Solving a Location-Allocation Problem with Logic-Based Benders' Decomposition

Mohammad M. Fazel-Zarandi and J. Christopher Beck

Department of Mechanical and Industrial Engineering
University of Toronto
Toronto, Ontario M5S 3G8, Canada
{fazel,jcb@mie.utoronto.ca}

Abstract. We address a location-allocation problem that requires deciding the location of a set of facilities, the allocation of customers to those facilities under facility capacity constraints, and the allocation of the customers to trucks at those facilities under per truck travel-distance constraints. We present a hybrid approach that combines integer programming and constraint programming using logic-based Benders' decomposition. Computational performance against an existing integer programming model and a tabu search approach demonstrates that the Benders' model is able to find and prove optimal solutions an order of magnitude faster than an integer programming model while also finding better feasible solutions in less time for the majority of problem instances when compared to the tabu search.

1 Introduction

Location-routing problems are well-studied, challenging problems in the area of logistics and fleet management [1]. The goal is to find the minimum cost solutions that decides on a set of facilities to open, the allocation of clients and vehicles to each facility, and finally the creation of a set of routes for each vehicle. Given the difficulty of this problem, Albareda-Sambola et al. [2] recently introduced a *location-allocation* problem which simplifies the routing aspect by assuming a full truckload per client. Multiple clients can be served by the same vehicle if the sum of the return trips is less than the maximum travel distance of the truck.

In this paper, we develop a logic-based Benders' decomposition [3] for the location-allocation problem . We compare our approach empirically to an integer programming (IP) model and to a sophisticated tabu search [2]. Our experimental results demonstrate an order of magnitude improvement over the IP model in terms of time required to find and prove optimality and significant improvement over the tabu search approach in terms of finding high-quality feasible solutions with small CPU time. To our knowledge, this is a first attempt to solve a location-allocation problem using logic-based Benders' decomposition.

2 Problem Definition and Existing Approaches

The capacity and distance constrained plant location problem (CDCPLP) [2] considers a set of capacitated facilities, each housing a number of identical vehicles for serving clients. Clients are served by full return trips from the facility. The same vehicle can be used to serve several clients as long as its daily workload does not exceed a given total driving distance. The goal is to select the set of facilities to open, determine the number of vehicles required at each opened site, and assign clients to facilities and vehicles in the most cost-efficient manner. The assignments must be feasible with respect to the facilities' capacities and the maximum distance a vehicle can travel.

Formally, let J be the set of potential facilities (or sites) and I be the set of clients. Each facility, $j \in J$, is associated with a fixed opening cost, f_j, and a capacity, b_j (e.g., a measure of the volume of material that a facility can process). Clients are served by open facilities with a homogeneous set of vehicles. Each vehicle has a corresponding fixed utilization cost, u, and a maximum total daily driving distance, l. Serving client i from site j generates a driving distance, t_{ij}, for the vehicle performing the service, consumes a quantity, d_i, of the capacity of the site, and has an associated cost, c_{ij}. The available vehicles at a site are indexed in set K with parameter $\overline{k} \geq |K|$ being the maximum number of vehicles at any site. Albareda-Sambola et al. formulate an integer programming (IP) model of the problem as shown in Figure 1, where the decision variables are:

$$p_j = \begin{cases} 1, \text{if facility } j \text{ is open} \\ 0, \text{otherwise} \end{cases}$$

$$z_{jk} = \begin{cases} 1, \text{if a } k\text{th vehicle is assigned to site } j \\ 0, \text{otherwise} \end{cases}$$

$$x_{ijk} = \begin{cases} 1, \text{if client } i \text{ is served by the } k\text{th vehicle of site } j \\ 0, \text{otherwise} \end{cases}$$

The objective function minimizes the sum of the costs of opening the facilities, using the vehicles, and the travel. Constraint (1) ensures that each client is served by exactly one facility. The driving distance limits are defined by constraint (2). Constraint (3) limits the demand allocated to facility j. Constraints (4) and (5) ensure that a client cannot be served from a site that has not been opened nor by a vehicle that has not been allocated. Constraint (6) states that at a site, vehicle k will not be used before vehicle $k-1$.

