A HEURISTIC ALGORITHM FOR SOLVING CAPACITATED FACILITY LOCATION PROBLEMS USING A GREEDY-BASED ITERATIVE LP RELAXATION PROCEDURE

A Thesis

by

HIROMICHI YAMAMOTO

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	V. Jorge Leon
Committee Members,	Andrew L. Johnson
	Senthil Gunasekaran
Head of Department,	Mark Lawley

August 2018

Major Subject: Industrial Engineering

ABSTRACT

A new methodology to solve the capacitated facility location problem (CFLP) is presented. This optimization problem can be explicitly formulated and solved as a mixed integer program (MIP); however, because binary variables are used, obtaining exact solutions can be computationally intensive. This issue is apparent for solving large-scale problems, where the problem complexity is known to increase exponentially in the number of location variables. The proposed approach will instead solve the problem in a heuristic manner, returning an approximate solution rather than an exact one. A linear program (LP) relaxation to the problem is solved, while iteratively fixing select binary location variables to 0 or 1 until a feasible solution is obtained. Experimental results show that the proposed methodology can be effective in obtaining solutions in a fraction of CPU (central processing unit) time compared to exact methods. The quality of the solution is also shown to be extremely close to optimal for problems with relatively high fixed cost parameters. An application to a real-life problem is also explored to validate the practicality of the proposed methodology.

Not only does the algorithm offer a new approach to solving the CFLP, but it also presents a fast approximation method which can be applied to solve MIP models in general. Additional ideas for improving the algorithm are also presented.

CONTRIBUTORS AND FUNDING SOURCES

This work was supervised by a thesis committee consisting of Professor V. Jorge Leon (advisor) of the Departments of Engineering Technology & Industrial Distribution and Industrial Engineering, Professor Andrew L. Johnson of the Department of Industrial Engineering, and Professor Senthil Gunasekaran of the Department of Engineering Technology & Industrial Distribution. All work conducted for the thesis was completed by the student independently.

Graduate study at Texas A&M University was partially supported by a research assistantship position from the Department of Engineering Technology & Industrial Distribution.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

The facility location problem is an optimization problem seeking to find an optimal placement of facilities in order to minimize the total costs of the network. This paper specifically considers the Capacitated Facility Location Problem (CFLP). In this problem we are given a set I of customers, with each customer $i \in I$ having demand d_i to be served. We are also given a set J of potential locations where a facility $j \in J$ can be opened. These facilities each have fixed cost f_j and capacity q_j components associated with them. Assigning demand to be served from facility j to customer i costs c_{ij} per unit. The CFLP objective is to select the best combination of facilities to be located which minimizes the sum of fixed and variable (e.g., transportation) costs. Each customer demand must be fully met while facility capacities may not be violated.

The following decision variables are introduced.

$$x_j = \begin{cases} 1, & \text{if facility } j \text{ is selected to operate} \\ 0, & \text{otherwise} \end{cases}$$

$$y_{ij} = \text{volume of demand served to customer } i \text{ from facility } j$$

The problem is formulated as follows.

$$\text{Minimize } \sum_{j \in J} f_j x_j + \sum_{i \in I} \sum_{j \in J} c_{ij} y_{ij} \quad (1)$$

$$\text{Subject to } \sum_{i \in I} y_{ij} \leq q_j x_j, \quad \forall j \in J \quad (2)$$

$$\sum_{j \in J} y_{ij} \geq d_i, \quad \forall i \in I \quad (3)$$

$$y_{ij} \leq q_j x_j, \quad \forall i \in I, \forall j \in J \quad (4)$$

$$x_j \in \{0,1\}, \quad \forall i \in I, \forall j \in J \quad (5)$$

$$y_{ij} \geq 0, \qquad \forall i \in I, \forall j \in J \quad (6)$$

Objective function (1) minimizes the global cost, which is the sum of the fixed cost component (first term) of opening the facilities and the variable cost component (second term) of serving all customer demand point-facility location combinations. Constraints (2) ensure the total volume handled at each facility does not exceed its capacity. Constraints (3) force all demand at each customer to be met. Constraints (4) are redundant constraints of (2), but give tighter bounds to the feasible region. The "strong" problem formulation enabled by constraints (4) is preferred over the "weak" formulation (e.g., no redundant constraints) in many previous works because it reduces the gap of the LP (linear program) relaxation relative to the optimum integer solution (Teixeira et al. (2006)). Finally, constraints (5) and (6) define the decision variables. It is also assumed that all parameters, including unit costs, demands, and capacities take nonzero values.

The CFLP can be solved to optimality as a mixed-integer program (MIP) as modeled above. Commercial software such as AMPL, Gurobi, and CPLEX is capable of solving these problems effectively. However, solving MIPs is computationally intensive in contrast to solving LPs; this is especially an issue for larger problem instances, as computation times can increase exponentially with the addition of more variables and constraints to the problem. Indeed, these problems have been proven to be NP-hard and the worst-case runtime is $O(2^n)$, where n is the number of binary (i.e., 0-1) location variables (Francis et al. (1983)). The non-deterministic nature of solving CFLPs and MIPs is limiting in practice, and therefore justifies the development of heuristics or approximation methods as alternative solving methods.

The literature offers several approaches to overcoming this obstacle in an attempt to speed up problem solve times. The complicating element is the fixed cost component, because this requires the need for the binary variables to indicate whether a facility will be opened or not in the optimal network topology. Nagurney (2010) formulates the supply chain optimization problem strictly as a flow problem, by making the assumption that facility capacities can be freely "bought" or "sold" in the open market. Lu et al. (2014) offers an approach that reflects "economies of scale" within the objective function – specifically, an S-shaped cost function is used to approximate a step function to capture the fixed cost component. These techniques allow for the elimination of the complicating binary variables from the formulation altogether. However, this type of approach can be limiting: in the former, not explicitly considering the effects of fixed cost could result in solutions failing to reflect the realities of business expenditures such as warehouse rent and other startup costs. In the latter method, the problem formulation may result in a nontrivial objective function (e.g., nonlinear and nonconvex), potentially leading to solving issues.

There also exist procedures that keep the MIP problem formulation intact, but apply heuristics to limit problem size and speed up computation times. Marmolejo et al. (2015) applies a Benders decomposition technique to solve a distribution network optimization problem. Other work such as that of Angelelli et al. (2012) and Gustaroba and Speranza (2014) implements a Kernel search framework to solve the MIP by only considering "promising" locations, which are determined to have high chances of being open in the optimal network topology. Incorporating LP relaxation approximations are also well-explored alternatives. Methods developed by Murray and Shanbhag (2006) and Melo et al. (2014) solve the LP relaxation to the problem to obtain an initial feasible solution, which is improved upon by local neighborhood searching. Thanh et al. (2010) uses LP relaxation and rounding of key decision variables to fix as many binary variables as possible, until the problem is reduced to a small enough MIP that can be solved using exact methods. Although these methodologies are shown to yield quality solutions (i.e., small optimality gap) while reducing problem complexity, invoking the use of MIP solvers eliminates the hope for any further savings in computational time.

In this paper, new heuristic technique to solve the CFLP model is introduced. The proposed algorithm will iteratively solve the LP relaxation to the problem – at each step, hard variable fixing is implemented to set “promising” and “unpromising” binary location variables to 1 and 0, respectively. The “promising” and “unpromising” variables are determined based on the location variable values in the most recent solution. The other variables are kept relaxed until subsequent iterations take place. The procedure is repeated until a valid binary solution vector is obtained. A feasibility recovery process is implemented in the case that an infeasible solution is encountered. To limit problem runtimes, the algorithm will invoke a timeout protocol in the case that too many infeasible iterations are observed.

Two contributions of the proposed work are that it offers: (1) a new approach to solving the CFLP, and (2) a fast approximation method to solve MIP problems in general. Because MIP solvers are not utilized in the proposed technique, it is expected that computation times are substantially improved, especially for large-scale instances. Performed experiments demonstrate three takeaways: (1) runtimes do not grow exponentially with increases in problem complexity, allowing for larger problems to be solved effectively, (2) the obtained solutions are of good quality for certain scenarios, and (3) the approach is capable of solving real-life problems.

The paper is structured as follows. Section 2 is devoted to outlining a detailed description of the proposed algorithm. The framework and results of the experimentation process are provided in Section 3. Section 4 explores an application of the technique to solving a warehouse network optimization exercise based on a real-life problem. To conclude, Section 5 addresses final remarks and some areas for future research.

2. THE HEURISTIC ALGORITHM

The detailed procedure of the proposed methodology (referred to as *heuristic algorithm* or simply *algorithm* throughout this paper), as well as its runtime complexity, is described in Section 2.1 and 2.2, respectively.

