Pràctica
Cerca Local

Roger Almató Baucells
Òscar Garcia Toledo
Farners Vallespí Soro

ÍNDEX

1. Part Descriptiva

1.1. Anàlisi del problema

Se'ns presenta un problema envers el canvi climàtic i les emissions de CO2. La idea és que d'una manera rotativa durant un període de temps, una persona utilitzi el seu cotxe per traslladar altres persones al seu lloc de treball. D'aquesta manera hi hauria cotxes amb més places ocupades, reduint el tràfic, el consum de combustible i la emissió de contaminants. Per tal de resoldre el problema, s'estableixen els paràmetres següents:

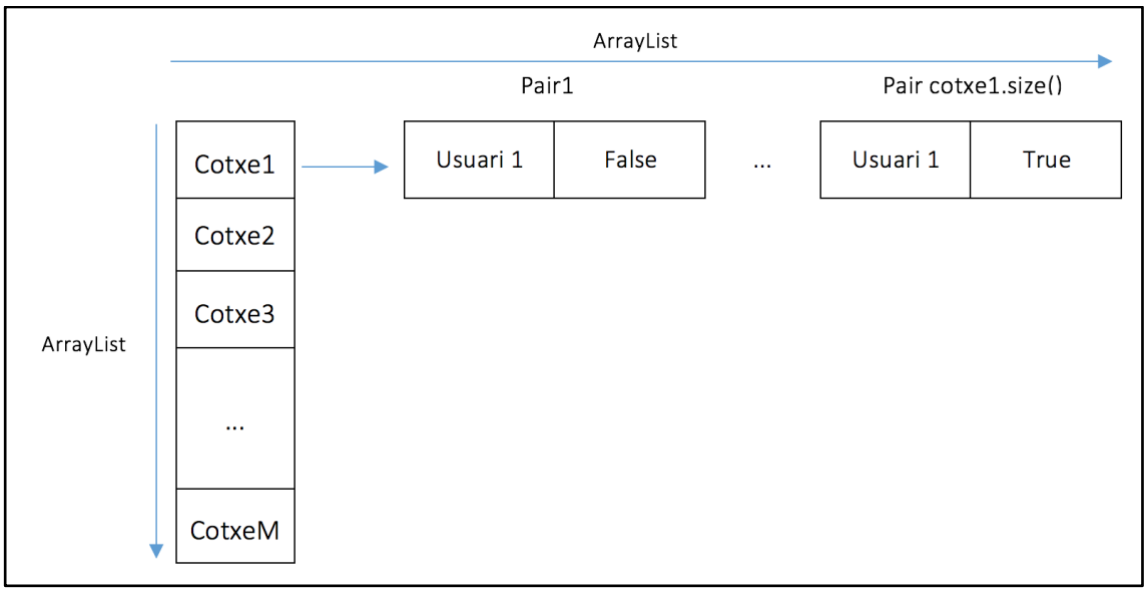
- N persones apuntades
- M persones que poden conduir
- M < N
- Ciutat és un quadrat de 10x10 kilòmetres
- Illa urbana té 100x100 metres
- Càlcul de la distància $d(i,j) = |i_x - j_x| + |i_y - j_y|$

1.2. Implementació de l'estat

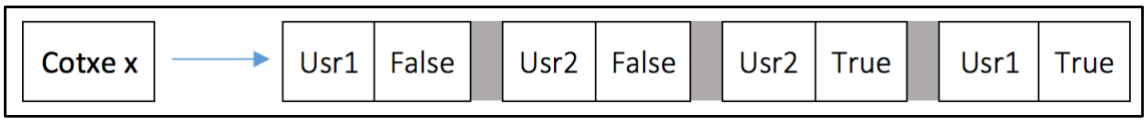
La nostra implementació de l'estat (Practica1Board.java) està definida amb els paràmetres següents:

```
1.      private static int nUsuarios;
2.      private static int nConductores;
3.      private static int seed;
4.      private static Usuarios usuarios;
5.      private ArrayList<ArrayList<Pair<Integer,Boolean>>> solucio;
```

- **nUsuarios:** enter estàtic amb el nombre d'usuaris (valor N del problema).
- **nConductores:** enter estàtic amb el nombre de conductors (valor M del problema).
- **seed:** valor de la llavor que utilitzarà la classe Usuarios (facilitada amb la pràctica), per generar la configuració de usuaris inicial.
- **solucio:** un ArrayList dins d'un altre ArrayList que conté parells (enter, booleà). Aquesta estructura de dades ens permetrà definir la solució inicial, i la seva evolució en els diferents estats fins arribar a la solució final. Gràfic de l'estructura:



Tal com podem observar en la representació, disposem d'un ArrayList que tindrà tamany M (cotxes que tenim inicialment). Al mateix temps, cada cotxe (element de l'ArrayList) conté un ArrayList de parells enter-booleà. Aquests parells els utilitzarem per conèixer qui és el conductor del vehicle, i per representar l'ordre en què recollim i deixem cada usuari.