Albareda-Sambola et al. compare the IP performance to that of a three-level nested tabu search. The outermost level decides the open facilities, the middle level, the assignment of clients to facilities, and the innermost level, the assignment of clients to trucks. Tabu search is done on each level in a nested fashion: first neighborhoods that open, close, and exchange facilities are used to find a feasible facility configuration, then, using that configuration, the client assignment neighborhoods are explored, and finally the truck assignment is searched

$$\min \sum_{j \in J} f_j p_j + u \sum_{j \in J} \sum_{k \in K} z_{jk} + \sum_{i \in I} \sum_{j \in J} c_{ij} \sum_{k \in k} x_{ijk}$$

$$
\begin{aligned}
\text{s.t.} \; & \sum_{j \in J} \sum_{k \in K} x_{ijk} = 1 && i \in I && (1)\\
& \sum_{i \in I} t_{ij} x_{ijk} \leq l \cdot z_{jk} && j \in J, k \in K && (2)\\
& \sum_{i \in I} \sum_{k \in K} d_i x_{ijk} \leq b_j p_j && j \in J && (3)\\
& z_{jk} \leq p_j && j \in J, k \in K && (4)\\
& x_{ijk} \leq z_{jk} && i \in I, j \in J, k \in K && (5)\\
& z_{jk} \leq z_{jk-1} && j \in J, k \in K \backslash \{1\} && (6)\\
& x_{ijk}, p_j, z_{jk} \in \{0, 1\} && i \in I, j \in J, k \in K && (7)
\end{aligned}
$$

Fig. 1. An IP model of the CDCPLP [2].

over. Search then returns (i.e., as in a nested-loop) to the client assignments and eventually back to the facility openings. Computational results showed strong performance for the tabu search: it was able to find close-to-optimal solutions within a few minutes of CPU time.

3 A Logic-Based Benders' Decomposition Approach

In Benders' decomposition [3], a problem is partitioned into a master problem and a subproblem, which are solved iteratively until the optimal solution is found. When the subproblem is infeasible subject to current master solution, a cut that eliminates at least the current master solution is added to the master problem. The cut ensures that all future solutions are closer to being feasible.

The CDCPLP can be decomposed into a location-allocation master problem (LAMP) and a set of truck assignment subproblems (TASPs). The LAMP is concerned with choosing the open facilities, allocating clients to these sites, and deciding on the number of trucks at each site. The TASP assigns clients to specific vehicles and can be modeled as a set of independent bin-packing problems: clients are allocated to the trucks so that the total-distance constraint on each truck is satisfied. We use IP for the master problem and CP for the subproblems.

The Location-Allocation Master Problem An IP formulation of LAMP is shown in Figure 2 where p_j is as defined above and:

$$x_{ij} = \begin{cases} 1, \text{if client } i \text{ is served by site } j \\ 0, \text{otherwise} \end{cases}$$

$$numVeh_j : \text{number of vehicles assigned to facility } j$$

$$
\begin{aligned}
\min \sum_{j \in J} f_j p_j + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} + u \sum_{j \in J} numVeh_j & & \\
\text{s.t.} \sum_{j \in J} x_{ij} = 1 & \quad i \in I & (8) \\
\sum_{i \in I} t_{ij} x_{ij} \leq l \cdot \overline{k} & \quad j \in J & (9) \\
t_{ij} x_{ij} \leq l & \quad i \in I, j \in J & (10) \\
\sum_{i \in I} d_i x_{ij} \leq b_j p_j & \quad j \in J & (11) \\
numVeh_j \geq \left\lceil \frac{\sum_{i \in I} t_{ij} x_{ij}}{l} \right\rceil & \quad j \in J & (12) \\
cuts & & (13) \\
x_{ij} \leq p_j & \quad i \in I, j \in J & (14) \\
x_{ij}, p_j \in \{0, 1\} & \quad i \in I, j \in J & (15)
\end{aligned}
$$

Fig. 2. An IP model of the LAMP.

