Trabajo Práctico 2

Logística centralizada de primera milla

Introducción

En el siguiente trabajo buscamos evaluar distintas estrategias para optimizar las operaciones de una red de depósitos. Se nos contrató para asistir a la start-up ThunderPack por lo que tendremos que encontrar buenas soluciones para su problema de logística.

Implementación

Para la implementación del trabajo práctico se utilizaron tres estructuras de datos (tres clases):

- ReadInstance: Estructura que se encarga de obtener y almacenar los datos de los archivos.
- Solución: Estructura que se encarga de almacenar una solución para el problema de GAP. Su método más importante es el Assign():
    - Assign se encarga principalmente a asignar a un vendedor a un determinado depósito, en esta misma se calculan todos los aspectos relacionados con esta asignación (Objective_value, Capacidades Restantes del depósito utilizado, cantidad de vendedores asignados). Como complemento, esta función se encarga de que en el caso de que como parámetro se pase un vendedor que ya está asignado, también se calculen primeramente los aspectos relacionados a la desasignación para luego reasignar.
- MetaHeuristica: Estructura que se encarga de procesar la instancia para generar una solución. Principalmente contiene a las heurísticas, los operadores de búsqueda local y la metaheurística.

Para compilar los archivos y correr el main se debe ejecutar en la terminal:

```
make
.\tp_2
```

Heurística 0 - Los vendedores eligen

Se conoce que el modelo empleado actualmente por la empresa consta de que los vendedores elijan a qué depósito llevar sus paquetes. Decidimos simular esta estrategia en la Heurística 0 para poder comparar posteriormente si nuestras alternativas serán mejores o no.

Heurística 1 - Los depósitos eligen

Para nuestra primera heurística pensamos en revertir los roles de la estrategia actual. En esta oportunidad cada depósito elegirá a sus vendedores por cercanía hasta saturar la capacidad (o no poder cubrir alguna de las demandas restantes). Así cada depósito se irá llenando uno por vez hasta que todos los vendedores tengan un depósito asignado.

- Pros: los depósitos podrán saber de antemano con qué demanda estarán trabajando.
- Cons: los últimos depósitos probablemente tengan asignaciones menos eficientes al tener menos vendedores para elegir.

Heurística 2 - Capitanes de equipo

Para nuestra segunda heurística decidimos compensar el problema de la anterior y esta vez los depósitos irán eligiendo un vendedor cada uno. Creemos que de esta forma se realizarán asignaciones más justas al ir turnándose como lo hacen los capitanes de equipo al ir eligiendo sus integrantes.

- Pros: asignaciones más justas.
- Cons: tarda más en ejecutarse.

Operador de Búsqueda Local 1 - Swap

Como primer operador elegimos Swap porque consideramos que como para nuestras dos heurísticas quienes eligen son los depósitos, Swap permitiría que dos vendedores intercambien destinos ahorrando distancias recorridas.

Operador de Búsqueda Local 2 - Relocate

Elegimos Relocate como segundo operador porque existe la posibilidad de que no hayan intercambios posibles (Swap no útil) pero que igual hubiese lugar para que un vendedor cambie de depósito.

Metaheurística - Variable Neighborhood Descent

Con el propósito de combinar la posibilidad de que vendedores puedan intercambiar depósitos entre ellos tanto como la posibilidad de que ya no hayan intercambios por realizar, decidimos implementar VND para nuestra heurística con N1: Swap y N2: Relocate. Así la solución se irá refinando

Experimentación y discusión

Para realizar la experimentación construimos el archivo *experimentacion_gap.csv* donde almacenamos los resultados de las heurísticas y el resultado de aplicar la metaheurística a cada una. En cada instancia evaluada pintamos de verde al mejor resultado y en azul en caso de empate.

Input	Heuristica 0	MetaH 0	Heuristica 1	MetaH 1	Heuristica 2	MetaH 2
gap_a/a05100	1719	1699	1932	1694	1712	1694
gap_a/a05200	3259	3238	3800	3235	3289	3235
gap_a/a10100	1370	1362	1732	1358	1416	1362
gap_a/a10200	2657	2629	3290	2631	2698	2623
gap_a/a20100	1177	1169	1589	1162	1275	1161

