Universitat Politècnica de Catalunya

Facultat d'Informàtica de Barcelona

Pràctica de Cerca Local

Intel·ligència Artificial

Edgar Moreno Martínez
Laia Pomar Pallarès
Maria Prat Colomer

Q1 2021-2022

Índex

1 Introducció

Aquest document conté la documentació de la pràctica de cerca local de l'assignatura d'Intel·ligència Artificial. L'objectiu d'aquesta pràctica és resoldre un problema utilitzant algoritmes de cerca local i realitzar una sèrie d'experiments per avaluar els resultats obtinguts. El desenvolupament de la pràctica l'hem fet en Java, utilitzant les classes i funcionalitats de la llibreria d'algoritmes AIMA, juntament amb d'altres que hem implementat específicament per aquest projecte.

Aquesta pràctica planteja un problema sobre centres de distribució inspirat en una situació pràctica i real, que descriurem en detall més endavant. Utilitzarem els algoritmes de Hill Climbing i Simulated Annealing per obtenir solucions del problema donada una noció d'optimalitat. És important tenir en compte que no volem obtenir necessàriament la solució òptima, ja que en la majoria de casos és impossible de trobar-ne en un temps raonable. Per això, definirem funcions heurístiques que, a partir d'un estat inicial, ens permetin millorar la qualitat de la nostra solució fins a arribar a una situació satisfactòria.

Els processos dissenyats per resoldre el problema tenen un conjunt de paràmetres que poden ser modificats i afinats per variar-ne el comportament. Amb l'objectiu de trobar quin algoritme i quins valors pels paràmetres dels mètodes s'adeqüen més a diferents situacions, dissenyarem i realitzarem un conjunt d'experiments que ens permetin analitzar la qualitat de les solucions, el nombre de passos realitzats i el temps d'execució dels diferents escenaris.

2 Descripció del problema

El problema consisteix en planificar, diàriament, les rutes d'un conjunt de camions cisterna d'una companyia de distribució. Els camions han d'abastir un conjunt de gasolineres que tenen dipòsits de gasolina buits i que necessiten ser omplerts.

2.1 Enunciat del problema

La companyia de distribució opera en una àrea geogràfica que representarem amb un mapa. El mapa consisteix en una graella de $100 \times 100\text{km}^2$, on els elements del problema poden estar col·locats en les posicions amb coordenades enteres. Farem servir la distància de Manhattan per determinar les distàncies entre elements (i.e. la distància entre dos punts és la suma dels valors absoluts de les diferències de les seves coordenades). En aquesta àrea hi ha C centres de distribució i G gasolineres.

Una gasolinera disposa d'un cert nombre de dipòsits. Quan un dipòsit es buida, la gasolinera envia una petició a la companyia de distribució perquè el torni a emplenar. Una gasolinera pot enviar una petició per cada un dels seus dipòsits. Per tant, podem tenir més d'una petició pendent d'una mateixa gasolinera.

Un centre de distribució disposa d'un únic camió cisterna. Un camió cisterna té capacitat per omplir dos dipòsits abans d'haver de tornar al centre de distribució per ser emplenat de nou. Cada camió cisterna pot fer, com a màxim, V viatges i a cada viatge pot emplenar, com a màxim, 2 dipòsits. També pot fer un viatge on només s'empleni un dipòsit. Considerem que omplir un dipòsit o un camió cisterna és instantani. A més, un camió cisterna no pot fer més de K km en un dia. El cost de que un camió cisterna recorri un quilòmetre és R i voldrem minimitzar la distància recorreguda pels camions cisterna per tal de minimitzar el cost dels trajectes.

A l'inici del dia, la companyia de distribució disposa d'una llista de les peticions que encara no s'han atès. Cada petició no atesa l'ha feta una certa gasolinera el mateix dia o bé fa un cert nombre de dies. Si la companyia no atén una petició durant el dia, la petició segueix pendent el dia següent però se n'incrementa el nombre de dies que porta esperant ser atesa. La companyia de distribució cobra per cada dipòsit que emplena. Per una valor D fixat, el preu del dipòsit disminueix amb el nombre de dies que una petició espera ser atesa de la següent manera:

$$f(x) = \begin{cases} 1.02D & \text{si} \quad \text{dies} = 0 \\ (1 - \frac{2^{\text{dies}}}{100})D & \text{si} \quad \text{dies} > 0 \end{cases}$$

Si dies $= 0$, vol dir que la petició s'ha fet el mateix dia i que, per tant, fa poc que espera ser atesa.

Donada la disposició dels elements en el mapa i una llista de peticions pendents, volem trobar quina ruta ha de seguir cada camió cisterna. L'objectiu serà obtenir el major benefici econòmic. Per aquest motiu, cal considerar els diners que perdem quan deixem una petició pel dia següent i en baixa el seu preu.

Si pensem en la situació global, no podem determinar quina serà l'assignació òptima, ja que no sabem quines peticions rebrem el dia següent. Per això, considerarem que volem maximitzar el benefici econòmic del dia actual tenint en compte també la baixada de preu dels dipòsits no atesos i, més endavant, establirem mètriques que ens permetin qualificar la qualitat d'una assignació.

2.2 Dades del problema

Les dades que tenim del problema són el nombre i coordenades de les gasolineres, el nombre i coordenades dels centres de distribució, el valor de les constants i una llista de peticions pendents. Cada petició conté

la gasolinera que l'ha feta i quants dies fa que està pendent. Si no indiquem el contrari, considerarem $V = 5$, $K = 640$ km, $R = 2$ i $D = 1000$.

La infraestructura per la generació d'instàncies del problema utilitza una *seed* per generar les dades. Això ens serà útil a l'hora de fer els experiments i ens permetrà generar instàncies aleatòries donats els valors de les constants. A més a més, el generador d'instàncies assigna entre 0 i 3 peticions per gasolinera, així que una gasolinera tindrà com a molt 3 peticions pendents.

Les gasolineres i els centres de distribució que crea el generador d'instàncies estan col·locats aleatòriament en l'àrea geogràfica seguint una distribució uniforme. Això fa que pugui haver-hi elements que tinguin les mateixes coordenades, és a dir, no podem assumir que elements diferents estiguin situats en posicions diferents.

2.3 Anàlisi inicial del problema

Aquest problema compleix les característiques d'un problema de cerca local:

- Volem aconseguir una solució bona en un temps raonable. Descartem obtenir la solució òptima per dues raons. En primer lloc, hi ha moltes solucions possibles i explorar-les totes per poder trobar l'òptima no es pot fer en un temps raonable. Com que no coneixem un algorisme que ens permeti fer una exploració sistemàtica i trobar l'òptim, necessitem explorar diferents opcions i escollir la millor i això comporta un cost temporal molt elevat donades les dimensions de les instàncies del problema. Les funcions heurístiques que definim i volem optimitzar ens permetran retallar l'espai de cerca i descartar solucions que no cal explorar. En segon lloc, la noció d'optimalitat és, en part, arbitrària. Com que les peticions arriben periòdicament, pot ser que el que considerem l'òptim amb la informació que tenim en un determinat moment, no ho sigui quan arribin noves peticions. Això ens dona cert marge de llibertat a l'hora d'escollir una solució subòptima.

- Tenim uns criteris definits per avaluar la qualitat d'una solució i podem transformar-los en funcions heurístiques. Cal fixar-se en que no tenim una funció de cost, per tant, no hi ha un criteri global per determinar la qualitat d'una solució. Per això, no tenim una manera directa de comparar les diferents heurístiques, però definirem una funció de cost que ens servirà de guia quantitativa per valorar els resultats dels experiments. És important tenir en compte que la funció que definirem, que quantificarà el benefici econòmic, no determina la qualitat de la solució.

- Podem començar amb una solució inicial i millorar-la progressivament a través de diferents estratègies, que seran els operadors que ens permeten moure'ns entre solucions.

En aquest problema, l'espai d'estats està format per totes les possibles assignacions de peticions pendents als centres de distribució. Cada petició pendent es pot assignar a un centre de distribució considerant les restriccions del problema o deixar-la sense assignar. A més, una vegada s'han fet les assignacions, cada camió cisterna pot processar les que li corresponen amb moltes ordenacions diferents.

Podem aproximar la mida de l'espai de solucions. Cada centre pot atendre un màxim de $2V$ peticions, ja que pot fer un màxim de V viatges i, a cada viatge, atendre com a molt 2 peticions. Podem tenir un total de $3G$ peticions i en total tenim C centres. Per tant, sense tenir en compte les restriccions del problema i considerant que un centre també té l'opció de no completar tots els seus viatges, podríem tenir fins a un total de $(3G + 1)^{2VC}$ solucions diferents. Per tant, és evident que l'espai de solucions és molt gran i és infactible explorar totes les opcions. Per fer aquesta aproximació no hem tingut en compte que només un centre pot ser assignat a cada petició ni que no tots els centres poden atendre una certa petició. Tot i així, ens permet fer-nos una idea de la magnitud de l'espai de solucions.

3 Definició d'una funció de cost

Les heurístiques ens permeten aproximar la qualitat de les solucions, però no són una funció de cost. Per tal de valorar la qualitat de les solucions i tenir una mètrica per comparar diferents heurístiques, definirem la següent funció de cost, a la que anomenarem **benefici**. Aquesta funció ens permet quantificar el benefici econòmic obtingut durant el dia donada una assignació. Tot i no ser una funció de cost real, ens ajudarà a l'hora de dur a terme i analitzar els resultats dels experiments.

Definim el benefici com la suma del preu cobrat per servir els dipòsits atesos en l'assignació menys el cost dels quilòmetres recorreguts al servir les peticions. Podem escriure-ho de la següent forma:

$$\text{benefici} = \sum_{p \in P_A} \text{preu de } \mathbf{p} \text{ avui} - R \sum_{c \in Centres} \text{km}(\mathbf{c})$$

on P_A és el conjunt de peticions ateses, $Centres$ és el conjunt de centres de distribució, R és el cost per quilòmetre i km($\mathbf{c}$) són els quilòmetres recorreguts pel camió del centre $\mathbf{c}$.

És important ser conscient que el nostre objectiu no és maximitzar aquesta funció, és a dir, no és una funció de cost real. Només és una mètrica per comparar els diferents enfocaments. Per exemple, és evident que ens convé prioritzar les peticions antigues i que no volem obviar-les, ja que això ens està fent perdre diners.

4 Representació de l'estat

Una vegada entesa l'estructura del problema i els elements de l'estat cal pensar en com representar-ho per aconseguir una implementació eficient i satisfactòria. Queda clar a partir de la descripció del problema que hi ha dos elements que cal representar:

1. La ruta que realitzen els camions segons l'assignació de l'estat.
2. Els dipòsits servits de cada gasolinera segons l'assignació de l'estat.

Una alternativa possible seria guardar els viatges que fa cada camió i a partir d'aquesta informació, si és necessari, calcular quines peticions han sigut servides. Fins i tot, es podria guardar només informació sobre les gasolineres i a partir d'aquí deduir els viatges que fan els camions.

