Modelling and solving the periodic delivery planning problem as a multiple-choice integer program

Abdelkader Sbihi

Ecole de Management de Normandie, Axe Logistique-Terre-Mer-Risque
30 rue de Richelieu, 76087 Le Havre Cedex - France
a.sbihi@em-normandie.fr

In this paper we consider a logistical problem that is the periodic delivery planning problem (PDP) (Cachon [1] et Gaur and Fisher [3]). It consists of a single stockpoint where a single stock keeping unit namely SKU is stored. We assume that the time is divided into equal length bins or intervals such that each period length is one weekday. The critical period is determined by the delivery planning. The aim is to find the number of the replenishment actions to be planned from the backroom to the frontroom for a certain SKU and the number of the order lines and quantities per weekday for a given delivery plan. Also, the planned replenishments have been transformed into idle replenishments.

We developed an adapted perturbed large neighborhood search algorithm (APLNS) for the PDP. It combines reactive tabu search with some specific neighbourhood search strategies to approximately solve the problem. The tests were conducted on randomly generated instances. The results outperform those of the Cplex solver and demonstrate the high quality of the two approach versions.

The problem objective is to find the number of deliveries per week, the weekdays that corresponds to servicing the stockpoint and the amount of SKU's to deliver to the store for each period of delivery in order to minimize the logistical costs. We have considerd idle costs replenishment per store based on SKU level. The transportation costs are based on the set of SKU's deliveries to the stockpoint that are assumed to have the same delivery planning. The model aims to find the optimal delivery schedules for the products of the stores.

The problem has been modeled and derived as a multiple-choice integer program (Nauss [4], Sbihi [2]) as follows:

$$(PDP)\begin{cases} \text{Minimize } Z(x) = \displaystyle\sum_{i=1}^{m}\sum_{j=1}^{n} v_{ij}^{p} x_{ij} + v_{\delta}\sum_{k} z_k & \\ \text{s.c:} \quad \displaystyle\sum_{i}\sum_{j} w_{ijk} x_{ij} < C, & \forall k \ (1) \\ \displaystyle\sum_{j} x_{ij} = 1, & \forall i \ (2) \\ \displaystyle\sum_{j}^{n} x_{ij} < N y_j, & \forall i \ (3) \\ \displaystyle\sum_{j}^{n} y_j \leq M, & (4) \\ \displaystyle\sum_{j} r_{jk} y_j \geq z_k, & \forall k \ (5) \\ x_{ij}, y_j, z_k \in \{0,1\}, & (6) \end{cases}$$

Constraint (1) ensures that the delivery amount depends on the planning assignment and must fit with the vehicle. Constraint (2) ensures that each SKU is assigned to exactly one and only one planning. Constaint (3) ensures that each SKU can only be assigned to active plannings. Constraint (4) ensures a maximum number of active plannings M. Constraint (5) ensures that the delivery days are determined by the active plannings.

Problem data

$$x_{ij} = \begin{cases} 1 & \text{if SKU } i \text{ is assigned to planning } j \\ 0 & \text{otherwise} \end{cases}$$

$$y_j = \begin{cases} 1 & \text{if planning } j \text{ is in use by the store} \\ 0 & \text{otherwise} \end{cases}$$

$$z_k = \begin{cases} 1 & \text{if the delivery occurs on weekday } k \\ 0 & \text{otherwise} \end{cases}$$

$$r_{jk} = \begin{cases} 1 & \text{if delivery on weekday } k \text{ occurs according to planning } j \\ 0 & \text{otherwise} \end{cases}$$

v_{ij}^p: idle replenishment costs for SKU i if the delivery occurs according to schedule j
v_δ: the transportation costs per delivery, a setup cots
N: number of active plannings
w_{ijk}: quantity to deliver on weekday k for SKU i according to delivery planning j
C: vehicle capacity
M: keeps track of the setups, i.e., a contant to ensure that we never exceed the total demand in weekday k

1 Solution approach

In our approach, we develop an adapted perturbed large neighborhood search (APLNS) to solve the PDP. It consists to react on the search by downgrading the solution taking account of these estimations improvements addressing the neighborhood search. The algorithm is a self-tuning heuristic. The main idea is: (i) to generate an initial solution; (ii) to operate some neighbourhood search selections; (iii) to make a choice regarding the aspiration criteria until the entire neighbourhood is searched and (iv) then to update the current solution.

1.1 A greedy procedure

The algorithm selects one item at the first planning and iteratively completes the partial solution by picking one item per planning action. The resulting string (0–1 vector) is the initial solution. In these early steps, we do not focus too much on the quality of the built solution but on quickly finding a feasible one.

1.2 A search objective

A diversification device is applied by disrupting (perturbing) the objective function to downgrade the cycling solution and skip to unexplored new search areas. The process is dynamically adjusted to select solution component to downgrade. If all previous solutions have failed to improve the current solution, the last best known solution is stored in a list banning last known best solution value and performs some downgrading operations. On the other hand, when the infeasibility is authorized, the disturbed search is combined with a high level of flexibility for exploring much larger areas of the search space. The feasibility gap can be measured/controlled by a penalty term in the objective function.

The approach takes into account both feasible and infeasible solutions that are inserted into the solutions list. A compromise between feasibility and infeasibility is governed by a penalty factor for non-feasibility. To explore the solution space depends on the penalty factor, which is adjusted dynamically, and a balance between feasibility/infeasibility is provided by the list of forbiddance. It can be seen as a path relinking to improve the solution.

2 Conclusion

We proposed an adaptive neighbourhood search-based reactive tabu to solve the PDP. The problem is not well known, but is very relevant in real-life applications. The two perturbed adaptive neighborhood search versions are approximate-based algorithms using both simple and modified reactive neighbourhood local search. The first approach of APLNS is a local search combining downgrading and releasing strategies. The downgrading strategy is used to diversify the search, and the releasing

technique is introduced to escape to local optima while the second version introduces a memory forbiddance list, which replaces the releasing strategy.

Computational results show that the first version yields good solutions within a very short computing time, while the second one yields high-quality solutions, reaching the optimal/best solutions for several instances, within a reasonable CPU time. We also used an infeasibility controlled allowance for a more efficient search.

References

1. Cachon, G. (2001). Managing a Retailer's Shelf Space, Inventory and Transportations. Manufacturing & Service Operations Management, 3(3), 211–229
2. Sbihi, A. (2007). A best first exact algorithm for the Multiple-choice Multidimensionnal Knapsack Problem. Journal of Combinatorial Optimization, 13, 337351
3. Gaur, V., Fisher, M.L. (2004). A Periodic Inventory Routing Problem at a Supermarket Chain. Operations Research, 52(6), 813–822
4. Nauss, R.M. (1978). The 0-1 Knapsack Problem with Multiple Choice Constraints, European Journal of Operational Research, 2(2), 125–131