Si observem la representació anterior sobre el Cotxe x, podem identificar:

- El conductor del Cotxe x és el usr1 ja que és el primer usuari que el cotxe agafa.
- L'ordre en què es realitzarà el trajecte serà:
    1. Agafem Usr1 (booleà del parell a False).
    2. Agafem Usr2 (booleà del parell a Fasle).
    3. Deixem Usr3 (booleà del parell a True).
    4. Deixem Usr4 (booleà del parell a True).

Hem pensat que aquesta seria una configuració adequada per diferents motius:

- En Java, un ArrayList és molt versàtil cosa que ens permet mutar l'estat de la solució amb facilitat, ja que per exemple, podem inserir ,eliminar i modificar elements amb facilitat i eficiència.
- Els pairs (enter - booleà) que ens permet definir l'ordre dels usuaris en el trajecte del cotxe amb facilitat.

1.3. Operadors

Un cop estudiat el problema i definit la implementació de l'estat, hem creat 3 operadors diferents:

```
1. public boolean CanviarUsuarisCotxe(Usuario usrCotxe1, Usuario
   usrPass1, Usuario usrCotxe2, Usuario usrPass2);
2. public boolean MoureUsuari(int usrCotxe1ID, int usrPass1ID, int
   usrCotxe2ID, int orig, int dest);
3. public boolean AgafarAbans(Usuario usrCotxe, Usuario usr);
```

1. **Intercanviar dos usuaris de cotxes diferents:** Donats dos conductors (usrCotxe1 i usrCotxe2) i dos passatgers (usrPass1 i usrPass2) dels cotxe 1 i 2 respectivament, intercambiar-los. És a dir, el usrPass1 que era passatger del cotxe 1, és mogut cap el cotxe 2 a on passarà a ocupar la posició del usrPass1. El mateix amb usrPass2.
2. **Moure un usuari d'un cotxe a un altre:** Donats dos conductors (usrCotxe1ID i usrCotxe2ID) i un passatger del cotxe 1 (cotxe del conductor usrCotxe1ID), moure el passatger del cotxe a 1 al cotxe 2. En cas de que l'usuari a moure sigui el conductor del vehicle, un cop realitzat el moviment s'eliminarà el cotxe buit.
3. **Cambiar la posició en què agafem o deixem un usuari en el trajecte d'un cotxe:** Donats un conductor (usrCotxe) i un passatger (usr) del cotxe on usrCotxe és conductor, agafar el passatger una posició abans dins del trajecte del cotxe.

Mitjançant aquests tres operadors podem realitzar una ramificació total dels possibles estats successors dins l'espai de solucions. Això ens permetrà mitjançant els algorismes de cerca local de Hill Climbing i Simulated Annealing arribar a solucions molt òptimes que analitzarem en l'apartat d'experimentació.

1.4. Funcions Heurístiques

Tal com se'ns explica en l'enunciat de la pràctica s'han de crear uns heurístics per tal de minimitzar la distància recorreguda pels conductors i la segona, a més a més, minimitzar el número de conductors.

Pel primer heurístic, hem tingut en compte tal com diu l'enunciat que la distància que pot recórrer cada cotxe és de 30 kms. D'aquesta manera ens assegurem que el temps de cada cotxe serà inferior a 1 hora (velocitat mitjana 30km / hora). Per tant, la funció quedarà definida de la següent manera:

- Si la distancia que recórrer un conductor és més de 30 km, elevarem a 3 la distància i retornarem el resultat.
- En cas contrari, retornarem la distancia.

Observem doncs que quan la distancia d'algun conductor és superior a 30 el sumatori de totes les distancies creix exponencialment. D'aquesta manera ens mourem cap a la zona de solucions reduint les distàncies dels conductors.

Pel què fa el segon heurístic, per tal de reduir el número de conductors, afegirem un factor de multiplicació de 100x. D'aquesta manera si una solució redueix un conductor, reduirem per 100 aquest factor.

Amb aquesta funció podrem reduir la distància (factor més rellevant) i en cas de que es pogui, el número de conductors.