2.1. Heuristic algorithm steps

The problem is first formulated as a standard CFLP model as presented in Section 1. The binary constraints are then removed, converting the problem into a LP relaxation problem. This problem is then solved, which yields an initial solution. A check is conducted to see if all candidate location variables in the solution are binary – this is the first of two termination conditions. If there are non-binary values in the solution vector, the algorithm must continue; otherwise, the algorithm terminates. The second termination condition, where the number of encountered infeasible iterations is evaluated, is explained at the end of this subsection.

In the "variable fixing" sequence, there are three procedures: (1) permanently fixing variables to 1, (2) tentatively fixing variables to 0, and (3) permanently fixing variables to 0. The algorithm will first search through all location variables (e.g., candidate facility location) not equaling 0. The algorithm will choose one "most promising" variable and set this to be permanently opened (fixed to 1) for the rest of the algorithm. This location is identified as the non-binary variable x_j (which has not yet been fixed by the algorithm) having the highest value. If there is a tie, the tiebreaker will be the fixed cost – the location with the lower fixed cost is chosen. If there still is a tie, the selection will be arbitrary (to be precise, the location with the smallest index value j is selected).

Similar to the "most promising" variable fixing process, the algorithm also looks for one "least promising" variable. The non-binary location variable (also which has not yet been fixed) having the lowest value will be tentatively set to be closed (fixed to 0) for the rest of the algorithm. Tiebreaker rules also apply – in the case of a tie, the location with the higher fixed cost will be set to 0.

The last component is the "permanent" fixing of variables to 0. For all facilities where the LP relaxation solution equals 0, they will be permanently closed (set to 0) for the rest of the algorithm. Unlike the other two variable fixing processes, more than a single facility variable can be fixed in one iteration.

Once the applicable variable values are all set, the modified LP relaxation problem is solved. The solver returns either a "feasible" or "infeasible" solution status. If feasible, the algorithm repeats iterations as necessary with the remaining "unfixed" location variables until a termination condition is met. However, if the result is "infeasible," we must backtrack. The location which was tentatively fixed to be closed in component (2) of the variable fixing stage will be opened (i.e., set to 1) for the rest of the algorithm. This modified LP relaxation problem will then be solved, and the resulting feasible solution will be further evaluated by the algorithm.

When an infeasible iteration is encountered, this means the problem has encountered a capacity limitation. As the algorithm artificially forces certain location variables to be closed, this could remove too much available capacity from the network and potentially lead to constraint violations. To resolve this issue, the location that was tentatively selected to be closed in that particular iteration is "re-opened," thus adding the previously removed capacity back into the problem. Because infeasible iterations can only be caused by capacity violations, the algorithm is guaranteed to recover feasibility once the re-modified problem is solved.

The number of these “backtracking” occurrences are tracked in order to cap the maximum number of infeasible iterations allowed by the algorithm. The rationale is that if infeasible iterations are being observed, the gap between total demand and total available network capacity should be diminishing, and thus the algorithm is approaching termination. To avoid exploring through too many infeasible solutions and thereby reducing computation times, a threshold *T* is set at

$$T = \left\lceil \frac{\sqrt{|J|}}{2} \right\rceil,$$

where $|J|$ is the total number of candidate locations in the problem. Threshold *T* is a heuristic parameter, and the value specified above is a default value to be used for the experiments discussed in Sections 3.2 and 3.3. The algorithm’s behavior under other levels of *T* will be examined in Section 3.4.

If the cumulative number of infeasible iterations exceeds *T*, then the algorithm will automatically timeout, and all remaining nonzero variables are fixed to 1. This modified LP relaxation will then be solved for a final time, leading to the termination of the algorithm. Figure 1 provides a visual overview of the algorithm steps, discussed in detail above.

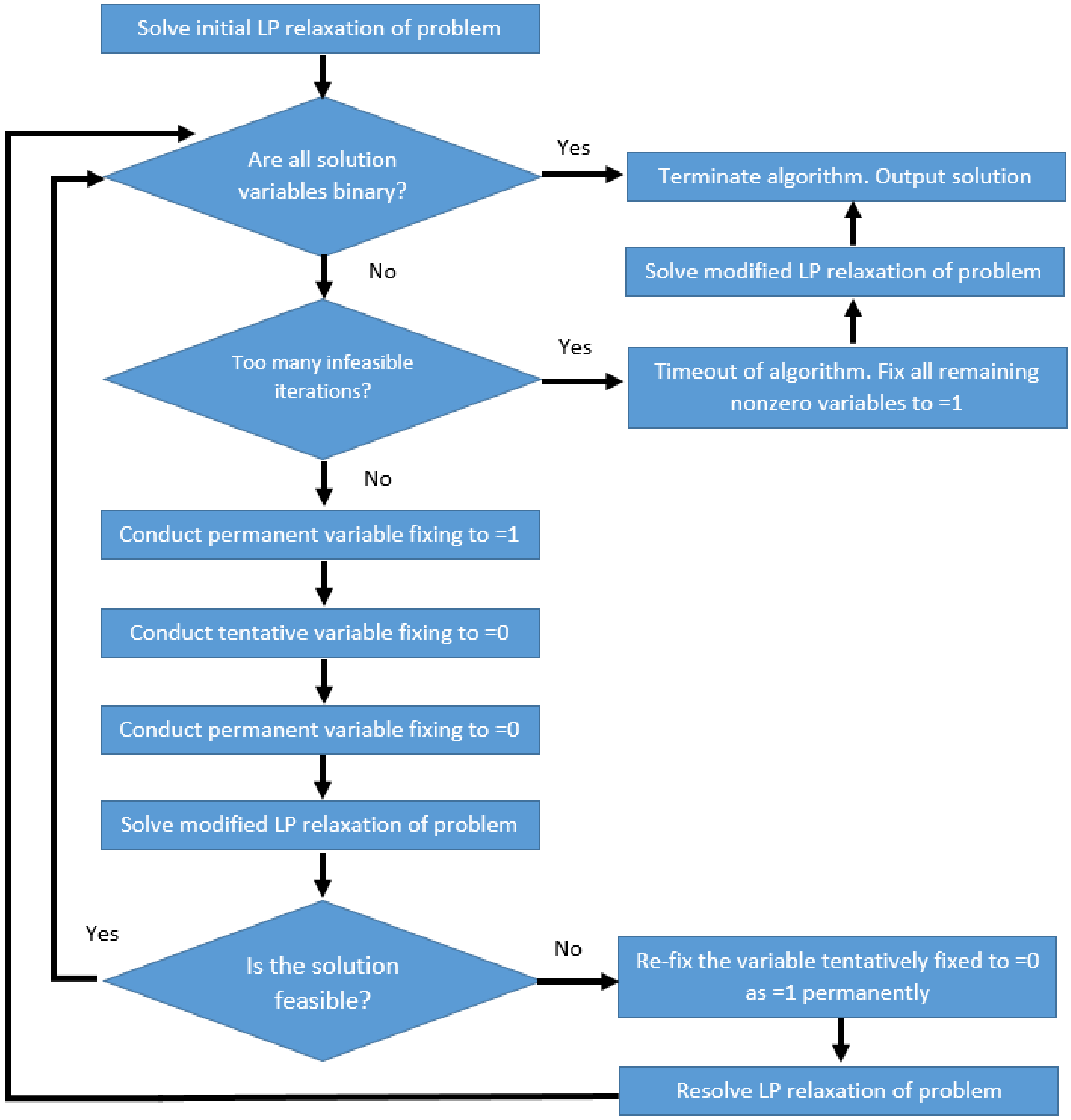

Figure 1. The detailed steps of the heuristic algorithm.

2.2. Algorithm complexity

The worst-case runtime of the proposed algorithm is estimated to be polynomial. The three-step sequence of variable fixing occurs in $O(n) + O(n) + O(n) = O(3n) = O(n)$ time, where *n* is the number of variables to the LP problem. Linear programming is known to run in polynomial time $O(L\sqrt{m+n})$, where *m* is the number of constraints to the LP problem, and L is defined by the number of bits required to store all entries of the problem (Renegar (1988)). In case the iteration encounters an infeasible solution, the re-fixing of one location variable of complexity $O(1)$ and an additional solve of the LP model taking $O(L\sqrt{m+n})$ time is implemented. Thus, each algorithm iteration has a worst-case bound of $O(n) + O\left(L\sqrt{m+n}\right) + O(1) + O\left(L\sqrt{m+n}\right) = O\left(n + 1 + 2L\sqrt{m+n}\right) = O\left(n + L\sqrt{m+n}\right)$.

Because a minimum of two location variables are fixed to 0 and 1 within each iteration, this procedure can be repeated a maximum of $O(n/2)$, or $O(n)$ times. Hence, the algorithm complexity is $O(n)O\left(n + L\sqrt{m+n}\right) = O\left(n^2 + nL\sqrt{m+n}\right)$, which is indeed polynomial in n. This result implies that the developed heuristic is predicted to outperform exact solving methods in terms of worst-case algorithm complexity, where solving to optimality results in exponentially growing worst-case solve times as discussed in Section 1. Experimental results explored in the next section will validate this prediction.

