les o no) on l'entrada $[i, i]$ és el grau del node i.

Un altre concepte és la matriu laplaciana, que simplement fa referència a la matriu resultant de restar la matriu de graus menys la matriu d'adjacència.

Per simplicitat, d'ara endavant ens referirem als grafs simples no dirigits i ponderats com a grafs, ja que serà l'utilitzat en el transcurs del projecte.

3.4 Arbre d'Steiner

Per fer arribar l'aigua des de l'origen fins als seus destins amb la mínima longitud de canonades (i per tant, cost) possible, haurem de resoldre l'algoritme de l'arbre mínim d'Steiner. Es tracta d'un problema d'optimització combinatòria (camp de l'optimització matemàtica que s'encarrega de trobar un objecte òptim dins d'un conjunt finit d'objectes) que, de forma generalitzada, consisteix en generar la interconnexió òptima de diferents objectes donada una funció objectiu.

Tot i que hi ha diferents variants (arbre mínim d'Steiner euclidià, arbre mínim d'Steiner rectilini), a nosaltres la que ens interessa és l'arbre mínim d'Steiner en grafs (d'ara endavant, arbre d'Steiner), és a dir, a partir d'un graf $G = (V, E)$ amb arestes que tenen associat un pes positiu, i donat $S \subseteq V$ (anomenats terminals), trobar un arbre de G que tingui pes mínim, que englobi tots els nodes de S, i que pot incloure altres nodes que no estan a S però que serveixen per connectar els vèrtexs de S. [25]

El problema de l'arbre d'Steiner és una generalització de dos altres problemes coneguts en la teoria de grafs. Per una banda, si el conjunt S té només dos nodes, el problema es redueix a resoldre el problema del camí més curt (amb pesos positius). Per altra banda, si tots els nodes de G es troben a S, l'arbre d'Steiner és equivalent a l'arbre d'expansió mínim (d'ara endavant MST a partir de les seves sigles en anglès Minimum Spanning Tree), això és, donat un graf, trobar un subgraf mínim d'aquest que sigui un arbre que contingui tots els vèrtexs del graf inicial. [1]

3.4.1 Implicacions de P vs NP

Els problemes del camí més curt i el de MST es troben dins del conjunt P. En la teoria de computació, els problemes de tipus P són aquells que es poden resoldre en una màquina seqüencial determinista en un temps que és polinòmic respecte la mida de les entrades. Això és equivalent a dir que la solució d'un problema de tipus P es pot verificar en temps polinòmic, ja que donada la solució d'un problema, podem resoldre aquest mateix problema i mirar si s'obté el mateix resultat. Per tant, és fàcil veure que el conjunt P es troba dins del conjunt NP (problemes dels quals es pot verificar la seva solució en temps polinòmic). També hi ha un altre subconjunt de NP, l'anomenat NP-complet. Es diu que un problema X dins d'NP és NP-complet si tots els altres problemes dins d'NP son reduïbles en temps polinòmic a X.

En canvi, a partir d'un graf trobar l'arbre d'Steiner és un problema NP-difícil. És a dir, donat un problema X, aquest és NP-difícil si tots els problemes dins d'NP-complet són reduïbles en temps polinòmic a X.

Tot això implica que, a menys que P=NP, ni els problemes NP-complet ni NP-difícil es poden resoldre en temps polinòmic, tal com es pot veure a la figura 3.2. [18]

Per tant, ara per ara és impossible trobar la solució exacta de l'arbre d'Steiner en temps polinòmic (és a dir, de forma "ràpida"). Tot i això, existeixen aproximacions (heurístiques) que, de forma eficient, garanteixen una certa ràtio. Actualment, el millor algorisme existent proporciona un resultat que té un cost de, com a molt, 1.39 vegades el cost de l'arbre d'Steiner òptim. En aquest projecte veurem una comparativa d'algunes d'aquestes aproximacions.

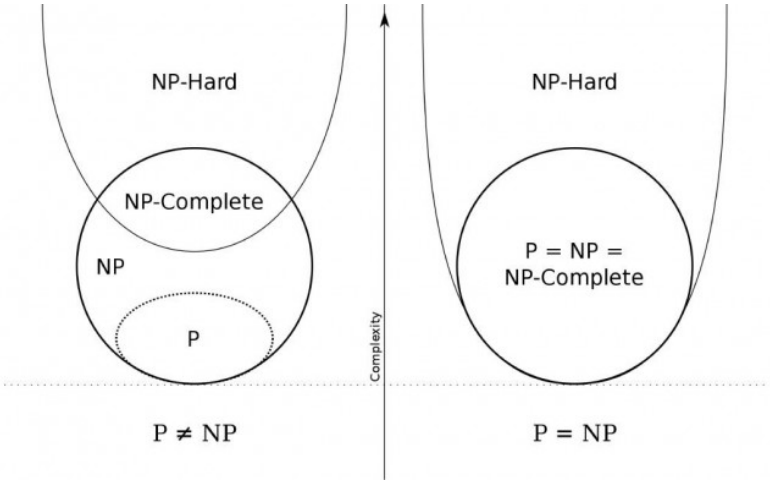


Figura 3.2: Implicacions de P=NP

3.4.2 Ant Colony Optmization

Per tal d'entendre un dels algorismes utilitzats per resoldre el problema de l'arbre d'Steiner, necessitem parlar dels algorismes de Swarm Intelligence, que són mètodes computacionals que optimitzen un problema intentant iterativament millorar una solució candidata pel que fa a una determinada mesura de qualitat. Aquests algorismes estan inspirats en el comportament social col·lectiu d'organismes com animals i plantes que solucionen diferents tipus de problemes utilitzant la intel·ligència col·lectiva.

En concret, l'algorisme que farem servir és l'Ant Colony Optmization (ACO), una heurística inspirada en el comportament de les formigues per trobar camins òptims. Suposem una formiga, que surt de la seva colònia (formiguer) per anar a buscar menjar. Quan en troba, n'agafa el màxim que pot i torna a la colònia. Mentre torna va deixant feromones al terra (una substància química que produeixen a través de les glàndules exocrines). Com més menjar porti i de més bona qualitat sigui aquest (entre d'altres factors) més feromones deixarà. Aquestes feromones, amb el temps, es van dissolent. Les següents formigues que surtin de la colònia oloraran aquestes feromones de la primera formiga, seguiran el mateix camí i arribaran fins al menjar. Les formigues també segueixen un comportament aleatori, és a dir, si hi ha diferents camins a triar, no sempre trien el camí amb més feromones. Amb el temps, totes les formigues acaben seguint el camí òptim.

Per aplicar l'ACO, es creen formigues artificials per trobar la solució òptima. Per tal de resoldre un problema amb ACO, cada formiga genera una solució (o un pas per arribar a aquesta). En el següent pas, tots els camins trobats per les formigues es comparen entre ells, i a partir d'aquí, el valor de les feromones en el camí s'actualitza. En la següent iteració, una nova generació de formigues aniran seguint les feromones deixades per les primeres formigues, i això

es va repetint durant un nombre d'iteracions determinat. [2]

3.4.2.1 Selecció de l'aresta

Per seleccionar la següent aresta en el seu camí, cada formiga ha de considerar la longitud de cada aresta que pot recórrer des de la posició actual, així com el nivell de feromones corresponent. Per la formiga a, la probabilitat P_{ij}^a de moure's des l'estat i al j depèn de la combinació de dos valors, el nivell de feromona τ_{ij} del moviment, i l'atractivitat η_{ij}:

$$P_{ij}^a = \frac{\tau_{ij}^\alpha \eta_{ij}^\beta}{\sum_{k \notin Tour_a} \tau_{ik}^\alpha \eta_{ik}^\beta} \tag{3.1}$$

On α i β són paràmetres per controlar el nivell de feromones i l'atractivitat, respectivament. El paràmetre $\eta = 1/d_{ij}$ és la visibilitat des del vèrtex i al vèrtex j, on d_{ij} és el cost del vèrtex $[i, j]$, i $Tour_a$ indica el camí fet per la formiga a.

3.4.2.2 Actualització de les feromones

Un cop s'ha completat una iteració, s'actualitzen les feromones, incrementant o decrementant la intensitat d'aquesta segons si el camí és bo o no. D'aquesta manera:

$$\tau_{ij}(t+1) = (1-e)\tau_{ij}(t) + \sum_{s=1}^{n_{ants}} \Delta\tau_{ij}^a + b \tag{3.2}$$

on e és el coeficient d'evaporació de les feromones, $\tau_{ij}(t)$ és el nivell de feromones en l'estat $[i, j]$ en l'instant de temps actual, i b és un bonus que val 0.2 si l'aresta $[i, j]$ es troba en el millor camí trobat de forma global, 0 altrament. Pel que fa $\Delta\tau_{ij}^a$ (3.3), aquesta és el canvi d'intensitat de feromona produït per la formiga a en l'estat $[i, j]$, que val:

$$\Delta\tau_{ij}^a = \begin{cases} \frac{1}{L_a} & \text{si la formiga } a \text{ passa per l'aresta } [i,j] \\ 0 & \text{altrament} \end{cases} \tag{3.3}$$

on L_a és la longitud del camí recorregut per la formiga a.

3.4.3 Generalització de l'arbre d'Steiner

Una generalització del problema de l'arbre d'Steiner és el Prize Collecting Steiner Tree (PCST), on donat un graf $G = (V, E)$, un pes no negatiu $c(e)$ per cada aresta $e \in E$, un benefici no

negatiu $p(v)$ per cada node $v \in V$, i un vèrtex arrel determinat $v_0 \in V$, podem trobar quatre problemes d'optimització que es basen en l'escenari de PCST, tal com es diu a [11].

- Problema de minimització de Goemans-Williamson (minimització GW): trobar un subarbre $T' = (V', E')$ de G tal que minimitza el pes de les arestes menys el benefici dels vèrtexs que no estan a T', és a dir, que minimitza:

$$GW(T') = \sum_{e \in E'} c(e) + \sum_{v \notin V'} p(v)$$

- Problema de maximització del valor net (Net Worth Maximization problem): trobar un subarbre T' tal que maximitzi el benefici dels nodes de T' menys el cost de les arestes, és a dir, que maximitza:

$$NW(T') = \sum_{v \in V'} p(v) - \sum_{e \in E'} c(e)$$

- Problema de la quota (quota problem): donat un $Q > 0$, trobar un subarbre T' tal que $\sum_{v \in V'} p(v) \geq Q$ i que minimitzi $\sum_{e \in E'} c(e)$
- Problema del pressupost: donat un $B > 0$, trobar un un subarbre T' tal que $\sum_{e \in E'} c(e) \leq B$ i que maximitzi $\sum_{v \in V'} p(v)$

A diferència de l'arbre d'Steiner, el PCST és útil en situacions en què s'ha de decidir si servir una sèrie de nodes o no. És a dir, en aquest últim, tot i que tens una sèrie de nodes els quals has d'abastar, no estàs obligat a fer-ho, sinó que el nombre de nodes que abastes són part del problema de decisió.

3.5 Clusterització

Clusteritzar o agrupar consisteix en fer una partició de les dades en diferents subconjunts (anomenats clústers) a partir de característiques comunes i és una de les aplicacions més populars dins del que es coneix com a aprenentatge no supervisat (aprenentatge automàtic quan les dades no estan etiquetades prèviament). Existeixen molts algorismes que resolen aquest problema, cadascun amb les seves característiques i limitacions, i entre ells difereixen significativament en la idea de què constitueix un grup i com trobar-los.

A continuació expliquem els 4 algorismes que hem seleccionat per implementar el model descentralitzat en el disseny de xarxa d'aigua regenerada. Com hem dit, hi ha molts algorismes, però els criteris que hem seguit alhora de seleccionar-ne uns o descartar-ne d'altres han estat:

- Que el número de clústers k sigui un paràmetre d'entrada, ja que l'eina a implementar ha de permetre a l'usuari que pugui seleccionar quants clústers vol.
- Que l'agrupament sigui estricte i no hi hagi solapament, és a dir, que cada observació o punt pertanyi només a un únic grup.
- Que no tingui en compte el soroll en les dades, és a dir, que no agrupi els *outliers* en un mateix grup.
- Atès que el que volem agrupar són punts d'un mapa, els quals tenim representats tants en forma de vectors d'atributs (coordenades GPS i elevació), com en un graf etiquetat, implementar tant algorismes basats en teoria de grafs com d'aprenentatge automàtic i profund, per poder comparar-los i veure com es comporten en el nostre cas.

3.5.1 Espectral

La clusterització espectral és una tècnica basada en la teoria de grafs, en què s'identifiquen clústers (o comunitats) de nodes en funció de la seva connectivitat. Fa servir els valors i vectors propis de la matriu de similitud de les dades per realitzar una reducció de dimensionalitat. Els termes de valors i vectors propis (o eigenvalues i eigenvectors) [6] fan referència a l'àlgebra lineal. Un eigenvector és un vector no nul que canvia com a molt per un factor escalar quan se li aplica una transformació lineal, tal que:

$$T(v) = \lambda v$$

De la mateixa manera, podem pensar en una matriu A com una funció per aplicar transformacions lineals. Així, si apliquem A a un vector v, tindrem un vector completament diferent. Per tant, els valors propis λ i vectors propis v d'una matriu A són aquells tals que:

$$Av = \lambda v$$

Si calculem els valors propis reals de la matriu laplaciana d'un graf, i els ordenem de petit a gran, podem començar a obtenir informació sobre aquest graf. Per exemple, tenim tants valors propis igual a 0 com nombre de components connexes té el graf. Per tant, si tenim un graf connex, aquest només tindrà un eigenvalue igual a 0. Un altre valor propi important és el primer valor diferent de 0. Aquest indica el que es coneix com a spectral gap, i ens dóna informació sobre la densitat del graf (com de fortament connectat està). El segon valor propi obtingut és el valor propi de Fiedler, i ens diu el nombre de talls que hem de fer per separar el graf en dues components connexes (i el seu vector propi associat ens dóna informació sobre

on hem de fer aquests talls). Per trobar el nombre de clústers ideal en què partir un graf ens hem de fixar en el primer gran salt entre els valors propis. Així, si de N valors propis en tenim X de petits i $N - X$ considerablement més grans, la divisió ideal del graf la podem fer en X clústers (això, òbviament, depèn del problema i del graf a tractar) [7]. En general, si volem dividir un graf amb N clústers aplicant la clusterització espectral, hem d'aplicar un mètode de clusterització a la matriu formada pels vectors propis associats als N valors propis més petits. En aquest cas, el mètode mètode utilitzat és la clusterització *k-means* (per més informació, veure [10]).

3.5.2 Agrupament jeràrquic

L'agrupament jeràrquic o basat en connectivitat, *hierarchical clustering* en anglès, agrupa la família d'algorismes que construeixen una jerarquia de grups o clústers. Els resultats de l'agrupament es presenten en el que s'anomena un dendrograma, tal com mostra l'exemple de la figura 3.3. Hi ha dues estratègies per aconseguir-ho:

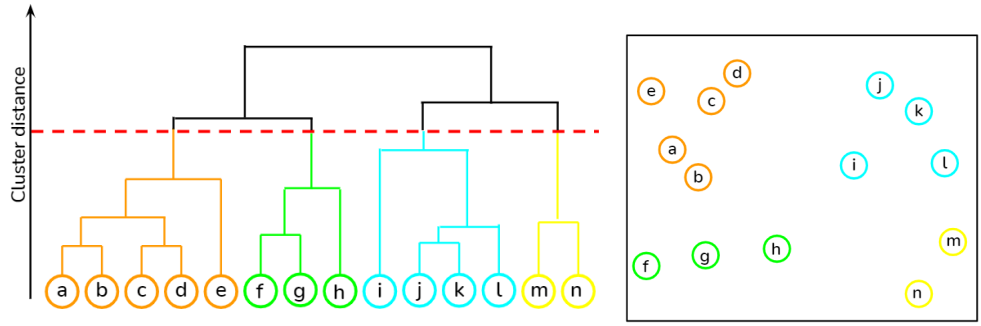
- **Aglomeratives** - Plantejament ascendent, en el que cada observació és un subclúster. Llavors entre totes les parelles de subclústers, s'uneixen els dos més propers. Aquest procés es va repetint fins que només queda un únic clúster, o s'arriba a un nombre K concret de clústers.
- **Divisives** - Plantejament descendent. Partim de que totes les observacions formen part del mateix clúster i s'aplica recursivament un K-means a cada clúster fins a obtenir tot l'arbre sencer, o bé K clústers.

En el nostre projecte hem utilitzat l'estratègia aglomerativa. A l'hora de fer l'agrupament de clústers en cada iteració, es requereix d'una mètrica que avaluï la distància entre parells d'observacions i un criteri d'enllaç que especifiqui la dissimilitud entre conjunts. [21]

- **Mètrica** - Les mètriques més utilitzades per calcular l'enllaç entre conjunts en l'algorisme jeràrquic són la distància euclidiana, la distància de Manhattan, la similitud del cosinus, etc. Nosaltres hem utilitzat la distància euclídia:

$$\|a - b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$$

- **Criteri d'enllaç** - Determina la distància entre conjunts d'observacions com una funció de les distàncies entre observacions dos a dos. El criteri que hem utilitzat és el que s'anomena criteri de Ward. Aquest mètode minimitza la suma de les diferències quadrades (variància) entre els clústers.



(a) Dendrograma tallat (b) Espai vectorial

Figura 3.3: Exemple d'algorisme aglomeratiu amb $k = 4$ clústers

3.5.3 Self Organizing Map - SOM

El *self-organizing map*, mapa autoorganitzat en català, és un tipus de xarxa neuronal artificial que s'entrena mitjançant un aprenentatge no supervisat per produir una representació discreta de l'espai de les dades d'entrada, conegut com a mapa. D'ara en endavant, ens referirem a aquest algorisme com a SOM.

Els SOMs difereixen d'altres xarxes neuronals essencialment en dos punts:

- Implementen el que es coneix com a aprenentatge competitiu, enlloc d'aplicar un aprenentatge automàtic que minimitza una funció de pèrdua mitjançant la retropropagació.
- Utilitzen una funció de veïnatge Θ per tal de preservar les propietats topològiques de l'espai d'entrada.

Com hem dit, en els SOMs s'entrena la xarxa amb **aprenentatge competitiu**. En aquest tipus d'aprenentatge, les neurones pugnen entre elles per tal de representar una classe o un patró d'entrada. Llavors, l'aprenentatge consisteix en reforçar les connexions de la unitat guanyadora, alhora que debilitar les altres perquè els pesos de la guanyadora s'assemblin cada vegada més al patró d'entrada. A grans trets, l'algorisme funciona de la següent manera:

1. Inicialitzar els pesos de les neurones de forma aleatòria amb valors petits.
2. Per cada vector d'entrada del dataset de dades:
   2.1. Per cada neurona del mapa:
      2.1.1. Buscar la similitud entre el vector d'entrada i el vector de pesos de la neurona a partir de la distància euclídia.
      2.1.2. Mantenir identificada la neurona amb la menor distància, la qual serà la BMU (*Best Matching Unit* en anglès).

2.2. Actualitzar els pesos de les neurones properes a la BMU i la pròpia BMU per aproparles al vector d'entrada.

3. Tornar al pas 2 mentre no s'hagin arribat al nombre d'iteracions predefinides.

La fórmula per actualitzar una neurona amb vector de pes $W_v(s)$ és:

$$W_v(s+1) = W_s(s) + \Theta(u, v, s) \cdot \alpha(s) \cdot (D(t) - W_v(s))$$

on s és la iteració actual, t és l'índex dins del conjunt entrant de dades, $D(t)$ és el vector d'entrada, u és l'index del BMU per $D(t)$, $\alpha(s)$ és el coeficient d'aprenentatge, $\Theta(u, v, s)$ és la funció de veïnatge entre la neurona u i la neurona v en la iteració s. Aquest procés s'anirà repetint per cada vector d'entrada fins a λ iteracions. [24]

3.5.4 Graph Neural Networks - GNNs

Els algorismes d'aprenentatge automàtic i profund processen dades representades en vectors o matrius de nombres. Quan aquests s'apliquen en grafs, aquests vectors no contenen cap tipus d'informació relacionada amb l'estructura del graf, que es perd. D'aquí la importància de les Graph Neural Networks (GNNs d'ara en endavant), que engloben mètodes i tècniques d'aprenentatge profund que pretenen resoldre problemes i fer inferència en grafs o dades que es poden representar mitjançant grafs.

Una GNN implementa l'intercanvi d'informació entre nodes del graf amb xarxes neuronals. L'intercanvi d'informació es fa de manera que cada node té un estat x que el representa com a concepte, el qual s'actualitza a mesura que es va passant "missatges" amb els nodes veïns fins a assolir l'estat d'equilibri o estat final. El procés inclou en primer lloc una funció de transició que agafa com a *input* el vector de característiques de cada node i de cada aresta, l'estat dels nodes veïns i el seu vector de característiques; i obté el nou estat del node com a sortida. En la figura 3.4 es pot veure un exemple d'una GNN de dues capes. [17, 9]

Les GNNs es consideren estat de l'art en algunes tasques d'aprenentatge supervisat o semisupervisat emprades en l'anàlisi de grafs com ara classificació de nodes i predicció d'arestes. D'altra banda, però, problemes d'aprenentatge no supervisat com el de la clusterització de grafs, no han assolit encara grans resultats utilitzant GNNs. Tsitsulin et al. [27] proposen un mètode nou per clusteritzar grafs amb atributs mitjançant GNNs, el qual anomenen DMoN (Deep Modularity Networks).