1.5. Elecció i generació de l'estat inicial

S'han plantejat dues estratègies per tal de generar la solució inicial:

- La primera estratègia s'insereixen els passatgers de manera seqüencial als cotxes. És a dir, primer passatger s'inserirà al primer cotxe, el segon passatger al segon cotxe i així successivament. En cas de qui hagi més usuaris que cotxes, quan s'acaben el cotxes es torna a començar l'assignació pel primer.
- La segona estratègia es divideix el tauler en quatre zones i s'agrupen els usuaris segons la zona de sortida i la d'arribada. D'aquesta manera obtenim 16 grups d'usuaris. Repartirem els passatgers en conductors del mateix grup i en cas de que no hi hagi prous conductors es repartiran els passatgers restants aleatòriament pels cotxes que no estiguin plens.

L'objectiu de la segona estratègia, tot i ser més costosa que la primera, és de generar una solució inicial molt més bona, és a dir, una solució que ens permet-hi arribar a l'estat final amb menys estats. Com que una de les heurístiques definida anteriorment és la de reduir la distància dels conductors, mitjançant la divisió del tauler en quatre zones podem obtenir una solució inicial teoricament millor que no pas amb la primera estratègia.

2. Part Experimental

En aquesta secció veurem els resultats i conclusions a les que hem arribat després de realitzar els experiments suggerits a la documentació. Totes les mesures de distància es fan en quilòmetres i les de temps estan en milisegons. Les taules resumint els resultats numèrics es troben a l'apèndix del treball per si cal comprovar alguna dada experimental o es vol veure d'on surten els resultats i les conclusions obtingudes.

Cal tenir en compte que els experiments s'han realitzat en ordinadors portàtils, cosa que vol dir que els resultats en quant al temps que triga en executar-se el programa pot ser prou major del que seria si s'hagués executat en un ordinador de sobretaula.

2.1. Conjunt d'operadors

Tal com s'ha comentat prèviament, podríem dir que disposen de 3 tipus de moviments:

1. Intercanvi d'usuaris entre cotxes
2. Agafar o deixar abans un usuari dintre del mateix cotxe
3. Moure un usuari de cotxe i si cal eliminar el cotxe del que se'l treu

Un cop classificats els operadors en 3, hem fet proves per mirar quin subconjunt d'operadors dóna millors resultats. Per això hem executat 10 vegades el programa, amb valors de seed entre 1000 i 1009 i amb els diferents subconjunts possibles d'operadors i hem pres nota de la distància, el nombre de cotxes resultants i el temps que ha trigat a resoldre cada problema.

Abans de començar, cal tenir en compte que l'únic operador que ens permet eliminar cotxes és el 3r, per tant tot apunta a que aquest hauria d'estar inclòs en el conjunt d'operadors que acabem utilitzant. El dubte tanmateix estaria en si ens surt a compte contemplar els altres 2 operadors.

º	{ 1 }	{ 2 }	{ 3 }	{ 1 , 2 }	{ 1 , 3 }	{ 2 , 3 }	{ 1 , 2 , 3 }
cotxes	100	100	76.1	100	76.8	76	77.2
dist	1205,9	1936,3	1830,4	1149,2	1119,6	1772,7	1073,4
temps	58306	54,9	3891,4	74635	93196	6051,5	109024

Observant els resultats obtinguts, representats a la taula anterior, hem decidit que el millor subconjunt d'operadors és el que conté tots els operadors plantejats, ja que és el que obté resultats amb valors per la distància total recorreguda menor.

Tal com esperàvem, utilitzant el 3r operador és quan s'obtenen els millors resultats.

Tot i això també volem puntualitzar el fet que si treballèssim amb els operadors 1 i 3 (és a dir, sense el d'agafar i deixar els passatgers abans) o amb el 1 i el 2 (sense el de moure usuaris d'un cotxe a un altre i per tant sense possibilitat d'eliminar cotxes) obtindríem resultats prou bons en comparació amb els que obtenim utilitzant els 3 moviments tardant una mica menys. Per tant és important recalcar que el 1r moviment ens ajuda molt a obtenir bons resultats.

2.2. Estratègia de generació de solucions inicials

En aquest apartat mirarem de decidir quin dels dos sistemes de generació de solucions inicials dóna millors resultats. Tindrem en compte que a primer cop d'ull el 2n generador sembla que hauria de ser el millor ja que de bon començament intenta distribuir la gent de manera que es redueixin el nombre de quilòmetres recorreguts pels cotxes.

Per tal de determinar quina de les dues estratègies per generar solucions inicials dóna millors resultats hem executat el programa 10 vegades amb cadascun dels generadors de solucions inicials utilitzant tots els moviments plantejats, tal com s'ha decidit en l'apartat anterior i amb 200 usuaris, 100 conductors i seeds amb valors entre 1000 i 1009.