3. EXPERIMENTAL ANALYSIS

This section is devoted to presentation and discussion of computational experiments. The tests were run on a PC Intel CORE i5 with 2.40 gigahertz 64-bit processor, 8.0 gigabytes of RAM, and Windows 10 64-bit as the Operating System. The algorithms were implemented in C++. Problems were solved with CPLEX 12.7, with all parameters set to their default values. Test cases were generated using R.

In Section 3.1 the testing environment of the general test cases is discussed. Section 3.2 summarizes computational findings of the test cases defined in Section 3.1. Findings from additional problem instances that test for various fixed cost levels, which was found to be a significant control parameter from the initial computational results, are presented in Section 3.3. To conclude the section, Section 3.4 discusses results from additional experiments testing for various levels of the infeasibility threshold *T*.

3.1. Testing environment

The heuristic algorithm was tested on 162 instances, ranging from small-scale (e.g., 10 candidate facilities and 21 customers) to large-scale (e.g., 124 facilities and 111 customers). The test cases were randomly generated as follows.

The random test case generator first determines the numbers of candidate facility and customer locations for each instance ($|J|$ and $|I|$ respectively), both random variables of a uniform distribution according to $\sim U(10{,}130)$. Each facility and customer point is assigned to a Cartesian coordinate location within a 2-dimension *[-1,1]* by *[-1,1]* field. It is assumed that each facility is identical with respect to capacity restrictions and fixed cost amounts, and each customer point uniformly has one unit of demand. The total available capacity (summed over all facilities) is defined by a uniformly distributed random variable as follows:

$$\sum_{j \in J} q_j \sim U(5 \times |I|, 15 \times |I|).$$

Lastly, transportation unit costs are determined based on the Euclidean distance between each customer-facility location combination.

Motivated to analyze the algorithm's performance under various real-world circumstances, two variants were implemented to the test case design. The first is coordinate location distribution. In practical applications, customer demand and facility location availability may vary across areas considered. Taking the region of North Texas, for example, population distribution patterns vary based on the geographical scale considered – within the Dallas-Fort Worth area, the population is spread out uniformly across the sprawling metropolitan region, whereas at the region-level, there is one significant "center of gravity" (e.g., the Metroplex) that the population centers around. To compare the effects of this parameter, three scenarios are evaluated: (1) "uniform" distribution, (2) "1-centroid" distribution, and (3) "2-centroid" distribution. In the "uniform" case, all coordinate locations are generated randomly within the Cartesian field according to $(x, y) \sim \big(U(-1,1), U(-1,1)\big)$. In the "1-centroid" scenario, one center of gravity point (x^0, y^0) is generated according to the same uniform distribution above. The rest are generated around this point following a normal distribution as follows:

$$(x, y) \sim \left(N\left(x^0, \left|\frac{x^0}{2}\right|\right), N\left(y^0, \left|\frac{y^0}{2}\right|\right)\right).$$

In the case that the point falls outside the defined coordinate field, the coordinate values are truncated to either *-1* or *1*. The "2-centroid" scenario follows the same distribution, except half of the points are generated around the first center of gravity, and the other half around a second.

Fixed cost is the second input parameter implemented. The CFLP solution can be significantly impacted by the fixed cost levels, influencing the optimal network topology to have more or less opened facilities based on overhead cost considerations. Three scenarios are evaluated: (1) "low," (2) "medium," and (3) "high" fixed cost levels. In each of the scenarios, all facilities will have fixed cost set to $f_j = |J|^{\alpha}$, where α is set to -0.5, 0.5, and 1.5 for the "low," "medium," and "high" cases, respectively.

Overviews of instances tested in this paper are shown in Tables 1 and 2: Table 1 aggregates the 162 test cases by the input scenario parameters (total of $3^2 = 9$), whereas Table 2 summarizes them by test case problem size (total of 18). The column "Problem Size" is defined as the total number of both binary (i.e., location) and non-binary (i.e., flow) variables considered in the problem, equal to $|J| \times (|I| + 1)$. This will be the metric evaluated when quantifying a particular test case's problem size.

Table 1. Overview of test cases summarized by input parameter.

Location distribution scenario	Fixed cost level scenario	Number of test cases evaluated
Uniform	Low	18
Uniform	Medium	18
Uniform	High	18
1-centroid	Low	18
1-centroid	Medium	18
1-centroid	High	18
2-centroid	Low	18
2-centroid	Medium	18
2-centroid	High	18
	Total	**162**

Table 2. Overview of test cases summarized by problem size.

$\|I\|$	$\|J\|$	Problem Size ($\|J\| \times (\|I\| + 1)$)	Number of test cases evaluated
21	10	220	9
15	17	272	9
13	24	336	9
14	23	345	9
13	26	364	9
38	39	1,170	9
36	50	1,850	9
59	35	2,100	9
44	58	2,610	9
65	55	3,630	9
63	63	4,032	9
77	77	6,006	9
71	99	7,128	9
70	112	7,952	9
89	89	8,010	9
77	121	9,438	9
106	107	11,449	9
111	124	13,888	9
		Total	**162**

3.2. Computational results

This section summarizes and comments on the computational results. All test cases are solved using two methods: the heuristic algorithm presented in this paper, and the CPLEX-MIP solver. The optimal solutions computed by CPLEX-MIP provide a benchmark to evaluate the heuristic algorithm performance in regards to two metrics: CPU runtime (measured in seconds) and solution quality (quantified as the optimality gap between the best found solutions from both methods). The optimality gap is reported as a percent (%) based on the following calculation:

$$Optimality\ gap = \frac{Heuristic\ Algorithm\ objective - CPLEXMIP\ objective}{CPLEXMIP\ objective} \times 100.$$

Because CPLEX may take very long to solve test cases to optimality, runtimes are capped at 3,600 seconds (i.e., 1 hour). If this threshold is exceeded, the best found solution is assumed to be the optimal solution for the particular test instance.

Table 3. Overview of experiment results summarized by problem size.

$\|I\|$	$\|J\|$	Problem size	Test cases considered	CPU (seconds) Algorithm			CPU (seconds) CPLEX-MIP			Optimality Gap (%)			"Timeout" test cases
				Min.	*Avg.*	*Max.*	*Min.*	*Avg.*	*Max.*	*Min.*	*Avg.*	*Max.*	
21	**10**	**220**	9	0.015	0.026	0.054	0.052	0.095	0.139	1.0%	21.3%	60.6%	0
15	**17**	**272**	9	0.017	0.023	0.032	0.103	0.215	0.262	2.2%	64.7%	252.7%	0
13	**24**	**336**	9	0.020	0.024	0.028	0.229	0.308	0.379	0.3%	22.2%	90.5%	0
14	**23**	**345**	9	0.023	0.029	0.060	0.244	0.285	0.354	0.0%	64.0%	230.8%	0
13	**26**	**364**	9	0.023	0.026	0.032	0.173	0.624	1.363	0.2%	29.6%	136.3%	0
38	**30**	**1,170**	9	0.025	0.044	0.067	0.108	0.806	1.327	0.2%	42.4%	184.6%	0
36	**50**	**1,850**	9	0.036	0.071	0.096	0.306	962.357	3610.090	0.0%	11.4%	59.7%	1
59	**35**	**2,100**	9	0.054	0.074	0.109	0.150	1.629	4.074	0.1%	8.8%	29.3%	0
44	**58**	**2,610**	9	0.052	0.085	0.106	9.499	2412.712	3615.920	0.0%	6.6%	21.4%	6
65	**55**	**3,630**	9	0.126	0.295	0.600	0.367	8.452	31.416	0.1%	14.5%	30.2%	0
63	**63**	**4,032**	9	0.161	0.236	0.293	0.527	234.545	1934.830	0.0%	14.8%	66.5%	0
77	**77**	**6,006**	9	0.267	0.378	0.454	0.766	144.332	1000.470	0.0%	25.0%	126.0%	0
71	**99**	**7,128**	9	0.286	0.388	0.477	30.828	1782.862	3671.130	0.0%	12.8%	67.2%	4
70	**112**	**7,952**	9	0.368	0.441	0.509	20.748	1629.630	3652.280	0.0%	5.9%	20.2%	3
89	**89**	**8,010**	9	0.319	0.448	0.539	1.291	315.806	1085.340	0.0%	14.9%	57.0%	0
77	**121**	**9,438**	9	0.412	0.855	1.231	3606.470	3612.949	3631.530	0.0%	13.6%	71.1%	9
106	**107**	**11,449**	9	0.501	0.564	0.637	2.158	1096.777	3618.210	0.0%	15.0%	51.7%	2
111	**124**	**13,888**	9	0.621	0.768	1.102	329.536	2989.585	3636.430	0.0%	20.2%	88.7%	6

Results are summarized in Table 3. One clear takeaway is the significant savings in CPU times offered by the heuristic algorithm. Figure 2 displays a scatterplot of solving runtimes using both methods (note the vertical axis is in logarithmic scale). As expected, it is apparent that using the CPLEX-MIP solver results in exponentially increasing CPU times as the problem size increases. Additionally, the results using the heuristic algorithm offer support to the proposition presented in Section 2.2 – the runtimes of the developed algorithm only appear to grow in a polynomial (perhaps even quasi-linear) fashion. Appendix A.1 offers results of all 162 test cases examined for this analysis.