Després de considerar les diferents alternatives hem decidit representar tota la informació a la vegada, tant de les gasolineres com dels camions. La implementació ha d'assegurar que en tot moment les dues bandes representen el mateix estat. Aquesta decisió permet fer consultes sobre l'estat actual amb la menor complexitat possible, tant per generar successors com per calcular funcions heurístiques. La disminució de la complexitat temporal compensa l'augment de la complexitat espacial.

L'estat constarà de 3 parts:

- **Informació estàtica:** la distància entre tots els punts rellevants és independent de l'estat i consta com un camp estàtic de la classe `Estat` en forma de `array` 2-dimensional. Per altra banda, la informació de `Gasolineras` i `CentrosDistribucion` tampoc varia durant el problema i també consten com a elements estàtics.
- **Rutes:** per cada camió guardem la ruta que fa en l'estat actual. La informació està guardada en una `array` amb una classe `Ruta` per cada camió. Aquesta classe té la informació de les parades en l'ordre en què les recorre el camió utilitzant un `ArrayList` que és actualitzat dinàmicament. A més, guarda el número de quilòmetres totals que ha de recórrer el camió en aquesta ruta i el número de viatges que fa (sortides del centre) per poder comprovar les restriccions de l'enunciat en temps constant.
- **Gasolineres:** les gasolineres guarden quines de les seves peticions han sigut servides i quin camió ha servit cada petició en una `array`. A més, guarden el número total de peticions i quantes han sigut servides, de nou per poder fer consultes en temps constant.

Per tant, tenim que el cost espacial de aquesta representació és $\Theta((G+C)^2)$ per la part estàtica i $\Theta(CV+G)$ per la part dinàmica.

Per assegurar una representació eficient i amb un codi llegible hi han altres classes de suport com pot ser una classe `Coordenades`. Totes les classes, a més dels constructors i els *setters* i *getters* indispensables, consten també de les funcions necessàries per encapsular el funcionament i, més important, poder assegurar la correctesa de l'estat representat en cada moment.

5 Operadors

Després de tenir la representació de l'estat ben definida passem a construir els operadors per generar els successors d'un estat. Definirem un conjunt d'operadors bàsics i després els combinarem per tenir operadors més complexos. Fixem-nos que qualsevol estat és una solució vàlida.

5.1 Operadors bàsics

- **AFEGEIX(i)**: afegeix una parada a la ruta del camió cisterna del centre de distribució i de la següent manera:

  1. Si no pot fer més viatges o no pot recórrer més quilòmetres, retorna al centre de distribució i s'atura. Si té la cisterna buida, retorna al centre de distribució i torna a començar el pas 1.
  2. Busca la gasolinera més propera a la posició actual amb una petició pendent.
  3. Si pot fer el trajecte a aquesta gasolinera (tenint en compte que ha de poder tornar al centre de distribució), el fa i omple el dipòsit amb la petició més antiga de la gasolinera.

- **ESBORRA(i)**: esborra l'última parada de la ruta del camió cisterna del centre de distribució i.

- **EMPLENA(i)**: emplena la ruta del camió cisterna del centre de distribució i fent **AFEGEIX(i)** successius fins que el camió s'atura.

- **BUIDA(i)**: buida tota la ruta del camió cisterna del centre de distribució i.

Cal observar que un camió cisterna sempre farà dos parades per viatge si és possible. Podríem argumentar que, en alguns casos, convé donar l'opció de fer una sola parada per viatge. No obstant, amb els paràmetres amb els que treballarem i tenint en compte que hi ha una restricció en el nombre de viatges que pot fer un camió, el benefici econòmic seria menor si un camió només fes una parada quan en podria fer dos.

5.2 Operadors formats per combinacions d'operadors bàsics

- **COMBINA(i, j)**: és equivalent a la seqüència d'operadors bàsics **BUIDA(i)** + **BUIDA(j)** + **EMPLENA(i)** + **EMPLENA(j)**. És a dir, esborra les rutes dels camions cisterna dels centres i i j i després les emplena. Aquest operador farà que, en certs casos, dos camions es redistribueixin les seves assignacions d'una forma més òptima. Després d'esborrar les rutes, la nova ruta assignada al camió i serà la que minimitza la distància que ha de recórrer tot maximitzant el nombre de dipòsits servits pel conjunt actualitzat de peticions pendents. Com que utilitzarem els operadors **COMBINA(i, j)** i **COMBINA(j, i)** per cada parella de camions, no donarem prioritat als camions amb menor índex a l'hora d'optimitzar les rutes individuals.

- **SERVEIX(N)**: obliga a atendre les N peticions no ateses més antigues. Per fer això, busquem la petició més antiga no atesa i fem que el camió cisterna més proper a la gasolinera de la petició la serveixi. Si és necessari, afegim una parada al centre de distribució o eliminem les últimes parades de la ruta del camió fins que la pugui servir. Un cop fet això per una petició, repetim el procés N vegades. Observem que aquest operador pot deixar rutes no acabades, és a dir, que podrien emplenar-se amb més viatges.

5.3 Justificació dels operadors escollits

A continuació, justifiquem l'elecció dels operadors que hem descrit abans:

- Els operadors ens permeten transformar una possible solució en una altra. En aquest problema, les solucions són les assignacions de peticions a centres de distribució. Per tant, té sentit que els operadors modifiquin les rutes dels centres de distribució. Les dues operacions bàsiques sobre una ruta són afegir i eliminar una parada. Tots els operadors s'han de basar en aquestes dues operacions i es diferenciaran en quin criteri utilitzen per esborrar o afegir parades.

- Com que volem minimitzar la distància recorreguda pels camions i volem prioritzar les peticions antigues, hem fet que l'operació bàsica d'afegir una parada a una ruta esculli la petició pendent que recorre menys quilòmetres i, en cas d'empat, escull la petició més antiga.

- A l'hora de pensar en els operadors, ens vam plantejar l'opció d'enfocar-ho des de les gasolineres enlloc de des dels centres. És a dir, ara generem els successors modificant les rutes dels centres. Una alternativa seria, per cada gasolinera, canviar els camions que les atenen. L'avantatge de fer-ho des dels centres és que és més natural i ens permet tenir rutes més eficients, ja que l'ordre de les peticions minimitza la distància recorreguda. A més a més, hem definit l'operador **SERVEIX(N)** per tal de tenir successors a partir de les peticions no ateses de les gasolineres.

5.4 Estudi del factor de ramificació dels operadors

A la Taula 1 s'indica el factor de ramificació de cada operador. El factor de ramificació dels 4 primers és el nombre de centres, ja que els nodes fills s'obtenen a partir d'aplicar cada operador a un dels C centres. El factor de ramificació de l'operador **COMBINA** és $C(C-1)$, ja que per cada centre, apliquem l'operador per la parella formada pel centre i per cada centre diferent a ell mateix. Observem que el factor no és $\binom{C}{2}$ sinó $2\binom{C}{2}$, ja que per cada parella de centres diferents, considerem els dos ordres possibles de combinació **COMBINA(i, j)** i **COMBINA(j, i)**. Finalment, per l'operador **SERVEIX(N)** només es genera un successor.

Operador	Factor de ramificació
AFEGEIX	C
ESBORRA	C
EMPLENA	C
BUIDA	C
COMBINA	$C(C-1)$
SERVEIX(N)	1

Taula 1: Factor de ramificació de cada operador.

5.5 Successors per a Simulated Annealing

Per un correcte funcionament de l'algorisme de Simulated Annealing hem de tenir successors de diferents tipus. Cada tipus de successors tindrà la mateixa probabilitat de ser representat, i un cop escollit el tipus s'agafarà un successor d'aquest tipus de manera aleatòria. Hem escollit 6 tipus de successors formats per combinacions dels operadors que hem definit:

- **COMBINA(i, j)**

- **COMBINA(i, j) + SERVEIX(N=10)**

- **BUIDA(i) + BUIDA(j) + AFEGEIX(i) + AFEGEIX(j)**
- **BUIDA(i) + BUIDA(j) + AFEGEIX(i) + AFEGEIX(j) + SERVEIX(N=10)**
- **AFEGEIX(i) + AFEGEIX(j)**
- **AFEGEIX(i) + AFEGEIX(j) + SERVEIX(N=10)**

La varietat de tipus de successors ens permetrà accedir a diferents nodes de l'espai de solucions aprofitant les característiques de l'algorisme de Simulated Annealing.

6 Descripció de les heurístiques

En la descripció del problema hem explicat que el nostre objectiu és trobar una assignació que maximitzi el benefici econòmic que obtenim en el dia actual tenint en compte les pèrdues que ens causa no atendre una petició.

Per tal d'aconseguir això hem definit diverses heurístiques i les hem analitzat per veure quina ens dona millor resultat.

6.1 Heurística 1

La primera heurística consisteix en maximitzar la suma del benefici menys les pèrdues que causa no atendre una petició avui. Aquesta heurística descriu el criteri de qualitat definit a l'enunciat de la pràctica. La fórmula és la següent:

$$\max \left(\sum_{p \in P_A} \text{preu de } \mathbf{p} \text{ avui} - \sum_{c \in Centres} R \cdot \text{km}(\mathbf{c}) - \sum_{p \in P \setminus P_A} (\text{preu de } \mathbf{p} \text{ avui} - \text{preu } \mathbf{p} \text{ demà}) \right)$$

on P és el conjunt de peticions, P_A el conjunt de peticions ateses, km($\mathbf{c}$) són els quilòmetres recorreguts pel camió del centre $\mathbf{c}$.

6.2 Heurística 2

Aquesta heurística simplement maximitza el nombre de dipòsits servits en una assignació.

$$\max \ \#\{\text{dipòsits servits}\}$$

6.3 Heurística 3

Una altra de les heurístiques que hem considerat és minimitzar el ràtio entre els quilòmetres recorreguts per cadascun dels camions i els dipòsits servits.

$$\min \ \frac{\sum_{c \in Centres} \text{km}(\mathbf{c})}{\#\{\text{dipòsits servits}\}}$$

on km($\mathbf{c}$) són els quilòmetres recorreguts pel camió del centre $\mathbf{c}$.

6.4 Heurística 4

La quarta heurística que hem definit minimitza el percentatge perdut al no atendre una petició durant el dia actual.

$$\min \sum_{p \in P_{NA}} (\text{percentatge de } \mathbf{p} \text{ avui} - \text{percentatge de } \mathbf{p} \text{ demà})$$

on P_{NA} és el conjunt de peticions no ateses.

6.5 Anàlisi incial de les heurístiques

Primer de tot cal que contemplem la possibilitat de ponderar alguns dels elements de les heurístiques. Està clar que en el cas de les heurístiques 2, 3 i 4 no té sentit afegir ponderacions ja que la 3 es tracta d'un ràtio, i tant en la 2 com en la 4 només hi actua un element. Pel que fa a la primera heurística, hem considerat que la ponderació necessària ja ve donada pel preu de cada element (en el cas de les peticions

pel valor que té un diposit, i en el dels quilometres pel cost que té recórrer un quilòmetre), i per tant, ja que tots els elements estan en la mateixa unitat no té massa sentit donar-li una ponderació especial a cap d'ells.