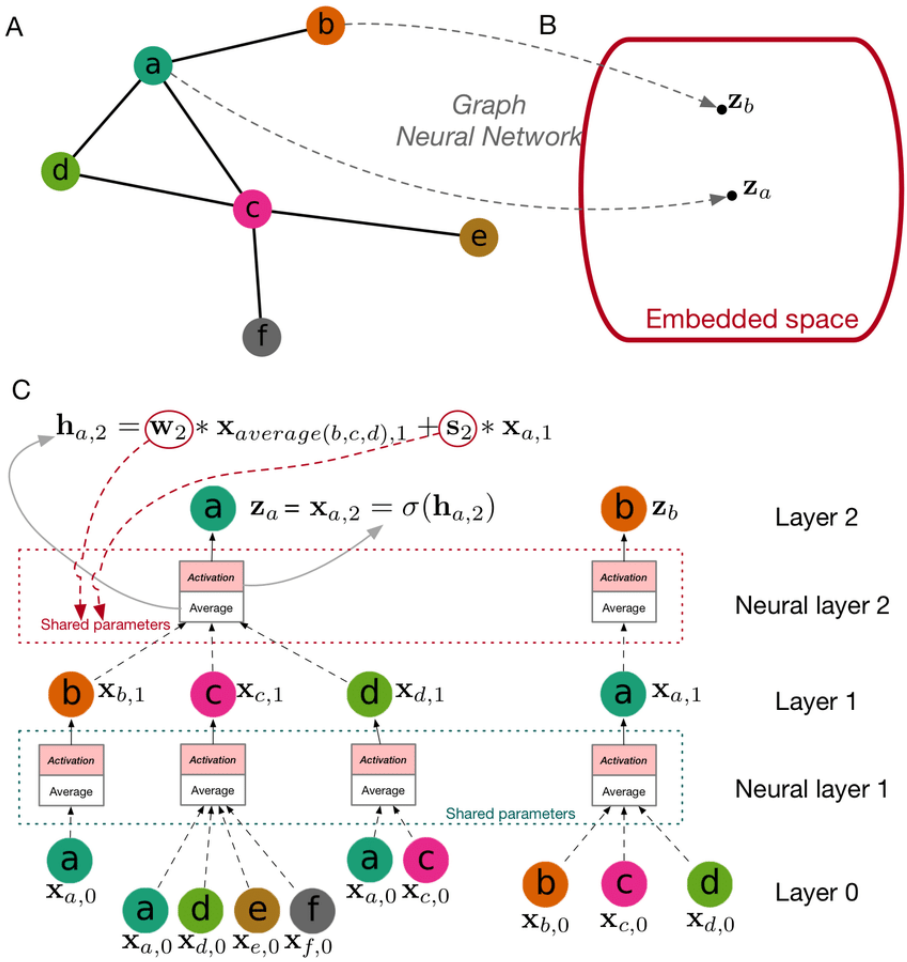


Figura 3.4: Representació d'una GNN de 2 capes. **(A)** Representació del graf. **(B)** Vectors de sortida, un per cada node del graf. **(C)** GNN pel graf de (A), amb els nodes **a** i **b** com a exemple. En cada capa, els nodes agreguen informació dels seus veïns per actualitzar la seva representació a partir de fer la mitjana dels seus veïns i aplicar la funció d'activació.

3.5.4.1 DMoN

DMoN està inspirat en la funció de modularitat i la seva optimització espectral.

La modularitat Q és una mètrica de l'estructura de xarxes o grafs. S'utilitza per mesurar la força de la divisió de la pròpia xarxa en clústers o comunitats. Grafs amb un valor de modularitat alt estan altament connectats entre els nodes que formen part del mateix clúster, mentre que les connexions entre nodes de diferents clústers són escasses. Pel nostre problema, ens ha semblat interessant, ja que si les connexions intraclúster són denses, pot afavorir en trobar un disseny més òptim de xarxa d'aigua regenerada.
Com que maximitzar la modularitat és un problema NP-difícil, el mètode DMoN aplica una relaxació espectral per tal de solucionar-ho de forma eficient. En problemes d'optimització, aplicar una relaxació consisteix en fer una aproximació d'un problema difícil a un de proper més fàcil de solucionar.
Malgrat s'han identificat problemes per detectar petits clústers en mètodes que optimitzen la modularitat, segueix essent una de les mètriques més utilitzades a l'hora d'avaluar la clusterització de grafs.

Els resultats del mètode DMoN, recollits a [27], mostren una millora en la clusterització de grafs si es té en compte la modularitat com a mètrica d'avaluació, en comparació a altres algorismes existents, alguns considerats també estat de l'art.

Aquest mètode de GNN, en concret, és el que hem implementat en el nostre treball, ja que els autors proporcionen un enllaç a un repositori online[1] amb la seva implementació que ens ha servit com a punt de partida. D'aquesta manera, ens hem pogut abstreure de la part més matemàtica que hi ha darrera les xarxes neuronals en general, i d'aquesta en particular.

3.6 Probabilitat bayesiana

La probabilitat bayesiana és una de les diferents branques del concepte de probabilitat, que l'interpreta com a grau de convicció personal, de manera que es diferencia de la percepció de probabilitat objectivista. El terme bayesià es refereix al matemàtic del s.XVIII i teòleg Thomas Bayes.

La probabilitat bayesiana pertany a la categoria de probabilitats probatòries: un paràmetre és vist com una variable aleatòria a la que, abans de l'evidència mostral, se li assigna una probabilitat a priori en base a un grau de creença amb respecte el comportament aleatori. Quan s'obté l'evidència mostral, la probabilitat a priori es modifica per una probabilitat a posteriori.

En els algorismes dinàmics per determinar els punts de mostreig assignem probabilitats

[1] github.com/google-research/google-research/tree/master/graph_embedding/dmon

bayesianes als nodes origen en base al nombre d'habitants connectats a cada node, ja que la nostra hipòtesi és que els nodes més contaminats són els que tenen més habitants connectats. [3]

4. Anàlisi i Requisits de sistema

Uns dels objectius d'aquest projecte és combinar múltiples fonts d'informació de forma automatitzada. Algunes d'aquestes dades s'obtenen del núvol, però d'altres s'han d'introduir manualment. Apart de totes aquestes dades, també es requereix el llenguatge de programació "Python 3", i que totes les llibreries descrites posteriorment estiguin instal·lades al sistema.

4.1 CLEaN-TOUR

4.1.1 Requisits

4.1.1.1 Graf de carrers

Els grafs de carrers de la regió de la qual es vol realitzar l'estudi s'obtenen des de l'API de OpenStreetMap, on les arestes representen els carrers i els nodes representen les interseccions entre ells. Aquesta regió ha de venir definida per un identificador, el qual ha de ser introduït per l'usuari, i tant pot ser el nom d'una ciutat, un codi postal, o qualsevol de les maneres descrites per *Nominatim*, el motor de cerca fet servir per OpenStreetMap.

4.1.1.2 Consum d'aigua i costos de construcció

Els consums d'aigua i els costos de construcció venen definits en les taules 1.1 i 1.2 respectivament, que ja hem vist anteriorment. Tot i que aquests són un valor per defecte, estan suficientment modularitzats perquè l'usuari les pugui modificar.

4.1.1.3 Informació d'ús de les parcel·les

A partir dels usos de cada parcel·la i dels consums mitjans d'aigua segons ús, podem estimar un consum total d'aigua diari. Aquests usos els podem obtenir a partir de la Seu Electrònica del Cadastre, des d'on hem obtingut els fitxers que inclouen la informació respectiva dels usos de terreny i immobles. Aquesta informació es divideix en diferents fitxers: els fitxers ".CAT", que

contenen informació dels immobles i terrenys tant rurals com urbans, i els fitxers ".DBF" que contenen informació dels jardins.

S'ha de tenir en compte que per cada municipi que tracta la depuradora, necessitem els seus respectius fitxers.

4.1.1.4 Nombre d'habitants

Com ja s'ha dit, la ciutat de Girona tenia 103.369 habitants l'any 2020. Utilitzant aquest valor, i sabent que el nombre d'habitatges habituals és de 38.245 s'ha deduït que, de mitjana, a cada llar hi viuen 2,7 persones (aplicant un lleuger factor de correcció, assumim que de mitjana n'hi viuen 2,4). Aquesta informació, que es complementa amb la informació de l'ús de les parcel·les. [22].

4.1.1.5 Elevació del terreny

Per tal de minimitzar els costos de distribució d'aigua, s'intenta que es desplaci per gravetat, ja que en cas contrari, s'hauria de bombejar. Per tant, per saber el pendent de la xarxa, necessitem saber l'elevació dels nodes, la qual s'obté de forma automàtica a partir de les dades de l'Instituto Geográfico Nacional (IGN).

4.1.1.6 Parcs i jardins públics

Com que les dades del Cadastre no inclouen informació sobre els jardins públics, sinó que només de les parcel·les privades, hem d'utilitzar les API's de Overpass i Shapely per tal d'obtenir les seves coordenades i superfícies.

4.1.2 Disseny de sistema

L'estructura del projecte CLEaN-TOUR s'ha implementat segons es pot veure a la figura 4.1.

Un cop l'usuari ha introduït el nom de la ciutat i els fitxers .CAT i .DBF, automàticament s'obté el graf de carrers a partir de OpenStreetMap. A continuació, s'executa l'*script* ja mencionat per obtenir un sol fitxer .JSON on, per cada parcel·la, s'especifica la localització, els usos, consums, m^2 de superfície i nombre d'habitants.. Aquesta informació s'incorpora al graf de carrers, i després s'hi afegeix el model d'elevacions obtingut a partir de l'API d'IGN.

A partir d'aquí, i donat un node origen (depuradora o dipòsit) i una llista de nodes destins (parcel·les on distribuir l'aigua), es pot calcular la xarxa d'aigua regenerada i generar les estadístiques desitjades.

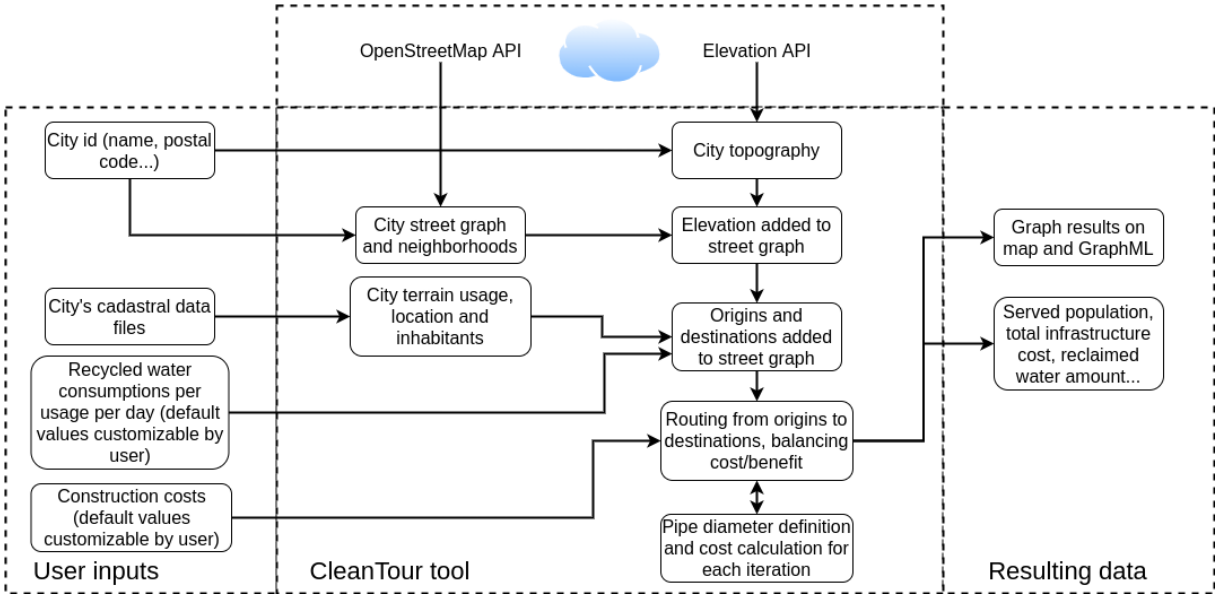


Figura 4.1: Diagrama de flux d'implementació del projecte CLEaN-TOUR

4.2 VIRWASTE

Les dades requerides per la implementació del programa es detallen a continuació.

4.2.1 Requisits

4.2.1.1 Obtenció de dades

Per poder generar un graf en format GraphML que contingui la informació necessària per implementar els algorismes, es van recollir les següents dades:

- Topologia de la xarxa de clavegueram amb atributs, facilitada per Aigües de Girona des de bases de dades GIS.
- Model digital d'elevacions (DME per les seves sigles en anglès), que és l'equivalent informatitzat de la cartografia clàssica d'elevacions.
- Informació cadastral de les parcel·les en format geojson.
- Indicadors demogràfics i socioeconòmics associats a cada parcel·la.

4.2.1.2 Construcció del graf i assignar indicadors

En primer lloc es van combinar els diferents fitxers GIS per obtenir un primer fitxer en format GraphML, en el qual les arestes representaven les canonades i els nodes els pous d'accés, inclosos els seus atributs. A continuació, es va fer una verificació i reconciliació de les dades: es van eliminar nodes i arestes desconnectades, i es van discutir possibles errors amb la companyia per assegurar precisió en el model. A continuació, a partir de les dades del cadastre i recollida en un fitxer geojson amb les geometries i dades alfanumèriques de les parcel·les, es va associar cada parcel·la al pou més proper seguint pendent negatiu assumint que l'aigua residual es transporta per gravetat. Per estimar la coordenada z (elevació) dels pous i de les parcel·les, es va utilitzar el DEM de l'Institut Cartogràfic i Geològic de Catalunya (ICGC) a una resolució de 2x2. Finalment, es va fer una estimació del nombre d'habitants de cada parcel·la, tal i com s'ha explicat en l'apartat 4.1.1.4, de manera que cada node del graf té un atribut "habitants" que agrega tots els que estan connectats a aquell node (o pou).

4.2.2 Disseny de sistema

A la figura 4.2 es pot veure l'esquema del sistema de l'eina. Després d'obtenir les dades i un cop es disposa del graf de la xarxa de clavegueram amb tota la informació i els indicadors en format d'atribut, s'executen els algorismes, per fases. En primer lloc el de posicionament de sensors estàtics, el qual cal indicar el nombre de punts de monitoratge que es volen tenir com a paràmetre d'entrada. Aquest algorisme posiciona els sensors i retorna també el el graf que cobreix cadascun dels sensors. Llavors, quan algun dels sensors detecti una càrrega viral, el segon algorisme es posa en marxa per tal de determinar el focus d'infecció, i es fa córrer únicament en aquella part de la xarxa coberta pel sensor.

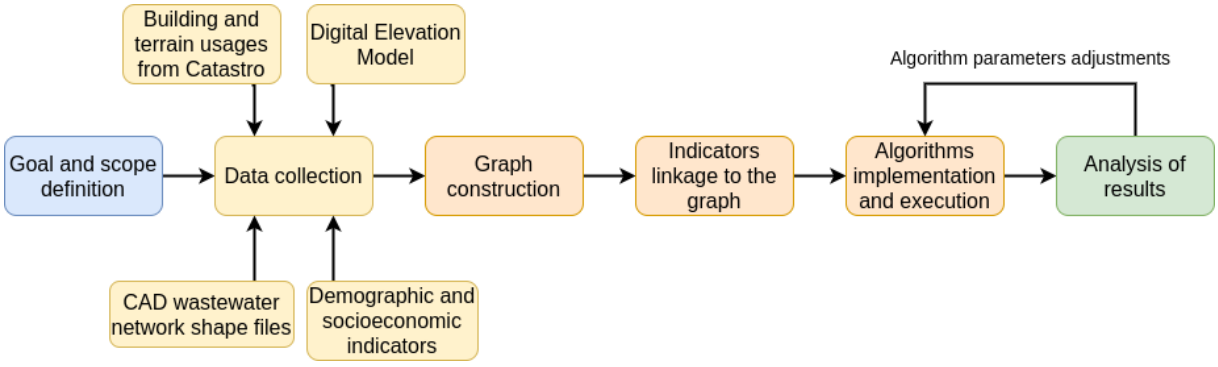


Figura 4.2: Diagrama de flux d'implementació del projecte VIRWASTE

5. Estudis i decisions

5.1 Python

La part ja implementada dels projectes de CLEaN-TOUR i VIRWASTE estava programada amb *Python*, per tant nosaltres també hem continuat amb aquest mateix llenguatge. El fet que sigui un dels més utilitzats fa que tingui un gran conjunt de biblioteques de codi lliure disponibles.

5.1.1 Jupyter Notebook

L'entorn de programació escollit ha sigut *Jupyter Notebook*, una eina web interactiva coneguda com a *interfície notebook* (o quadern computacional) ja que és una eina que permet executar codi i visualitzar dades i estadístiques de forma molt fàcil i intuïtiva.

5.1.2 Conda

Conda és un sistema de gestor de paquets de codi obert, desenvolupat originalment per resoldre els reptes difícils de gestió de paquets.

5.1.3 Biblioteques

De forma resumida, les principals llibreries que hem fet servir s'expliquen a continuació.

5.1.3.1 NetworkX

És una llibreria per crear, manipular i estudiar grafs i analitzar xarxes complexes. És adequada per operar amb grans xarxes del món real, ja que la seva estructura de dades està basada en l'ús de diccionaris, i es tracta d'un *framework* bastant eficient i escalable.

5.1.3.2 OSMnx

Es tracta d'un paquet per obtenir informació geoespacial des d'OpenStreetMap i modelar, projectar i visualitzar xarxes reals de carrers i altres geometries espacials.

5.1.3.3 Scikit-learn

Scikit-learn, més coneguda per sklearn, és una biblioteca que agrega suport en l'àmbit de l'aprenentatge automàtic. Disposa d'algorismes d'estadística, regressió i clustering i està dissenada per integrar-se conjuntament amb les biblioteques numèriques NumPy i SciPy.

5.1.3.4 Pandas

Pandas és una biblioteca escrita com una extensió de NumPy per a la manipulació i anàlisi de dades. En particular, ofereix estructures de dades i operacions per manipular taules numèriques i sèries temporals.

5.1.3.5 TensorFlow

TensorFlow és una biblioteca desenvolupada per Google, extensament utilitzada dins l'àmbit de l'aprenentatge profund i automàtic. Permet construir i entrenar xarxes neuronals per detectar i desxifrar patrons i correlacions, anàlegs a l'aprenentatge i raonament utilitzats pels humans.

5.1.3.6 Pyproj

Pyproj és una interfície per Python de PROJ, una llibreria per fer projeccions cartogràfiques i transformacions de coordenades desenvolupada pel Servei Geològic dels Estats Units.

5.1.3.7 GDAL

Llibreria per manipular dades geoespacials, tant ràster com vectorials.

5.1.3.8 Matplotlib

Es tracta d'una biblioteca per generar gràfiques i altres estadístiques.

5.1.3.9 pcst_fast

Framework extret d'un repositori online[1] per resoldre la variant de minimització de Goemans-Williamson del PCST (d'ara endavant, algorisme GW). Aquest algorisme es basa en una aproximació primal-dual, un concepte del camp de l'optimització, que diu que en un problema de programació lineal n'existeix un altre, anomenat problema dual, i les relacions entre ells poden ser útils per resoldre el problema original. En aquest treball, però, ens hem abstret de la seva implementació.

5.1.3.10 Minisom

Minisom és una biblioteca que implementa els Self Organizing Maps.

5.1.3.11 OWSLib

Biblioteca per obtenir dades a través dels serveis web de l'Open Geospatial Consortium (OGC), en el nostre cas, a través de la interfície Web Coverage Service (WCS), que permet realitzar peticions de cobertura geogràfica (dades ràster), per obtenir les elevacions.

5.1.3.12 Rasterio

Rasterio és una biblioteca que permet llegir i escriure els formats de dades GIS (GeoTIFF, etc).

[1] https://github.com/fraenkel-lab/pcst_fast

6. Implementació i proves

En aquest capítol s'hi descriuen els algorismes més rellevants que s'han desenvolupat en el projecte.

6.1 CleanTour

Per realitzar les proves dels algorismes s'ha treballat amb llindars de consum d'aigua (metres cúbics servits). Així, el llindar 0 està format pels nodes que consumeixen més de 0 m^3 diaris d'aigua. Pel llindar 10, els que en consumeixen més de 10, i així successivament. Aquesta informació es pot veure per Girona i Lloret a la figura 6.1. Pel que fa al graf sencer la primera ciutat està formada per 2646 nodes, mentre que Lloret per 1331.

	Girona	**Lloret**
Llindar 0	1961	1070
Llindar 10	477	385
Llindar 20	144	178
Llindar 25	94	134
Llindar 50	23	46
Llindar 75	13	25
Llindar 100	10	15
Llindar 125	7	11
Llindar 150	6	9
Llindar 175	5	6
Llindar 200	5	6

Taula 6.1: Nodes per llindar a Girona i Lloret de Mar

Per l'opció descentralitzada, s'ha treballat només amb els llindars que van des de 0 fins a $50m^3$ d'aigua, ja que per llindars superiors el nombre de destins és força petit i considerem que ja no té sentit plantejar una xarxa descentralitzada.

D'altra part, també hem definit el número de clústers k amb els quals provarem els algorismes pel disseny descentralitzat per cadascuna de les ciutats:

- Girona: $k \in K = \{3, 4, 5, 6, 7\}$
- Lloret de Mar: $k \in K = \{2, 3, 4, 5\}$

6.1.1 Obtenció del model d'elevacions

Hem modularitzat l'algorisme d'obtenció de les elevacions d'un graf de carrers, de manera que es pugui fer a partir de diferents serveis (IGN[1] o ICGC[2]), i amb diferents precisions (les acceptades per cada servei). L'algorisme es pot veure a 1.

Algorithm 1: Obtenció del model d'elevació

Input: Identificador de la zona a obtenir el model
Servei utilitzat per obtenir les elevacions (ICGC o IGN)
Precisió de l'obtenció de les dades (en cas de servei ICGC, la precisió pot ser 5 o 15, en cas d'IGN 5 o 25)

Output: Graf de carrers de la zona demanada, on cada node té com a atribut la seva elevació

1. Obtenir el graf de carrers de la zona sobre la qual volem obtenir la xarxa.
2. De cada node en podem obtenir les seves coordenades, en format EPSG:4326.
3. Obtenir una bounding-box del graf obtingut, és a dir, una àrea definida per dues longituds i dues latituds que engloba tots els nodes del graf.
4. Canviar de coordenades EPSG:4326 al format demanat pel servei que utilitzem (ICGC o IGN).
5. Ajustar la bounding-box segons la precisió ($m \times n$) del servei per tenir una matriu de píxels amb número de files i columnes múltiples de la precisió.
6. Fer la petició WCS al servei utilitzat amb les coordenades de sortida, la bounding-box i la precisió.
7. Rebre la petició i llegir-la.
8. Buscar les coordenades i l'elevació del centroide de cada píxel del ràster obtingut, i guardar-les en un KDTree.
9. Per cada node del graf de carrers, trobar el node més proper del KDTree i assignar-li la mateixa elevació, en forma d'atribut.