gap_a/a20200	2369	2351	2804	2346	2458	2341
gap_b/b05100	3419	3379	3161	3091	3052	3052
gap_b/b05200	5778	5616	5627	5522	5494	5494
gap_b/b10100	2411	2308	2320	2249	2175	2162
gap_b/b10200	5076	4808	4381	4325	4316	4299
gap_b/b20100	2182	1981	1956	1913	1950	1925
gap_b/b20200	3670	3465	3524	3370	3570	3239
gap_e/e05100	6699	5589	7053	6702	6516	5241
gap_e/e05200	17370	13432	13245	12737	11919	11440
gap_e/e10100	4332	4081	5782	4711	5486	4018
gap_e/e10200	10511	10051	9652	9432	8816	8472
gap_e/e10400	33785	33173	34239	34127	31047	30635
gap_e/e15900	81010	77697	69244	68869	64345	64289
gap_e/e20100	2869	2716	5895	2483	4001	2564
gap_e/e20200	8565	7745	9024	7499	8519	6739
gap_e/e20400	30369	29379	31538	31512	30196	27323
gap_e/e30900	65605	64791	62590	62509	56732	52218
gap_e/e40400	25353	23879	23263	22496	23566	20970
gap_e/e60900	58855	56450	48928	48016	43666	40252
gap_e/e201600	130901	127824	117012	116632	109191	109184
gap_e/e401600	117763	115951	107849	107682	101163	98730
gap_e/e801600	107630	104415	89187	89063	79212	73442

Observaciones

- Obtenemos mejores resultados post-Metaheurística cuando elegimos la Heurística 2. Creemos que esto se debe a que la construcción de la solución en H2 es más “aleatoria” que en H0 donde se van asignando uno por uno los vendedores y que en H1 donde van uno por uno los depósitos. Esta forma de asignar, al ir llenando las capacidades de los depósitos de forma medianamente pareja, deja más lugar a la MH para que realice los cambios que crea necesarios sin estancarse porque las capacidades de algunos pocos depósitos estén saturadas.

- Hay una coincidencia interesante entre las instancias *gap_b/b20100* y *gap_e/e20100* donde gana la Metaheurística asociada a H1 y además para *gap_a/a20100* queda en segundo lugar solo por un punto. A partir de esto nos inclinamos a creer que la distribución 20 depósitos - 100 vendedores favorece a H1.

- La Heurística 0 no logra mejoras competentes post-Metaheurística descubriendo que es la estrategia más débil de las 3 aun habiendo presentado mejores resultados que H1 y H2 pre-Metaheurística.

Conclusión

Para las instancias de GAP queda demostrado por la experimentación que la mejor estrategia es utilizar **Heurística 2 - Capitanes de equipo.** Alternativamente si la empresa tuviera siempre una distribución 20 depósitos - 100 vendedores le recomendamos utilizar la estrategia de **Heurística 1 - Los depósitos eligen**.

Análisis del caso real

En esta segunda etapa de experimentación construimos el archivo *experimentacion_real.csv* donde además de los resultados incluimos el tiempo de ejecución para tener un segundo parámetro de elección. Pintado en verde se encuentra el mejor resultado y en naranja el menor tiempo.

Input	**Heuristica 0**	**MetaH 0**	**Tiempo Tot0**
real/real_instance	788.2	714.5	63

Input	**Heuristica 1**	**MetaH 1**	**Tiempo Tot1**
real/real_instance	1770.3	727	417

Input	**Heuristica 2**	**MetaH 2**	**Tiempo Tot2**
real/real_instance	8857	705.6	303

Observaciones

- La Metaheurística es considerablemente mejor cuando emplea la Heurística 2. Esto era de esperarse a partir de la experimentación realizada para GAP.
- El tiempo de ejecución es muchísimo menor para la Heurística 0 y tiene sentido dado que la disminución en la función objetivo es muy poca (hubieron menos cambios realizados) haciendo que tarde menos.
- La Heurística 1 parece ser la más estable en el sentido de que no devuelve un valor excesivamente elevado pre-Metaheurística y post-Metaheurística logra un resultado no muy alejado del resto.

Conclusión

Con el análisis de la instancia real confirmamos que la mejor estrategia es la **Heurística 2 - Capitanes de equipo** y le recomendamos a ThunderPack que la utilice dado que es mejor que su estrategia actual por más que tarde unos segundos extra en computar la solución.

Apéndice - Mejora no implementada

Una mejora que no llegamos a implementar por falta de tiempo fue considerar soluciones infactibles al momento de aplicar búsquedas locales. Si bien esto expandirá el vecindario de manera indeterminada, el mismo se podría reducir utilizando ciertos criterios de aceptación para este tipo de soluciones infactibles. Básicamente se buscarían soluciones en donde se sobresaturan los depósitos, para luego intentar realizar una reasignación que lo vuelva a su estado factible. Esta idea podría tener sentido, ya que óptimamente voy a querer saturar lo más exactamente posible a los depósitos, sabiendo que hay una mayor cantidad de soluciones infactibles cerca del óptimo. Para estos casos infactibles también se podría aplicar una penalización para el porcentaje de la demanda que excede la capacidad del depósito, al estilo de lo que se hacía para los vendedores no asignados.