Analitzant aquestes heurístiques i tenint en compte quins són els criteris en que ens basem per avaluar la qualitat d'una assignació, veiem que el principal problema de les funcions heurístiques 2 i 3 és que no tenen en compte les pèrdues que provoca no atendre una petició en el dia actual. Això fa que prioritzi assignacions on es serveixen només les peticions més recents ja que són les que tenen el preu més alt, i per tant les pèrdues augmenten, ja que com més dies porta pendent una assignació més ràpid baixa el seu preu.

Pel que fa a l'heurística 4 el problema és el contrari, només té en compte els diners que es perden en no atendre una assignació avui, per tant l'assignació òptima segons aquest criteri serà aquella que ompli els dipòsits més antics, i no tindrà en compte el cost dels quilòmetres recorreguts pels camions.

En canvi la primera heurística té en compte tant el benefici com les pèrdues de les peticions no ateses, per tant, creiem que és la millor opció per avaluar la qualitat de les assignacions.

7 Estat inicial

Volem solucionar el problema amb algoritmes de cerca local. Això porta implícit que al començar a executar l'algorisme no és desitjable que ens trobem en un estat molt llunyà d'un possible òptim local. L'algorisme serà capaç de trobar òptims sigui quin sigui l'estat inicial proveït, però si aquest estat està aprop d'un òptim, el número d'iteracions serà perceptiblement menor. Considerant que cada iteració de l'algorisme pot tenir una complexitat elevada, aconseguir un estat inicial bo pot reduir enormement el temps d'execució.

La forma de generar aquest estat inicial tampoc pot tenir una complexitat massa elevada, ja que el que guanyem per una banda ho podríem perdre per l'altra. Així doncs, proposarem un algorisme voraç que generi un estat inicial ràpidament. També experimentarem amb un estat inicial buit per veure els possibles beneficis que podem obtenir.

7.1 Estat inicial buit

La primera opció és que l'estat inicial sigui l'estat buit. És a dir, cap de les peticions està atesa i cap camió cisterna s'ha mogut del seu centre de distribució. L'avantatge d'aquesta estratègia és que és molt poc costosa, ja que només ha d'inicialitzar les diferents classes que formen l'estat del problema. No obstant, necessitarà realitzar més passos per arribar a la solució final ja que estarà lluny de qualsevol òptim local.

7.2 Estat inicial generat amb un algorisme voraç

L'avantatge d'utilitzar una solució inicial bona és que no haurem de realitzar tants passos per arribar a la solució final. No obstant, això també vol dir que hi ha poc marge de millora i que és possible que les assignacions que s'han fet no siguin les millors i això faci que no s'arribi a solucions tan bones.

Anem a veure l'algorisme decidit i una justificació d'aquest.

Recordem que tenim dos objectius a la vegada: servir el màxim número de gasolineres gastant el mínim possible en gasolina. També és convenient prioritzar les peticions que portin més dies actives.

A priori sembla que volem assegurar-nos que succeeixin els viatges entre gasolineres i centres de distribució molt propers. Això no ens assegura que aquesta sigui la distribució que minimitza la distància per servir els centres, ni que es prioritzaran les peticions més antigues, però hauria de ser una solució prou bona.

Per aconseguir això dissenyarem un algorisme iteratiu que construeixi la solució pas a pas. Donat un estat parcial de l'algorisme, podem decidir quin serà el següent viatge computant totes les distàncies entre camions i gasolineres. Donada la parella que minimitza aquesta distància, enviarem aquest camió a aquesta gasolinera. Això tindria un cost temporal de $\mathcal{O}(\#\text{iteracions} \cdot G \cdot C) = \mathcal{O}(V \cdot C \cdot G \cdot C) = \mathcal{O}(C^2GV)$ on $C = \#\text{camions}, G = \#\text{gasolineres}, V = \#\text{viatges per camió}$.

Podem millorar aquesta complexitat evitant recalcular la distància mínima. Per aconseguir això podem tenir una cua de prioritat on, en tot moment, tenim la distància mínima a una gasolinera per cada centre.

Per tant, en cada iteració hem de extreure el mínim element de la cua de prioritat, fer el viatge indicat i recalcular la distància mínima entre aquest camió i una gasolinera qualsevol. Això només no ho podrem fer si la gasolinera desitjada ja ha sigut emplenada. No obstant, això només pot passar GC cops. Per tant, haurem de fer un màxim de $VC+GC$ iteracions. A cada iteració hem de visitar totes les gasolineres per recalcular la distància. Per tant tenim un algorisme que funciona en temps $\mathcal{O}(G(VC + GC))$.

A primera vista pot semblar que el factor G^2C és pitjor que el factor C^2GV de l'algorisme anterior, ja que sabem que $G \gg C, V$. Calcular la complexitat teòrica de l'algorisme és una tasca molt difícil i ens ajudarem d'un experiment per mesurar-la. Com podem veure en el següent gràfic, a la pràctica el factor

del que parlàvem no apareix, ja que l'aproximació que hem fet és bastant barroera. Per tant, hem vist experimentalment que aquesta segona versió té un temps d'execució de $\mathcal{O}(GVC)$, clarament millor que l'anterior $\mathcal{O}(C^2GV)$.

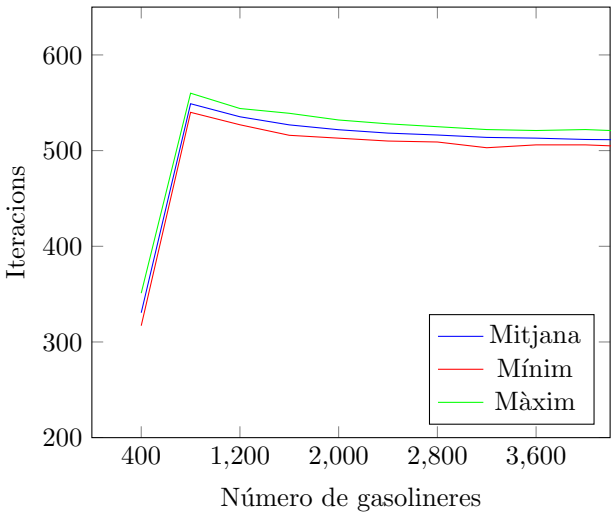

Figura 1: Iteracions de l'algorisme voraç segons el nombre de gasolineres. Per a cada nombre de gasolineres s'han fet 100 experiments i el nombre de centres estava fixat a 100. Podem observar com, excepte pels casos més petits, el nombre d'iteracions no depèn del nombre de gasolineres si fixem el nombre de centres. A més, podem observar una variància molt petita ja que no hi ha *outliers*.

8 Experiments

A continuació presentem els experiments que hem realitzat i que ens permeten analitzar les solucions que trobem.

8.1 Experiment 1

El primer experiment consisteix en determinar quin conjunt d'operadors dóna millors resultats per una funció heurística que optimitza el criteri de qualitat del problema. Per les raons que hem explicat a la descripció de les heurístiques, l'heurística escollida és l'heurística 1. Per realitzar aquest experiment, considerarem un escenari amb $C = 10$ i $G = 100$, és a dir, tenim 10 centres de distribució i 100 gasolineres i utilitzarem l'algorisme de Hill Climbing. Utilitzarem un estat inicial buit. En la Taula 2 es recullen els valors dels paràmetres que mantindrem constants. Els paràmetres que no apareixen a la taula tenen el valor per defecte descrit a la descripció de les dades del problema.

Paràmetre	**Valor**
C	10
G	100
Algorisme	Hill Climbing
Generació de l'estat inicial	Buit
Heurística	1

Taula 2: Paràmetres constants de l'Experiment 1

Per escollir el conjunt d'operadors a utilitzar farem servir l'estat inicial buit. Com que en un cert punt de l'algorisme els estats que considerem estaran completament plens, ens permetrà valorar els operadors tant per estats inicials incomplerts com per estats inicials complerts. La diferència estarà en el nombre de passos per arribar a la solució, ja que partint d'un estat inicial buit s'han d'omplir totes les rutes.

En la Taula 3 es recull la descripció de l'experiment. Els escenaris que estudiarem són combinacions lògiques dels operadors definits. Els operadors **AFEGEIX(i)** i **ESBORRA(i)** sempre apareixen junts perquè **ESBORRA(i)** no té sentit per si sol i la combinació ja inclou a **AFEGEIX(i)**. Les mesures del temps no es poden comparar entre experiments, ja que s'han realitzat en ordinadors diferents.

Per tots els escenaris, prendrem $N = 10$. Hem provat diferents valors i aquest és el que ens dóna millors resultats. No considerem prou rellevant incloure un estudi detallat de l'elecció del valor de N, ja que s'hauria de fer per cada combinació d'operadors i complicaria innecessàriament l'anàlisi.

La nostra hipòtesi és que els escenaris ens portaran a resultats bastant semblants, ja que **COMBINA(i, j)** és una combinació de **AFEGEIX(i)** i **ESBORRA(i)** i l'únic que canvia és l'ordre en el que es fan les assignacions dels centres. Per altra banda, creiem que **SERVEIX(N)** no millorarà molt les solucions perquè el possible avantatge es veurà compensat pel fet que les rutes no seran tan eficients.

Els resultats es mostren a les Figures 2, 3 i 4. Hem de tenir en ment que el benefici està fitat superiorment per 102000 €. Com que hem utilitzat les mateixes 1000 *seeds* per generar les instàncies del problema, el benefici obtingut per cada conjunt d'operadors sí que és comparable. Dels gràfics en podem extreure les següents conclusions:

- Podem descartar els conjunts S, AE i AES, ja que la qualitat de les solucions és molt inferior a la dels altres conjunts d'operadors. La resta de conjunts són els que tenen l'operador **COMBINA**. Per tant, veiem que aquest operador és molt útil i ens permet obtenir millors solucions. La nostra hipòtesi era que els operadors ens portarien a solucions semblants. Veiem que no és així, ja que l'operador **COMBINA** ens permet redistribuir les assignacions prèvies de millor manera.

Observació	L'elecció del conjunt d'operadors pot conduir a millors solucions i a arribar-hi més ràpidament.
Plantejament	Fixant els altres paràmetres del problema, provarem diferents combinacions d'operadors i observarem els resultats que obtenim.
Hipòtesi	Els conjunts d'operadors són equivalents a la pràctica, però les combinacions ens permeten arribar a les solucions més ràpidament.
Mètode	– Utilitzarem les mateixes 1000 *seeds* per realitzar 1000 repeticions per escenari. – Mesurarem el temps necessari per resoldre cada instància de cada escenari. – Mesurarem el nombre de pasos per resoldre cada instància de cada escenari. – Calcularem el benefici obtingut en la solució trobada per cada instància de cada escenari. – Els escenaris estudiats seran els següents: 1. **SERVEIX(N)** 2. **AFEGEIX** + **ESBORRA** 3. **COMBINA** 4. **AFEGEIX** + **ESBORRA** + **COMBINA** 5. **AFEGEIX** + **ESBORRA** + **SERVEIX(N)** 6. **COMBINA** + **SERVEIX(N)** 7. **AFEGEIX** + **ESBORRA** + **COMBINA** + **SERVEIX(N)**

Taula 3: Descripció de l'Experiment 1

– En termes de cost temporal, els conjunts d'operadors que porten a pitjors solucions són més ràpids. D'entre els 4 que donen bones solucions, C i CS són lleugerament més ràpids. Quan afegim l'operador **COMBINA** el nombre de passos disminueix molt, mentre que el temps no varia. Tot això mostra que l'operador **COMBINA** és equivalent a qualsevol combinació d'operadors **AFEGEIX** i **ESBORRA** i, a més, és més ràpid.