Constraint (8) ensures that all clients are served by exactly one facility. The distance limitations are defined by constraints (9) and (10). Constraint (11) limits the demand assigned to facility j. Constraint (12) defines the minimum number of vehicles assigned to each site. *cuts* are constraints that are added to the master problem each time one of the subproblems is not able to find a feasible solution. Initially, *cuts* is empty.

The cut for a given TASP j after iteration h is:

$$numVeh_j \geq numVeh^*_{jh} - \sum_{i \in I_{jh}} (1 - x_{ij}), \quad j \in J_h$$

where, $I_{jh} = \{i \mid x^h_{ij} = 1\}$ is the set of clients assigned to facility j in iteration h, J_h is the set of sites for which the subproblem is infeasible in iteration h, and $numVeh^*_{jh}$ is the minimum number of vehicles needed at site j to serve the clients that were assigned. Informally, the summation is the maximal decrease in the minimal number of trucks needed given the clients reassigned to other facilities: the largest possible reduction in reassigning one client is one truck. The form of this cut is directly inspired by the Benders' cut for scheduling with makespan minimization formulated by Hooker [4].

The Truck Assignment Subproblem Given the set of clients allocated (I_j) and the number of vehicles assigned to an open facility ($numVeh_j$), the goal of the TASP is to assign clients to the vehicles of each site such that the vehicle travel-distance constraints are satisfied. The TASP for each facility can be modeled as a bin-packing problem. A CP formulation of TASP is shown in Figure 3 where: *load* is an array of variables such that $load[k] \in \{0, ..., l\}$ is the total distance assigned to

$$\min \; numVehBinPacking_j$$

$$\text{s.t.} \quad pack(load, truck, dist) \qquad (16)$$

$$numVeh_j \leq numVehBinPacking_j < numVehFFD_j \qquad (17)$$

Fig. 3. A CP model of the TASP.

vehicle $k \in \{1, ..., numVehBinPacking_j\}$, $truck$ is an array of decision variables, one for each client $i \in I_j$, such that $truck[i] \in \{1, ..., numVeh_j\}$ is the index of the truck assigned to client i, and $dist$ is the vector of distances between site j and client $i \in I_j$. The pack global constraint (16) maintains the load of the vehicles given the distances and assignments of clients to vehicles [5]. The upper and lower bounds on the number of vehicles is represented by constraint (17).

Algorithm 1 shows how we solve the sub-problems in practice. We first use the first-fit decreasing (FFD) heuristic (line 3) to find $numVehFFD_j$, a heuristic solution to the sub-problem. If this value is equal to the value assigned by the LAMP solution, $numVeh_j$, then the sub-problem has been solved. Otherwise, in line 5 we solve a series of satisfaction problems using the CP formulation, setting $numVehBinPacking_j$ to each value in the interval $[numVeh_j..numVehFFD_j - 1]$ in increasing order.

4 Computational Results

We compare our Benders' approach to the IP and tabu search models in turn. Unless otherwise noted, the tests were performed on a Duo Core AMD 270 CPU with 1 MB cache, 4 GB of main memory, running Red Hat Enterprise Linux 4. The IP model was implemented in ILOG CPLEX 11.0. The Benders' IP/CP approach was implemented in ILOG CPLEX 11.0 and ILOG Solver 6.5.

IP vs. Benders' We generated problems following exactly the same method as Albareda-Sambola et al [2]. We start with the 25 instances of Barceló et al. [6] in

Algorithm 1: Algorithm for solving the TASP

```
   SolveTASP():
1  cuts = ∅
2  for each facility do
3      numVehFFD = runFFD()
4      if numVehFFD > numVeh_j then
5          numVehBinPacking = runCPBinPacking()
6          if numVehBinPacking > numVeh_j then
7              cuts ← cuts + new cut
8  return cuts
```

Problem Set	Uncorrelated						Correlated					
	IP		Benders'			Time	IP		Benders'			Time
	Time	% Uns.	Time	% Uns.	Iter	Ratio	Time	% Uns.	Time	% Uns.	Iter	Ratio
20 × 10	252	0	33	0	3.8	9.6	65	0	24	0	4.1	6.3
30 × 15	55303	23	17593	6	16	13.6	29514	12	8065	2	13.1	20.5
40 × 20	144247	75	72598	35	26.2	2.4	79517	38	28221	10	22.8	6.5
Overall	70553	34	30980	14	16.3	9.1	38447	17	12585	4	14.0	12.6