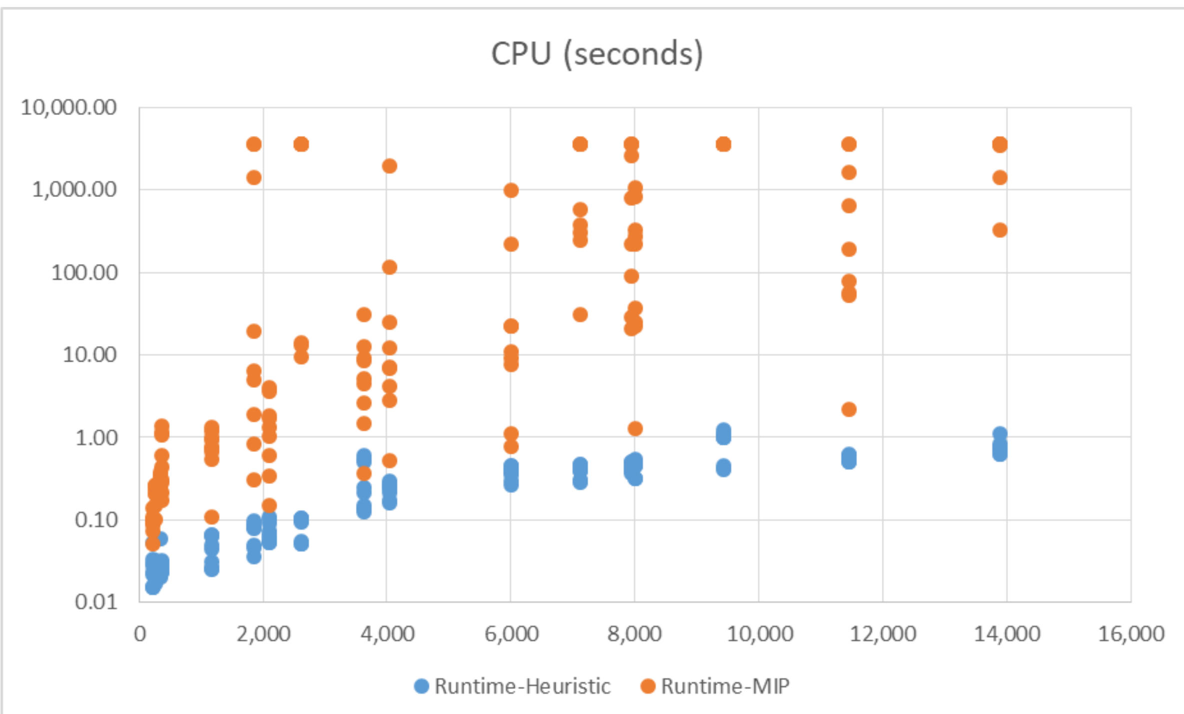

Figure 2. Computation time of heuristic and CPLEX-MIP plotted against problem size.

It is important to note that solve times using CPLEX-MIP was capped at 1 hour. This gives rise to two points: the first is that because an upper bound on allowed computation time is set, CPU times for “timeout” test cases are artificially truncated. If this restriction was not implemented, we can expect to see longer upper bounds on problem solve times using CPLEX-MIP, further amplifying the benefits of the algorithm as a fast approximation method. The second point is that as Figure 3 illustrates below, it is intuitively expected that the frequency of instances that “timeout” increases as the problem size gets larger. However, a strong relationship from the test cases cannot be derived – this depicts the “non-deterministic” nature of solving these types of problems to optimality. Being able to estimate problem solve time bounds is another aspect that the heuristic offers that exact methods cannot.

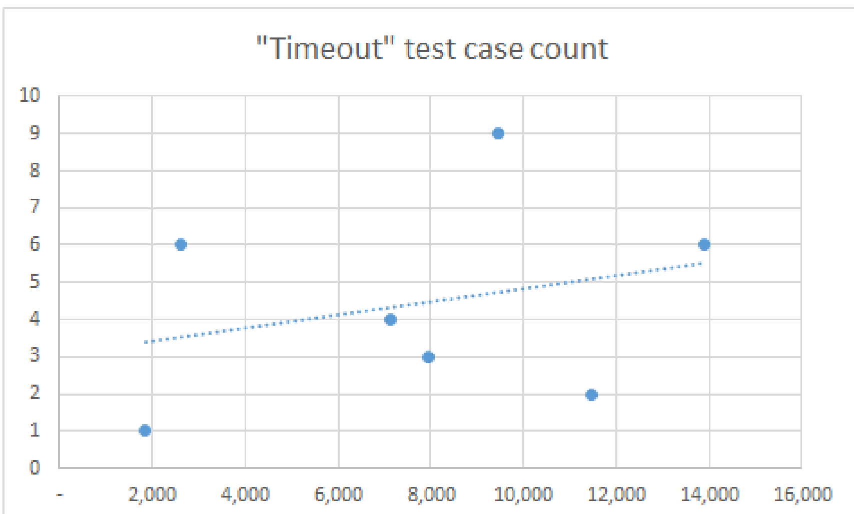

Figure 3. Number of “timeout” cases when using CPLEX-MIP to solve, plotted against problem size.

The second metric, solution quality, is also observed. Overall solution qualities of test cases vary significantly – Figure 4 displays a scatterplot showing the relationship between solution accuracy and problem size. Based on the results, the optimality gaps are not small enough to claim that the heuristic algorithm offers satisfactory solutions at a reliable rate. From these results, we can conclude that the benefit of the algorithm is producing a feasible solution (which can *potentially* offer “adequate” solutions) in a fraction of computation time. This benefit may be marginal when solving smaller instances, because in these cases CPLEX can provide optimal solutions while maintaining feasible CPU runtimes. Rather, the benefit is most reflected in the larger test cases, where instances that “timeout” using the CPLEX-MIP solver can be solved by the heuristic in seconds.

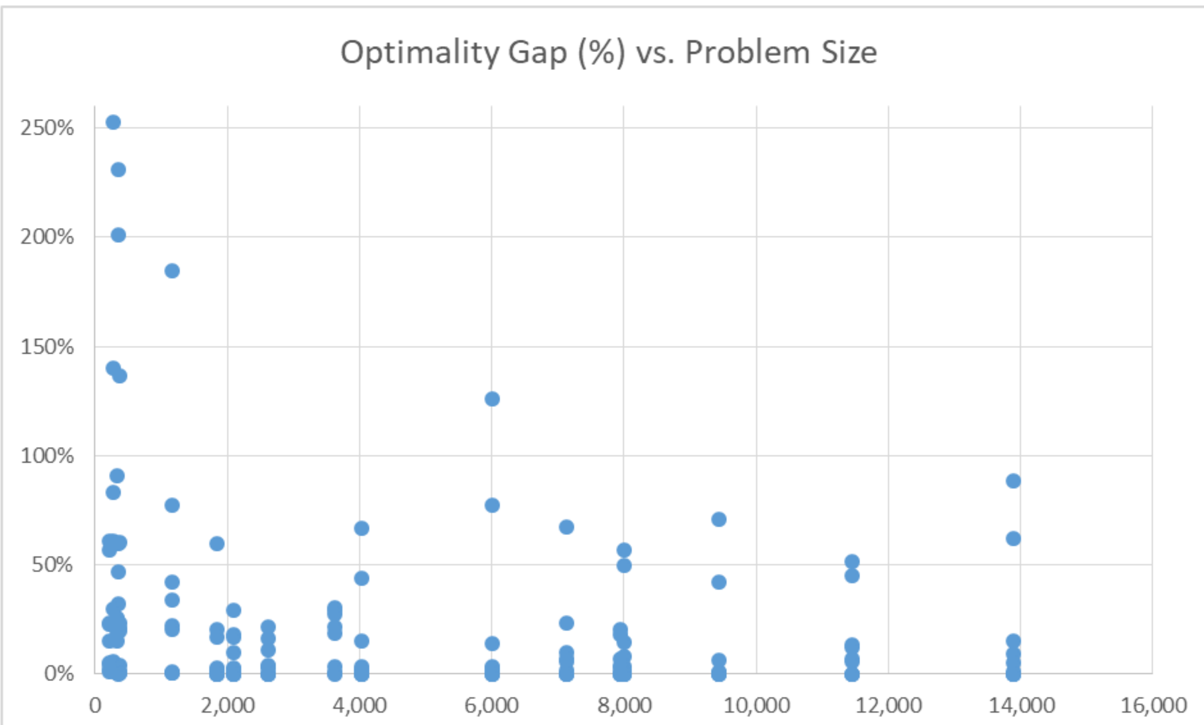

Figure 4. Optimality gap of test cases plotted against problem size.