– Els conjunts AE i AES tenen un nombre de passos més elevat, ja que cada expansió només canvia en una parada la ruta d'un sol centre. En canvi, els conjunts que tenen l'operador **COMBINA** fan menys passos perquè amb una sola expansió modifiquen i emplenen completament dues rutes. Pel que fa a les altres combinacions, totes tenen un nombre de passos semblant.

La conclusió dels gràfics és que els conjunts d'operadors C, AEC, CS i $AECS$ ens donen resultats semblants. Descartarem els conjunts AEC i $AECS$, ja que, com hem raonat, són equivalents als altres dos, l'algorisme prefereix els nodes generats per **COMBINA** i el factor de ramificació seria innecessàriament alt. A la Figura 5 podem veure els mateixos gràfics que abans però incloent només els conjunts C i CS. No és necessari realitzar algun altre anàlisi estadístic per veure que els dos conjunts són pràcticament equivalents. Així doncs, l'elecció del conjunt d'operadors entre aquests dos és bastant arbitrària i es pot argumentar en els dos sentits. Nosaltres utilitzarem el conjunt C, ja que la desviació estàndard del nombre de passos és menor.

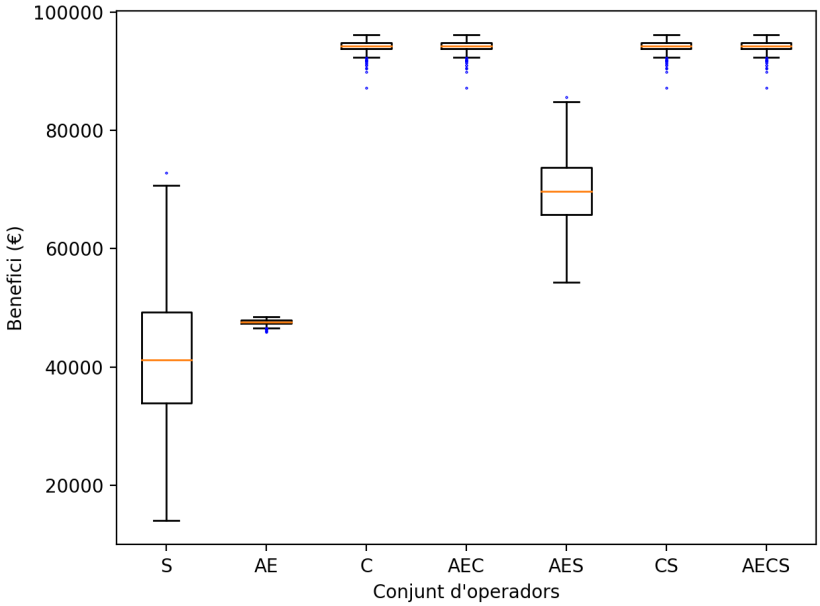

Figura 2: Benefici obtingut en funció del conjunt d'operadors utilitzat per l'experiment 1.

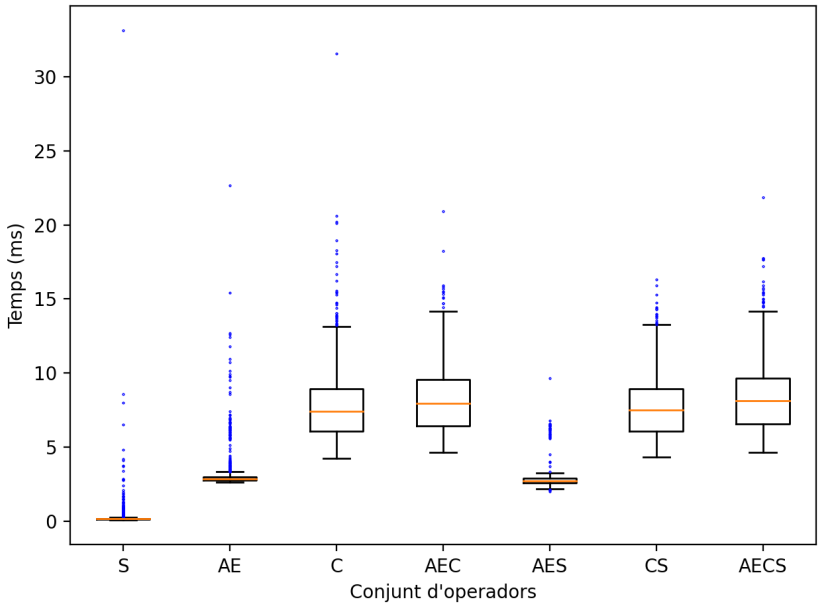

Figura 3: Temps en funció del conjunt d'operadors utilitzat per l'experiment 1.

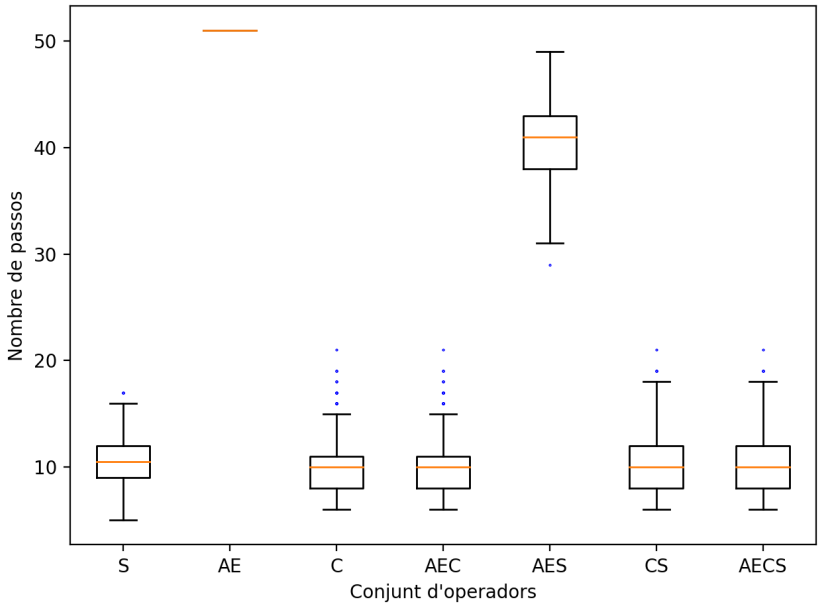

Figura 4: Nombre de passos en funció del conjunt d'operadors utilitzat per l'experiment 1.

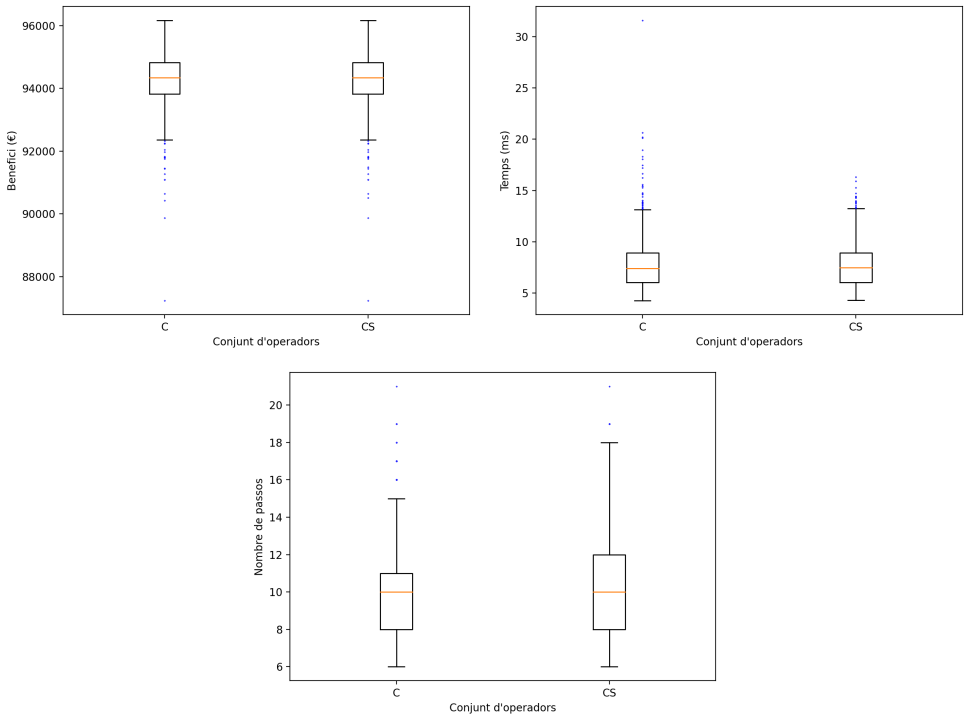

Figura 5: Comparació dels conjunts C i CS per l'experiment 1.

8.2 Experiment 2

L'objectiu del segon experiment és determinar quina és la millor manera de generar l'estat inicial. Utilitzarem els mateixos valors pels paràmetres que mantenim fixats que en l'experiment 1, amb l'excepció de la generació de l'estat inicial. També fixarem el conjunt d'operadors segons els resultats de l'experiment 1. En la Taula 2 es recullen els valors dels paràmetres que mantindrem constants. Els paràmetres que no apareixen a la taula tenen el valor per defecte descrit a la descripció de les dades del problema.

Paràmetre	**Valor**
C	10
G	100
Algorisme	Hill Climbing
Conjunt d'operadors	C
Heurística	1

Taula 4: Paràmetres constants de l'Experiment 2

En la Taula 3 es recull la descripció de l'experiment.

Observació	L'elecció de l'algorisme de generació de l'estat inicial pot conduir a millors solucions i a arribar-hi més ràpidament.
Plantejament	Fixant els altres paràmetres del problema, provarem diferents maneres de generar l'estat inicial i observarem els resultats que obtenim.
Hipòtesi	L'estat inicial generat amb l'algorisme voraç ens permet arribar a solucions tan bones com l'estat inicial buit però hi arribem més ràpidament.
Mètode	– Utilitzarem les mateixes 1000 *seeds* aleatòries per realitzar 1000 repeticions per escenari. – Mesurarem el temps necessari per resoldre cada instància de cada escenari. – Mesurarem el nombre de pasos per resoldre cada instància de cada escenari. – Calcularem el benefici obtingut en la solució trobada per cada instància de cada escenari. – Estudiarem les estratègies de generació d'estats inicials buida i voraç.

Taula 5: Descripció de l'Experiment 2

En les Figures 6, 7 i 8 es mostren els resultats de l'experiment. Hem de tenir en ment que el benefici està fitat superiorment per 102000 €. En aquest cas, veiem que per les 3 mètriques l'estratègia voraç és la millor. La mitjana dels beneficis obtinguts és més elevada, les solucions es troben en menys temps i amb, aproximadament, la meitat dels passos. Per tant, amb aquest experiment arribem a la conclusió que l'estratègia inicial generada amb l'algorisme voraç que hem dissenyat és molt propera a la solució final i, conjuntament amb els operadors, ens permet arribar a bones solucions inicials amb un nombre reduït de passos.