Table 1. The mean CPU time (seconds) and percentage of unsolved problem instances (% Uns.) for the IP and Benders' approaches and for the Benders' approach, the mean number of iterations. Overall indicates the mean results over all problem instances–recall that each subset has a different number of instances.

three sizes: 6 instances of size 20×10 (i.e., 20 clients, 10 possible facility sites), 11 instances of size 30×15, and 8 instances of size 40×20. The fixed facility opening cost, f_j, demands for each client, d_i, assignment costs, c_{ij}, and facility capacities, b_j, are extracted from Barceló et al. with the exception that the facility capacities are multiplied by 1.5 as they are very tight. Six different pairs of truck distance limit, l, and truck usage cost, u, values are then used to create different problem sets: $(40, 50), (40, 100), (50, 80), (50, 150), (100, 150), (100, 300)$. Finally, the travel distances, t_{ij}, are randomly generated based on the costs, c_{ij} in two different conditions. In the *correlated* condition: $t_{ij} = scale(c_{ij}, [15, 45]) + rand[-5, 5]$. The first term is a uniform scaling of c_{ij} to the integer interval $[15, 45]$ while the second term is a random integer uniformly generated on the interval $[-5, 5]$. In the *uncorrelated* condition, $t_{ij} = rand[10, 50]$. Overall, therefore, there are 300 problem instances: 25 original instances times 6 (l, u) conditions times 2 correlated/uncorrelated conditions.

Table 1 compares the mean CPU time in seconds required to solve each problem instance for each set. In all cases, 48 hours was used as a maximum time. The "Time Ratio" for a given instance is calculated as the IP run-time divided by the Benders' run-time. The mean over each instance in each subset was then calculated. For unsolved instances, the 48-hour time limit was used. As can be seen Benders' is able to solve substantially more problems than IP and, on average, has a run-time about an order-of-magnitude faster.

Figure 4 shows a scatter-plot of the run-times of each problem instances for both IP and the Benders' approach. Both axes are log-scale and the points below the $x = y$ line indicate lower run-time for the Benders' approach. On all but 26 of the 300 instances, the Benders' achieves equivalent or better run-time.

Tabu search vs. Benders' One of the weaknesses of a Benders' decomposition approach is that usually the first globally feasible solution found is the optimal solution. This means that cutting off runs due to a time-limit will result in no feasible solutions. For problems too large for a Benders' approach to find optimality, another algorithm is needed to find a good but not necessarily optimal solution. Metaheuristic techniques, such as tabu search, are widely used for location and routing problems for this purpose [7].

With our Benders' formulation, however, we have a globally feasible, suboptimal solution at each iteration. In generating a cut, we find the minimum number of trucks needed at each facility. This number of trucks constitutes a

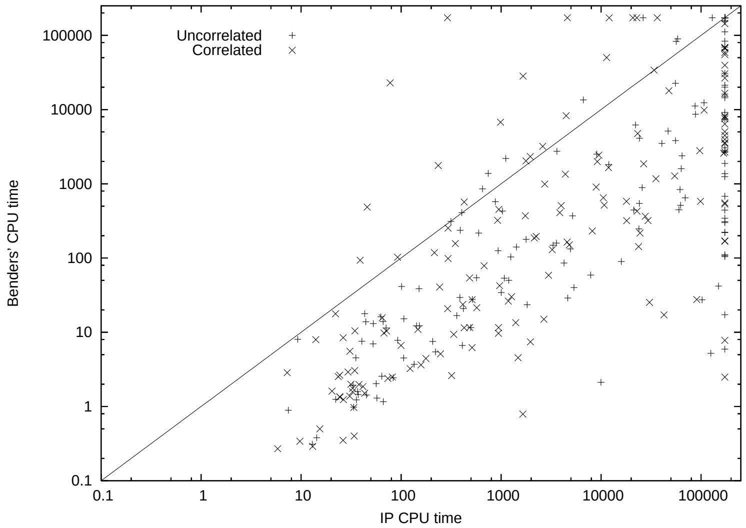

Fig. 4. Run-time of IP model (x-axis, log-scale) vs. Benders' IP/CP model (y-axis, log-scale) of the 300 problem instances. Points below the $x = y$ line indicate lower run-time for the Benders' model.

feasible solution even though fewer trucks were assigned in the master solution.[1] Thus, at the end of each iteration, we have a globally feasible solution.