En aquest projecte hem treballat amb el servei d'IGN perquè ens permet obtenir elevacions de tot Espanya.

[1] https://datos.gob.es/ca/catalogo/e00125901-spaignmdt
[2] https://www.icgc.cat/Administracio-i-empresa/Serveis/Geoinformacio-en-linia-Geoserveis/WMS-i-WCS-Elevacions/WCS-del-Model-d-Elevacions-del-Terreny

6.1.2 Encaminament

S'han utilitzat dos algorismes diferents per dissenyar la xarxa més curta possible (que com ja hem dit, consisteix en resoldre el problema de l'arbre d'Steiner) en la versió centralitzada. En aquests dos algorismes, l'entrada està formada (en part) per $G = (V, E, d)$ i $S \subseteq G$, on $G = (V, E)$ és el graf de carrers tal que la longitud de la canonada $E_i = d_i$, i S és una llista dels nodes destins que s'han de servir, entre els quals està inclòs l'origen (depuradora).

6.1.2.1 Ant Colony Optimization

Com que trobar l'arbre d'Steiner en un graf és una tasca computacionalment difícil, s'ha utilitzat l'aproximació de [23], que enlloc de fer servir l'ACO tal com s'ha explicat al capítol 3.4.2, proposa primer fragmentar el graf original en subgrafs fent servir alguna tècnica de clusterització (en concret la clusterització espectral), i en cada subgraf trobar l'arbre d'Steiner local. Un cop calculats aquests, els arbres obtinguts són combinats per obtenir una solució final. De forma resumida, els passos generals de l'algorisme els podem veure a l'algorisme 2.

Algorithm 2: Passos algorísmics ACO

Input: $G = (V, E)$: graf amb conjunt de nodes V, conjunt d'arestes E
S: llista dels nodes terminals
k: nombre de clusters amb què es vol dividir el graf

Output: $G' = (V', E')$ tal que G' és l'arbre d'Steiner de G amb S com a nodes terminals

1. Clusteritzar G aplicant clusterització espectral per obtenir k clusters
2. Per cada cluster, trobar l'arbre d'Steiner corresponent
3. Retornar el graf resultant de combinar cadascun dels arbres Steiner calculats anteriorment

El pas 2 de l'algorisme 2 es realitza a partir de moltes colònies de formigues independents. Es posiciona una colònia c per cluster, i cadascuna d'aquestes executa el seu propi ACO. Aquestes colònies, a la vegada, es divideixen en subcolònies, les quals es comuniquen després de cada iteració. De forma més precisa, cada subcolònia tria un vèrtex aleatori dins del seu subgraf com a formiguer, i a continuació s'aplica l'algorisme per trobar l'arbre d'Steiner mínim que connecta tots els terminals S_c dins c. Pel que fa al pas 3, simplement fa referència a una agregació dels arbres Steiner locals per obtenir un arbre global.

A partir de les equacions 3.1 i 3.2, s'aplica l'algorisme 3. Cada formiga a d'una colònia c comença el seu camí des d'un node terminal aleatori $s \in S$, de manera que $Tour_{ant} = \{random(S)\}$. Cada colònia c es va engrandint a mesura que s'afegeixen els vèrtexs de $Tour^c_{best}$ (millor camí trobat dins de la colònia c) al conjunt V_c (caus de la colònia c). Si dues colònies c_1

i c_2 comparteixen un vèrtex $v_1 \in V_{c_1}$ i $v1 \in V_{c_2}$, aquestes dues colònies es combinen en una de sola. Tot aquest procediment es va repetint fins que només queda una única colònia.

Algorithm 3: ACO independent

Input: $G = (V, E)$: graf amb conjunt de nodes V, conjunt d'arestes E
S: llista dels nodes terminals.
n_{ants}: Nombre de formigues per colònia
min_{ants}: Nombre mínim de formigues per cau
p_{ants}: Percentatge de formigues en el millor camí per fer més gran la colònia
n_{cycles}: Nombre de cicles abans d'assignar el millor camí
α: Paràmetre per controlar influència de les feromones
β: Paràmetre per controlar atractivitat del camí
e: Coeficient d'evaporació

Output: $G' = (V', E')$ tal que G' és l'arbre d'Steiner de G amb S com a nodes terminals

1. Per cada colònia $c \in C$ fer:
2. Triar un node terminal aleatori com a cau V_c
3. Inicialitzar colònia (algorisme 4)
4. Repetir:
5. Per cada colònia $c \in C$:
6. Per $m = 1$ fins a n_{cycles}:
7. Construir el tour $Tour^c_m$ aplicant l'algorisme 5
8. Trobar el millor tour de la colònia $Tour^c_{best}$, que és $min(cost(Tour^c_m))$
9. Si el percentatge de formigues que recorren el millor tour $\geq p_{ants}$:
10. Break (hem trobat una solució bona)
11. Afegir els vèrtexs de $Tour^c_{best}$ a la colònia c $(V_c \cup Tour^c_{best})$
12. Si dues subcolònies c_i i c_j comparteixen dos vèrtexs, és a dir, $V_{c_i} \cap V_{c_j} \neq \emptyset$ aleshores:
13. Combinar c_i i c_j per obtenir una nova subcolònia, és a dir, $V_{ck} = V_{c_i} \cup V_{c_j}$, i esborrar c_i i c_j
14. Per cada colònia $c \in C$:
15. Inicialitzar colònia (algorisme 4)
16. Fins que una colònia c tingui tots els terminals: $S \subseteq V_c$
17. Retornar el graf format per tots els nodes de V_c

6.1.2.2 Algorisme de Mehlhorn

A diferència de l'algorisme descrit al punt anterior, l'algorisme de Mehlhorn (algorisme 6), implementat segons [16], utilitza tècniques de teoria de grafs. Redueix el problema a trobar el

Algorithm 4: Inicialitzar colònia

Input: V_c: Caus de la colònia c
n_{ant}: Nombre de formigues de la colònia
min_{ant}: Nombre mínim de formigues

Input: Diu on estan les formigues de la colònia c

1. Inicialitzar cada cel·la de la matriu de feromones a 10^{-4}
2. Per cada vèrtex $v \in V_c$:
3. Posiciona $max(\frac{|V_c|}{n_{Ants}}, min_{ants})$ a v
4. Inicialitza els tours de les formigues posicionades de manera que $Tour_a = \{v\}$

Algorithm 5: Generar camins

Input: A_c: Formigues de la colònia c
V_c: Caus de la colònia c

Output: Per cada formiga $a \in A_c$, retorna $Tour_a$

1. Per cada formiga $a \in A_c$:
2. Repetir:
3. Moure's a un vèrtex v tal que $v \notin V_c$ i $v \notin Tour_a$ aplicant (3.1)
4. Afegir v al tour: $Tour_a \cup v$
5. Fins que $v \in S$
6. Actualitzar matriu de feromones seguint (3.2)

camí més curt entre dos nodes, i a aplicar el MST.

6.1.3 Optimització

En l'apartat 3.4.3, hem parlat de diferents variants del problema de l'arbre d'Steiner. Si recordem en concret el problema del pressupost, aquest retorna un arbre que maximitza el benefici dels nodes, però que té cost total (suma del pes de cadascuna de les arestes) inferior a un pressupost donat.

En el context d'aquest projecte, no ens interessa això, sinó que el que volem és que el cost total sigui el cost de construir tota la xarxa de canonades, i que maximitzi el consum d'aigua distribuït. Això implica, a efectes pràctics, que el fet de tenir una xarxa o una altra pot fer variar el pes d'una mateixa aresta. En resum, el benefici de cada node és el consum, el cost de cada aresta és la distància de la canonada, però el cost total del graf és el cost de construir la xarxa.

Algorithm 6: Algorisme de Mehlhorn

Input: $G = (V, E)$: graf amb conjunt de nodes V, conjunt d'arestes E
S: llista dels nodes terminals

Output: $G' = (V', E')$ tal que G' és l'arbre d'Steiner de G amb el S com a nodes terminals

1. Construir el graf $G_1 = (S, E_1)$ de la següent manera: per per cada $v \in V$, $s(v)$ és el node $s \in S$ més proper a v. Per cada $(u, v) \in E$, generar totes les tripletes$((s(u), s(v), d(s(u), u) + d(u, v) + d(s(v), v))$. Per últim, selecciona, per cada aresta de G_1 (dos primers elements de la tripleta), el seu cost mínim (últim element de la tripleta).
2. Trobar el MST G_2 de G_1
3. Construir un subgraf G_3 de G substituint cada aresta de G_2 pel seu camí mínim corresponent de G (si hi ha més d'un camí mínim, triar-ne algun a l'atzar)
4. Trobar el MST G_4 de G_3
5. Sigui G_5 el graf resultant de treure tots els nodes i arestes corresponents de G_4 tal que si un node v és un node fulla de G_5, $v \notin S$
6. Retornar G_5

6.1.3.1 Problema del pressupost a partir d'una cerca de força bruta

La primera idea per tal de resoldre el problema consisteix en una cerca exhaustiva per força bruta, la qual no és massa eficient, però que en canvi dóna la millor solució. El codi es pot veure a 7.

Algorithm 7: Problema del pressupost a partir de la força bruta

Input: $G = (V, E)$: graf amb conjunt de nodes V, conjunt d'arestes E
S: llista dels nodes terminals
r: node arrel de l'arbre, $r \in V$
B: pressupost màxim

Output: $G' = (V', E')$ tal que el cost de $G' < B$, i es maximitza $\sum_{v \in V'} p(v)$

1. Afegeix r a S
2. Per $i \in |S|$, trobar totes les possibles combinacions de i elements de S i guardar-les a A (cadascuna d'aquestes combinacions representa els nodes terminals als quals arribarà la xarxa)
3. Sigui $B \subseteq A$ tal que tots els elements de B tinguin el node r
4. Sigui $C \subseteq B$ tal que per $x \in C$, el cost de construir l'arbre d'Steiner a partir del graf G amb nodes terminals x sigui menor que B ($cost(Steiner(G, x)) \leq B$)
5. Sigui d l'element $x \in C$ tal que la suma del consum dels nodes de x és màxim
6. Retornar $Steiner(G, d)$

6.1.3.2 Problema del pressupost a partir de la minimització GW

Aquest algorisme (veure 8) es basa en la implementació de [12], on es proposa una solució per resoldre el problema del pressupost partir de l'algorisme GW. El seu funcionament es basa en aplicar un factor multiplicador α als beneficis de cada node, i amb aquests nous beneficis aplicar l'algorisme GW. Així, donat un pressupost B, es tracta de trobar un factor α tal que el cost de l'arbre trobat per l'algorisme GW sigui menor a B, i el cost de l'arbre resultant d'aplicar el factor $\alpha + \epsilon$ sigui superior a B.

Algorithm 8: Algorisme per resoldre el problema del pressupost a partir del problema de minimització GW

Input: $G = (V, E)$: graf amb conjunt de nodes V, conjunt d'arestes E
S: llista dels nodes terminals
r: node arrel de l'arbre, $r \in V$
B: pressupost màxim

Output: $G' = (V', E')$ tal que el cost de $G' < B$, i es maximitza $\sum_{v \in V'} p(v)$

1. $\alpha = 1$
2. Sigui c el conjunt de pesos de les arestes E
3. Sigui p el conjunt de beneficis dels nodes V
4. $T = GW(V, E, c, \alpha p)$
5. Mentre $cost(T) < B$:
6. $\alpha = 2\alpha$
7. $T = GW(V, E, c, \alpha p)$
8. $upper = \alpha$
9. $lower = 1$
10. Mentre $upper - lower \geq 0.1$:
11. $multiplier = (upper + lower)/2$
12. $T = GW(V, E, c, multiplier * p)$
13. Si $cost(T) > B$:
14. $upper = multiplier$:
15. O bé si $cost(T) < B$:
16. $lower = multiplier$:
17. Sinó:
18. break
19. Retorna T

6.1.3.3 Problema del pressupost amb implementació voraç

S'ha implementat l'algorisme a partir de *Greedy algorithm for the STPRBH* ([5]), però adaptant-lo al nostre problema. La idea de l'algorisme és anar construint un arbre iterativament a partir d'una arrel donada, mentre es compleixi la restricció de pressupost. A cada pas de l'algorisme, s'afegeix un camí que connecti un node no seleccionat amb la solució existent. L'algorisme para quan no es poden afegir més nodes. Pel que fa a la selecció del nou camí a afegir, es té en consideració tant el pes del camí com el benefici d'afegir el node destí. L'algorisme amb detall es pot veure a 9. Al ser un algorisme voraç, no es pot garantir que el resultat proporcionat sigui òptim.

6.1.4 Clusterització

Per fer la clusterització necessària pel disseny descentralitzat de la xarxa d'aigua regenerada, hem aplicat els mètodes explicats en l'apartat 3.5 al graf de carrers $G = (V, E)$. Per tots els algorismes i per simplicitat, s'han considerat com a dades d'entrada tots els nodes de V, independentment de si representen o no punts de subministrament de la futura xarxa d'aigua. Com a característiques de cada node (*features*), hem escollit les coordenades GPS i l'elevació sobre el terreny, ja que el que volem és agrupar nodes propers. Totes les dades s'han redimensionat de manera que oscillin entre 0 i 1 (normalització). La normalització és necessària ja que la majoria d'algorismes d'aprenentatge automàtic són sensibles a l'escalat de característiques.

Per poder fer una comparació entre algorismes i extreure'n resultats, s'ha implementat un script (veure algoritme 10) que, a grans trets, funciona de la següent manera: es defineixen les llistes K amb diferents números de clústers, i L, que conté els diferents llindars de consum per determinar a quins nodes s'ha d'arribar. Llavors s'aplica la clusterització per cadascun dels algorismes implementats i per cada $k \in K$ definides. Amb cada clusterització obtinguda, i per cadascun dels llindars $l \in L$ especificats, s'aplica una funció per avaluar el cost de la xarxa i després poder determinar quina és la opció més econòmica.

6.1.4.1 Espectral

Per aplicar la clusterització espectral, no hem implementat exactament la versió especificada al capítol 3.5.1 (versió sense normalitzar), sinó que, tal com es veu a l'algorisme 11, s'han aplicat alguns canvis, seguint [19]. D'aquesta manera, mentre la versió de la clusterització espectral sense normalitzar no acaba de funcionar correctament en el cas que els clústers no tinguin assignats un nombre de nodes similar, la versió utilitzada esmena aquest error. [30]

L'algorisme K-Means no l'hem implementat de zero, sinó que hem utilitzat el mètode que

Algorithm 9: Algorisme voraç pel problema del pressupost

Input: $G = (V, E)$: graf amb conjunt de nodes V, conjunt d'arestes E
S: llista dels nodes terminals
r: node arrel de l'arbre, $r \in V$
B: pressupost màxim

Output: $G' = (V', E')$ tal que el cost de $G' < B$, i es maximitza $\sum_{v \in V'} p(v)$

1. Sigui T un graf buit
2. Afegir node r a T, tal que $vertexs(T) = \{r\}$
3. $edges(T) = \emptyset$
4. $profit = 1$
5. Mentre $profit \neq 0$:
6. $\quad profit = 0$, $\overline{j} = -1$
7. $\quad$ Per tots els vèrtexs $j \notin vertexs(T)$:
8. $\quad\quad$ $cost_j$ = suma dels pesos de les arestes de la xarxa resultant d'unir graf T i vèrtex j pel camí més curt
9. $\quad\quad$ $constructionCost_j$ = cost de construcció de l xarxa resultant d'unir graf T i vèrtex j pel camí més curt
10. $\quad\quad$ $profit_j = p(j)^3 / cost_j$
11. $\quad\quad$ Si $constructionCost_j > B$:
12. $\quad\quad\quad$ $profit_j = 0$
13. $\quad\quad$ Si $profit_j \geq profit$:
14. $\quad\quad\quad$ $\overline{j} = j$
15. $\quad\quad\quad$ $profit = profit_j$
16. $\quad$ Si $profit \neq 0$:
17. $\quad\quad$ Sigui P el camí de cost mínim per unir T amb j
18. $\quad\quad$ $vertexs(T) = vertexs(T) \cup vertexs(P)$
19. $\quad\quad$ $edges(T) = edges(T) \cup edges(P)$
20. $\quad\quad$ $T = MST(T)$
21. Retornar T

proporciona la biblioteca Scikit-learn que en simplifica la implementació, de manera que només s'han de definir els paràmetres d'entrada. Per aquest mètode són:

- k: Número de clústers
- X: Matriu amb informació dels nodes a clusteritzar, de mida $m \times n$, on $m = |V|$ i $n =$ número de *features*. Per nosaltres, aquesta matriu és la matriu que anomenem Y, resultant

Algorithm 10: Avaluació de la clusterització per diferents mètodes i número de clústers

Input: $G = (V, E)$: graf amb conjunt de nodes V, conjunt d'arestes E
K: llista de número de clústers
L: llista de llindars de consum
F: llista d'algorismes de clusterització

Output: $D\{k : \{f : c\}\}$: diccionari en format CSV amb els costos, guardats per cada k i per cada f, el seu cost c.

1. Inicialitzar diccionari $D\{k : \{f : c\}\}$ per guardar els costos
2. Per cada algorisme $f \in F$:
3. Per cada $k \in K$:
4. Executar algorisme de clusterització $f(k)$.
5. Sigui $G_{k,f}$ el graf G inicial amb els nodes etiquetats segons l'assignació de clústers $f(k)$.
6. Per cada llindar de consum $l \in L$:
7. Calcular cost c de xarxa descentralitzada per $G_{k,f}$ i guardar-lo a D.
8. Retornar D amb tots els costos.

Algorithm 11: Clusterització espectral

Input: $G = (V, E)$: graf amb conjunt de nodes V, conjunt d'arestes E
k: nombre de clústers

Output: Per cada node de V, diu a quin clúster pertany

1. Calcular la matriu d'adjacència S a partir dels pesos de cada aresta
2. Calcular la matriu de graus D
3. Sigui L la matriu laplaciana normalitzada, tal que $L = D^{-1/2}SD^{1/2}$
4. Calcular els k eigenvalues més grans
5. Crear la matriu $V = [v_1, ..., v_k]$, on v_i és el vector propi (en forma de columna) corresponent al i-èssim valor propi
6. Sigui Y la matriu resultant de normalitzar cadascuna de les files de V per tal que tinguin norma euclídia $||x||_2 = 1$
7. Aplicar l'algorisme $k - means$ per clusteritzar amb k nombre de clústers la matriu Y
8. Assignar el vèrtex x_i al clúster j si i només si y_i s'ha assignat al clúster j

de normalitzar cadascuna de les files de la matriu de vectors propis (ordenats de menor a major), i quedant-nos només amb les k primers columnes.

6.1.4.2 Agrupament jeràrquic

El mètode d'agrupament jeràrquic que hem implementat és l'explicat a l'apartat 3.5.2. Però com a paràmetres d'entrada a l'algorisme, hi hem afegit el que s'anomenen **restriccions de connectivitat**. En el nostre cas, aquestes restriccions són la pròpia matriu d'adjacències del graf, la qual ja determina si dos nodes estan o no connectats entre si. El que s'aconsegueix amb aquestes restriccions són dues coses:

- Millorar el temps de còmput, sobretot quan el nombre de dades d'entrada és molt gran.
- Tenir en compte l'estructura del graf, ja que s'evita que en cada iteració s'uneixin en un mateix clúster nodes que no són adjacents entre si.

De la mateixa manera que hem fet amb el mètode anterior, aquí també hem utilitzat l'algorisme que ja té implementat la biblioteca Scikit-learn per aquest propòsit. Els paràmetres d'entrada que s'han de passar són:

- k: Número de clústers
- A: Matriu de restriccions de connectivitat, que com hem dit, és la matriu d'adjacències del graf, de mida $m \times m | m = |V|$.
- *ward*: Criteri d'enllaç
- *euclidean*: Mètrica pel càlcul de distàncies
- X: Matriu amb informació dels nodes a clusteritzar, de mida $m \times n$, on $m = |V|$ i $n =$ número de *features*.

Per aquest algorisme hem fet execucions amb vectors de *features* de mida 2, latitud i longitud en coordenades GPS dels nodes; i de mida 3, afegint també l'elevació dels nodes sobre el terreny.

6.1.4.3 SOM

Per clusteritzar a partir de SOMs hem utilitzat la biblioteca MiniSom, que que conté una implementació de codi obert, senzilla i minimalista dels *Self Organizing Maps* [28], i ens permet abstreure'ns de la implementació a baix nivell de la xarxa neuronal.

En primer lloc, cal inicialitzar un objecte de tipus SOM, amb els seus paràmetres que són la mida del mapa $x \times y$ o número de neurones, número de característiques (*features*) dels paràmetres d'entrada, *learning rate* α, funció de veïnatge Θ i una llavor per inicialitzar els pesos aleatòriament. Llavors es pot entrenar el SOM passant-li la matriu de dades i un número

d'iteracions a fer per cada vector d'entrada.
En la nostra implementació hem provat diverses execucions amb diferents valors pels paràmetres de l'objecte SOM.

- $x = 2$, $y = k$: Mida del mapa o número de neurones, on k és el nombre de clústers mínim que volem obtenir
- $\Theta = gaussian$: Funció de veïnatge
- $\alpha = 0.5$: coeficient d'aprenentatge
- 500 iteracions
- X: Matriu amb informació dels nodes a clusteritzar, de mida $m \times n$, on $m = |V|$ i $n =$ número de *features*.

Per aquest algorisme, finalment només hem fet execucions amb vectors de *features* de mida 2, corresponents a la latitud i longitud en coordenades GPS dels nodes. En fases inicials es va provar amb 3 característiques, afegint-hi també l'elevació sobre el terreny de cada node. Però els resultats que s'obtenien no eren els esperats, ja que el SOM agrupava nodes allunyats entre si (en quant a coordenades GPS), però que estaven a altituds similars. I nodes intermedis d'elevació diferent, els etiquetava a un altre clúster.