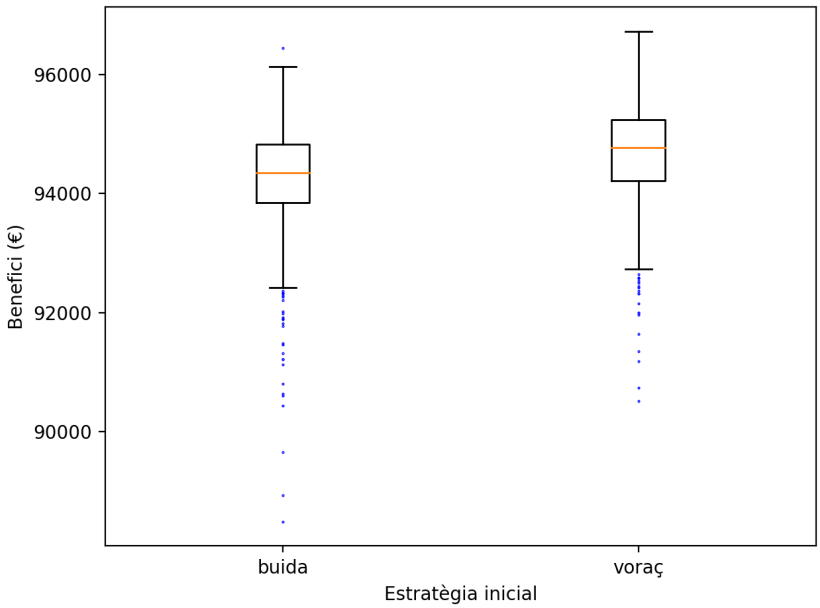

Figura 6: Benefici obtingut en funció de l'estratègia inicial utilitzada per l'experiment 2.

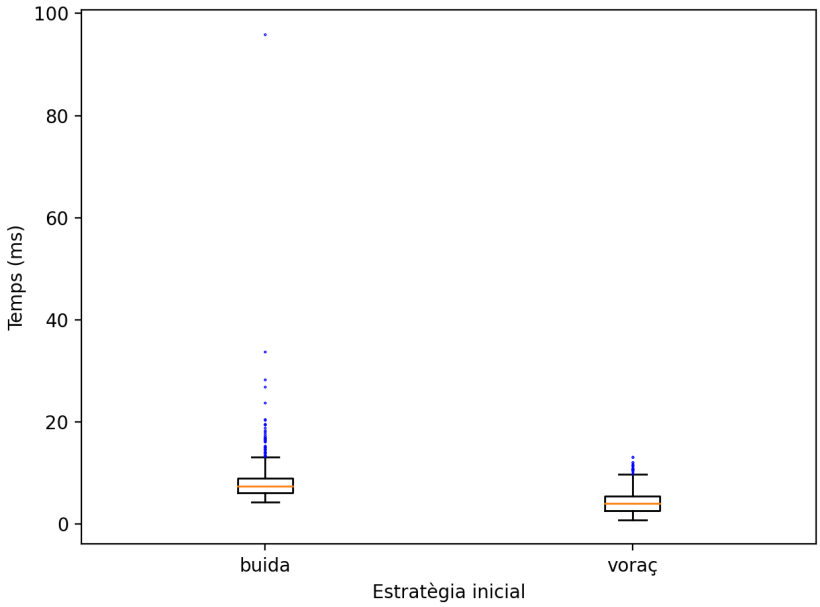

Figura 7: Temps en funció de l'estratègia inicial utilitzada per l'experiment 2.

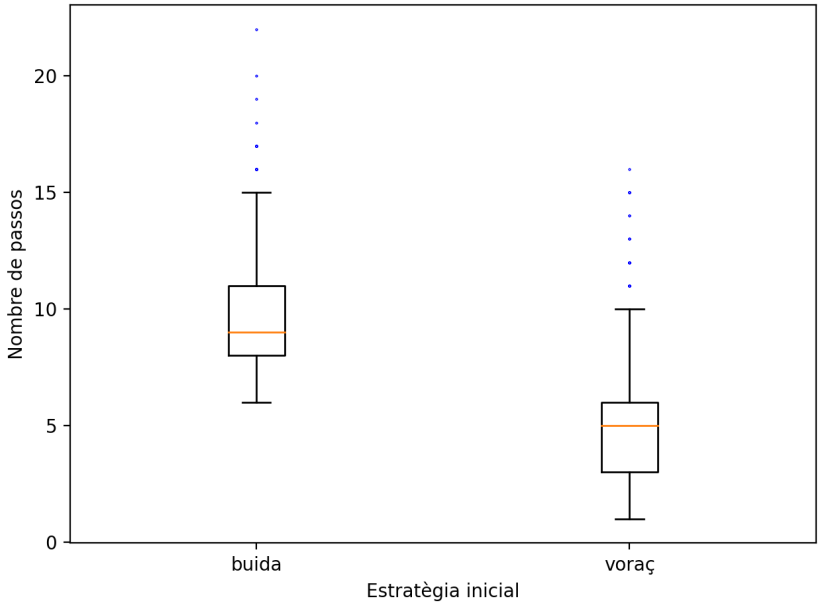

Figura 8: Nombre de passos en funció de l'estratègia inicial utilitzada per l'experiment 2.

8.3 Experiment 3

Ara hem de determinar els paràmetres de Simulated Annealing. Per començar a reduir l'espai de cerca dels paràmetres hem iterat per totes les parelles (λ, k) per $\lambda = \frac{10}{8^i}$ per $i \in \{0, ..., 19\}$ i $k = 5^i$ per $i \in \{0, 4\}$. Per tant $\lambda \in [10^{-16}, 10]$ i $k \in [1, 625]$. Per a cada parella hem fet 10 repeticions d'experiments generats aleatòriament (els mateixos 10 escenaris per a cada parella) amb un limit d'iteracions a 5000. En els següents gràfics es poden veure les mitjanes de cada parella tant dels beneficis aconseguits com de les iteracions fins estabilitzar. Hem d'anar amb cura interpretant els gràfics ja que els de beneficis representen un rang de valors petit que no comença en 0. Tot i així, és l'adequat ja que les possibles diferències entre les solucions és petita.

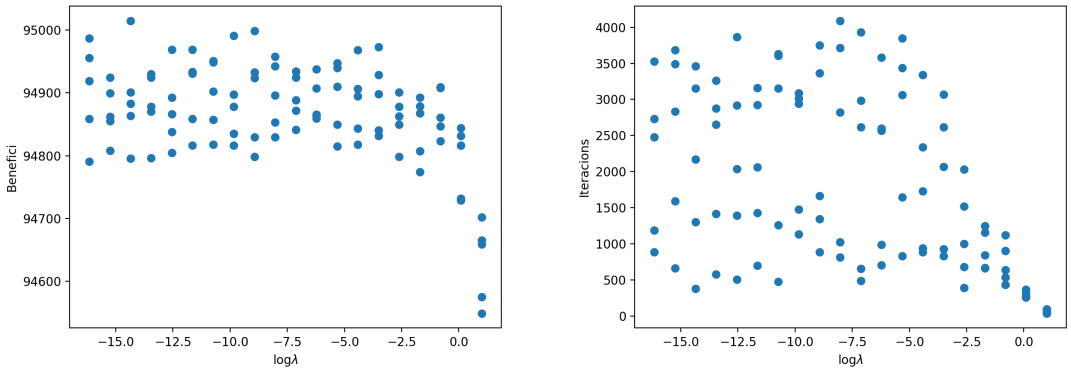

Figura 9: $\log \lambda$ vs Benefici (esquerra) i $\log \lambda$ vs Iteracions (dreta). Per a cada valor de λ podem veure les mitjanes per a cada valor de k utilitzat. S'observa com els valors més grans de λ necessiten menys iteracions però obtenen beneficis menors.

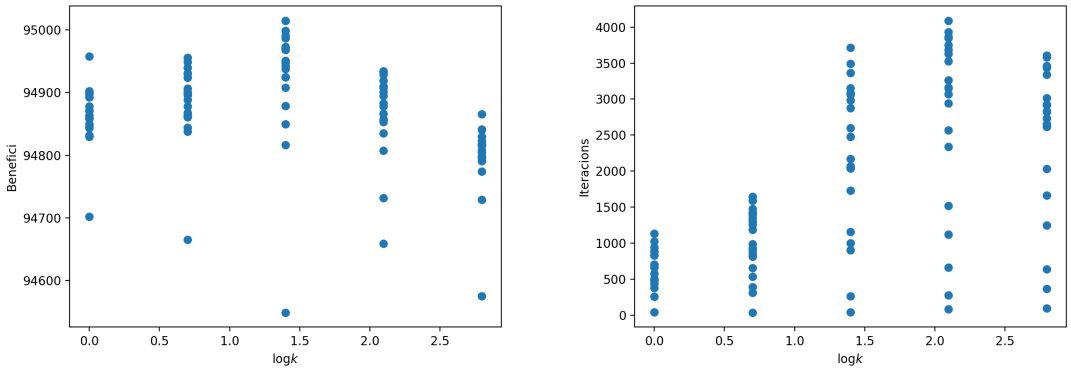

Figura 10: $\log k$ vs Benefici (esquerra) i $\log k$ vs Iteracions (dreta). Per a cada valor de k podem veure les mitjanes per a cada valor de λ utilitzat. S'observa com els valors més petits de k tarden moltes menys operacions en aconseguir uns beneficis similars o superiors a la resta.

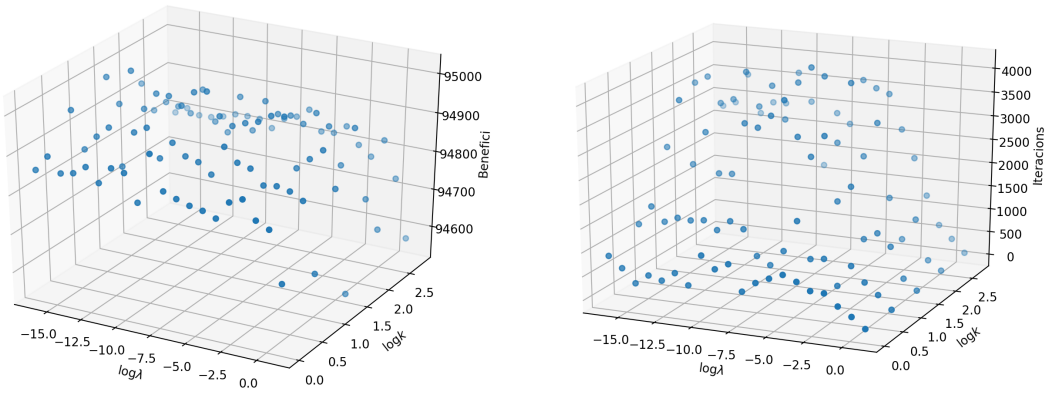

Figura 11: $\log k$ i $\log \lambda$ vs Benefici (esquerra) i $\log k$ i $\log \lambda$ vs Iteracions (dreta). Aquests gràfics donen una guia sobre com interpretar els anteriors. Recordem que tenim dos paràmetres i que els seus resultats estan interrelacionats. Podem veure que les tendències observades son correctes, per valors grans de λ i petits de k les iteracions disminueixen i per valors grans de λ disminueixen els beneficis.