Albareda-Sambola et al. used 19 medium and large problem instances to evaluate their tabu search approach, reporting run-times and bounds on the optimality gap. All instances are correlated and have (l, u) values of $(50, 80)$. We received these exact instances form the authors.[2] We believe that the run-time for our Benders' model to find the *first* feasible solution and the gap from optimality provide some basis for comparison.

Table 2 presents the mean and median time for the first iteration and mean percentage gap from optimal for the Benders' approach. This is compared to the run-time on a 2.4GHz Pentium IV and mean percentage gap from optimal reported by Albareda-Sambola et al.[3] The columns labeled "# Dom." indicate the number of problems in each set for which one approach was clearly dominant with respect to both lower CPU time and lower % gap. As can be seen, on average the Benders' approach is 13.6% worse than Tabu search with respect to the mean CPU time to find a feasible solution but finds solutions with substantially smaller

[1] This is not true when the minimal number of trucks required at a facility is greater than $\bar{k}$. This did not occur in any of our experiments.

[2] We would like to thank Maria Albareda-Sambola for providing these instances.

[3] Albareda-Sambola et al. presented the cost of their best solution and bounds on the percentage gap. As we found the optimal solutions we were able to calculated the exact gap from optimality for the tabu search.

Problem Set	Benders'				Tabu			
	Time		% Gap	# Dom.	Time		% Gap	# Dom.
	Mean	Median	Mean		Mean	Median	Mean	
30 × 15	60.4	13.5	2.07	4	60.5	66.1	4.12	0
40 × 20	185.4	86.2	1.82	6	148.7	163.0	10.41	0
Overall	113.0	39.5	1.96	10	97.6	78.8	6.77	0

Table 2. The mean and median CPU time (seconds), the mean percentage gap from optimal and the bounds of that gap for Tabu, and the number of instances for which each approach dominated the other.

optimality gaps. However, Benders' exhibits three run-time outliers that obscure the results. Out of 19 problem instances, Benders' finds a better solution faster than tabu on 10 instances while tabu search was not able to find a better solution faster than Benders' for any instance.

5 Conclusion

In this paper, we presented a novel logic-based Benders' decomposition approach to a location-allocation problem. Our approach was able to substantially outperform an existing IP model by finding and proving optimality, on average, more than ten times faster. Our approach also performed better than an existing tabu search in finding good, feasible solutions in a short time.

Acknowledgments This research was supported in part by the Natural Sciences and Engineering Research Council, Canadian Foundation for Innovation, Ontario Ministry for Research and Innovation, Alcatel-Lucent, and ILOG, S.A.

References

1. Drezner, Z.: Facility Location. A Survey of Applications and Methods. Springer Series in Operations Research. Springer Verlag, New York (1995)
2. Albareda-Sambola, M., Fernández, E., Laporte, G.: The capacity and distance constrained plant location problem. Computers & Operation Research **36**(2) (2009) 597–611
3. Hooker, J.N., Ottosson, G.: Logic-based benders decomposition. Mathematical Programming **96** (2003) 2003
4. Hooker, J.: A hybrid method for planning and scheduling. Constraints **10** (2005) 385–401
5. Shaw, P.: A constraint for bin packing. In: Principles and Practice of Constraint Programming - CP 2004. (2004) 648–662
6. Barceló, J., Fernández, E., K., J.: Computational results from a new lagrangean relaxation algorithm for the capacitated plant locating problem. European Journal of Operational Research **53** (1991) 38–45
7. Laporte, G.: Location-routing problems. In Golden, B.L., Assad, A.A., eds.: Vehicle routing: methods and studies. Amsterdam : North-Holland (1988) 163–197