To determine heuristic performance under various scenarios, results are now analyzed with respect to the input parameters. The optimality gap metric will be the primary focus of the subsequent analyses. The summary statistics offered in Table 4 point to the location distribution parameter having no significant effect in respect to solution quality. However, Table 5 shows that test cases with “high” fixed cost parameters perform well, offering optimality gaps within a range of 0% (i.e., optimal) to 4.9%. It is conjectured that the algorithm performs better for this scenario because higher fixed costs will influence the optimal solution to have less opened facilities in the final network topology. This minimizes the set of potentially “promising” locations that the algorithm must consider to opened, thus increasing the likelihood of arriving at a near-optimal solution within the heuristic search tree. Another reason could be in the algorithm framework. Because the heuristic greedily focuses on the binary location variable values when determining which locations to be opened or closed, the objective function value becomes significantly influenced by the fixed cost component ($f_j x_j$ in the model formulation). Hence, the solution accuracy performance becomes dependent on the fixed cost test case parameter – as the results indicate, this appears beneficial when solving problems with “high” fixed costs but problematic for the “medium” and “low” cases. Additional experiments will be conducted in Section 3.3 for further evaluation.

3.3. Additional test cases controlling for the fixed cost parameter

In order to validate the results observed in Section 3.2, 60 additional test cases are examined. As we assume the location distribution parameter has no effect on solution outcomes, the additional cases take on the “uniformly distributed” scenario. Table 6 offers summary statistics of these additional test cases. The results further validate the explored conjecture: excluding the outlier test case having a 91.2% optimality gap, the remaining 19 cases of the “High” fixed cost level have minimum, average, and maximum optimality gaps of 0.0%, 0.4%, and 2.7%, respectively. (The full results are available in Appendix A.2.) The algorithm may be an effective alternative to using a commercial MIP solver when solving large-scale problems

Table 4. Overview of experiment results summarized by the location distribution parameter.

Location distribution scenario	Test cases considered	CPU (seconds) Algorithm			CPU (seconds) CPLEX-MIP			Optimality Gap (%)			"Timeout" test cases
		Min.	*Avg.*	*Max.*	*Min.*	*Avg.*	*Max.*	*Min.*	*Avg.*	*Max.*	
Uniform	54	0.015	0.256	1.006	0.074	780.557	3652.280	0.0%	19.9%	252.7%	9
1-centroid	54	0.015	0.263	1.061	0.103	803.873	3638.090	0.0%	22.5%	140.1%	10
2-centroid	54	0.016	0.277	1.231	0.052	947.899	3671.130	0.0%	25.5%	230.8%	12

Table 5. Overview of experiment results summarized by the fixed cost level parameter.

Fixed cost level scenario	Test cases considered	CPU (seconds) Algorithm			CPU (seconds) CPLEX-MIP			Optimality Gap (%)			"Timeout" test cases
		Min.	*Avg.*	*Max.*	*Min.*	*Avg.*	*Max.*	*Min.*	*Avg.*	*Max.*	
Low	54	0.022	0.235	1.102	0.052	1290.405	3671.130	0.0%	33.2%	126.0%	17
Medium	54	0.022	0.282	1.231	0.090	598.249	3615.010	0.6%	34.2%	252.7%	6
High	54	0.015	0.279	1.115	0.087	643.673	3615.200	0.0%	0.5%	4.9%	8

Table 6. Overview of additional experiment results summarized by the fixed cost level parameter.

Fixed cost level scenario	Test cases considered	Optimality Gap (%)			"Timeout" test cases
		Min.	*Avg.*	*Max.*	
Low	20	0.6%	11.3%	31.3%	4
Medium	20	0.2%	48.0%	151.9%	0
High	20	0.0%	5.0%	91.2%	1

with relatively high facility fixed costs. Problems where facility consolidation is expected to be the optimal strategy could be an area of effective application.

3.4. Modifying the infeasibility threshold

The conducted experiments show the heuristic performs well for problems of the "high" fixed cost parameter. In an effort to improve solution accuracy for the other test cases, specifically the "low" and "medium" fixed cost problems, modification of the infeasibility threshold *T* is explored. Relaxing *T* to a higher value enables the algorithm to explore more solutions and potentially leading to better solutions. For the 60 additional test cases generated for Section 3.3, four values of *T* were implemented as follows.

$$T = \begin{cases} \frac{\sqrt{|J|}}{2} \ [default\ setting] \\ \sqrt{|J|} \\ 2\sqrt{|J|} \\ \frac{|J|}{2} \ [i.e.,\ no\ threshold] \end{cases}$$

It was found that in 4 out of the 60 cases, the revised *T* values led to solution accuracy improvements. Tradeoff curves showing runtime performance and optimality gap for test cases *A*, *B*, and *C* (of the "low" fixed cost parameter) and test case *D* (of the "medium" fixed cost parameter) are depicted in Figures 5 through 8 below. Whether modifying the *T* value (resulting in longer computation times) being worth the solution accuracy improvement is up to interpretation. However, because runtime performance is still very fast (at less than a few seconds for all cases) and only seems to be growing linearly, this modification can be an easy way to marginally improve the algorithm's solution accuracies with minimal effort.

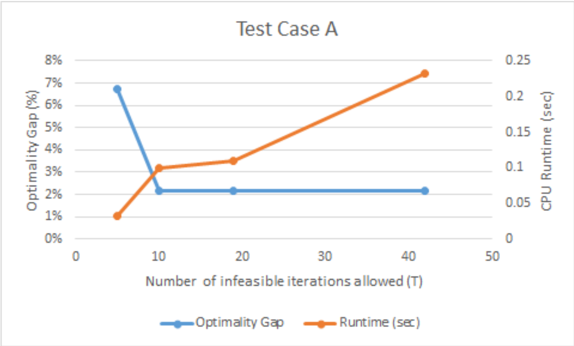

Figure 5. Runtime vs. optimality gap tradeoff curve for test case A.

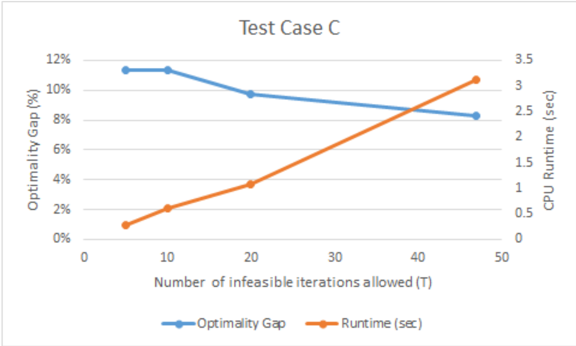

Figure 7. Runtime vs. optimality gap tradeoff curve for test case C.

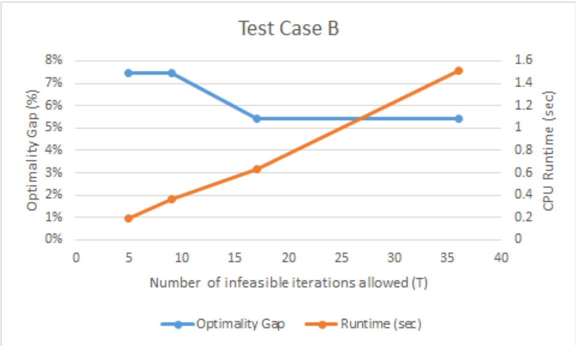

Figure 6. Runtime vs. optimality gap tradeoff curve for test case B.

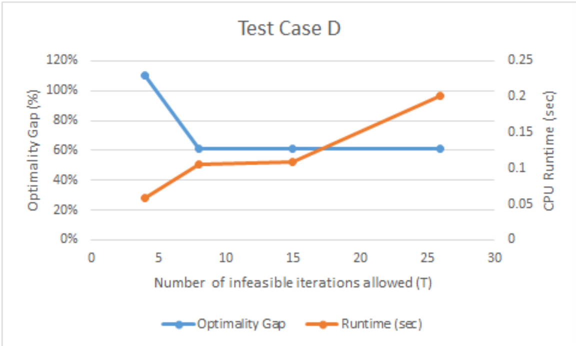

Figure 8. Runtime vs. optimality gap tradeoff curve for test case D.

4. APPLICATION TO A REAL-WORLD PROBLEM

The results from Section 3 illustrate the algorithm's potential as an effective method to solve CFLPs. However, because simplifying assumptions were made for the test cases considered thus far (such as using uniform fixed costs and capacities for all candidate facilities), the heuristic still has not proven its effectiveness for solving more realistic problems observed in industry. To examine whether the developed algorithm has practical applicability, a business problem was modeled to perform additional computational tests on. The considered problem is a CFLP adapted from a 2017 study of a warehouse network optimization project, conducted for an industrial materials distributor in the United States. The instance was slightly modified to fit the scope of the considered problem type for this paper, and the data altered for confidentiality.