De les figures obtenim que els millors valors de k són 5 i 25. Obtenen els millors beneficis amb un nombre de iteracions mínim, sobretot per $k = 5$. Per la λ la conclusió més clara és que els valors més alts no donen bons resultats (tot i que tarden molt poques iteracions en convergir). La resta de punts semblen distribuïts de forma més aleatòria. Amb aquests resultats la segona iteració la farem amb només els valors de $k = 1, 5, 11, 25$ i $\lambda = \frac{10}{8^i}$ per $i \in \{6, ..., 15\}$.

Procedint amb aquests paràmetres de la mateixa manera que abans obtenim el següent:

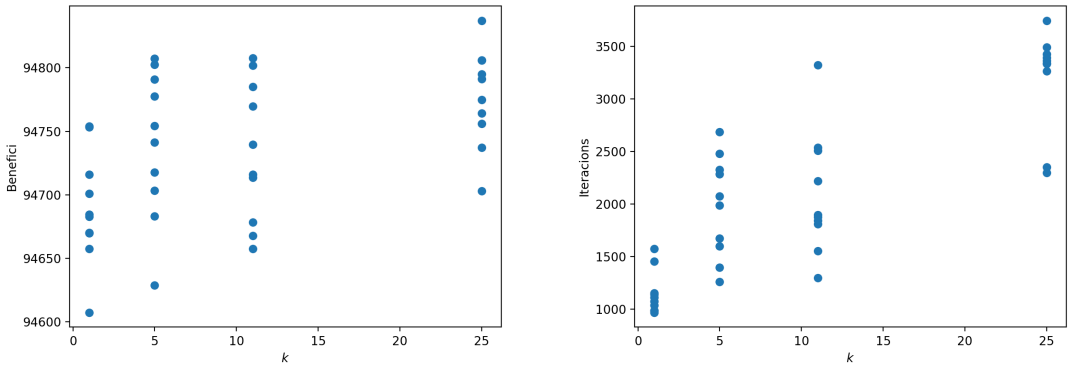

Figura 12: k vs Benefici (esquerra) i k vs Iteracions (dreta). Per a cada valor de k podem veure les mitjanes per a cada valor de λ utilitzat. S'observa que els resultats son menors si $k = 1$ i de la resta el mínim nombre d'iteracions s'aconsegueix amb $k = 5$

Per a la λ tornem a obtenir un núvol de punts. Dels gràfics anteriors podem veure que tenint uns resultats similars el nombre d'iteracions per $k = 25$ és clarament superior. $k = 1$ necessita menys iteracions però té pitjors resultats. Tot i que $k = 5$ i $k = 11$ es comporten de forma similar, utilitzarem $k = 5$ pel menor nombre d'iteracions. Un cop fixat k anem a determinar la millor λ. Repetint per tant l'experiment per aquesta k fixada obtenim el següent. Aquest cop hem utilitzat 40 repeticions per a cada valor per intentar minimitzar el possible error o soroll en l'experiment:

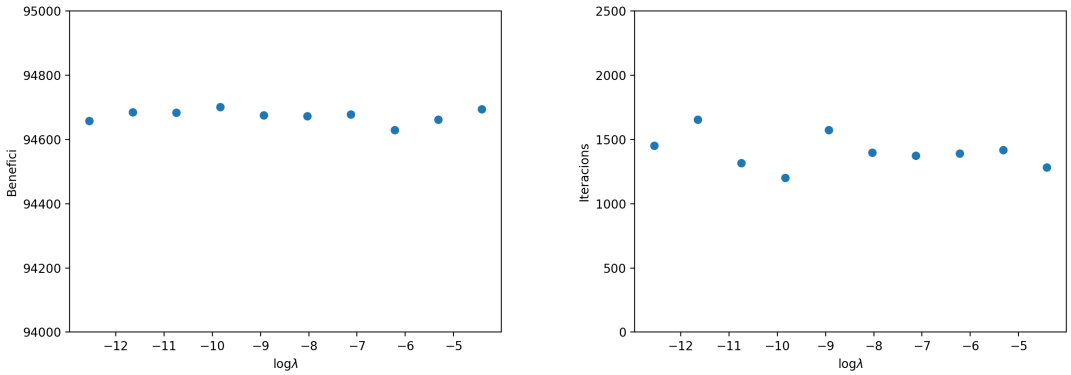

Figura 13: $\log \lambda$ vs Benefici (esquerra) i $\log \lambda$ vs Iteracions (dreta). Per a cada λ ara només tenim un valor ja que hem fixat k.

Observem que la influència en els beneficis i el nombre de iteracions per la λ en aquest interval és mínima. Com que sí havíem observat abans que valors més grans de λ donaven menys iteracions, hem decidit que utilitzarem $\lambda = 10^{-5}$.

Per tant els paràmetres decidits per l'algorisme de Simulated Annealing són $(k, \lambda) = (5, 10^{-5})$.

8.4 Experiment 4

Anem a explorar ara com varia el temps d'execució a l'augmentar la grandària tot respectant la proporció $1:10$ entre centres i gasolineres. Per fer-ho agafarem valors de $C = 10k$ per $k \in \{1, ..., 20\}$ (recordem que C és el nombre de centres). Per cada valor de C farem 20 iteracions generades aleatòriament i cada escenari el resoldrem tant per Hill Climbing com per Simulated Annealing. Finalment, calcularem la mitjana dels temps obtinguts per a cada C tant per generar les solucions inicials, com per executar Hill Climbing, com per executar Simulated Annealing. A més, també calcularem la mitjana dels beneficis per cada algorisme.

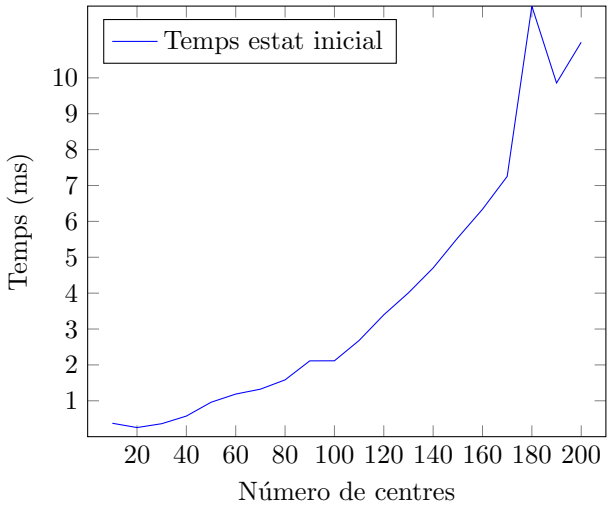

Figura 14: Número de centres vs temps d'execució per generar l'estat inicial. El més rellevant d'aquest gràfic és que observem que excepte l'*outlier* per $C = 180$ els temps creixen quadràticament respecte la mida del problema. Això confirma la hipòtesi de que el temps d'aquest algorisme és $\mathcal{O}(GVC)$, ja que aquí tenim $G = 10C$ i per tant esperem que el creixement sigui quadràtic respecte C.

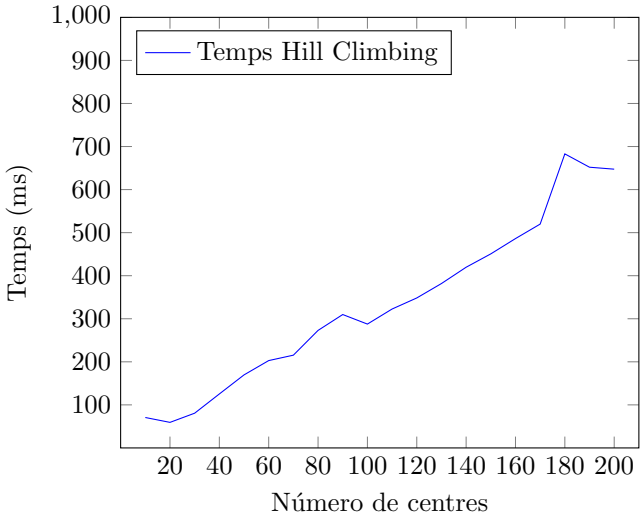

Figura 15: Número de centres vs temps d'execució de Hill climbing. Observem que el temps d'execució creix linealment amb la mida del problema.

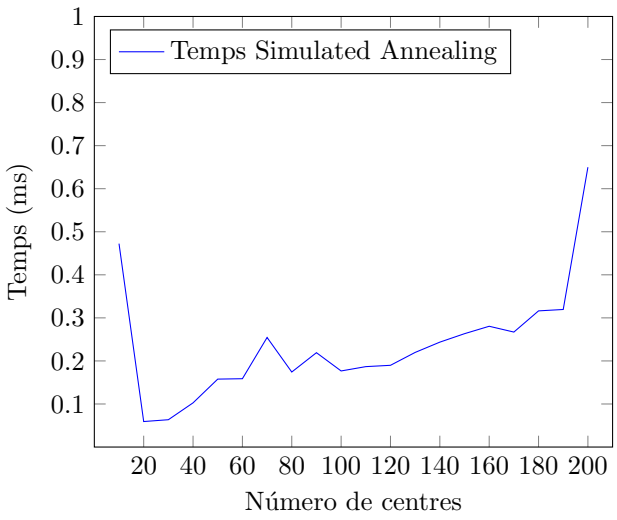

Figura 16: Número de centres vs temps d'execucució per Simulated Annealing. Excepte pels dos *outliers* observem que el cost creix lleugerament però amb una pendent poc pronunciada.

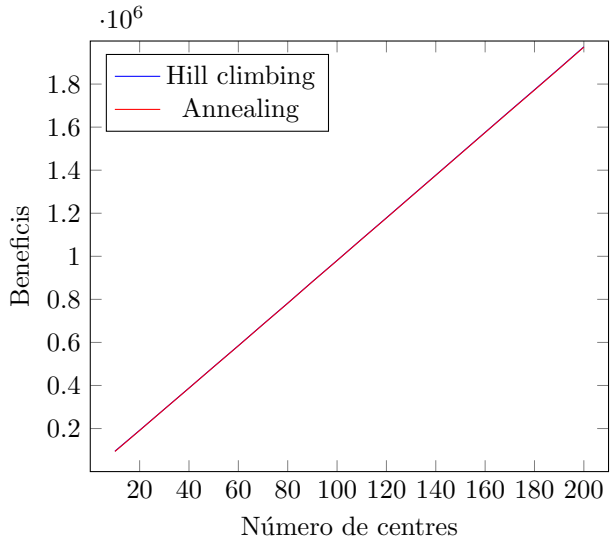

Figura 17: Número de centres vs beneficis per Hill Climbing i Simulated Annealing. Observem que com esperàvem a l'augmentar el nombre de centres i gasolineres els beneficis creixen de manera proporcional.