6.1.4.4 GNN

Per últim, hem aplicat una clusterització amb aprenentatge profund no supervisat amb una GNN. Ho hem fet a partir de la implementació del mètode que s'explica a [27] anomenat DMoN que, com hem comentat en l'apartat 3.5.4.1, clusteritza els nodes del graf d'entrada a partir d'optimitzar-ne (maximització) la seva modularitat Q. Els paràmetres d'entrada són:

- X: Matriu amb informació dels nodes a clusteritzar, de mida $m \times n$, on $m = |V|$ i $n =$ número de *features*.
- A: Matriu d'adjacències normalitzada.
- k: Número de clústers.
- $epochs = 500$
- $\alpha = 0.001$: Coeficient d'aprenentatge

Aquesta xarxa neuronal consta de dues capes:

1. **GCN** - Capa encarregada de fer el que s'anomena un *soft assignment*. En clusterització, els *soft assignments* són assignacions que es fan de cada node a cadascun dels clústers amb un pes o probabilitat p. La capa aplica la següent fórmula:

$$C = softmax(GCN(\hat{A}, X))$$

   tal que els paràmetres d'entrada a la xarxa són la matriu d'adjacències normalitzada $A = D^{-1/2}(A)D^{-1/2}$ i la matriu de *features* X; i el paràmetre de sortida és la matriu d'assignacions (soft) $C \in 0, 1^{n \times k}$, on $n = |V|$ i $k =$ número de clústers. Aquesta matriu s'obté a partir d'una funció d'activació *softmax*.

2. **DMoN** - Capa encarregada d'optimitzar la modularitat de la xarxa a partir de la matriu d'assignacions C de la capa anterior i la matriu d'adjacències del graf (no normalitzada). Retorna la matriu d'assignacions C optimitzada.

Per últim, per cada node $v \in V$, cal passar de *soft* a *hard assigment*, de manera que $k(v) =$ assignació soft de màxima probabilitat.

Per aquest algorisme hem fet execucions amb vectors de *features* de mida 2, latitud i longitud en coordenades GPS dels nodes; i de mida 3, afegint també l'elevació dels nodes sobre el terreny. Com que en cada execució s'obté un resultat de clusterització una mica diferent, per cada valor de k i per cada vector de característiques diferent (de mida 2 o 3), s'han realitzat 121 iteracions (amb els seus 500 epochs cadascuna). De manera que, a cada node, li assignem finalment l'etiqueta del clúster que més vegades es repeteix en les 121 iteracions, que és el nombre d'iteracions a partir de les quals es considera que no hi ha influència de l'atzar.

6.2 Sensor placement

Com hem explicat a l'apartat 1.2 d'objectius, de la part de posicionament de sensors només hem implementat el que anomenem algorismes dinàmics. Prèviament s'ha implementat un algorisme per posicionar sensors estàtics a la xarxa de clavegueram, des del qual s'obté el conjunt S de punts on col·locar els sensors i, per cada node sensor $s \in S$, el subgraf $G(s)$ de la part de xarxa que cobreix. Llavors, un node s i $G(s)$ serveixen de paràmetres d'entrada als algorismes dinàmics que hem implementat seguint la proposta de Larson et al. [14]. La seva aproximació consisteix en assignar probabilitats Bayesianes d'infecció, segons criteris i consideracions professionals, a tots els nodes origen (nodes de la xarxa que tenen alguna parcel·la amb habitants assignada) i aplicar una cerca binària per determinar els punts de mostreig utilitzant les probabilitats. La nostra implementació va un pas més enllà, i les probabilitats que assignem als nodes origen són en funció dels habitants que hi ha assignats, ja que creiem que a més població, més probabilitats

de detectar gent infectada. S'assumeix, a més, que existeix algun tipus de mètode d'anàlisi portable i ràpid (resultats en menys de 24 hores) per tal de garantir que es poden executar diverses iteracions "mostra-test" en un període curt de temps.

Hem implementat dos algorismes:

i algorisme **Pacient Zero** (PZ) - assumeix que només hi ha un únic cas de COVID-19 en una comunitat (el pacient zero), i minimitza la seqüència de pous d'on s'han d'extreure mostres per localitzar aquest pacient zero.

ii algorisme **Hot Spot** (HS) - assumeix que moltes persones ja estan infectades i intenta localitzar el clúster o focus d'infecció de la xarxa d'aigües residuals amb major presència de traces de SARS-CoV-2.

6.2.1 Algorisme HS

L'algorisme HS funciona de la següent manera: a cada iteració es busca el node que té probabilitat Bayesiana d'infecció més alta, que serà el punt de mostreig. Després s'extreu una mostra, es testeja, i s'obté un valor de càrrega viral $F' \leq F$. Si F' té un valor alt en comparació a l'obtingut al punt de mostreig anterior, sabem que el focus d'infecció està corrent amunt i descartem l'altra part de la xarxa. Altrament, la part de xarxa corrent amunt al punt de mostreig es descarta i es continua buscant per l'altra part. D'aquesta manera, en cada iteració reduïm aproximadament a la meitat el nombre d'habitants que queden al graf. Per simplificar la simulació, considerem que el valor F' és booleà. L'algorisme para quan s'assoleix la regla d'aturada, que l'usuari ha de definir prèviament. En el nostre cas, l'algorisme acaba quan s'ha localitzat un focus important de 3000 habitants o menys. Es pot veure la implementació de l'algorisme a 12.

Tant l'algorisme HS com el PZ comparteixen tres passos previs, els quals definim a continuació. Donat $G(s) \leftarrow \{V(s), E(s)\}$, c_v és la població associada al node $v \in V(s)$:

1. **Crear nodes artificials:** Tots els nodes origen han de tenir $deg^-(v) = 0$. Per aconseguir-ho, per cada node interior (anomenat "node original") amb població associada, crear un nou node (anomenat "node artificial") amb la mateixa població associada i connectat al seu "node original". Des de qualsevol "node artificial" es pot obtenir fàcilment el seu "node original". Sigui A el conjunt amb tots els nodes artificials de $G(s)$.

   1.1. Crear conjunt buit A

   1.2. $\forall v \in V(s)$ tal que $deg^-(v) > 0$ i $c_v > 0$, afegir nou node v' i aresta (v', v) a $G(s)$, assignar $c_{v'} \leftarrow c_v$, definir $original(v') \leftarrow v$, afegir v' a A

   1.3. Retornar A

Algorithm 12: Algorisme dinàmic de selecció de punts de mostreig

Input: $G(s) = \{V(s), E(s)\}$: Graf dirigit amb conjunt de nodes V, arestes E i un únic node $s \in V \mid deg^+(s) = 0$, corresponent al *sensor fixe*.
$\forall v \in V(s), c_v$: ciutadans de v.
F: Càrrega viral detectada al *sensor fixe* s.
Output: $T \mid T \subset V(s)$: Nodes del focus d'infecció

1. $A \leftarrow$ Crear nodes artificials
2. $P \leftarrow$ Simplificar i normalitzar
3. Definir la *regla d'aturada*.
4. Mentre no s'arribi a la *regla d'aturada*:
5. Propagar probabilitats
6. Trobar node $t \in (V(s) \backslash A)$ que $t \leftarrow \min_{v \in (V(s) \backslash A)} \mid probability(v) - F/2 \mid$. En cas d'empat, escollir el node amb probabilitat més gran.
7. Sigui G' el subgraf de $G(s)$ al qual pertany t i tots els antecessors de t.
8. Extreure i analitzar mostra del punt t.
9. $F' \leftarrow$ *càrrega viral detectada*
10. Si $F' \geq F/2$ llavors $G(s) \leftarrow G'$, $F \leftarrow F'$
11. Altrament, $G(s) \leftarrow G(s) \backslash G'$, $F \leftarrow F - F'$
12. $P \leftarrow$ Simplificar i normalitzar el graf
13. $T \leftarrow V(s) \backslash A$
14. Retornar T

2. **Simplificar i normalitzar:** Eliminar tots els nodes que no tenen cap utilitat en el càlcul per tal de simplificar el graf i després assignar les probabilitats Bayesianes als nodes origen a partir de la seva població associada.

   2.1. $\forall v \in V(s)$, si $deg^-(v) = 0$ i $c_v = 0$, eliminar node

   2.2. $\forall v \in V(s)$ amb $deg^-(v) = 1$, $deg^+(v) = 1$ i $c_v = 0$, afegir aresta a $G(s)$ des del predecessor de v fins al successor de v. Eliminar el node v

   2.3. Sigui $P \subset V(s)$ el conjunt de tots els nodes $v \in V(s)$ tals que $deg^-(v) = 0$

   2.4. Normalitzar P:

$$\forall p \in P,\ probability(p) \leftarrow \frac{c_p}{\sum_{p \in P} c_p}$$

   2.5. Retornar P

3. **Propagar probabilitats:** Propagar les probabilitats des dels nodes origen a la resta de

nodes, de manera que la probabilitat de cada node interior és la suma de probabilitats del node que té corrent amunt. Sigui P el conjunt de nodes obtinguts després de **simplificar i normalitzar** $G(s)$.

3.1. $\forall v \in (V(s) \backslash P)$, i W són tots els predecessors de v:

$$probability(v) \leftarrow \sum_{w \in W} \frac{probability(w)}{deg^{+}(w)}$$

6.2.2 Algorisme PZ

L'algorisme del PZ és una instància especial de l'algorisme HS, en la qual la regla d'aturada s'assoleix quan només queda un únic node origen. Aquest node representa el pacient zero.

6.2.3 Simulacions

L'algorisme s'executa tant per tota la xarxa de clavegueram de Girona G, considerant que només hi ha un únic sensor estàtic que es troba a la depuradora; com en una xarxa reduïda $G(s)$, que és la part de xarxa coberta per un sensor s (que no està localitzat a la depuradora). Llavors, hem considerat 4 casos:

- **Cas 1** - tota la xarxa de Girona G. L'assignació de probabilitats als nodes origen es fa en funció de la població.
- **Cas 2** - tota la xarxa de Girona G. L'assignació de probabilitats es fa de forma aleatòria seguint una distribució unitària de probabilitat.
- **Cas 3** - subgraf $G(s)$. L'assignació de probabilitats es fa en funció de la població.
- **Cas 4** - subgraf $G(s)$. L'assignació de probabilitats es fa de forma aleatòria seguint una distribució unitària de probabilitat.

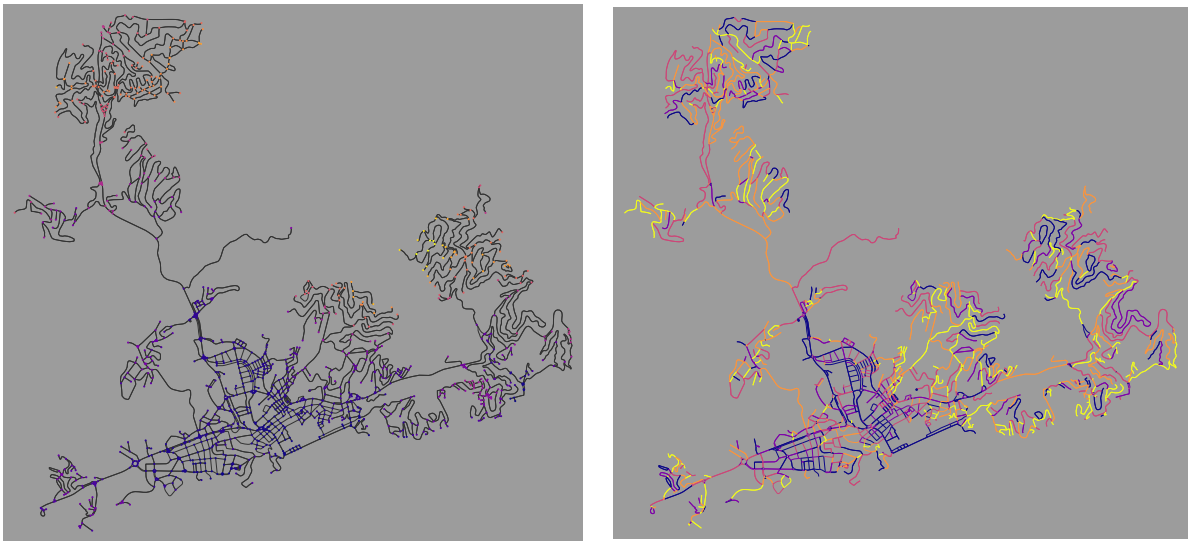
Els casos 2 i 4 serien comparables a la metodologia proposada a [14]. La xarxa sencera G consta de 9718 pous, 338km de canonades, i cobreix una població de 151,248 habitants. Pel que fa al $G(s)$ seleccionat, inclou 1099 pous i cobreix una població de 44,102 habitants.

7. Resultats

7.1 CleanTour

7.1.1 Obtenció de les elevacions

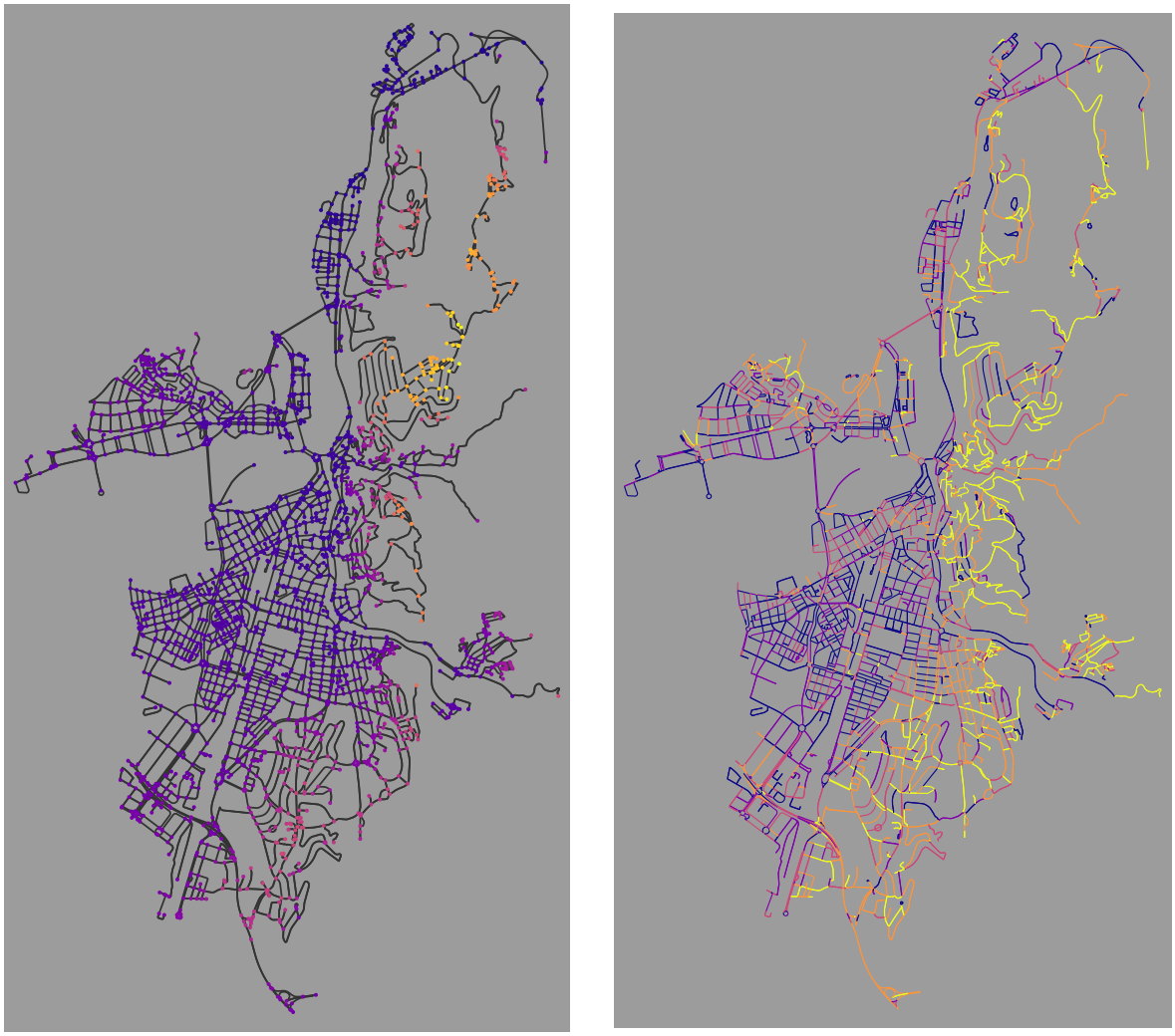
Per tal de comprovar que el model d'elevacions obtingut fos correcte, a les figures 7.1 i 7.2 hem representat la xarxa de carrers de Lloret de Mar i Girona respectivament, de dues maneres diferents: en la primera hem assignat el color de cada node segons la seva elevació (com més groc, més altitud), mentre que en la segona, hem afegit colors a les arestes, de manera que com més groga sigui aquesta, més pendent hi ha entre els nodes que uneix.



(a) Elevació dels nodes

(b) Pendent de les arestes

Figura 7.1: Model d'elevació de la xarxa de Lloret de Mar



(a) Elevació dels nodes

(b) Pendent de les arestes

Figura 7.2: Model d'elevació de la xarxa de Girona

7.1.2 Encaminament

En aquest apartat es comparen els algorismes d'ACO (2) i de Mehlhorn (6), juntament amb l'algorisme de Kou[13] i el de Takahashi[26], els quals ja estaven implementats per altres membres del grup. A la taula 7.1 es pot veure una comparació entre tots aquests algorismes, tant de la seva complexitat temporal com del pitjor resultat garantit respecte l'arbre d'Steiner òptim (ràtio).

Algorisme	**Temps de còmput**	**Ràtio**
Kou	$O(\|V\|\|E\|log\|V\|)$	$2-2/t$
Takahashi	$O(\|S\|\|V\|^2)$	$2-2/\|S\|$
Mehlhorn	$O(\|V\|log\|V\|+E)$	$2-2/t$
ACO	-	-

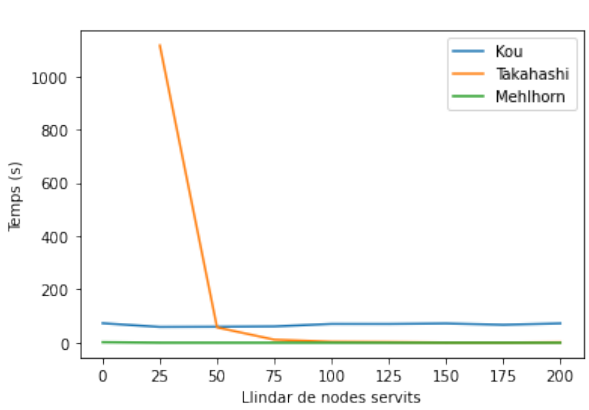
Taula 7.1: Comparació tècnica dels diferents algorismes. Donat $G=(V,E)$ trobar l'arbre d'Steiner amb terminals S. t fa referència al nombre de fulles de l'arbre d'Steiner, i per tant, $t \leq |S|$

A les figures 7.3a, 7.3b, 7.3c i 7.3d es pot veure tant el temps d'execució com els costos resultants de la xarxa després d'executar els diferents algorismes per calcular l'arbre d'Steiner sobre el graf de Girona i Lloret de Mar portant l'aigua des de la depuradora fins als destins corresponents (versió centralitzada).

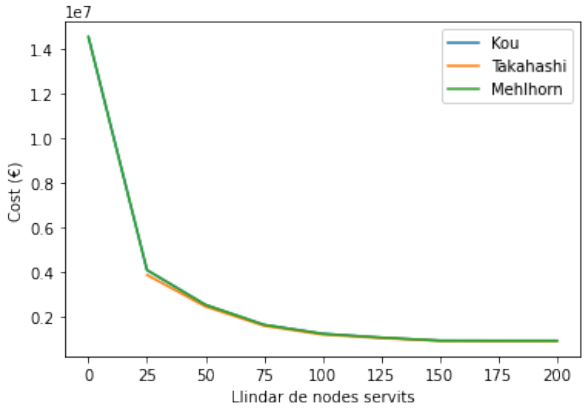
Com es podia concloure de la taula 7.1, l'algorisme de Mehlhorn és el més eficient (com a molt tarda 2 segons en resoldre la instància amb més nodes), seguit per l'algorisme de Kou (que tal com s'ha implementat, el seu temps d'execució no depèn del nombre de nodes terminals, sinó només del graf base), el qual en el cas de Girona tarda al voltant d'1 minut. Pel que fa al cas de l'algorisme de Takahashi, és considerablement més lent que els altres (amb 24 hores no ha resolt les proves amb llindar 0). Tot i això, el seu resultat és lleugerament millor (els arbres resultants que dóna tenen un cost més baix), mentre que el cost de les xarxes proporcionades per l'algorisme de Kou i Mehlhorn són equivalents. No s'han mostrat els resultats de l'algorisme ACO ja que no ha sigut capaç d'acabar cap execució.

7.1.3 Optimització

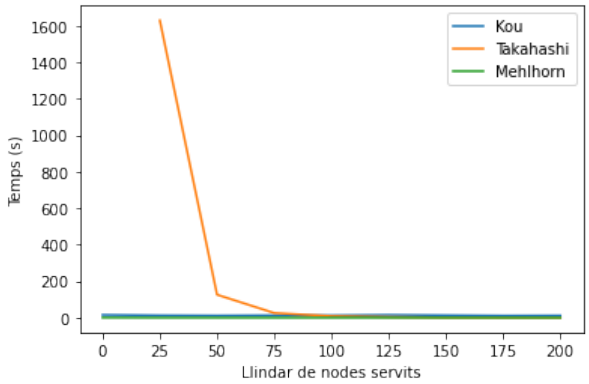
A les figura 7.4 es pot veure els metres cúbics d'aigua distribuïts i temps d'execució dels algorismes d'optimització per força bruta (7) i la implementació voraç (9) del graf de Girona amb un pressupost de 1 milió d'€. La versió implementant minimització GW (8) no trobava resultats correctes, motiu pel qual no s'ha representat. Es pot veure com la implementació per força bruta troba solució en un espai de temps considerablement ràpid per llindars superiors a 75, però amb llindars inferiors l'execució no acaba. Pel que fa a la l'algorisme voraç, la solució proporcionada s'aproxima bastant a la solució exacta, i fins i tot i amb pocs nodes terminals la solució donada



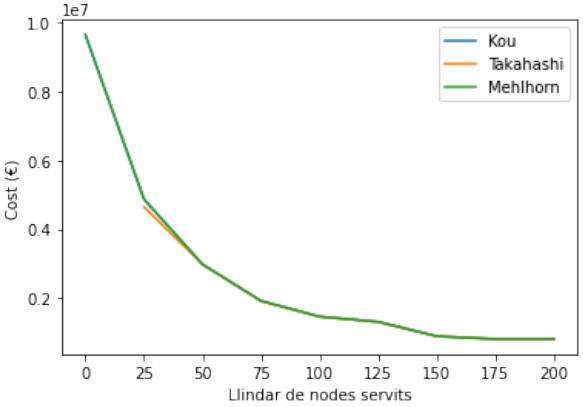
(a) Temps d'execució algorismes sobre Girona



(b) Cost construcció xarxa sobre Girona



(c) Temps d'execució algorismes sobre Lloret



(d) Cost construcció xarxa sobre Lloret

Figura 7.3: Comparació temps i resultat dels algorismes d'encaminament

coincideix amb l'òptima.