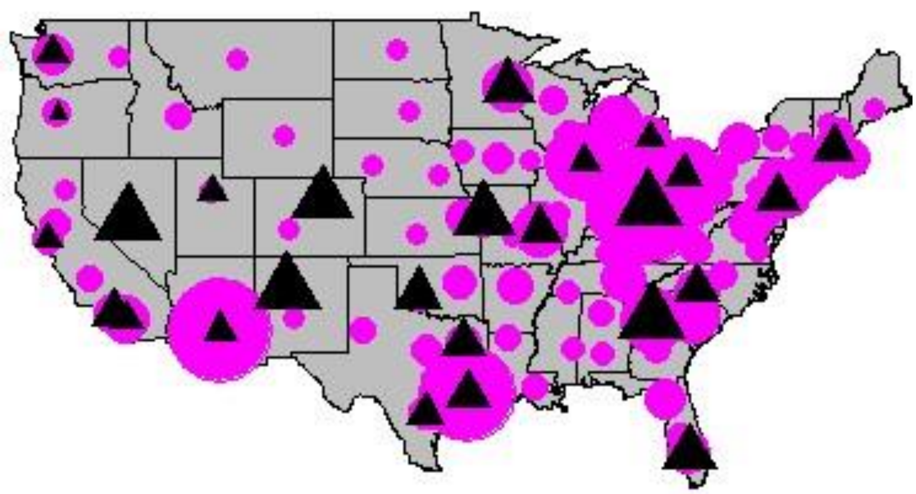

Figure 9. Map of all candidate warehouse locations (black triangles) and customer demand (magenta circles).

In this problem, the distributor is looking to optimize its warehouse network across the continental United States. There are 28 potential warehouse locations to select from, each with varying facility capacities and fixed costs. The optimal set of warehouses must satisfy all customer demand points, which are aggregated by the first 2-digit prefixes of US postal zipcodes. Figure 9 visualizes the customer and demand nodes geographically.

Two types of costs are considered: fixed and transportation. The distributor seeks to find the optimal network topology of warehouse placement and customer assignment in order to minimize total annual expenditures. For simplicity, all products that the distributor handles are considered to be of one type, with tons being the universal measure of volume. Distances are computed using the great circle method between geocoordinates of zipcodes. Similar to the experiments conducted in Section 3, three different cases of fixed cost levels were examined: "Standard," "High," and "Low." In the "High" scenario, the fixed costs were amplified by a factor of 1.5 relative to the "Standard" case, and for the "Low" scenario, scaled down by a factor of 1.5. Refer to Appendix B.1 for all problem parameters.

Results from the three runs are summarized in Table 7. As expected from the findings in Section 3, the heuristic algorithm indeed reduces the problem solving runtimes. Additionally, the solution accuracy is best for the "High" test case at an optimality gap of 5.6%, followed by the "Standard" (24.9%) and "Low" (47.1%) instances.

Table 7. Summary of warehouse problem solutions.

Problem Type	**Algorithm**		**CPLEX-MIP**		**Optimality Gap (%)**
	Total cost ($)	***CPU (seconds)***	***Total cost ($)***	***CPU (seconds)***	
Standard	$4,349,580	0.93	$3,483,260	3.43	24.9%
High	$4,770,720	0.53	$4,516,170	5.11	5.6%
Low	$4,044,090	0.11	$2,749,920	12.82	47.1%

Furthermore, detailed results of opened warehouse locations and their handled volumes for the “High” problem are summarized in Table 8 (results from the other two scenarios are available in Appendix B.2). It is observed that the solutions produced by both the algorithm and CPLEX-MIP are very similar, as validated by the maps of Figures 10 and 11. These results provide a promising outlook for the practicality of the developed heuristic, showing the algorithm is indeed capable of selecting “promising” candidate facility sites that are also selected in the true optimal solution set.

Table 8. Opened warehouses in Algorithm and CPLEX-MIP solutions for the “High” problem scenario.

Warehouse location	Fixed cost ($)	Capacity (tons)	Volume handled in Algorithm solution (tons)	Volume handled in CPLEX-MIP solution (tons)
Atlanta, GA	$750,000	15,000	8,248	13,896
Cincinnati, OH	$750,000	15,000	13,506	15,000
Cleveland, OH	$450,000	9,000	9,000	
Dallas, TX	$600,000	10,000		10,000
Detroit, MI	$300,000	7,000	7,000	7,000
Houston, TX	$600,000	10,000	8,142	
Tucson, AZ	$450,000	8,000	8,000	8,000
		Total	**53,896**	**53,896**

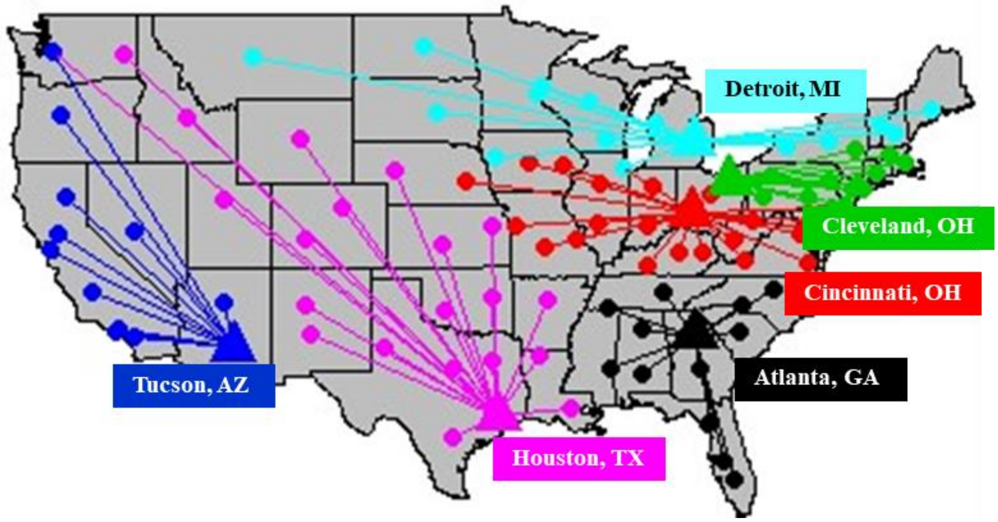

Figure 10. Optimal network topology given by algorithm solution for the “High” problem scenario.

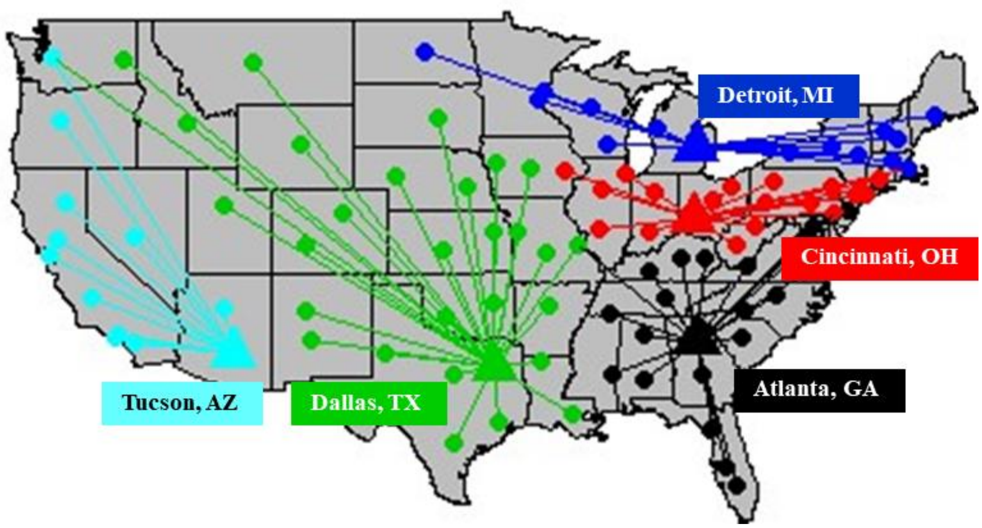

Figure 11. Optimal network topology given by CPLEX-MIP solution for the “High” problem scenario.

5. CONCLUSIONS AND FUTURE WORK

The algorithm presented in this paper is a straightforward and applicable heuristic incorporating greedy-based variable fixing and iterative LP relaxation solving to solve CFLPs. The methodology offers an alternative to solving the MIP to optimality – which can take exponential CPU time due to the existence of binary location variables in the formulation – by providing a heuristic solution in a fraction of required CPU time while attaining acceptable solution accuracy in certain scenarios. The experiments offer validation of the computational benefits, with additional insight that the heuristic performs best for problem instances involving facilities having relatively high fixed cost levels. The promising results from solving the warehouse network problem in Section 4 also boost confidence in the algorithm being a viable method for solving similar problems in an industry setting.