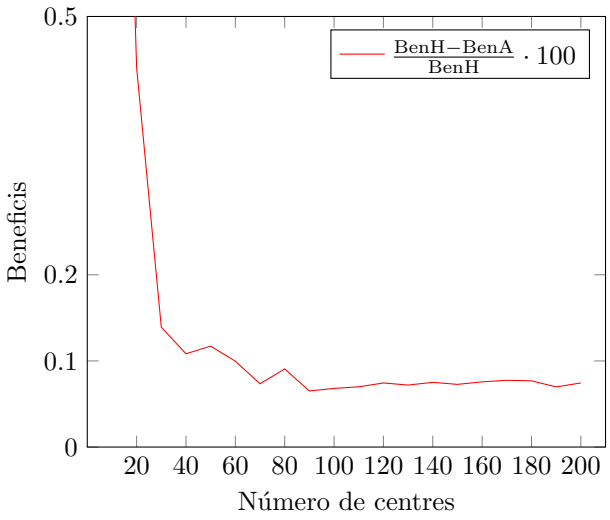

Figura 18: Nombre de centres vs $\frac{\text{BenH}-\text{BenA}}{\text{BenH}} \cdot 100$ on BenH són els beneficis amb Hill Climbing i BenA els beneficis amb Simulated Annealing. Podem interpretar aquesta quantitat com quant de millor ho fa Hill Climbing respecte el Simulated Annealing. Observem que les diferències es situen per sota del 0.2%.

Dels gràfics anteriors podem treure diverses conclusions. Primer, com ja esperàvem, Simulated Annealing és molt més ràpid tant en termes absoluts (uns 1000 cops) com asimptòticament. Això era fàcilment esperable ja que tot i que Simulated Annealing fa moltes més iteracions, a cada iteració genera un sol estat. En canvi Hill Climbing genera tots els successors i aplica la funció heurística a cadascun d'ells. Per una altra banda, Hill Climbing dona resultats lleugerament superiors consistentment, però les diferències són mínimes. Hem observat que per Simulated Annealing el temps d'execució no s'ha disparat i els resultats no han empitjorat respecte Hill Climbing. D'això podem deduir que els paràmetres utilitzats segueixen donant uns paràmetres raonablement bons i, per tant, es comporten bé tot i el canvi de mida del problema.

Els resultats sobre els temps eren els esperables ja que la potència de Simulated Annealing rau en que no cal generar tot l'espai de successors cada vegada.

8.5 Experiment 5

En el 5è experiment volem comprovar com varien els beneficis i els quilòmetres recorreguts si en comptes de tenir 10 centres amb un camió cadascun, en tenim 5 i a cada centre hi ha 2 camions. Per estudiar com afecta aquest canvi, farem 100 iteracions amb 100 seeds diferents.

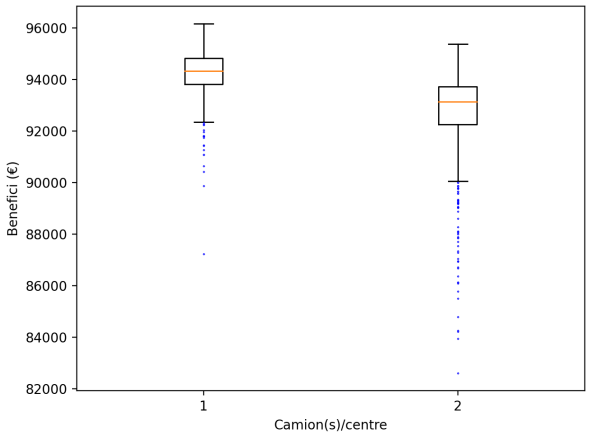

Figura 19: Benefici obtingut per 10 camions en funció dels camions per centre

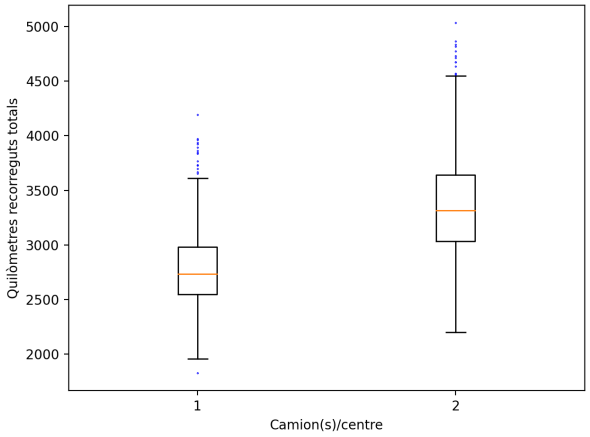

Figura 20: Quilòmetres recorreguts per 10 camions en funció dels camions per centre

En les figures 19 i 20 veiem la comparació dels beneficis i quilòmetres recorreguts dels dos escenaris. Podem comprovar que en el cas en que tenim 10 centres amb 1 camió cadascun els beneficis són lleugerament més elevats, probablement perquè, com podem comprovar a la figura 20, els quilòmetres recorreguts en aquest cas són força menys que en el cas en que els 10 camions estan repartits en 5 centres.

D'aquests resultats podem deduir que, com que en el segon escenari els camions estan menys repartits per el mapa (ja que estan agrupats de dos en dos), han de recórrer més distància per tal d'atendre les peticions de les gasolineres, i això fa que el benefici disminueixi degut al cost extra de fer més quilòmetres.

8.6 Experiment 6

En aquest experiment estudiarem com afecta el cost de recórrer un quilòmetre amb el número de peticions ateses. Per fer-ho anirem doblant aquest cost, fent 100 iteracions per cada cas, i compararem les peticions mitjanes ateses per cadascun dels valors.

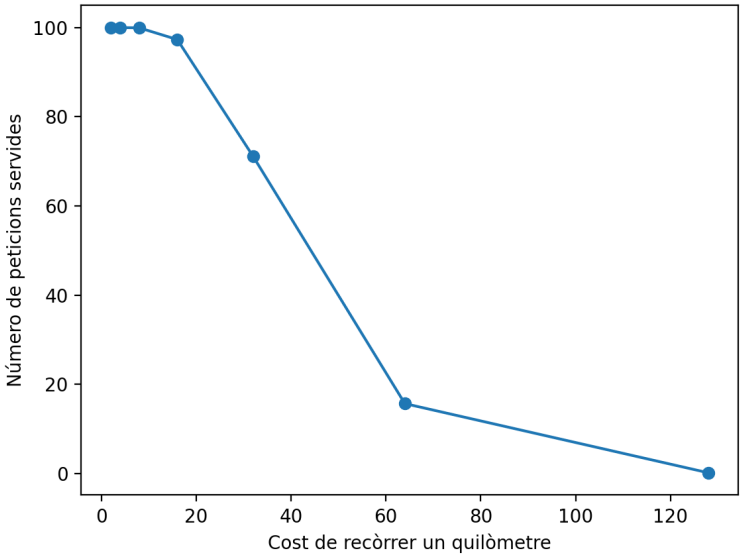

Figura 21: Peticions ateses vs Cost per quilòmetre

A la figura 21, podem comprovar que el número de peticions ateses es manté més o menys estable en els quatre primers casos (és a dir quan el cost és 2, 4, 8 i 16 respectivament), però passat aquest punt disminueix ràpidament fins arribar a 0 quan el cost per quilòmetre arriba a 128.

Observant aquests resultats podem veure que, en les condicions inicials del problema, el valor d'un dipòsit és bastant més elevat que el cost dels trajectes dels camions, fet que permet que hi hagi un marge prou ampli pel que fa al preu per quilòmetre. Però si seguim augmentant el seu preu, només surt a compte atendre les peticions més properes per evitar pèrdues.

Anem a estudiar ara si el cost del quilòmetre té algun efecte sobre la proporció de peticions ateses tenint en compte el dies que porten pendents. Per fer-ho veurem quines són les proporcions en les iteracions que hem fet per obtenir els resultats de la figura 21.

Figura 22: Proporció de peticions ateses segons el dies que porten pendents

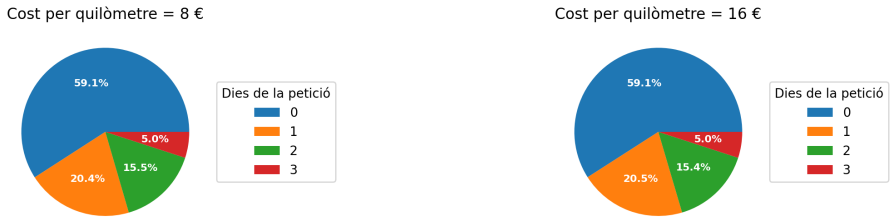

Figura 23: Proporció de peticions ateses segons el dies que porten pendents

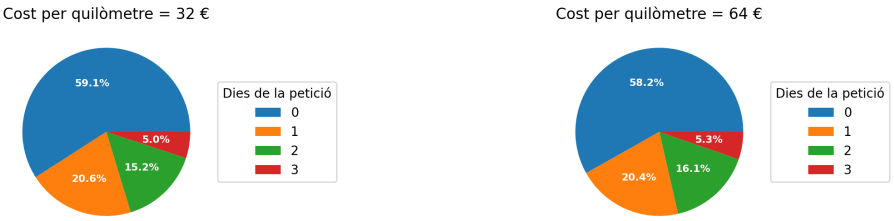

Figura 24: Proporció de peticions ateses segons el dies que porten pendents

Les figures 22, 23 i 24, ens mostren aquestes proporcions per diversos costos per quilòmetre recorregut. Al veure aquests gràfics és evident que l'efecte que té aquest cost sobre la proporció de peticions ateses és inexistent, ja que tots són pràcticament idèntics. Això es degut a que els nostres operadors donen prioritat a les peticions més antigues independentment d'aquest cost. Tot i això, ens pot semblar contradictori que, si s'atenen primer les peticions més antigues, les peticions ateses amb un percentatge més elevat sigui les que són noves; però per entendre aquest fet n'hi ha prou en veure que a l'hora de generar una petició hi ha una probabilitat del 60% de que la petició sigui d'avui, un 20% de que porti un dia pendent, un 15% de que en porti 2 i un 5% de que en porti 3, que són quasi exactament les proporcions que veiem en els nostres resultats.

8.7 Experiment 7

Ara volem comprovar l'efecte que té sobre els beneficis una petita variació de les hores de feina d'un camió, o el que és equivalent, els quilòmetres màxims que pot recórrer cada camió. Per fer-ho avaluarem els beneficis aconseguits quan els camions tenen un horari de 7 hores, després quan tenen un horari de 8 hores i finalment de 9, i compararem els resultats.