(a) m^3 d'aigua servits

(b) Temps d'execució

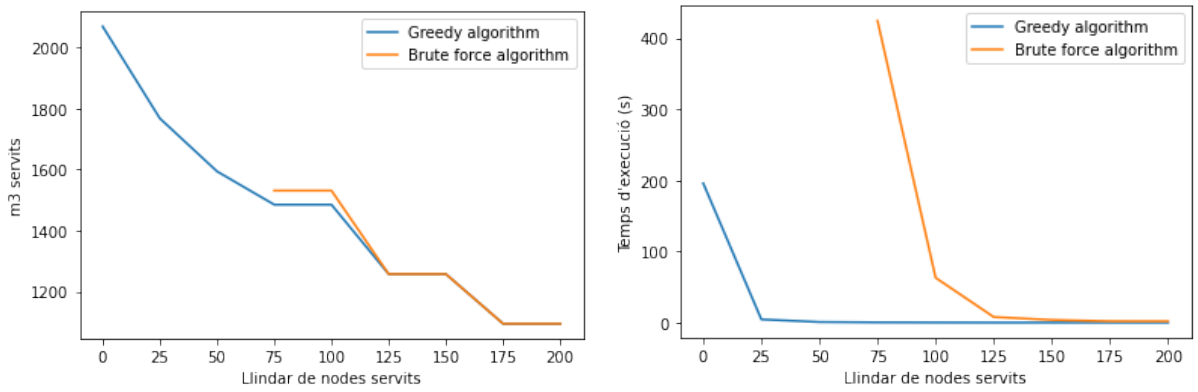


Figura 7.4: Comparació algorismes d'optimització en la ciutat de Girona, per un pressupost d'1 milió d'€

7.1.4 Estadístiques complementàries

A mode d'exemple, a la figura 7.5 es pot veure la xarxa de carrers de Girona, on les arestes de color representen el recorregut de la xarxa d'aigua regenerada més curta (realitzada amb l'algorisme de Mehlhorn), que distribueix aigua als nodes que consumeixen més de 50 m^3 d'aigua diaris (llindar 50). Els nodes verds representen els nodes destins, el node taronja representa la depuradora, i els nodes blaus les vàlvules. Com més intens és el color de l'aresta, més diàmetre té la canonada que representa. L'ús de les canonades per aquest llindar es pot veure més en detall a 7.6. Òbviament, com que per la depuradora surt tot el volum d'aigua a distribuir, el diàmetre de les canonades sortint l'origen és superior al de les canonades que distribueixen als destins.

Pel que fa al cas d'ús de la ciutat de Girona, si ens centrem en l'anàlisi de com varia l'ús i les característiques de la xarxa en funció del llindar, podem veure (figura 7.7) com en el cas dels llindar 50 i 75 (i fins a 200), gairebé la majoria d'aigua es dedica a conreus, tot i que també hi ha una part que es fa servir en jardins privats, camps de futbol, altres complexes esportius i en residències. Entre els llindars 0 i 25, però, es pot veure com es dispara l'ús d'aigua tant en llars com en jardins públics. Aquesta afirmació es veu validada en les figures 7.8a, 7.8c i 7.8d. Entre els llindars 50 i 25, però sobretot 25 i 0 es pot veure com augmenta la demanda de consum d'aigua, seguint la mateixa tendència el nombre de parcel·les on servim aigua, i el nombre d'habitants vinculats a aquestes.. Per tant, tot i que cada casa té un consum relativament petit, el consum global en habitatge és molt gran (però també és molt més car arribar a tots aquests destins). Tal com mostra la figura 7.8b, és més elevat el cost d'obrir i tancar la rasa que no pas el material de les canonades en si.

Aquesta tendència d'usos que hem comentat per Girona no es manté pel cas de Lloret de Mar (figura 7.9), on la major presència d'hotels i urbanitzacions fa que el consum d'aigua en hotels i jardins sigui més elevat que en el cas de la ciutat de Girona.

7.1.5 Clusterització

A continuació es fa una avaluació dels algorismes utilitzats en la clusterització, a partir del cost que representa construir una xarxa nova d'aigua regenerada descentralitzada.

S'han recollit els resultats dels costos de les diferents execucions i algorismes en diferents diagrames i gràfiques. D'aquesta manera podem tenir una representació visual de les dades i afavorir-ne la seva comprensió, alhora que facilita poder extreure conclusions.

7.1.5.1 Visualització resultats

- **Mapa de calor** - Visualitza la relació entre dues variables, una en cada eix, a través de variacions de color. Observant aquests canvis es poden trobar patrons per una o ambdues variables, i veure si existeix alguna correlació entre elles. Per cadascuna de les ciutats, Girona i Lloret de Mar, tenim 5 mapes (un per cada llindar de consum) que relacionen l'algorisme (eix Y) i el número de clústers (eix X) en funció del cost per aquell llindar concret. Dels algorismes que s'ha aplicat la clusterització amb 2 i 3 *features*, s'ha agafat el que té el cost menor. Per consultar els mapes de Girona, veure figura 7.10, i figura 7.15 pels de Lloret.
- **Diagrama de punts i línies** - S'utilitza sovint per visualitzar una tendència en les dades. Hem fet tres representacions d'aquest tipus:
  1. Diagrames que contenen exactament la mateixa informació que els mapes de calor, però aquesta representació ens permet veure si hi ha algorismes que segueixen la mateixa tendència o similar per cada llindar, a mesura que s'incrementa el número de clústers (veure figura 7.11 pels de Girona, i figura 7.16 pels de Lloret).
  2. Ídem anterior, però la variable independent és el llindar (i no la k), i estan agrupats per k. Aquesta representació s'ha fet per veure si els algorismes obtenen costos que segueixen una tendència similar quan, per el mateix número de clústers, s'incrementa el llindar de consum (i decreix el número de destins). Veure figura 7.12, i figura 7.17 pels de Lloret.

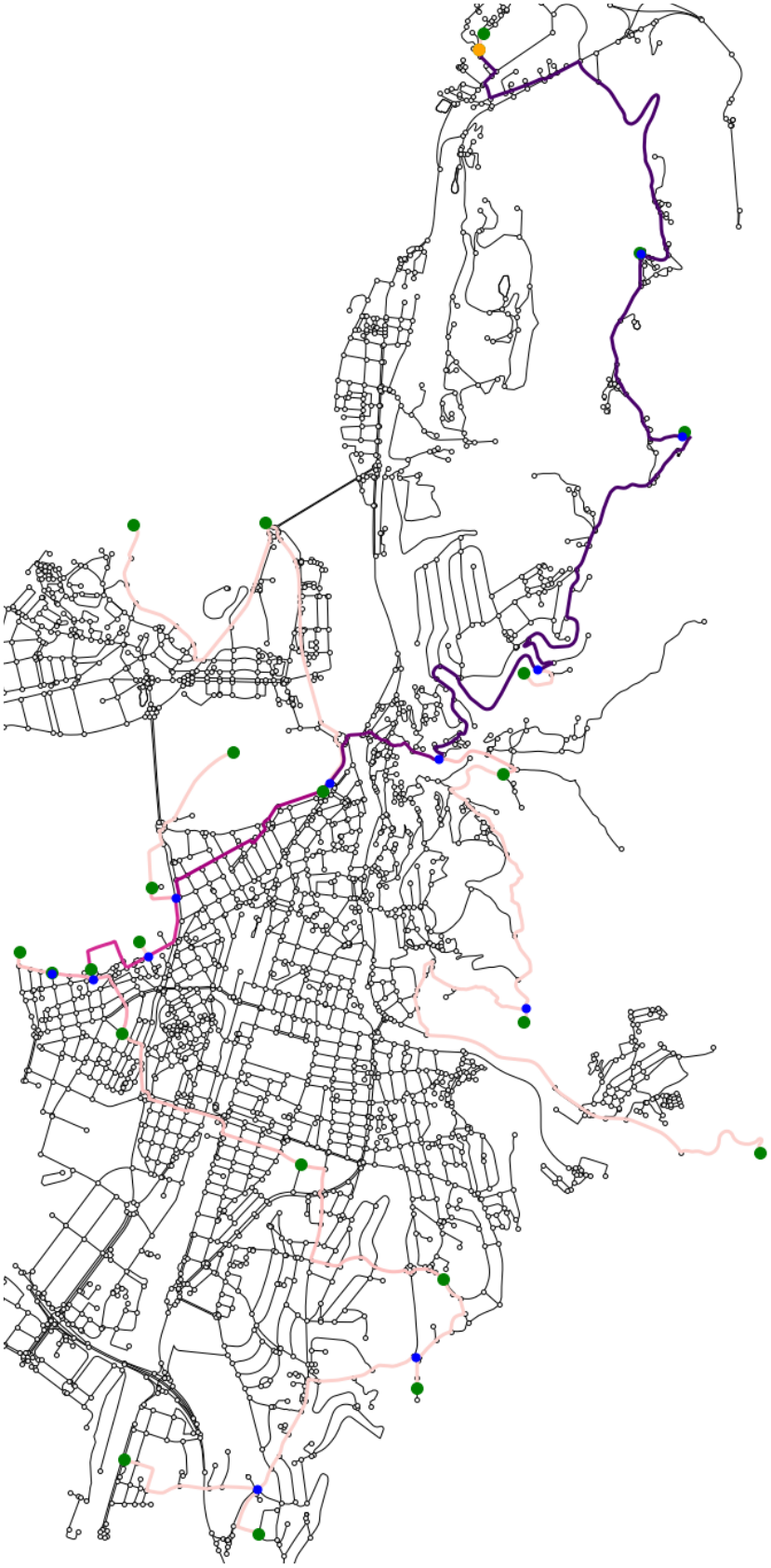


Figura 7.5: Xarxa de Girona amb nodes a partir de llindar 50

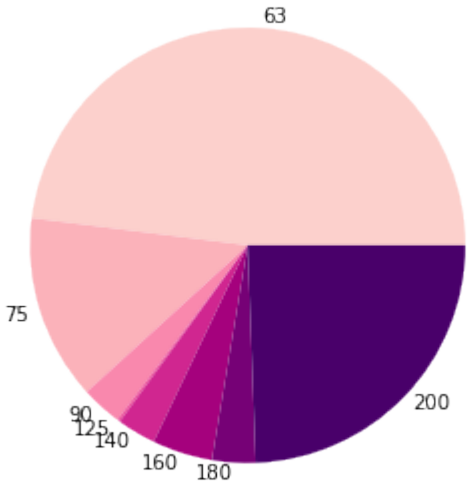


Figura 7.6: Diàmetre de les canonades (mm) utilitzades en llindar 50

3. Representació per comparar, amb el mateix algorisme, els costos que s'obtenen quan es clusteritza amb 2 o 3 *features* i per diferents k, agrupats per llindars. Veure figura 7.13, i figura 7.18 pels de Lloret.

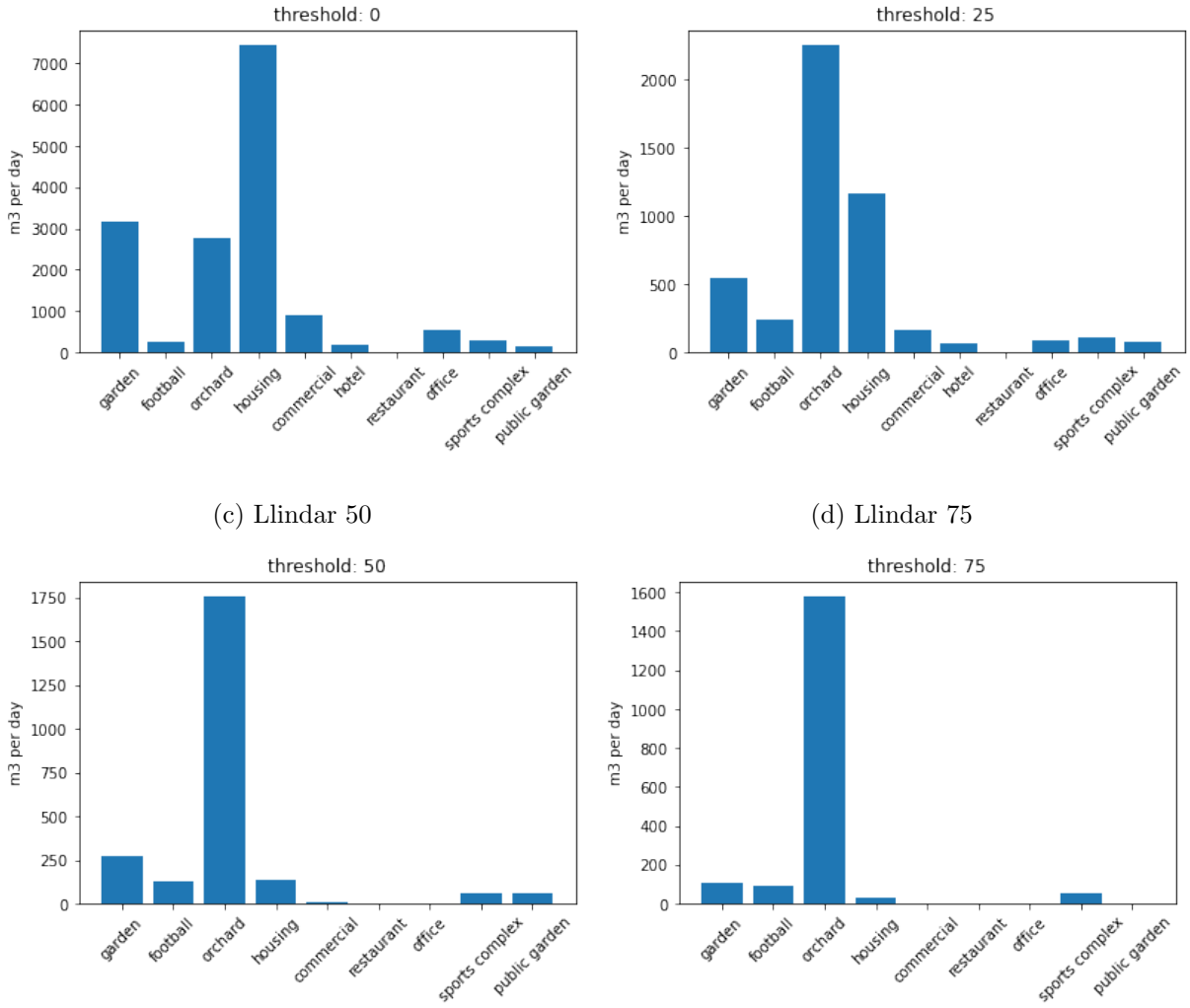
7.1.5.2 Interpretació dades

De la representació gràfica de les dades observem el següent:

- Tots els algorismes segueixen la mateixa tendència decreixent pel que fa a costos per un mateix número de clústers, quan el llindar de consum augmenta. Això veiem que passa tant a Lloret com Girona.
- A la ciutat de Lloret, a excepció del llindar 0, s'observa que l'augment de costos que es produeix quan s'incrementa el número de clústers és molt similar entre els algorismes de clusterització espectral i l'aglomeratiu. A Girona, però, els algorismes que guarden similitud en aquest sentit són l'aglomeratiu i la GNN.
- Pel que fa a si els costos són més econòmics si es clusteritza amb 2 o 3 *features*, no s'observa cap patró clar, però sí que es veu que a Girona hi ha molt poca diferència en el resultat per l'algorisme espectral, mentre que a Lloret l'algorisme que presenta menys diferències en aquest sentit és l'aglomeratiu.
- En conjunt, l'algorisme aglomeratiu és el que guanya més vegades, un 56%, però és sobretot a Lloret on millor funciona, guanyant en un 80% dels casos, per un 36% a Girona. En segon lloc hi ha l'espectral amb un 31%, que guanya a Girona amb un 48% dels casos per

(a) Llindar 0

(b) Llindar 25



(c) Llindar 50

(d) Llindar 75

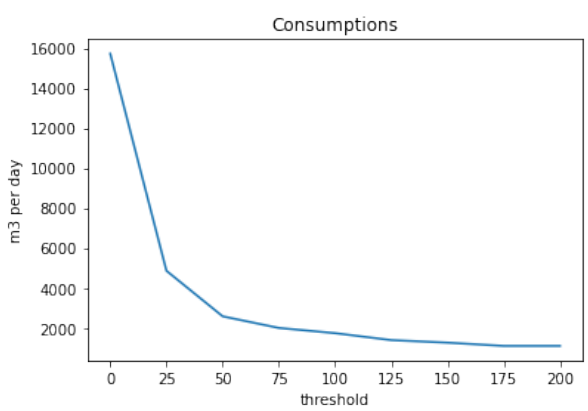
Figura 7.7: Consum d'aigua a Girona segons usos

un 10% a Lloret. Després tenim empatats amb un 6.5% la GNN i el SOM. Si ens fixem en el número de característiques (amb o sense elevació), un 49% de les vegades guanya la clusterització sense tenir en compte l'elevació, i un 51% tenint-la en compte.

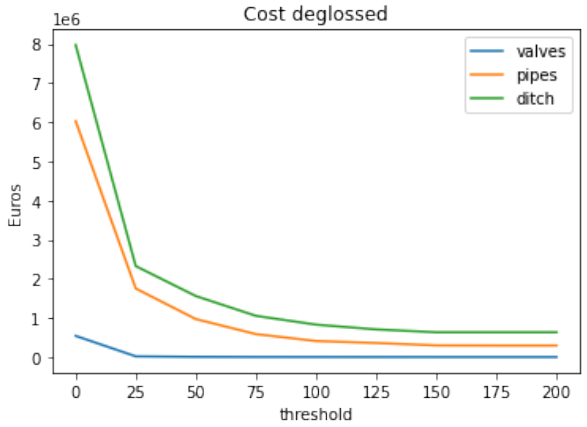
7.2 Sensor placement

Els resultats obtinguts en les simulacions per cadascun dels algorismes es poden veure a la figura 7.20, que mostra la distribució de la freqüència de punts de mostreig requerits per cada cas. Estan representades en un diagrama de violí, de manera que permet visualitzar no només la distribució, sinó la seva densitat.

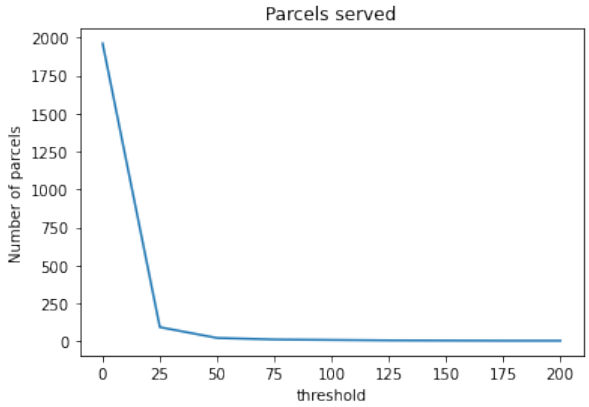
(a) Consum



(b) Costos desglossats



(c) Nombre de parcel·les



(d) Població servida

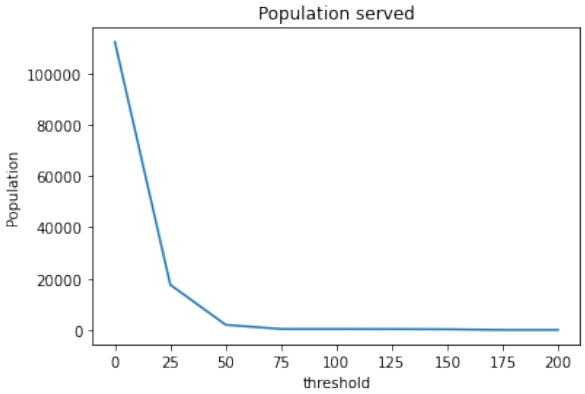


Figura 7.8: Estadístiques ciutat de Girona

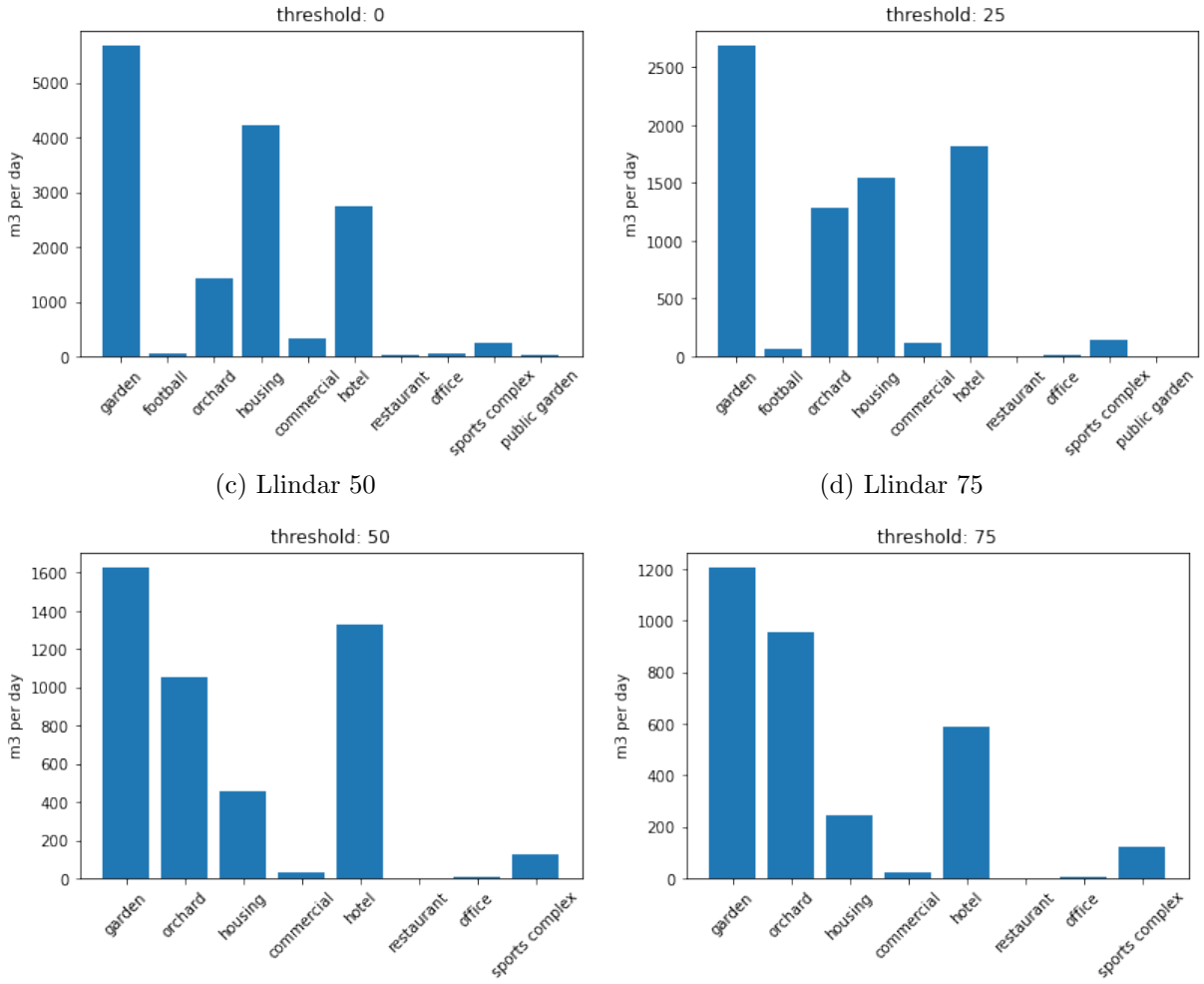
7.2.1 Algorisme PZ

Quan s'executa l'algorisme PZ a tota la xarxa amb probabilitats en funció de la població (cas 1), el nombre de punts de mostreig per identificar el primer individu responsable de l'abocament de traces genètiques del SARS-CoV-2 a la xarxa, oscil·la entre 8 i 15. Per contra, quan les probabilitats s'assignen aleatòriament (cas 2), el rang és més petit, entre 10 i 14 iteracions. Si ens fixem en la mediana de les distribucions, però, es necessitaria una iteració menys quan s'assignen les probabilitats en funció de la població. Aquest guany podria ser major en ciutats amb major variabilitat en la densitat de població entre barris.