Un cop fetes totes les proves hem obtingut que tant la quantitat de cotxes utilitzats en la solució final com el temps que triga en acabar l'execució són molt similars i no influeixen en la tria del generador. Tanmateix el 1r generador troba solucions que ens mitjana recorren menys distància que les trobades utilitzant el 2n generador. En aquest cas, amb 200 usuaris i 100 conductors, ens hem trobat que es recorren un 70 km menys en mitjana, dada prou significativa que ens fa decantar-nos per fer servir el 1r generador abans que el segon.

	Número de cotxes	**Distància en km**	**Temps d'execució**
1r generador	77,2	1070,83	109024,8
2n generador	76,8	1141,2	106800,1

2.3. Paràmetres Simulated Annealing

A l'hora d'executar Simulated Annealing cal tenir en compte 4 paràmetres:

1. steps: quantitat d'iteracions que s'executen.
2. stiter: quantitat d'iteracions per pas de temperatura.
3. k: constant que multiplica la temperatura en quan es calcula el paràmetre per mirar la probabilitat de canviar d'estat.
4. lambda: quantificador de la velocitat a la que es redueix la temperatura.

Per tal de decidir quins valors són millors per les execucions realitzarem diversos experiments utilitzant els operadors i el generador de solucions inicials decidits en els apartats anteriors. Utilitzarem l'escenari generat pel seed 1000, 200 usuaris i 100 conductors.

Inicialment buscarem com orientar els paràmetres individualment per tal que els resultats millorin i després ho posarem en comú per trobar la millor combinació de valors pels 4 paràmetres. Les dades mostrades a la taula 3.3 són la mitjana de 10 execucions per cada conjunt de valors.

Un cop mirats fetes totes les proves podem concluir que :

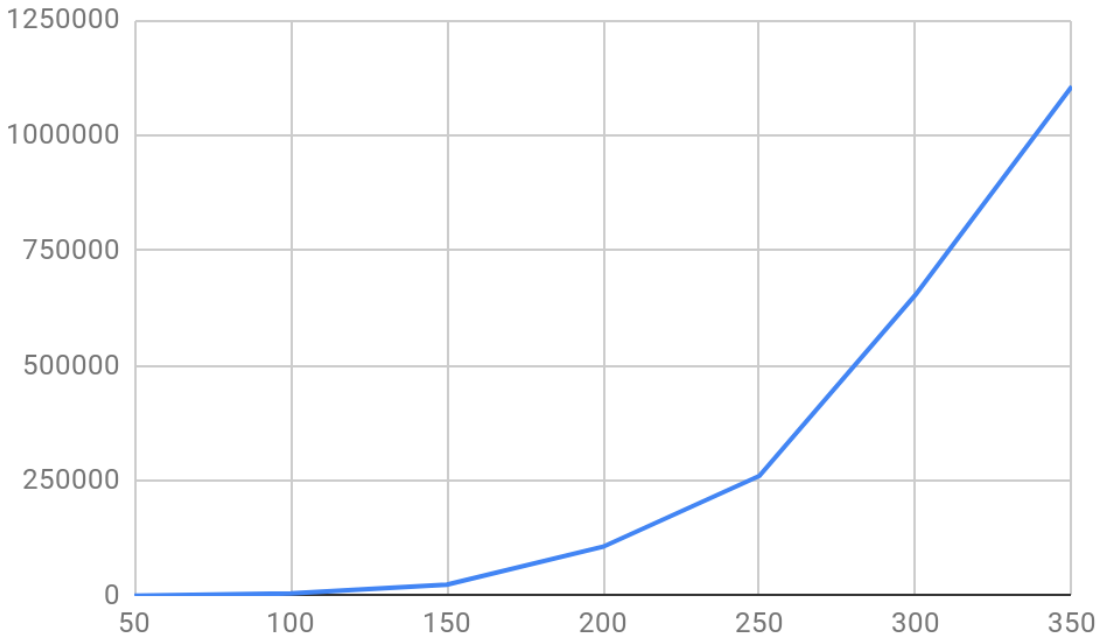
- com més gran sigui el valor de steps, menys distància es recorre, tanmateix la distància mitjana deixa de reduir-se a una velocitat significativa a partir de 50.000 steps, per tant, per evitar que no es gasti més temps per millorar tant poc, ens quedem amb aquest valor com l'òptim.
- el valor de stiter no sembla influir de manera clara sobre els resultats obtinguts, per tant hem decidit quedar-nos amb el valor 1000 ja que no semblava desproporcionat tenint en compte el valor escollit pel paràmetre steps.
- com més petit és el valor de la K, millor. Per tant ens quedem amb K = 1.
- com més disminueix el valor de lambda, menys distància total es recorre en la solució obtinguda. Tanmateix la distància deixa de disminuir de manera significativa a partir de lambda = 0,001, per tant ens quedem amb aquest valor.