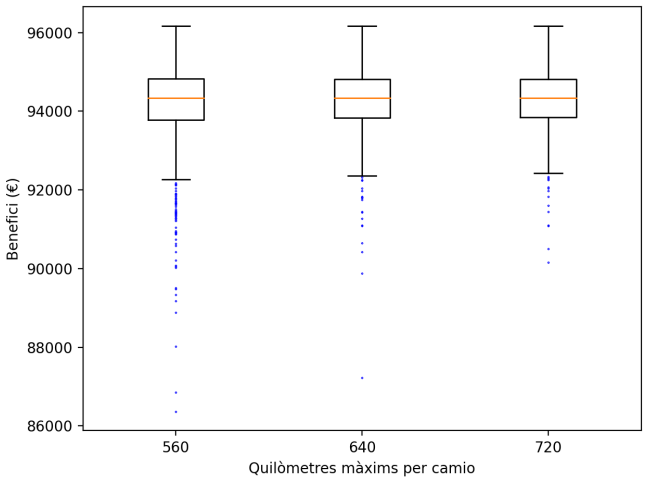

Figura 25: Benefici obtingut depenent del quilòmetres màxims que pot recórrer un camió

Era esperable que l'escenari on cada camió pot fer com a màxim 560 km diaris fos el que prestés uns beneficis més baixos, seguit pel de 640 i que el de 720 km fos el que tingués beneficis més elevats. Però, observant la figura 25 veiem que els tres casos presenten uns resultats molt similars. Això és degut a que la restricció de 5 viatges diaris acaba sent més forta que la dels quilòmetres, i que per tant, en general, quan un camió ha fet 5 viatges, no ha gastat tots els quilòmetres que podia recórrer.

En canvi, si repetim el mateix experiment però prèviament reduïm el nombre de camions a 5 i augmentem el número de viatges màxim que pot fer cada camió diàriament, veiem que els beneficis augmenten conforme ho fan les hores de feina de cada camió.

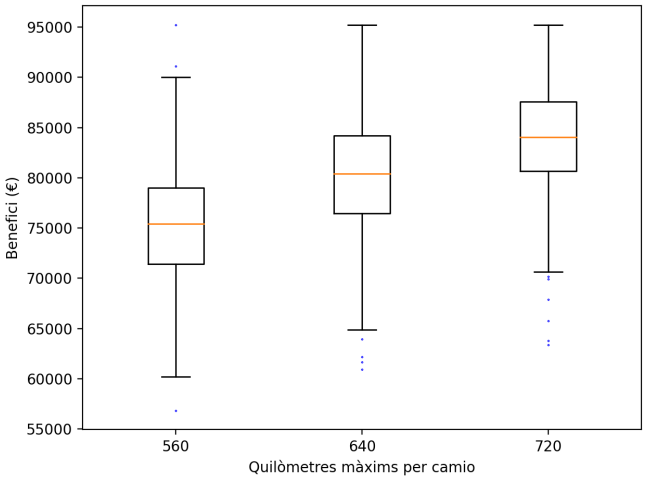

Figura 26: Benefici obtingut per 5 camions (amb un límit de 10 viatges per camió) depenent dels quilòmetres màxims que poden recórrer

8.8 Experiment 8

Aquest experiment és l'experiment especial i consisteix en calcular el benefici que s'obté en un escenari amb 10 centres de distribució, un camió per centre i 100 gasolineres i el temps que es tarda en trobar la solució. Utilitzarem la *seed* 1234 per generar la instància del problema. Els operadors, l'estratègia per generar l'estat inicial i l'heurística són els que hem escollit mitjançant els experiments 1 i 2 i es mostren a la Taula 6.

Paràmetre	**Valor**
Seed	1234
C	10
G	100
Algorisme	Hill Climbing
Conjunt d'operadors	C
Generació de l'estat inicial	Voraç
Heurística	1

Taula 6: Paràmetres constants de l'Experiment 8

Tal i com vam demostrar a la classe de laboratori, obtenim un benefici de 94892 €. El temps d'execució en un dels nostres ordinadors és de 4.574083 ms.

9 Conclusions

En conclusió, en aquest projecte hem encarat un problema de distribució d'un recurs com un problema de cerca local i hem experimentat amb diferents estratègies per obtenir solucions. Per mitjà dels experiments, hem pogut justificar raonadament l'elecció dels valors de diferents paràmetres i veure com varien el benefici econòmic obtingut, els passos realitzats i el temps d'execució en funció de l'escenari triat.

A l'experiment 1 hem conclòs que el millor conjunt d'operadors per l'algorisme de Hill Climbing és el format per l'operador **COMBINA**, que redistribueix les rutes de dos centres de distribució. A l'experiment 2, hem vist que l'estratègia de generació de l'estat inicial mitjançant l'algorisme voraç que hem dissenyat ens dóna molt bons resultats. En l'experiment 3 hem trobat que els millors paràmetres per l'algorisme de Simulated Annealing són $(k, \lambda) = (5, 10^{-5})$. En l'experiment 4, hem comprovat que l'algorisme de Simulated Annealing és molt més ràpid que l'algorisme de Hill Climbing i que el temps d'execució es comporta molt millor quan augmenta la mida del problema. En aquest experiment també hem observat les tendències dels dos algorismes quan variem la mida del problema. En l'experiment 5, hem vist que és millor tenir un camió per centre que dos camions per centre. En l'experiment 6, hem comprovat quin és el marge del preu per quilòmetre per tal que les peticions ateses no baixin significativament, i hem vist que aquest cost no afecta la proporció de peticions ateses segons els dies d'antiguitat que tenen. En l'experiment 7, hem vist que la restricció dels viatges màxims és més forta que la dels quilòmetres màxims, i que per tant si fem una petita variació d'aquest últim paràmetre, el seu efecte sobre els beneficis és gairebé nul. Per últim, l'experiment 8 va ser un incentiu per treballar continuadament en la pràctica.

Aquesta pràctica ens ha ajudat a comprendre millor els algorismes de Hill Climbing i Simulated Annealing. Hem vist que l'elecció de l'estratègia inicial, l'heurística i els operadors és rellevant en el seu funcionament i marca la diferència en la qualitat i eficiència de les solucions obtingudes. En retrospectiva, creiem que potser hauríem d'haver escollit uns altres conjunts d'operadors per l'algorisme de Simulated Annealing. Degut a l'important component aleatori de l'algorisme, potser hauria sigut adequat considerar operadors menys encarats a millorar la solució i que, simplement, estiguessin pensats per donar-nos més llibertat a l'hora d'explorar l'espai de solucions.

Tot i així, hem obtingut resultats lògics i bons i aquestes reflexions fan palès que després d'aquest projecte tenim més intuïció pel que fa als algorismes de cerca local i hem après a enfrontar-nos a un problema d'aquest tipus utilitzant una gran varietat d'eines i anàlisis estadístics.

10 Treball d'innovació

Pel treball d'innovació hem escollit els algorismes de recomanació que utilitza Spotify, una plataforma de reproducció de música via streaming.

10.1 Descripció del tema

Els usuaris d'Spotify poden obtenir recomanacions de cançons segons la seva activitat a la plataforma. Per tal de fer aquestes recomanacions, Spotify utilitza algorismes d'IA sobre les dades obtingudes de les cançons que l'usuari ha marcat com les que li agraden, l'historial de reproducció, les llistes de reproducció personals i els artistes als que segueix. Aleshores, utilitza diferents tècniques d'IA, com poden ser *Collaborative filtering*, *Natural Language Processing* i *Convolutional Neural Networks*.

10.2 Repartiment del treball entre els membres del grup

La primera tasca que ens vam assignar va ser la d'obtenir referències per començar a treballar i assegurarnos que podíem trobar suficient informació sobre el tema escollit. Entre els tres membres de l'equip, vam recollir una llista de fonts amb informació sobre com Spotify utilitza l'IA.

Una vegada plantejat el tema, ens vam repartir la recerca del treball de la següent manera:

Edgar:

- Impacte de l'ús d'IA en Spotify
- Impacte de l'ús d'IA en els usuaris d'Spotify

Laia:

- Descripció d'Spotify
- Explicació de l'innovació d'Spotify i comparació amb altres sistemes

Maria:

- Descripció de les tècniques d'IA que utilitza Spotify (*Collaborative filtering*, *Natural Language Processing* i *Convolutional Neural Networks*)
- Descripció de l'ús que fa Spotify de les tècniques d'IA descrites (amb l'ajuda de la **Laia**)

Hi ha una part del treball que ens l'hem repartit entre dues ja que combina la informació de parts en les que treballarem les dues. A mesura que anem avançant amb el treball acabarem de cal·librar el repartiment perquè sigui equilibrat.

10.3 Llista de referències

- Descripció d'Spotify
  - https://www.spotify.com/us/ [27/09/2021]
  - https://en.wikipedia.org/wiki/Spotify [27/09/2021]
- Descripció de les tècniques d'IA que utilitza Spotify (*Collaborative filtering*, *Natural Language Processing* i *Convolutional Neural Networks*)
  - https://en.wikipedia.org/wiki/Collaborative_filtering [14/10/2021]

- https://developers.google.com/machine-learning/recommendation/collaborative/basics [14/10/2021]
- https://medium.com/@pcnavarr/analisis-collaborative-filtering-recommender-systems-94bea27088d [14/10/2021]
- https://en.wikipedia.org/wiki/Natural_language_processing [14/10/2021]
- https://www.ibm.com/cloud/learn/natural-language-processing [14/10/2021]
- https://en.wikipedia.org/wiki/Convolutional_neural_network [14/10/2021]
- https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b [14/10/2021]
- https://cs231n.github.io/convolutional-networks/ [14/10/2021]

– Descripció de l'ús que fa Spotify de les tècniques d'IA descrites

- https://outsideinsight.com/insights/how-ai-helps-spotify-win-in-the-music-streaming-world/ [27/09/2021]
- https://www.datasciencecentral.com/profiles/blogs/6448529:BlogPost:1041799 [27/09/2021]
- https://developer.spotify.com/?_ga=2.235761473.1564095249.1635699260-145426574.1635442709 [22/10/2021]

– Explicació de l'innovació d'Spotify i comparació amb altres sistemes

- https://dl.acm.org/doi/10.1145/3209219.3209223 [16/10/2021]
- http://arno.uvt.nl/show.cgi?fid=136352 [16/10/2021]
- http://ambertsai.me/pdf/music_recommendation.pdf [16/10/2021]
- https://par.nsf.gov/servlets/purl/10088296 [16/10/2021]
- https://www.researchgate.net/publication/318511102_Recommender_System_Based_on_Collaborative_Filtering_for_Spotify%27s_Users [18/10/2021]

– Impacte de l'ús d'IA en Spotify

- https://www.forbes.com/sites/bernardmarr/2017/10/30/the-amazing-ways-spotify-uses-big-data-ai-and-mach?sh=469f35df4bd2 [17/09/2021]
- https://www.northeastern.edu/graduate/blog/spotify-big-data/ [17/09/2021]

– Impacte de l'ús d'IA en els usuaris d'Spotify

- https://www.analyticssteps.com/blogs/how-spotify-uses-machine-learning-models [17/09/2021]
- https://www.aidataanalytics.network/data-monetization/articles/data-visualization-monetization-and-person [17/09/2021]

10.4 Dificultats que ens hem trobat

A l'hora de buscar informació sobre el tema escollit, la dificultat més gran que ens hem trobat és que no hi ha gaire informació pública oficial per part d'Spotify. Tot i així, sí que hem trobat articles i un gran nombre de pàgines no oficials que descriuen amb les tècniques utilitzades per Spotify.

L'altra dificultat és que hi ha poca informació a la xarxa sobre l'impacte de l'ús de l'IA en Spotify. No obstant, sí que podem trobar informació sobre l'impacte de l'ús de l'IA en els usuaris d'Spotify.