L'execució de l'algorisme PZ al subgraf resulta en la reducció de dues iteracions a l'hora d'identificar el pacient zero (cas 3 en comparació al cas 1, i cas 4 respecte el cas 2). En general, l'estratègia que requereix menys punts de mostreig és la que representa el cas 3: partir d'una xarxa reduïda fruit d'aplicar l'algorisme de selecció de sensors estàtics, i assignar les probabilitats en funció dels habitants. Els resultats es mostren a la figura 7.20a.

(a) Llindar 0

(b) Llindar 25



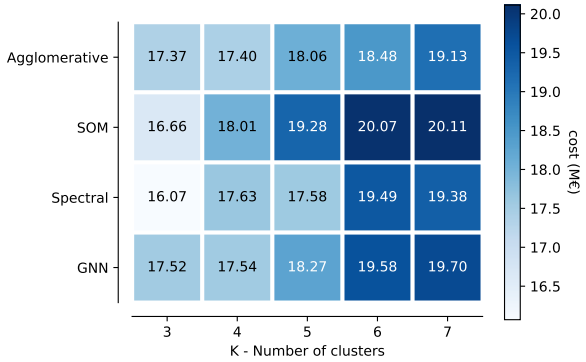
(c) Llindar 50

(d) Llindar 75

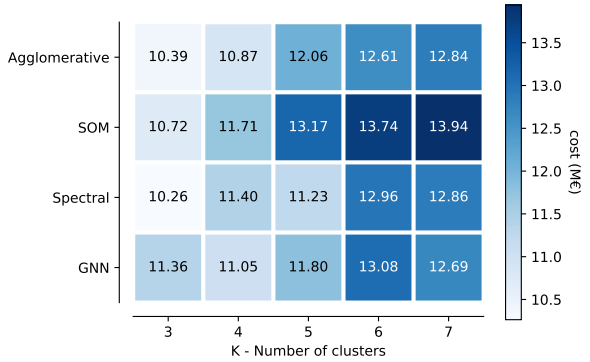
Figura 7.9: Consum d'aigua a Lloret segons usos

7.2.2 Algorisme HS

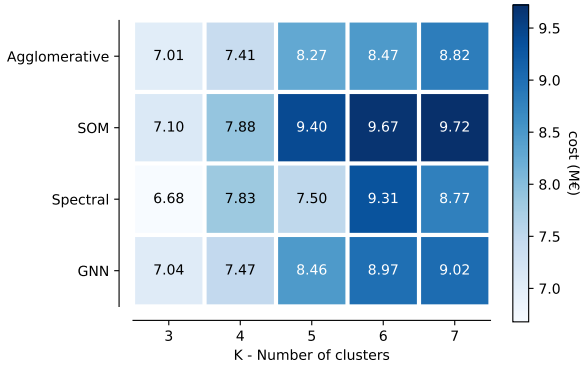
Com ja hem explicat a l'apartat 6.2.1, fixem 3000 habitants infectats com a la *regla d'aturada* de l'algorisme HS. Si ho comparem amb l'algorisme PZ, per cada cas es redueix en 5 el nombre d'iteracions. Es pot veure a la figura 7.20b que quan s'assignen probabilitats en funció dels habitants, es requereix una mostra menys (entre casos 2 i 1, i entre casos 4 i 3). A més, també resulta en una distribució més densa en comparació als casos amb probabilitat aleatòries. Això, però, és just el contrari del que passa amb l'algorisme PZ. L'opció més favorable, alhora que conservadora, seria la del cas 3: major certesa que amb 4 o 5 iteracions s'identificaria el focus d'infecció. En aquest sentit, el benefici de partir de la xarxa reduïda obtinguda de l'algorisme de selecció de sensors és que ens estalviem 1 o 2 iteracions.



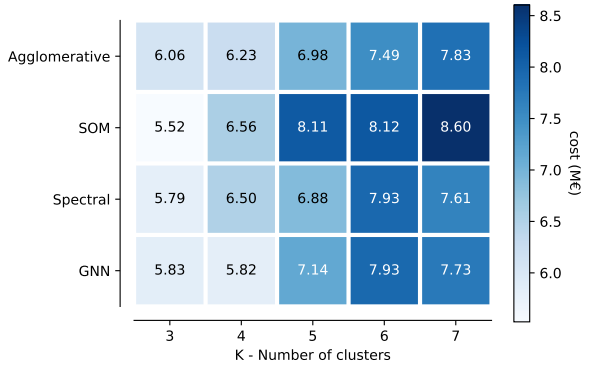
(a) Llindar 0



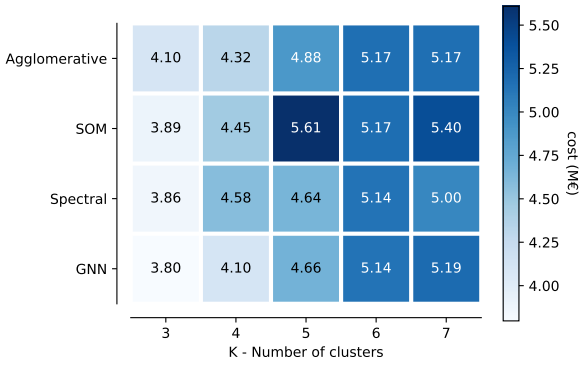
(b) Llindar 10



(c) Llindar 20

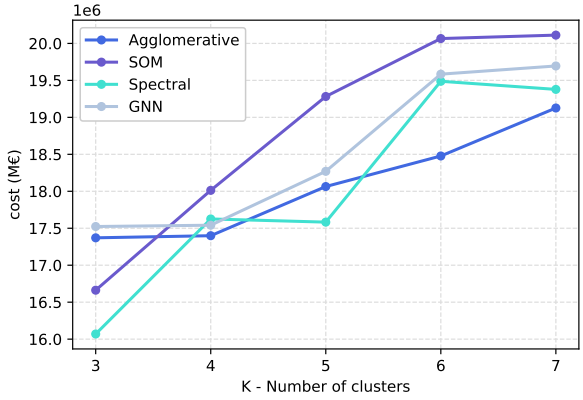


(d) Llindar 25

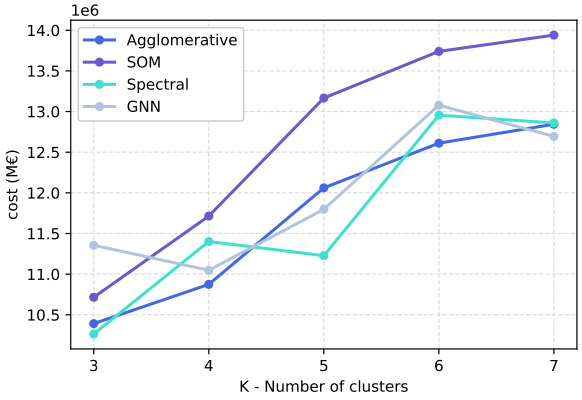


(e) Llindar 50

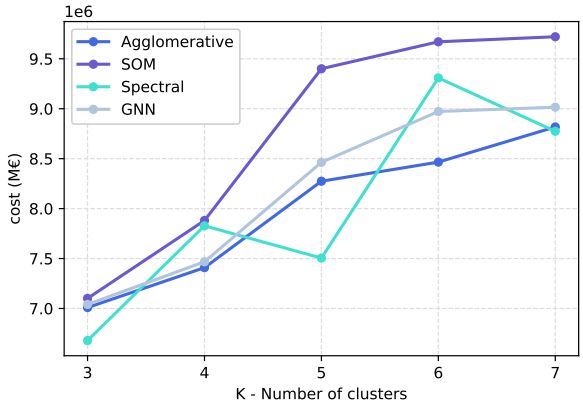
Figura 7.10: Mapa de calor dels costos de l'opció descentralitzada a Girona



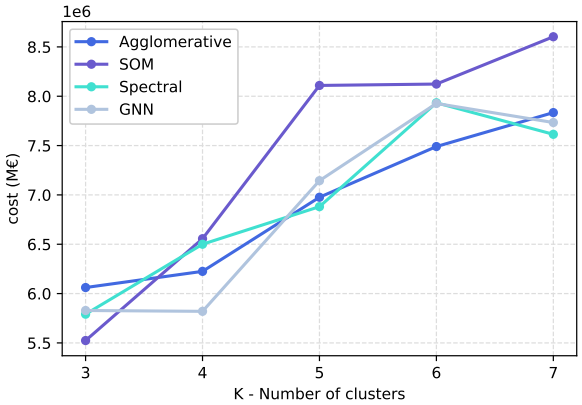
(a) Llindar 0



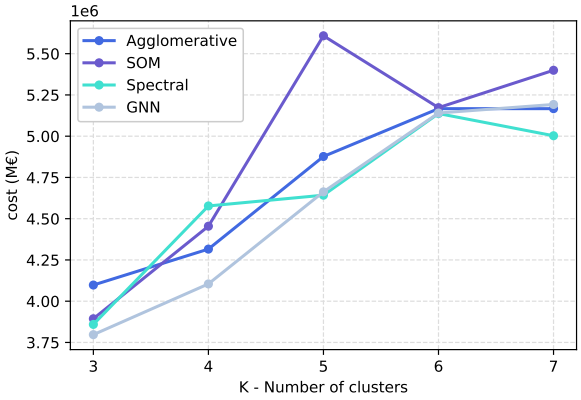
(b) Llindar 10



(c) Llindar 20

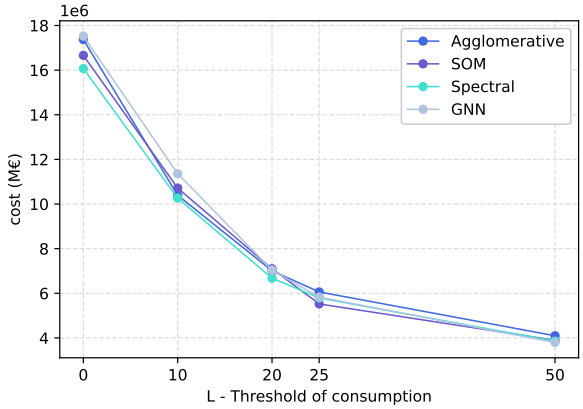


(d) Llindar 25

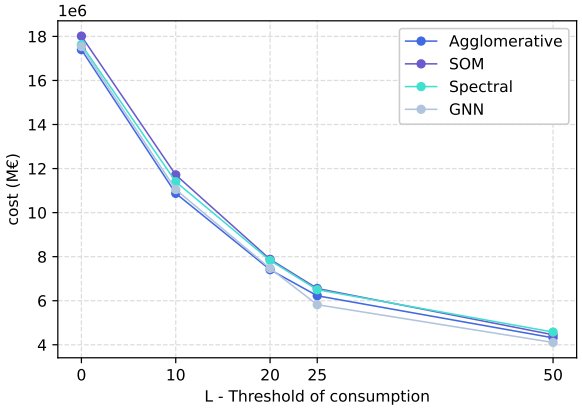


(e) Llindar 50

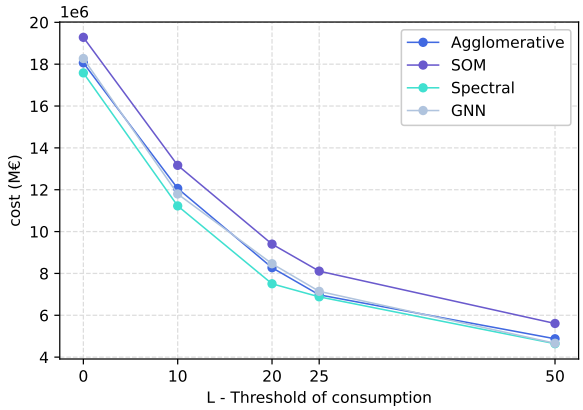
Figura 7.11: Gràfic de línies dels costos de l'opció descentralitzada a Girona segons k i agrupats per llindars de consum



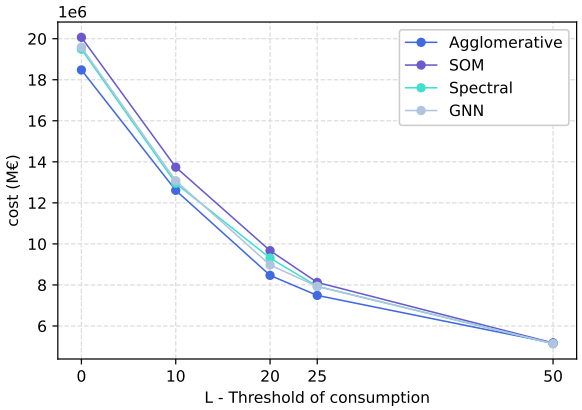
(a) $k = 3$



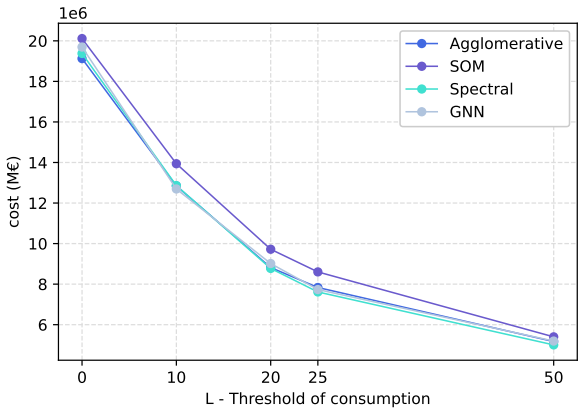
(b) $k = 4$



(c) $k = 5$

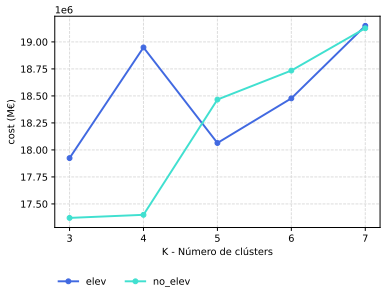


(d) $k = 6$

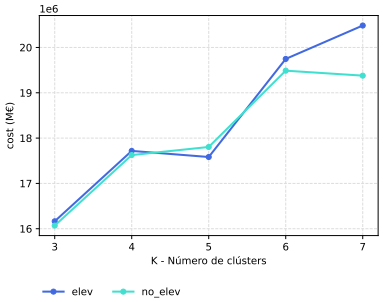


(e) $k = 7$

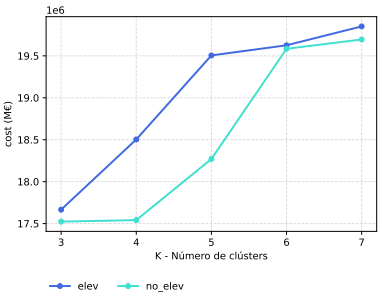
Figura 7.12: Gràfic de línies dels costos de l'opció descentralitzada a Girona segons llindar i agrupats per número de clústers k