The savings in computation times are apparent from the experiments run for this paper, but there is significant room for improvement in the solution accuracy aspect. Though not explored in this paper, combining the heuristic and CPLEX-MIP solving techniques could be a promising idea. Because the proposed algorithm returns a solution in minimal CPU time, problems may be first solved using the algorithm, and the resulting solution could be provided as a "warm start" input to the CPLEX-MIP solver, which will then solve the problem to optimality.

Currently the algorithm only considers the binary location variable values in the variable fixing process. In future work, the algorithm could be revisited so that other elements are also considered when making greedy-based branching decisions. As discussed in Section 3.2, the current algorithm framework presumably leads to the over-prioritization of the fixed cost component when deciding which locations are "promising" or not. It is conjectured that a more sophisticated heuristic rule, such as one that considers the tradeoffs between both fixed and variable cost influences, may be more appropriate. For instance, shadow prices, reduced costs, and non-binary (i.e., flow) variable values could be additional metrics to consider. Evaluating additional information hopefully will result in obtaining better solutions for problems of various types.

Although the paper only explores a limited scope of the applicability of our method, we predict that this framework can be applied to solve not only just CFLPs, but to a broader scope of MIPs as well. The algorithm's main benefit is being able to produce heuristic solutions very quickly, so applications requiring the fast and scalable reproduction of solutions are potential use cases. Some examples could be determining optimal power grid usage that can reflect instantaneous changes in load demand, or enabling efficient computation of service patterns in the sharing economy (such as with ridesharing in Uber). Cloud manufacturing could also be another use case for the methodology. As introduced by Wu et al. (2013) and Wu et al. (2015), this paradigm enables system models to access a shared collection of various manufacturing resources. The developed heuristic enables the required scalability in solving manufacturing-related MIP models that is required to efficiently reflect tremendous amount of input information being updated instantaneously.

Due to the straightforward framework and flexibility of the algorithm, we believe the developed heuristic has a strong potential for further exploration in both theory and practice.

REFERENCES

Angelelli, E., Mansini, R., & Speranza, M. G. (2012). Kernel Search: a new heuristic framework for portfolio selection. *Computational Optimization and Applications*, *51*, 345-361.

Francis, R.L., McGinnis, L. F., & White, J.A. (1983). Locational analysis. *European Journal of Operations Research*, *12(3)*, 220-252.

Gustaroba, G. & Speranza, M. G. (2014). A heuristic for BILP problems: The Single Source Capacitated Facility Location Problem. *European Journal of Operational Research*, *238*, 438-450.

Lu, D., Gzara, F., & Elhedhli, S. (2013). Facility Location with economies and diseconomies of scale: models and column generation heuristics. *IIE Transactions*, *46(6)*, 585-600.

Marmolejo, J. A., Soria, I., & Perez, H. A. (2015). A Decomposition Strategy for Optimal Design of a Soda Company Distribution System. *Mathematical Problems in Engineering*, *2015*, 7 pages.

Melo, M.T., Nickel, S., & Saldanha-da-Gama, F. (2014). An efficient heuristic approach for a multi-period logistics network redesign problem. *TOP*, *22(1)*, 80-108.

Murray W. & Shanbhag U. V. (2006). A Local Relaxation Method for Nonlinear Facility Location Problems. In: Hager W.W., Huang SJ., Pardalos P.M., Prokopyev O.A. (eds) Multiscale Optimization Methods and Applications. Nonconvex Optimization and Its Applications, vol 82. Springer, Boston, MA.

Naguney, A. (2010). Optimal supply chain network design and redesign at minimal total cost and with demand satisfaction. *Int. J. Production Economics*, *128*, 200-208.

Renegar, J. (1988). A polynomial-time algorithm, based on Newton's method, for linear programming. *Mathematical Programming*, *40*, 59-63.

Teixeira, J. C., Antunes, A. P., Júdice, J. J. & Martins, P.C. (2006). Solving facility location models with modern optimization software: the weak and strong formulations revisited. Retrieved from <https://www.semanticscholar.org/paper/Solving-facility-location-models-with-modern-%3A-the-Teixeira-Antunes/ce1f0fa41a069d23a0c71f7289a491d462214705?tab=abstract>.

Thanh, P. N., Péton, O., & Bostel, N. (2010). A linear relaxation-based approach for logistics network design. *Computers & Industrial Engineering*, *59*, 964-975.

Wu, D., Greer, M. J., Rosen, D. W., & Schaefer, D. (2013). Cloud manufacturing: Strategic vision and state-of-the-art. *Journal of Manufacturing Systems*, *32*, 564-579.

Wu, D., Rosen, D. W., Wang, L., & Schaefer, D. (2015). Cloud-based design and manufacturing: A new paradigm in digital manufacturing and design innovation. *Computer-Aided Design*, *59*, 1-14.

APPENDIX A.

DETAILED COMPUTATIONAL RESULTS FOR TEST CASES CONSIDERED IN SECTION 3

A.1. Full results of experiments run for Section 3.2

Table 9. Results for all 162 test case experiments conducted.

Location distribution scenario	Fixed cost level scenario	$\lvert I \rvert$	$\lvert J \rvert$	Problem size	CPU (seconds) Algorithm	CPU (seconds) CPLEX-MIP	Optimality Gap (%)	"Timeout" occurrence (*)
2-centroid	Medium	36	50	1,850	0.087	6.347	1.9%	
2-centroid	Medium	59	35	2,100	0.074	1.827	2.6%	
2-centroid	Medium	38	30	1,170	0.044	0.925	41.9%	
2-centroid	Medium	44	58	2,610	0.104	9.499	4.0%	
2-centroid	Medium	65	55	3,630	0.247	4.487	30.2%	
2-centroid	Low	36	50	1,850	0.049	3610.090	20.3%	*
2-centroid	Low	59	35	2,100	0.103	0.346	18.1%	
2-centroid	Low	38	30	1,170	0.063	0.752	77.5%	
2-centroid	Low	44	58	2,610	0.052	3615.920	16.2%	*
2-centroid	Low	65	55	3,630	0.537	1.486	18.4%	
2-centroid	High	36	50	1,850	0.096	0.822	0.0%	
2-centroid	High	59	35	2,100	0.055	1.327	0.1%	
2-centroid	High	38	30	1,170	0.026	1.327	0.2%	
2-centroid	High	44	58	2,610	0.095	3609.140	0.1%	*
2-centroid	High	65	55	3,630	0.140	5.125	0.2%	
1-centroid	Medium	36	50	1,850	0.081	1.881	1.0%	
1-centroid	Medium	59	35	2,100	0.054	3.577	1.9%	
1-centroid	Medium	38	30	1,170	0.031	1.200	22.2%	
1-centroid	Medium	44	58	2,610	0.097	13.170	3.5%	
1-centroid	Medium	65	55	3,630	0.126	8.686	3.6%	
1-centroid	Low	36	50	1,850	0.046	1451.840	59.7%	
1-centroid	Low	59	35	2,100	0.109	0.614	29.3%	
1-centroid	Low	38	30	1,170	0.066	0.535	33.9%	
1-centroid	Low	44	58	2,610	0.052	3610.400	21.4%	*
1-centroid	Low	65	55	3,630	0.501	2.614	21.3%	
1-centroid	High	36	50	1,850	0.087	0.306	0.0%	
1-centroid	High	59	35	2,100	0.054	4.074	0.1%	
1-centroid	High	38	30	1,170	0.025	0.986	0.2%	
1-centroid	High	44	58	2,610	0.100	3615.200	0.1%	*
1-centroid	High	65	55	3,630	0.149	31.416	0.1%	
Uniform	Medium	36	50	1,850	0.081	19.221	2.7%	
Uniform	Medium	59	35	2,100	0.067	1.697	17.1%	
Uniform	Medium	38	30	1,170	0.050	0.681	184.6%	
Uniform	Medium	44	58	2,610	0.105	14.172	2.5%	
Uniform	Medium	65	55	3,630	0.216	9.126	29.1%	
Uniform	Low	36	50	1,850	0.036	3565.780	16.8%	*

Table 9. (Continued)