Així doncs el conjunt de paràmetres que sembla ser el que obté millors resultats amb el Simulated Annealing i el problema que hem plantejat nosaltres són steps = 50000, stiter = 1000, K = 1 i Lambda = 0,001, que en mitjana ens retornen un plantejament final on s'utilitzen 70,4 cotxes i es recorren 954,66 km i la solució s'obté en poc menys de 3 segons.

2.4. Evolució temps d'execució

En aquest apartat comparem la evolució del temps d'execució en funció del nombre de persones apuntades a l'aplicació. La intenció inicial era realitzar tandes de 10 execucions amb quantitats d'usuaris que anèssin incrementant de 100 en 100 a cada tanda, tanmateix ens hem trobat que en els nostres portàtils no es pot executar per a 400 usuaris i 200 conductors, per tant hem hagut de fer càlculs intermitjos per tal d'entreveure, a menor escala, el comportament del temps en funció de la quantitat d'usuaris. Així doncs hem fet 10 execucions del programa per cada situació augmentant el nombre d'usuaris de 50 en 50 començant des de 50 i utilitzant la 1a heurística i Hill Climbing.

A continuació adjuntem una gràfica amb la representació de la relació del nombre d'usuaris envers la mitjana de temps en milisegons que es triga a resoldre el problema amb Hill Climbing per cadascun dels casos.



Amb aquesta gràfica podem deduir que la funció sembla ser exponencial o quadràtica respecte el nombre d'usuaris, tanmateix com que no hem pogut fer proves amb nombres elevats en els nostres portàtils, no podem precisar més els nostres resultats en quant a això.

2.5. Funció heurística amb Hill Climbing

Per estudiar els resultats d'aplicar la primera funció heurística o la segona, compararem els resultats obtinguts d'aplicar hill climbing 10 vegades amb cadascuna utilitzant els seeds entre 1000 i 1009 amb l'escenari inicial amb 200 usuaris i 100 conductors.

Un cop obtinguts els resultats de les execucions podem veure que amb la 1a heurística s'obtenen la solució final recorre menys distància, uns 65 km de mitjana menys que amb la 2a heurística en el cas amb 200 usuaris i 100 conductors, mentre que amb la 2a heurística els resultats obtinguts utilitzen 4 cotxes menys de mitjana amb el mateix cas.

Cal notar també que utilitzant la 1a heurística el temps d'execució en mitjana és superior en 20 segons respecte a quan s'utilitza la 2a heurística.

	Número de cotxes	**Distància en km**	**Temps d'execució**
1a heurística	77,2	1070,83	109024,8
2a heurística	73,3	1135,33	90147,2

2.6. Funció heurística amb Simulated Annealing

Per a estudiar els resultats de les heurístiques amb Simulated Annealing, realitzarem 4 execucions per cada seed entre 1000 i 1009 per extreure mitjanes i obtenir la mitjana total a partir d'aquestes, degut a la natura aleatoria del Simulated Annealing, sobre l'escenari amb 200 usuaris i 100 conductors, amb els paràmetres que hem trobat òptims prèviament: steps = 50000, stiter = 1000, K = 1 i lambda = 0,001.

		1a Heurística			2a Heurística	
	#cotxes	Distància	Temps	#cotxes	Distància	Temps
Mitjana Total	**87**	**938.5**	**2389,5**	**83,5**	**949,77**	**2381,7**

Com podem observar, en general els temps son bastant similars, i es nota com la 2a heurística consegueix minimitzar la quantitat de cotxes a canvi d'una distància recorreguda major.

2.7. Proporció de conductors respecte la mida

Per a trobar la proporció òptima de conductors farem servir la següent metodologia:

Utilitzarem Simulated Annealing amb els paràmetres trobats en l'experiment anterior, amb seed 1000, i com a valor de N farem servir els mateixos de l'experiment del temps d'execució.

Per cada valor de N, realitzarem una cerca binària sobre M, on el rang de valors va de 1 a N. El valor que s'assignarà al nombre de conductors sera la meitat de l'interval actual, i degut a la natura aleatoria del Simulated Annealing, executem 5 iteracions i obtenim la mitjana dels resultats per més consistència. D'aquests resultats, analitzem el percentatge de cotxes que no arriben a temps: si és molt proper a 0, busquem pel subinterval dret, i en cas contrari, pel subinterval esquerre. Aquest algorisme para quan els extrems de l'interval tenen 1 de diferencia. Un cop finalitzada la cerca, agafem el valor d'M més petit, i amb aquest obtenim la proporció. A continuació una mostra de resultat d'aquest procediment:

	M	Temps (ms)	Distància	%Cotxes no solució	Ratio
N = 50	25	722.75	22.782.501	0.0	4,166666667
	12	687.5	25.130.002	0.0	
	6	1007.5	286.075	79	
	9	654.5	279.575	31	
	10	686.0	279.525	17,5	
	11	911.75	297.05	14	

Realitzant això per cada valor de N podem extreure que la proporció adequada és aproximadament M = N/4,1. Si executem amb aquesta proporció, el temps es veu consistentment reduït pel simple fet de que ens apropem més rapidament a la solució. Com aquesta proporció s'ha trobat en base al valor de M més petit que produeix una solució , la qualitat també es manté.

3. Apèndix

Taula 3.1: Mostra els resultats obtinguts provant els operadors amb 10 exemples diferents

Seed		{ 1 }	{ 2 }	{ 3 }	{ 1 , 2 }	{ 1 , 3 }	{ 2 , 3 }	Tots
1000	**cotxes**	100	100	73	100	78	74	78
	dist	1285	2026	1877	1248	1179	1810	1153
	temps	62182	166	6732	63665	106589	9708	112256
1001	**cotxes**	100	100	77	100	77	78	78
	dist	1241	1979	1893	1124	1130	1828	1071
	temps	49156	44	3572	89145	98690	4851	103588
1002	**cotxes**	100	100	78	100	77	77	78
	dist	1167	1864	1764	1119	1092	1732	1042
	temps	54213	50	3508	77778	92824	5046	106804
1003	**cotxes**	100	100	75	100	76	74	77
	dist	1167	1897	1795	1111	1096	1729	1058
	temps	54213	39	3986	65773	82604	5985	97449
1004	**cotxes**	100	100	77	100	75	77	76
	dist	1164	1914	1773	1109	1075	1759	1040
	temps	54791	21	3284	74855	98512	3966	120866
1005	**cotxes**	100	100	76	100	74	76	75
	dist	1178	2001	1897	119	1094	1804	1077

	temps	59098	57	3519	82490	95889	6417	109206
1006	**cotxes**	100	100	80	100	79	79	78
	dist	1161	1902	1834	1131	1116	1758	1057
	temps	65074	55	2168	72572	85978	4660	102074
1007	**cotxes**	100	100	74	100	77	74	76
	dist	1239	1894	1740	1156	1136	1722	1082
	temps	57835	40	5420	72289	88541	6260	106525
1008	**cotxes**	100	100	76	100	78	74	78
	dist	1220	1912	1863	1192	1145	1791	1070
	temps	70187	39	3622	70045	94300	8496	120696
1009	**cotxes**	100	100	75	100	77	75	78
	dist	1223	1912	1858	1183	1130	1794	1084
	temps	62766	38	3203	72763	88024	5126	110782
Mitja	**cotxes**	**100**	**100**	**76.1**	**100**	**76.8**	**76**	**77.2**
	dist	**1205,9**	**1936,3**	**1830,4**	**1149,2**	**1119,6**	**1772,7**	**1073,4**
	temps	**58306**	**54,9**	**3891,4**	**74635**	**93196**	**6051,5**	**109024**

Taula 3.2: Mostra els resultats de comparar els 2 generadors de solucions inicials

	1r generador (el més senzill)			**2n generador (el més complex)**		
seed	**#cotxes**	**distància**	**temps**	**#cotxes**	**distància**	**temps**
1000	78	1153,2	112256	76	1152,6	122676
1001	78	1071,1	103588	72	1148,3	101733

1002	78	1042,8	106804	82	1181,7	101455
1003	77	1058,6	97449	79	1114,1	100439
1004	76	1040,2	120866	71	1078,8	108811
1005	75	1077,0	109206	71	1144,2	110906
1006	78	1057,7	102076	78	1189,0	100695
1007	76	1082,9	106525	82	1138,7	115643
1008	78	1070,5	120696	81	1105,2	103990
1009	78	1084,3	110782	76	1159,4	101653
Mitjana	**77,2**	**1070,83**	**109024,8**	**76,8**	**1141,2**	**106800,1**

Taula 3.3: Resultats de la prova de diferents paràmetres per la execució del Simulated Annealing