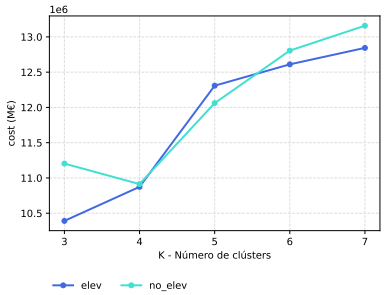
(a) Agglomerative - Llindar 0



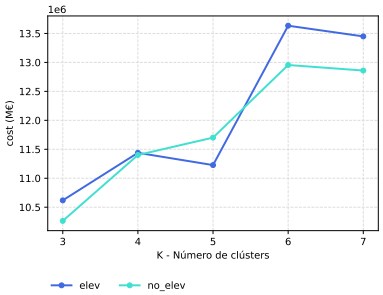
(b) Spectral - Llindar 0



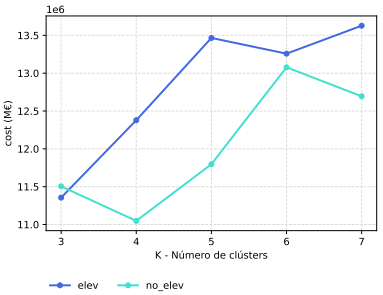
(c) GNN - Llindar 0



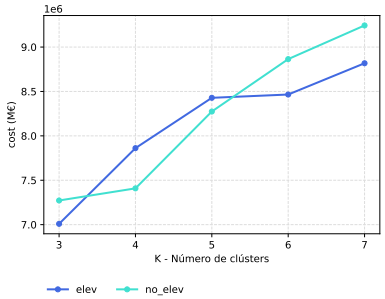
(d) Agglomerative - Llindar 10



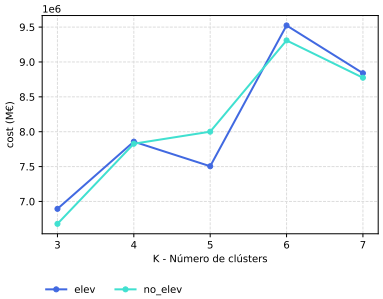
(e) Spectral - Llindar 10



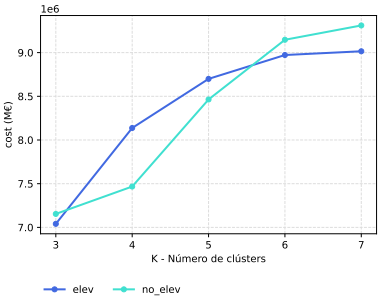
(f) GNN - Llindar 10



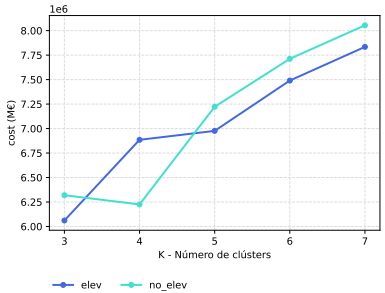
(g) Agglomerative - Llindar 20



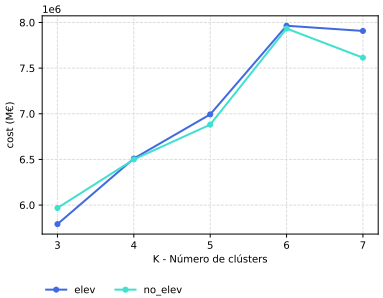
(h) Spectral - Llindar 20



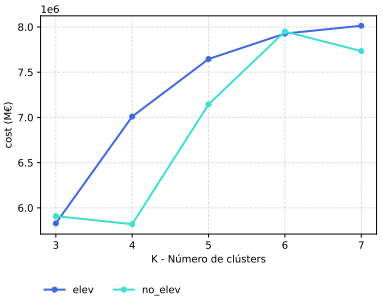
(i) GNN - Llindar 20



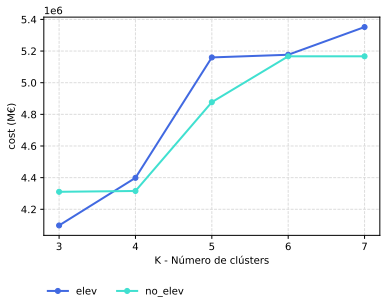
(j) Agglomerative - Llindar 25



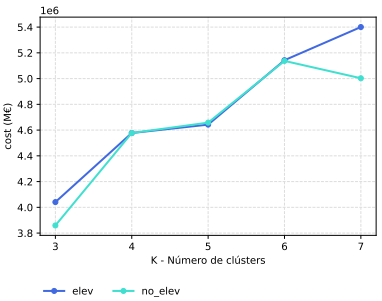
(k) Spectral - Llindar 25



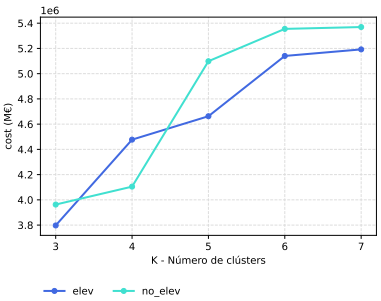
(l) GNN - Llindar 25



(m) Agglomerative - Llindar 50

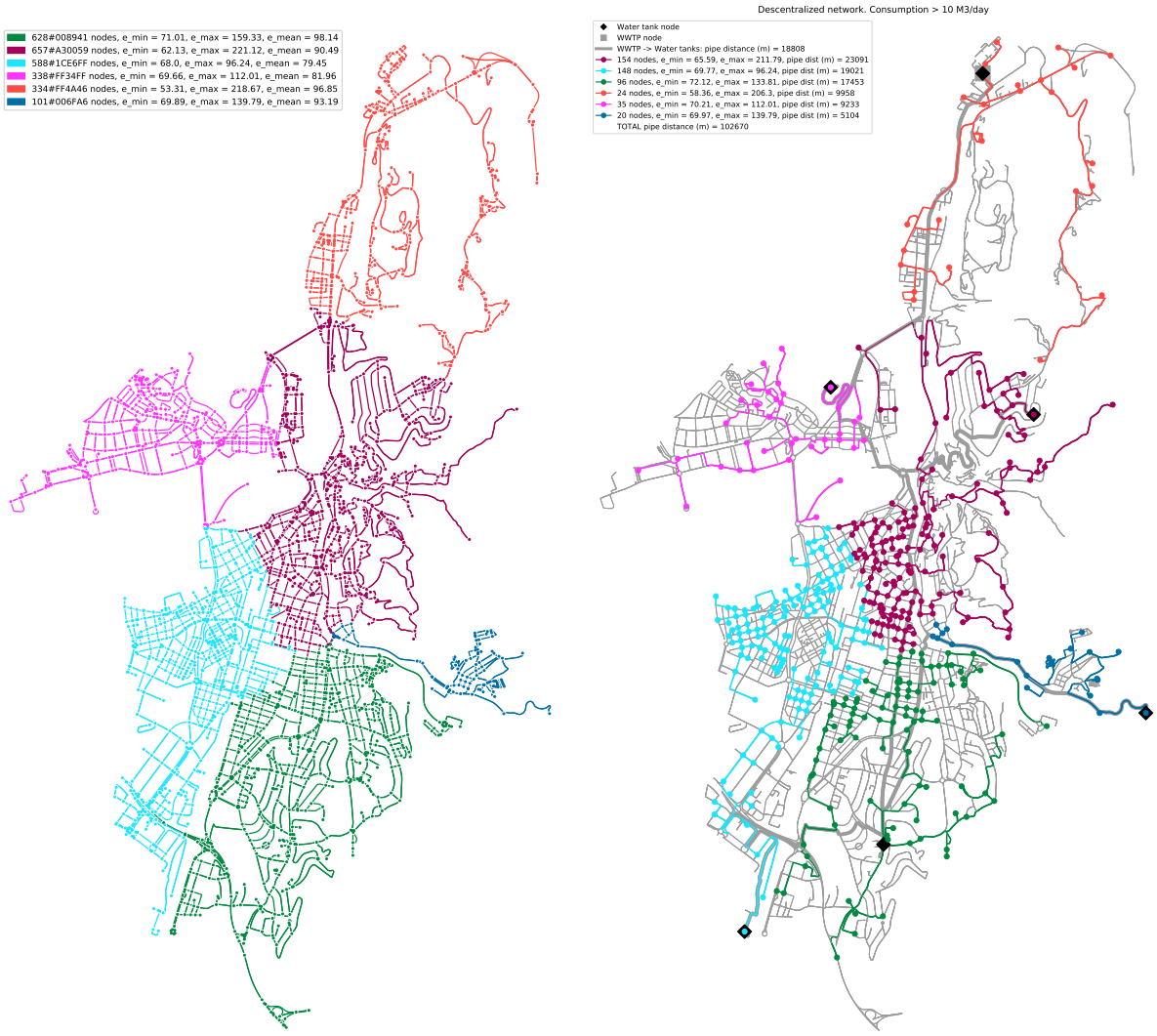


(n) Spectral - Llindar 50



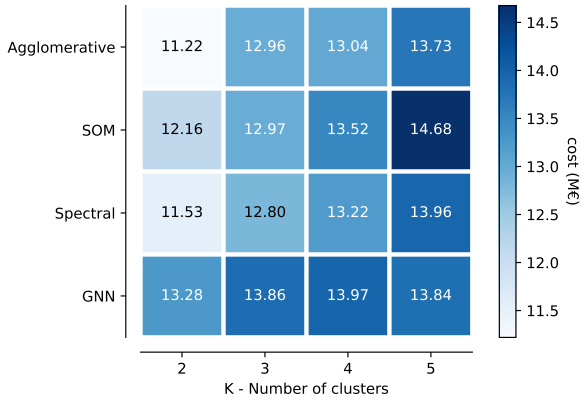
(o) GNN - Llindar 50

Figura 7.13: Gràfic de línies dels costos de l'opció descentralitzada a Girona agrupats per mateix algorisme però diferents *features* a l'hora de clusteritzar

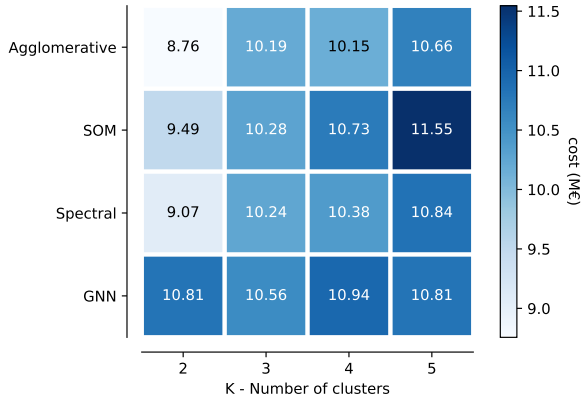


(a) Clusterització (b) Disseny xarxa descentralitzada

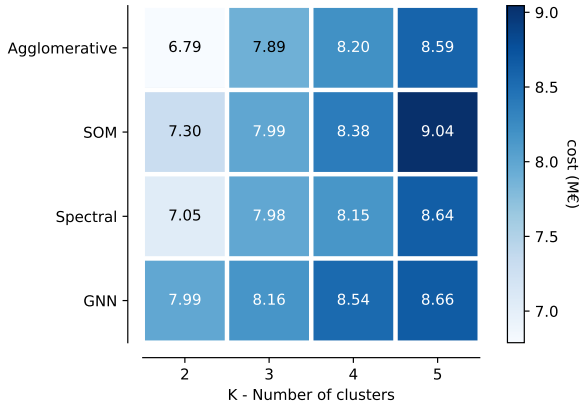
Figura 7.14: Exemple xarxa descentralitzada a Girona per $k = 6$ i llindar 10.



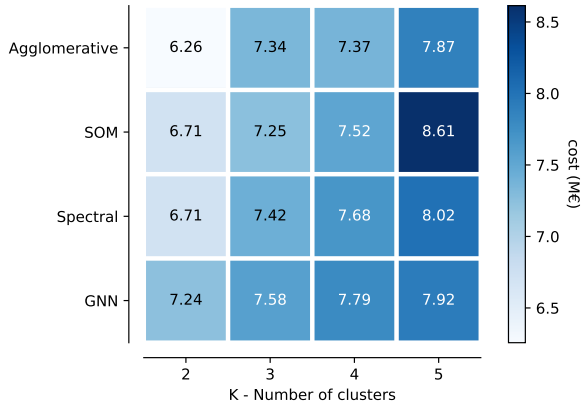
(a) Llindar 0



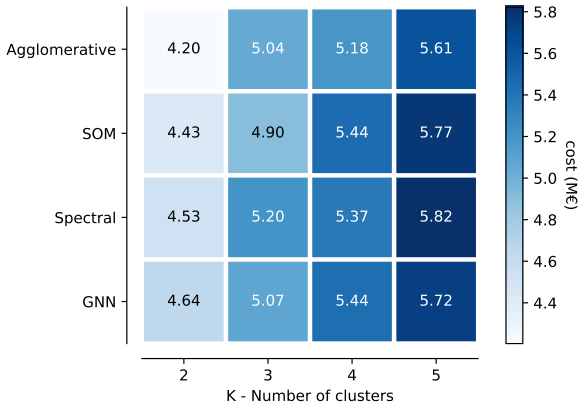
(b) Llindar 10



(c) Llindar 20

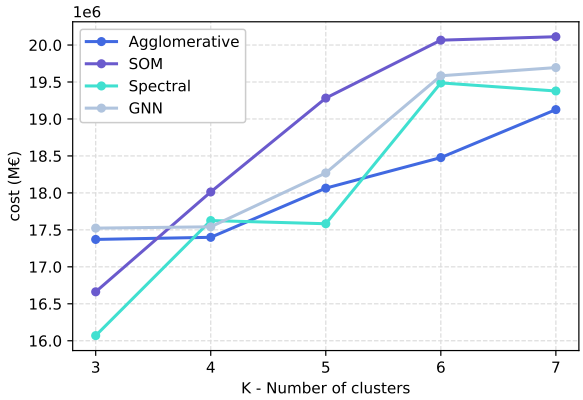


(d) Llindar 25

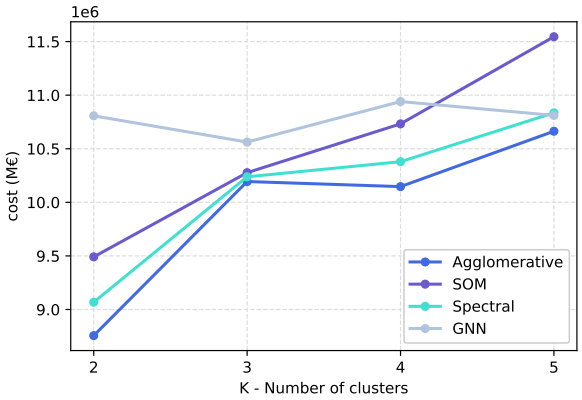


(e) Llindar 50

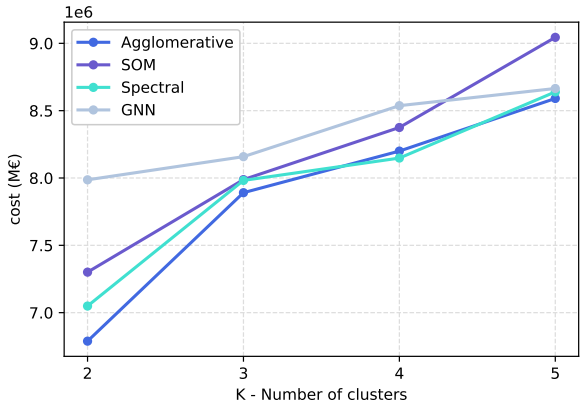
Figura 7.15: Mapa de calor dels costos de l'opció descentralitzada a Lloret de Mar



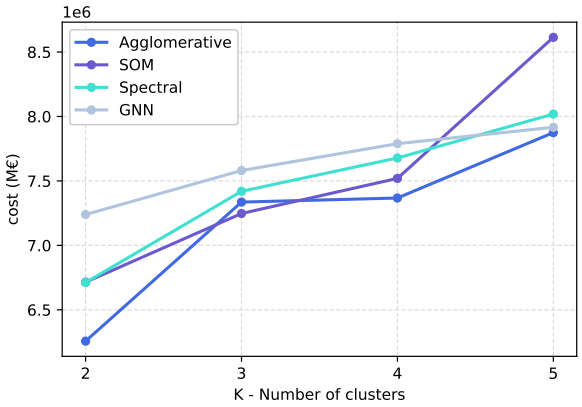
(a) Llindar 0



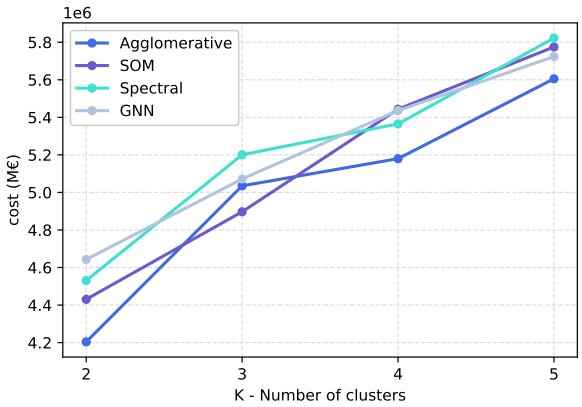
(b) Llindar 10



(c) Llindar 20

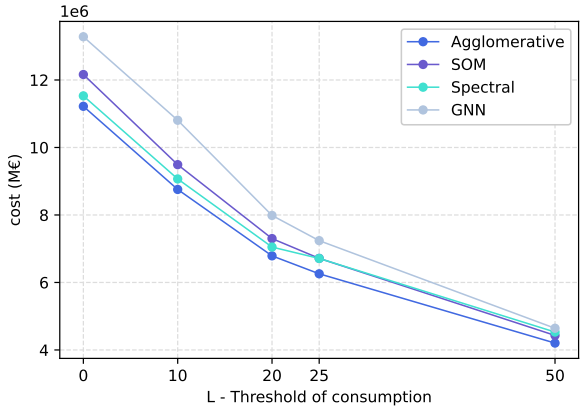


(d) Llindar 25

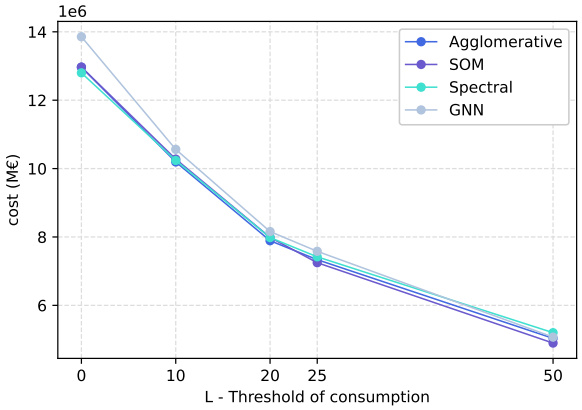


(e) Llindar 50

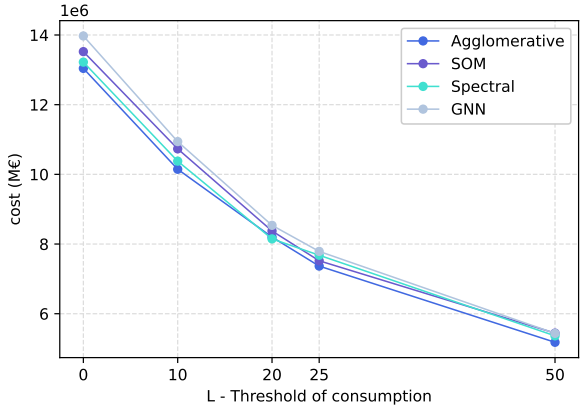
Figura 7.16: Gràfic de línies dels costos de l'opció descentralitzada a Lloret de Mar segons k i agrupats per llindars de consum



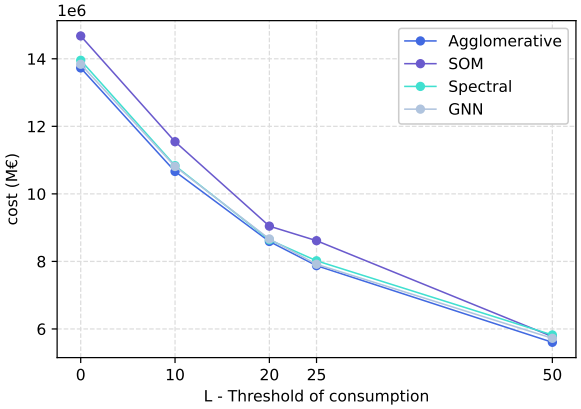
(a) $k = 2$



(b) $k = 3$

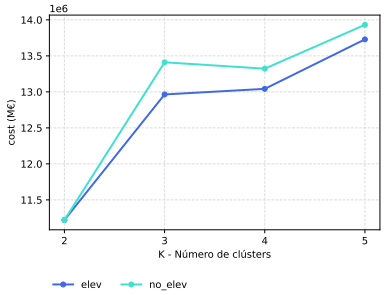


(c) $k = 4$

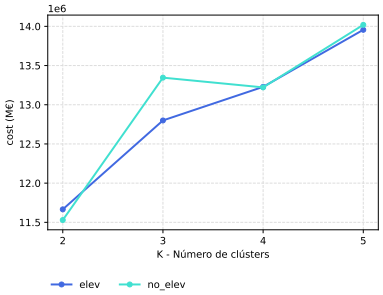


(d) $k = 5$

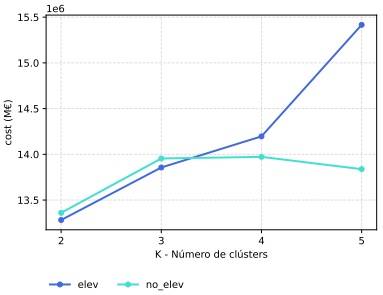
Figura 7.17: Gràfic de línies dels costos de l'opció descentralitzada a Lloret de Mar segons llindar i agrupats per número de clústers k



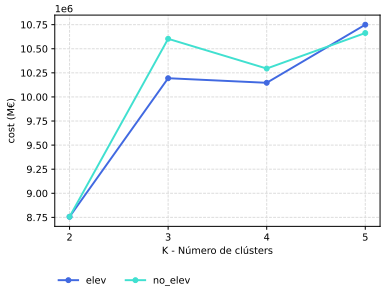
(a) Agglomerative - Llindar 0



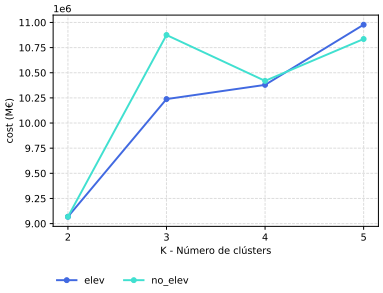
(b) Spectral - Llindar 0



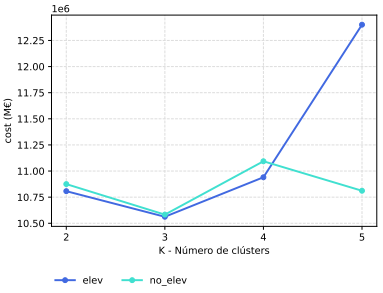
(c) GNN - Llindar 0



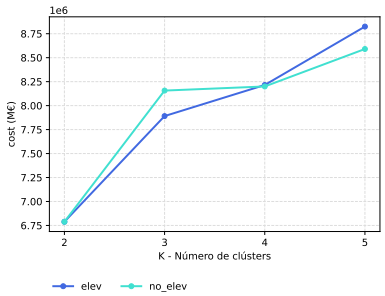
(d) Agglomerative - Llindar 10



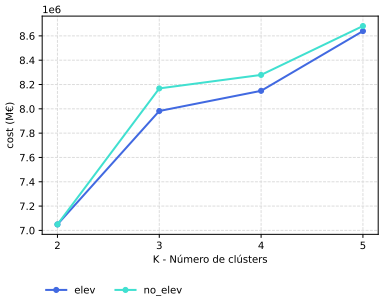
(e) Spectral - Llindar 10



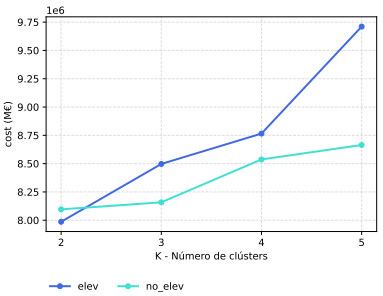
(f) GNN - Llindar 10



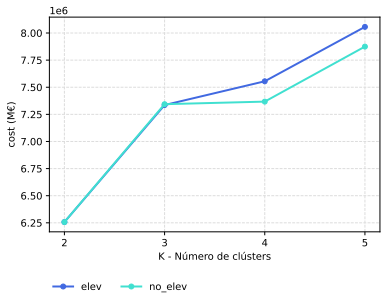
(g) Agglomerative - Llindar 20



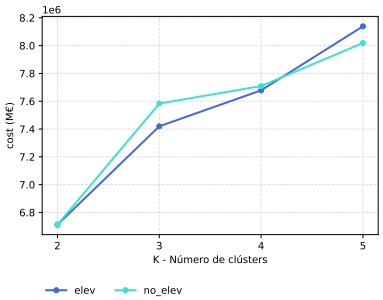
(h) Spectral - Llindar 20



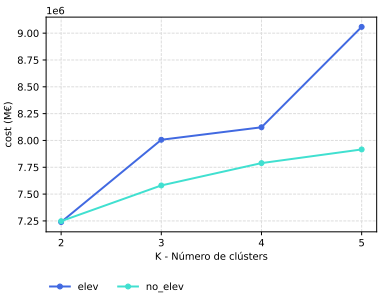
(i) GNN - Llindar 20



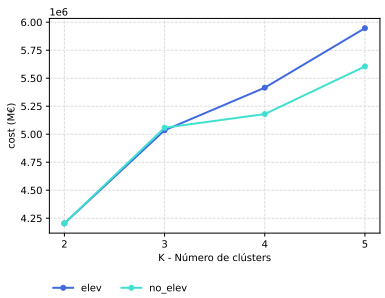
(j) Agglomerative - Llindar 25



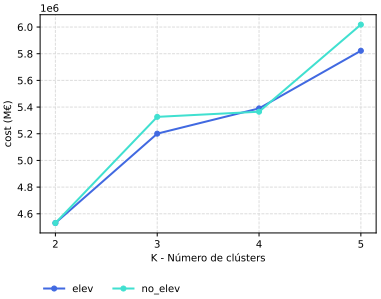
(k) Spectral - Llindar 25



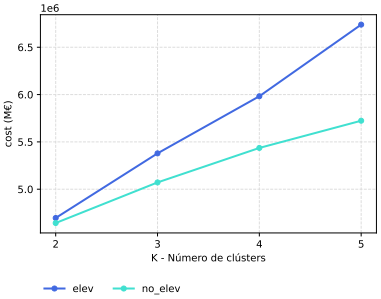
(l) GNN - Llindar 25



(m) Agglomerative - Llindar 50

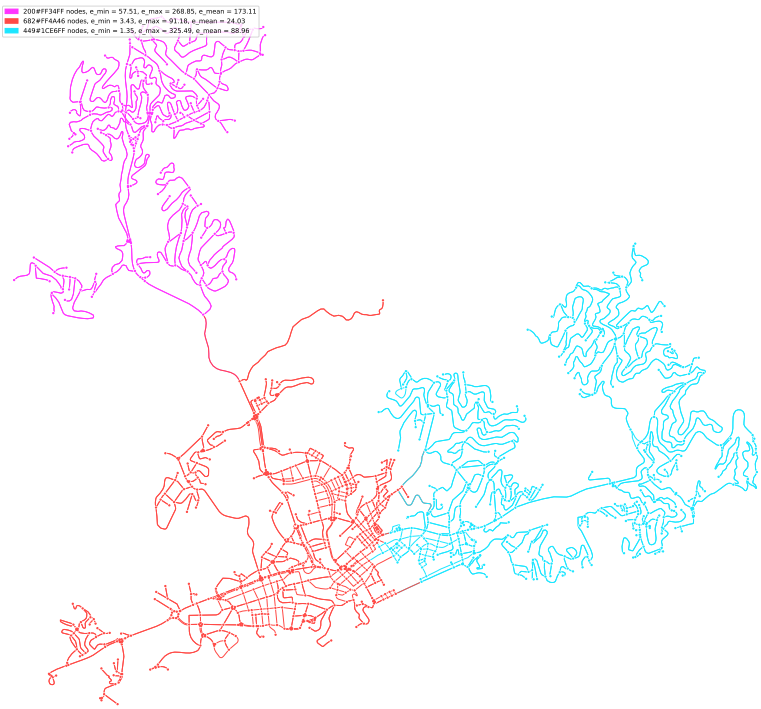


(n) Spectral - Llindar 50

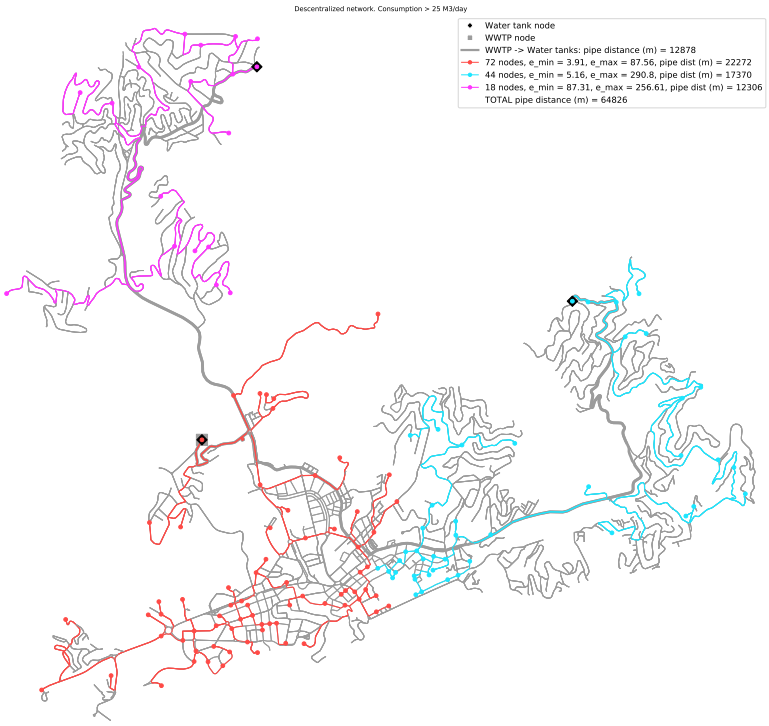


(o) GNN - Llindar 50

Figura 7.18: Gràfic de línies dels costos de l'opció descentralitzada a Lloret de Mar agrupats per mateix algorisme però diferents *features* a l'hora de clusteritzar