Steps	**Stiter**	**K**	**Lambda**	**# cotxes**	**distància**	**temps**
100	**100**	**5**	**0,001**	94,6	1987,9	10,6
1000	**100**	**5**	**0,001**	82,2	1535,11	80,4
5000	**100**	**5**	**0,001**	84,4	1289,9	316,9
10000	**100**	**5**	**0,001**	76,8	1106,8	594,6
50000	**100**	**5**	**0,001**	72,8	957,62	3748,0
100000	**100**	**5**	**0,001**	84,4	953,4	7513,8
50000	**10**	**5**	**0,001**	80,2	986,8	3401,4
50000	**100**	**5**	**0,001**	73,0	954,98	3469,6
50000	**1000**	**5**	**0,001**	78,0	980,92	3386,8
50000	**5000**	**5**	**0,001**	74,6	1007,6	3649,2
50000	**10000**	**5**	**0,001**	66,2	998,72	3502,0
50000	**1000**	**1**	**0,001**	70,4	954,66	2967,6
50000	**1000**	**3**	**0,001**	76,2	963,32	3141,4
50000	**1000**	**5**	**0,001**	72,6	971,5	3121,2
50000	**1000**	**10**	**0,001**	86,0	985,4	4080,6
50000	**1000**	**25**	**0,001**	82,4	1005,7	3704,6
50000	**1000**	**1**	**0,0001**	68,8	947,92	2845,0

50000	**1000**	**1**	**0,0005**	72,7	960,61	3143,5
50000	**1000**	**1**	**0,001**	**70,4**	**954,66**	**2967,6**
50000	**1000**	**1**	**0,005**	70,7	956,8	3434,2
50000	**1000**	**1**	**0,01**	79,2	958,56	3274,6
50000	**1000**	**1**	**0,05**	78,1	1053,3	1057,3
50000	**1000**	**1**	**0,1**	76,7	1121,62	486,4

Taula 3.4: Temps d'execució del programa utilitzant Hill Climbing i variant el nombre d'usuaris i el de conductors contemplats en cada cas

	N = 50 M = 25	**N = 100 M = 50**	**N = 150 M = 75**	**N = 200 M = 100**	**N = 250 M = 125**	**N = 300 M = 150**	**N = 350 M = 175**
1000	665	6501	31737	112256	296190	793300	1260021
1001	453	5735	19311	103588	308219	667270	1021638
1002	488	5120	23898	106804	242672	699931	1143161
1003	311	5442	22559	974499	243340	610466	1131864
1004	351	5273	22672	120866	262291	625857	1143718
1005	387	4698	21058	109206	266878	586842	1065920
1006	238	4824	25578	102076	227755	616719	1145954
1007	423	5416	23552	106525	253545	667815	1064402
1008	267	4689	22321	120696	267112	648512	1016917
1009	298	4272	31810	110782	238083	622104	1075621
Mitjana:	388,1	5198	24449,6	106800,1	260608,5	65388,6	1106921,6

Taula 3.5: Comparació dels resultats obtinguts utilitzant les dues heurístiques

	1a Heurística			**2a Heurística**		
seed	**#cotxes**	**distància**	**temps**	**#cotxes**	**distància**	**temps**
1000	78	1153,2	112256	70	1220,4	98268
1001	78	1071,1	103588	76	1126,2	91254
1002	78	1042,8	106804	73	1067,0	98924

1003	77	1058,6	97449	71	1118,3	77743
1004	76	1040,2	120866	73	1112,2	88021
1005	75	1072,0	109206	72	1129,0	89361
1006	78	1057,7	102076	80	1110,5	87372
1007	76	1082,9	106525	71	1187,0	86781
1008	78	1070,5	120696	73	1135,7	97763
1009	78	1084,3	110782	74	1147,0	85985
Mitjana	**77,2**	**1070,83**	**109024,8**	**73,3**	**1135,33**	**90147,2**

Taula 3.6: Resultats de les dos heuristiques amb diferents seeds empleant Simulated Annealing

		1a Heuristica			2a Heuristica	
Seed	#cotxes	Distància	Temps	#cotxes	Distància	Temps
1000	74	965,6	3063	78	965,7002	2424
1000	74	961,3	1823	68	992,90015	2164
1000	94	957,7	2491	87	975,8999	2408
1000	75	974,4	2179	77	977,3001	1945
Mitjana 1000	**74,5**	**963,45**	**2335**	**77,5**	**976,6**	**2286**
1001	100	956	2868	100	967,6	2652
1001	85	911,8	2020	100	964,0002	2806
1001	98	947,9	2532	86	955,5002	2055
1001	90	999	2542	85	963,9001	2405
Mitjana 1001	**94**	**951,95**	**2537**	**93**	**963,95015**	**2528,5**
1002	88	942,3	2413	63	950,39984	1502
1002	83	912,6	2225	88	952,1998	2311
1002	67	879,9	1833	78	920,2998	1977

1002	85	895,2	2044	60	947,4999	1577
Mitjana 1002	**84**	**903,9**	**2134,5**	**70,5**	**948,94987**	**1777**
1003	100	940,8	2737	84	966,8001	2694
1003	91	936,5	2380	100	943,8996	3124
1003	64	927,3	1915	78	941,2999	1967
1003	86	910,2	2003	78	935,10004	2023
Mitjana 1003	**88,5**	**931,9**	**2191,5**	**81**	**942,59975**	**2358,5**
1004	100	918,3	2956	81	917,9999	1999
1004	99	953,3	2654	100	927	2944
1004	71	927,4	1850	100	916,89984	2632
1004	85	915,2	2144	83	907,1001	2335
Mitjana 1004	**92**	**922,85**	**2399**	**91,5**	**917,44987**	**2483,5**
1005	63	906,9	2936	100	948,3	2785
1005	70	919,6	1845	81	961,7998	2879
1005	100	942,9	3129	94	922,9	2476
1005	100	983,7	3321	100	935,9001	2683
Mitjana 1005	**85**	**931,25**	**3032,5**	**97**	**942,10005**	**2734**
1006	100	994,8	3000	100	1009,1003	3144
1006	79	960,7	1844	100	994,90015	3326
1006	100	993,5	2951	69	967,80005	1775
1006	57	947,8	3157	82	957,20013	1978
Mitjana 1006	**89,5**	**977,1**	**2975,5**	**91**	**981,3501**	**2561**
1007	92	947	2910	100	969,8001	2811
1007	100	927,9	2872	100	937,1001	3081
1007	83	934	2103	83	946,7001	2327
1007	100	959	3122	63	935,49994	1723
Mitjana 1007	**96**	**940,5**	**2891**	**91,5**	**941,9001**	**2569**
1008	69	947,4	2100	62	916,2	1558
1008	75	931,7	1955	54	971,5	1885
1008	71	908,8	1994	81	927,89996	2105
1008	67	911,3	1892	71	946,1	1789
Mitjana 1008	**70**	**921,5**	**1974,5**	**66,5**	**936,99998**	**1837**
1009	88	916,1	2295	75	946,39984	2066
1009	88	941,5	2431	95	952,3001	2929
1009	76	913,5	1975	100	976,1999	2760

1009	95	960,6	2677	**72**	**969,4999**	**2105**
Mitjana 1009	**88**	**928,8**	**2363**	**85**	**960,9**	**2432,5**
Mitjana Total	**87**	**938,575**	**2389**	**83**	**949,77481**	**2381.75**

Taula 3.7: Resultats de la cerca binària per obtenir el valor de la M òptim.

	M	Temps (ms)	Distància	%Cotxes no solució	Ratio
N = 50	25	722.75	22.782.501	0.0	4,166666667
	12	687.5	25.130.002	0.0	
	6	1007.5	286.075	79	
	9	654.5	279.575	31	
	10	686.0	279.525	17,5	
	11	911.75	297.05	14	
N = 100	50	668.0	42.099.997	0.0	4,545454545
	25	697.5	475.325	0.0	
	12	1503.75	58.327.496	83	
	18	912.0	536.625	26	
	21	888.75	558.3	8	
	23	748.5	496.45	0.0	
	22	807.75	483.675	0.0	
N = 150	75	1261.75	686.325	0.0	4,054054054
	37	1073.5	75.922.504	0.0	
	18	2203.5	852.275	79	
	27	1882.75	864.25	38	
	32	1251.5	787.525	2	
	34	1358.75	768.975	1	
	35	1343.0	76.017.505	1	
	36	1209.25	706.2	0.0	
N = 200	100	2330.0	97.450.006	0.0	4,166666667
	50	1611.25	1.006.625	0.0	
	25	2399.5	1.161.775	85	
	37	2211.25	1135.05	32	
	43	1989.5	1129.45	10	

	46	2021.25	1137.4	7	
	48	1946.75	10.494.249	0.0	
	47	1780.25	10.881.499	2	
N = 250	125	4863.25	12.772.998	0.0	4,032258065
	62	2243.75	13.298.501	0	
	93	2912.5	1.259.325	0.0	
	77	2513.25	12.637.749	0.0	
	69	2297.0	13.101.001	0.0	
	65	2182.25	1319.5	0.0	
	63	2717.5	1.363.425	1	
	64	2477.25	13.677.001	1	
N = 300	150	4737.75	15.589.752	0.0	3,75
	75	3499.0	16.922.999	3	
	112	4115.75	15.469.001	0.0	
	93	3158.75	15.542.249	0.0	
	84	4745.25	15.723.752	0.0	
	79	3192.5	1.664.425	1	
	81	2923.75	15.978.999	0.0	
	80	2838.75	16.174.502	0.0	

Ratio Median
4,11036036