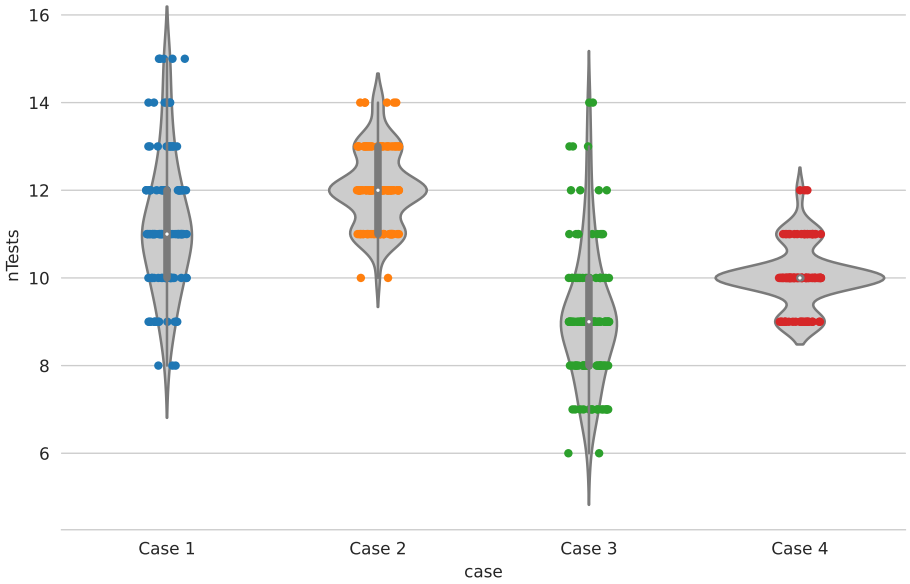


(a) Clusterització

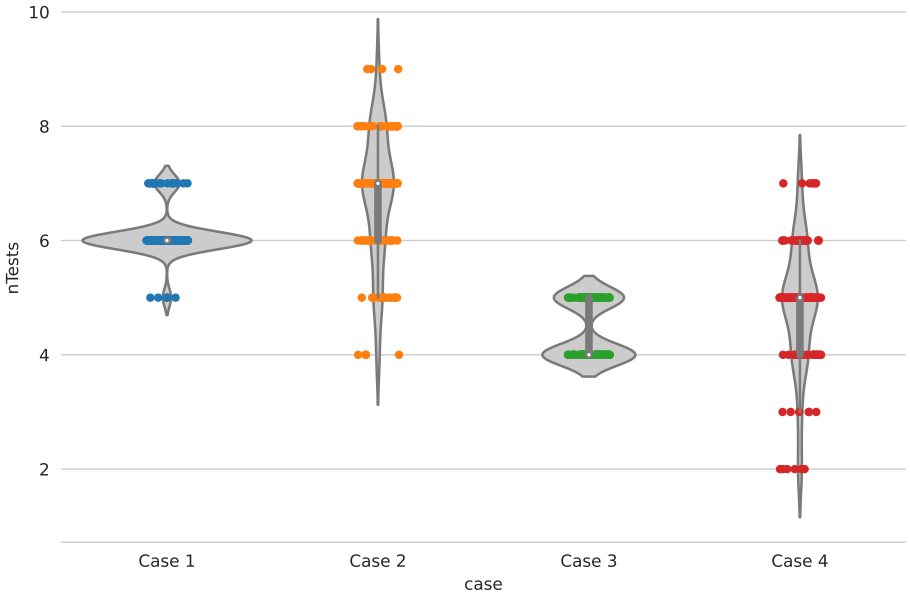


(b) Disseny xarxa descentralitzada

Figura 7.19: Exemple xarxa descentralitzada a Lloret de Mar per $k = 3$ i llindar 25.



(a) Algorisme Pacient Zero (PZ)



(b) Algorisme Hot Spot (HS)

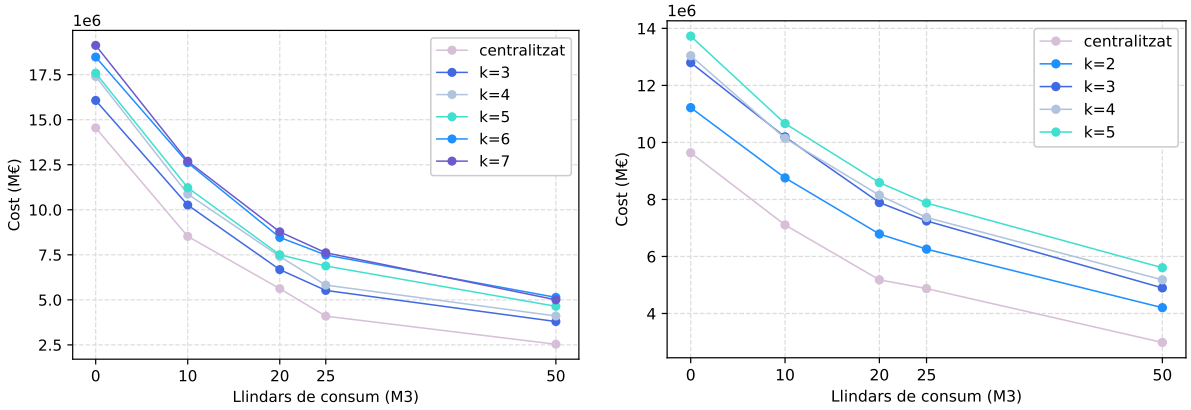
case 1 – tota la xarxa de Girona amb assignació de probabilitats Bayesianes en funció de la població.
case 2 – tota la xarxa de Girona amb assignació aleatòria de probabilitats Bayesianes.
case 3 – subgraf de la xarxa i probabilitats Bayesianes com case 1.
case 4 – subgraf de la xarxa i probabilitats Bayesianes com case 2.

Figura 7.20: Distribució de punts de mostreig requerits pel algorismes dinàmics.

8. Conclusions

De l'estudi que hem fet i amb els resultats obtinguts i el seu anàlisi, n'extraiem les següents conclusions:

- Tal com s'observa a la figura 8.1, tant per Girona com per Lloret i per qualsevol llindar de consum, l'opció de xarxa centralitzada sempre és més econòmica que la versió descentralitzada. Igualment, observem que en tots els casos, la tendència que segueixen els costos és pràcticament la mateixa. L'explicació que donem a aquest fet és que en l'opció descentralitzada hi ha uns costos extres respecte la centralitzada, que són els dels dipòsits. De totes maneres, l'opció centralitzada, malgrat ser més econòmica, no sempre es pot aplicar degut a que hi ha distàncies molt llargues i s'ha de bombejar a una pressió massa alta. Es poden veure els costos desglossats per Girona a la taula 8.1, i a la taula 8.2 per Lloret de Mar.



(a) Girona (b) Lloret de Mar

Figura 8.1: Costos opció centralitzada vs. descentralitzada

- Pel que fa als algorismes d'encaminament, hem vist que l'algorisme de Mehlhorn és molt més ràpid que el de Takahashi, però aquest últim dóna millors resultats (menor cost) en les proves implementades. Tot i que en el nostre cas la diferència de preu no és molt elevada, en d'altres pot ser més gran. De totes maneres, com veiem a la figura 7.1, la

Llindar de consum (M^8)	Cost xarxa (M€)					
	centr.	k=3	k=4	k=5	k=6	k=7
0	14.548	16.070	17.399	17.583	18.477	19.126
10	8.525	10.264	10.874	11.227	12.611	12.697
20	5.623	6.679	7.409	7.505	8.466	8.774
25	4.095	5.524	5.820	6.881	7.490	7.614
50	2.536	3.797	4.105	4.642	5.137	5.003

Taula 8.1: Costos opció centralitzada i descentralitzada per diferents k a Girona

Llindar de consum (M^8)	Cost xarxa (M€)				
	centr.	k=2	k=3	k=4	k=5
0	9.638	11.220	12.800	13.041	13.729
10	7.106	8.756	10.194	10.147	10.663
20	5.181	6.788	7.891	8.148	8.590
25	4.873	6.257	7.247	7.367	7.875
50	2.984	4.204	4.897	5.180	5.605

Taula 8.2: Costos opció centralitzada i descentralitzada per diferents k a Lloret de Mar

ràtio de l'algorisme de Mehlhorn és millor. Per tant, l'algorisme a utilitzar depèn de la implementació i la xarxa.

- Dels algorismes de clusterització no en podem treure una conclusió clara, més enllà de que els que millor funcionen i permeten dissenyar xarxes més econòmiques en la majoria de casos, són l'aglomeratiu (per Lloret de Mar) i l'espectral (per Girona). Aquestes dues ciutats presenten topologies força diferents, ja que Lloret està molt més escampada i descentralitzada en comparació a Girona. Caldria fer proves en d'altres ciutats de topologies similars a aquestes per poder determinar si hi ha un patró en aquest sentit.

- Quant als algorismes dinàmics implementats per determinar els punts de mostreig en aigües residuals, la millora dels algorismes implementats respecte aproximacions existents existeix, i es deu al fet de beneficiar-nos d'aplicar un primer posicionament estàtic de sensors, fet que permet analitzar només una part de la xarxa, així com pel fet d'utilitzar la informació dels habitants connectats als nodes alhora de seleccionar els punts.

- Hem vist que integrar les dades de diferents capes permet fer un anàlisi més exhaustiu del problema. Si bé per aquests dos projectes ho haguéssim pogut fer de forma manual, el fet d'automatitzar aquest procés d'obtenció i integració de dades, en facilitarà l'ús d'aquestes en projectes similars.

8.1 Dificultats

- Hi ha hagut un procés previ de recerca important, en el qual hem llegit diferents articles acadèmics, i no hi estem acostumats.
- Els objectius que ens vam proposar inicialment, i que estaven definits al full de projecte, eren molt oberts i poc concrets. Llavors, a l'hora de començar, no sabíem per on enfocar-ho.
- Malgrat tenim moltes dades com a resultats, ens és complicat saber si són correctes.
- Alguns dels criteris del projecte CLEaN-TOUR canviaven mentre estàvem fent el TFG i ens ha fet haver de tirar enrere en algun punt, o modificar els paràmetres d'alguna funció.

9. Treball futur

El projecte desenvolupat ha assolit els objectius plantejats. Tanmateix, hi ha aspectes que es podrien millorar i aspectes nous que es podrien realitzar en un futur. Així doncs, contemplem algunes possibles millores:

- Validar el disseny de la xarxa d'aigua regenerada, juntament amb els consums predits per cada destí, amb EPANET, per confirmar que tot el sistema és vàlid segons el pendent i pressions.
- Aplicar un model matemàtic per calcular la interacció i la dissolució de les traces de SARS-CoV-2 amb altres contaminants i partícules presents en l'aigua residual per posicionar els sensors estàtics d'una manera més correcta.
- Fer la clusterització a partir dels nodes destins, per tal de tenir uns millors resultats en casos on s'estableixi un llindar de consum alt i el número de nodes disminueix.
- Aplicar els algorismes de clusterització en altres ciutats per veure si es pot determinar que hi ha un algorisme que afavoreix dissenys descentralitzats més econòmics en la majoria de casos que la resta.
- Integrar els algorismes implementats al simulador de recerca de xarxes[1] del grup BCDS.

[1] https://nrs2.udg.edu/

10. Manual d'usuari

Per instal·lar totes les llibreries i material necessari per executar el codi, hem creat un entorn virtual (anomenat "tfg"), el qual és un ambient creat amb l'objectiu d'aïllar recursos com biblioteques i entorns d'execució per evitar conflictes.

Aquest manual està pensat per un SO Ubuntu on ja hi han instal·lats Python i Conda. L'ordre és el següent:

```
$ conda create --name tfg python=3.7
$ activate tfg
$ conda install gdal
$ conda install -c anaconda jupyter
$ conda install -c conda-forge owslib
$ conda install -c anaconda scikit-learn
$ conda install -c conda-forge rasterio
$ conda install -c conda-forge matplotlib
$ conda install -c anaconda networkx
$ conda install -c conda-forge osmnx
$ conda install -c conda-forge overpy
$ conda install -c conda-forge utm
$ conda install -c conda-forge rasterio
$ conda install -c conda-forge owslib
$ conda install -c conda-forge pandas
$ conda install -c conda-forge dbfread
$ conda install -c conda-forge pyproj
$ conda install -c conda-forge shapely
$ pip install MiniSom
$ pip install tensorflow
```

En el cas de la ciutat de Girona, els fitxers d'informació del Cadastre s'anomenen "GIRONA.CAT", "GIRONA-RUSTIC.CAT" "CONSTRU-Girona.DBF". L'script per tractar les dades (anomenat m2parcela) s'hauria de cridar de la següent manera:

```
$ python m2parcela GIRONA.CAT GIRONA-RUSTIC.CAT
  CONSTRU-Girona.DBF "nom fitxer sortida"
```

Bibliografia

[1] *Árbol recubridor mínimo*. URL: https://es.wikipedia.org/wiki/%C3%81rbol_recubridor_m%C3%ADnimo. (accessed: June 2021).

[2] Awan-Ur-Rahman. *Introduction to Ant colony optimization(ACO)*. URL: https://towardsdatascience.com/the-inspiration-of-an-ant-colony-optimization-f377568ea03f. (accessed: June 2021).

[3] *Bayesian probability*. URL: https://en.wikipedia.org/wiki/Bayesian_probability. (accessed: June 2021).

[4] *Clavegueram*. URL: https://ca.wikipedia.org/wiki/Clavegueram. (accessed: May 2021).

[5] Alysson M. Costa, Jean-François Cordeau, and Gilbert Laporte. "Fast heuristics for the Steiner tree problem with revenues, budget and hop constraints." In: *European Journal of Operational Research* 190.1 (2008), pp. 68–78. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2007.06.012. URL: https://www.sciencedirect.com/science/article/pii/S0377221707005516.

[6] *Eigenvalues and eigenvectors*. URL: https://en.wikipedia.org/wiki/Eigenvalues_and_eigenvectors. (accessed: June 2021).

[7] William Fleshman. *Spectral Clustering*. URL: https://towardsdatascience.com/spectral-clustering-aba2640c0d5b. (accessed: June 2021).

[8] Martín Gullón. *Xarxes de clavegueram*. URL: http://www3.udg.edu/publicacions/vell/electroniques/gestio_aigues_residuals/pag/Capitol2.htm. (accessed: May 2021).

[9] Shanon Hong. *An Introduction to Graph Neural Network(GNN) For Analysing Structured Data*. URL: https://bit.ly/2R9muTM. (accessed: May 2021).

[10] Xin Jin and Jiawei Han. "K-Means Clustering." In: *Encyclopedia of Machine Learning*. Ed. by Claude Sammut and Geoffrey I. Webb. Boston, MA: Springer US, 2010, pp. 563–564. ISBN: 978-0-387-30164-8. DOI: 10.1007/978-0-387-30164-8_425. URL: https://doi.org/10.1007/978-0-387-30164-8_425.

[11] David S. Johnson, Maria Minkoff, and Steven Phillips. "The Prize Collecting Steiner Tree Problem: Theory and Practice." In: *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '00. San Francisco, California, USA: Society for Industrial and Applied Mathematics, 2000, pp. 760–769. ISBN: 0898714532.

[12] David S. Johnson, Maria Minkoff, and Steven Phillips. "The Prize Collecting Steiner Tree Problem: Theory and Practice." In: *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '00. San Francisco, California, USA: Society for Industrial and Applied Mathematics, 2000, pp. 760–769. ISBN: 0898714532.

[13] L. Kou, G. Markowsky, and L. Berman. "A Fast Algorithm for Steiner Trees." In: *Acta Inf.* 15.2 (June 1981), pp. 141–145. ISSN: 0001-5903. DOI: 10.1007/BF00288961. URL: https://doi.org/10.1007/BF00288961.

[14] Richard C. Larson, Oded Berman, and Mehdi Nourinejad. "Sampling manholes to home in on SARS-CoV-2 infections." In: *PLoS ONE* 15 (2020). ISSN: 19326203. DOI: 10.1371/journal.pone.0240007.

[15] *Lesson 2. GIS in Python: Intro to Coordinate Reference Systems in Python.* URL: https://www.earthdatascience.org/courses/use-data-open-source-python/intro-vector-data-python/spatial-data-vector-shapefiles/intro-to-coordinate-reference-systems-python/. (accessed: June 2021).

[16] Kurt Mehlhorn. "A faster approximation algorithm for the Steiner problem in graphs." In: *Information Processing Letters* 27.3 (1988), pp. 125–128. ISSN: 0020-0190. DOI: https://doi.org/10.1016/0020-0190(88)90066-X. URL: https://www.sciencedirect.com/science/article/pii/002001908890066X.

[17] Amal Menzli. *Graph Neural Network and Some of GNN Applications.* URL: https://neptune.ai/blog/graph-neural-network-and-some-of-gnn-applications. (accessed: May 2021).

[18] Eliran Natan. *P vs. NP — What is the Difference Between Solving a Problem and Recognizing its Solution?* URL: https://www.cantorsparadise.com/p-vs-np-what-is-the-difference-between-solving-a-problem-and-recognizing-its-solution-921c4c0df561. (accessed: June 2021).

[19] A.Y. Ng, Michael Jordan, and Y Weiss. "On Spectral Clustering: Analysis and an Algorithm." In: vol. 2. Nov. 2001.

[20] Joan Nunes. *Sistema de referència espacial.* URL: https://www.icgc.cat/Ciutada/Informa-t/Diccionaris/Sistema-de-referencia-espacial#Base_de_dades_i_codificaci%C3%B3_EPSG. (accessed: June 2021).

[21] Prasad Pai. *Hierarchical clustering explained.* URL: https://towardsdatascience.com/hierarchical-clustering-explained-e59b13846da8. (accessed: May 2021).

[22] *Població de Girona.* URL: https://www.idescat.cat/emex/?id=170792. (accessed: May 2021).

[23] Markus Prossegger and Hamid Bouchachia. "Ant colony optimization for Steiner tree problems." In: Jan. 2008, pp. 331–336. DOI: 10.1145/1456223.1456292.

[24] *Self-organizing map.* URL: https://en.wikipedia.org/wiki/Self-organizing_map. (accessed: May 2021).

[25] *Steiner tree problem.* URL: https://en.wikipedia.org/wiki/Steiner_tree_problem. (accessed: June 2021).

[26] H. Takahashii and A. Matsuyama. "An approximate solution for the Steiner problem in graphs." In: *Math Japonica* (1980).

[27] Anton Tsitsulin et al. "Graph Clustering with Graph Neural Networks." In: *CoRR* abs/2006.16904 (2020). arXiv: 2006.16904. URL: https://arxiv.org/abs/2006.16904.

[28] Giuseppe Vettigli. *MiniSom: minimalistic and NumPy-based implementation of the Self Organizing Map.* 2018. URL: https://github.com/JustGlowing/minisom/.

[29] *What is an SRID?* URL: https://desktop.arcgis.com/en/arcmap/10.3/manage-data/using-sql-with-gdbs/what-is-an-srid.htm. (accessed: June 2021).

[30] Vasileios Zografos and Klas Nordberg. *Class lectures in Introductory Course on Spectral Clustering*. May 2012.