Location distribution scenario	Fixed cost level scenario	|*I*|	|*J*|	Problem size	CPU (seconds) Algorithm	CPU (seconds) CPLEX-MIP	Optimality Gap (%)	"Timeout" occurrence (*)
Uniform	Low	59	35	2,100	0.092	0.150	10.1%	
Uniform	Low	38	30	1,170	0.067	0.108	20.1%	
Uniform	Low	44	58	2,610	0.056	3613.710	11.2%	*
Uniform	Low	65	55	3,630	0.600	0.367	27.5%	
Uniform	High	36	50	1,850	0.079	4.930	0.1%	
Uniform	High	59	35	2,100	0.061	1.047	0.1%	
Uniform	High	38	30	1,170	0.026	0.743	0.9%	
Uniform	High	44	58	2,610	0.106	3613.200	0.0%	*
Uniform	High	65	55	3,630	0.135	12.764	0.2%	
2-centroid	Medium	14	23	345	0.030	0.263	230.8%	
2-centroid	Medium	21	10	220	0.054	0.099	56.5%	
2-centroid	Medium	13	26	364	0.032	0.615	136.3%	
2-centroid	Medium	13	24	336	0.027	0.379	25.8%	
2-centroid	Medium	15	17	272	0.032	0.259	83.4%	
2-centroid	Low	14	23	345	0.026	0.267	46.7%	
2-centroid	Low	21	10	220	0.023	0.052	15.1%	
2-centroid	Low	13	26	364	0.025	0.173	19.6%	
2-centroid	Low	13	24	336	0.025	0.229	15.0%	
2-centroid	Low	15	17	272	0.024	0.150	61.0%	
2-centroid	High	14	23	345	0.060	0.253	2.9%	
2-centroid	High	21	10	220	0.016	0.087	4.9%	
2-centroid	High	13	26	364	0.024	1.146	0.7%	
2-centroid	High	13	24	336	0.024	0.283	0.4%	
2-centroid	High	15	17	272	0.019	0.256	4.8%	
1-centroid	Medium	14	23	345	0.025	0.282	32.2%	
1-centroid	Medium	21	10	220	0.030	0.103	60.6%	
1-centroid	Medium	13	26	364	0.028	0.288	20.9%	
1-centroid	Medium	13	24	336	0.028	0.340	23.5%	
1-centroid	Medium	15	17	272	0.023	0.208	140.1%	
1-centroid	Low	14	23	345	0.023	0.323	59.6%	
1-centroid	Low	21	10	220	0.033	0.139	23.5%	
1-centroid	Low	13	26	364	0.027	0.433	60.4%	
1-centroid	Low	13	24	336	0.026	0.265	18.4%	
1-centroid	Low	15	17	272	0.022	0.204	29.6%	
1-centroid	High	14	23	345	0.023	0.354	2.2%	
1-centroid	High	21	10	220	0.015	0.108	1.0%	
1-centroid	High	13	26	364	0.023	1.071	0.9%	
1-centroid	High	13	24	336	0.022	0.295	0.4%	
1-centroid	High	15	17	272	0.017	0.257	3.0%	
Uniform	Medium	14	23	345	0.024	0.254	201.0%	
Uniform	Medium	21	10	220	0.022	0.090	23.0%	
Uniform	Medium	13	26	364	0.029	0.215	23.3%	
Uniform	Medium	13	24	336	0.026	0.327	90.5%	
Uniform	Medium	15	17	272	0.025	0.262	252.7%	
Uniform	Low	14	23	345	0.023	0.329	0.0%	
Uniform	Low	21	10	220	0.028	0.074	4.6%	
Uniform	Low	13	26	364	0.023	0.312	4.1%	
Uniform	Low	13	24	336	0.022	0.316	25.1%	

Table 9. (Continued)

Location distribution scenario	Fixed cost level scenario	|*I*|	|*J*|	Problem size	CPU (seconds) Algorithm	CPU (seconds) CPLEX-MIP	Optimality Gap (%)	"Timeout" occurrence (*)
Uniform	Low	15	17	272	0.024	0.103	5.7%	
Uniform	High	14	23	345	0.024	0.244	0.4%	
Uniform	High	21	10	220	0.015	0.103	2.1%	
Uniform	High	13	26	364	0.025	1.363	0.2%	
Uniform	High	13	24	336	0.020	0.338	0.3%	
Uniform	High	15	17	272	0.022	0.233	2.2%	
2-centroid	Medium	71	99	7,128	0.402	3601.570	5.6%	*
2-centroid	Medium	70	112	7,952	0.403	29.125	1.3%	
2-centroid	Medium	77	121	9,438	1.231	3606.700	0.7%	*
2-centroid	Medium	106	107	11,449	0.614	189.689	6.7%	
2-centroid	Medium	111	124	13,888	0.738	3600.110	1.2%	*
2-centroid	Low	71	99	7,128	0.291	3671.130	23.5%	*
2-centroid	Low	70	112	7,952	0.399	3620.290	18.0%	*
2-centroid	Low	77	121	9,438	0.412	3611.790	71.1%	*
2-centroid	Low	106	107	11,449	0.527	3618.210	51.7%	*
2-centroid	Low	111	124	13,888	1.102	3636.430	62.2%	*
2-centroid	High	71	99	7,128	0.475	248.827	0.1%	
2-centroid	High	70	112	7,952	0.470	20.748	0.0%	
2-centroid	High	77	121	9,438	1.115	3606.470	0.0%	*
2-centroid	High	106	107	11,449	0.558	56.401	0.0%	
2-centroid	High	111	124	13,888	0.765	3500.070	0.0%	*
1-centroid	Medium	71	99	7,128	0.477	582.388	10.1%	
1-centroid	Medium	70	112	7,952	0.485	221.062	3.3%	
1-centroid	Medium	77	121	9,438	1.005	3613.730	1.0%	*
1-centroid	Medium	106	107	11,449	0.637	654.941	13.5%	
1-centroid	Medium	111	124	13,888	0.825	3600.090	9.1%	*
1-centroid	Low	71	99	7,128	0.308	3611.140	67.2%	*
1-centroid	Low	70	112	7,952	0.368	3638.090	20.2%	*
1-centroid	Low	77	121	9,438	0.421	3615.780	42.3%	*
1-centroid	Low	106	107	11,449	0.512	1619.040	45.3%	
1-centroid	Low	111	124	13,888	0.621	3611.710	88.7%	*
1-centroid	High	71	99	7,128	0.458	30.828	0.0%	
1-centroid	High	70	112	7,952	0.501	89.887	0.0%	
1-centroid	High	77	121	9,438	1.061	3608.270	0.0%	*
1-centroid	High	106	107	11,449	0.580	3600.160	0.1%	*
1-centroid	High	111	124	13,888	0.756	329.536	0.0%	
Uniform	Medium	71	99	7,128	0.398	309.137	1.7%	
Uniform	Medium	70	112	7,952	0.443	2583.050	3.4%	
Uniform	Medium	77	121	9,438	1.006	3615.010	0.6%	*
Uniform	Medium	106	107	11,449	0.501	52.175	6.0%	
Uniform	Medium	111	124	13,888	0.701	1421.270	5.2%	
Uniform	Low	71	99	7,128	0.286	3616.180	6.8%	*
Uniform	Low	70	112	7,952	0.389	3652.280	6.7%	*
Uniform	Low	77	121	9,438	0.449	3631.530	6.4%	*
Uniform	Low	106	107	11,449	0.543	2.158	11.9%	
Uniform	Low	111	124	13,888	0.623	3605.580	15.3%	*
Uniform	High	71	99	7,128	0.395	374.560	0.0%	
Uniform	High	70	112	7,952	0.509	812.137	0.0%	

Table 9. (Continued)

Location distribution scenario	Fixed cost level scenario	|*I*|	|*J*|	Problem size	CPU (seconds) Algorithm	CPU (seconds) CPLEX-MIP	Optimality Gap (%)	"Timeout" occurrence (*)
Uniform	High	77	121	9,438	0.994	3607.260	0.0%	*
Uniform	High	106	107	11,449	0.606	78.216	0.1%	
Uniform	High	111	124	13,888	0.778	3601.470	0.1%	*
2-centroid	Medium	63	63	4,032	0.235	1934.830	1.6%	
2-centroid	Low	89	89	8,010	0.450	1085.340	49.7%	
1-centroid	Medium	77	77	6,006	0.454	1000.470	1.4%	
1-centroid	Medium	89	89	8,010	0.539	845.571	7.8%	
1-centroid	Low	89	89	8,010	0.323	328.750	57.0%	
2-centroid	Medium	89	89	8,010	0.449	273.362	3.2%	
2-centroid	Medium	77	77	6,006	0.397	224.272	3.5%	
Uniform	Medium	89	89	8,010	0.490	223.278	1.4%	
2-centroid	High	63	63	4,032	0.216	117.606	0.0%	
Uniform	High	89	89	8,010	0.509	37.189	0.0%	
2-centroid	High	89	89	8,010	0.465	25.268	0.0%	
Uniform	High	63	63	4,032	0.258	24.642	0.1%	
2-centroid	High	77	77	6,006	0.377	22.471	0.0%	
1-centroid	High	89	89	8,010	0.489	22.205	0.1%	
Uniform	High	77	77	6,006	0.407	22.091	0.0%	
1-centroid	High	63	63	4,032	0.271	12.131	0.0%	
1-centroid	Low	77	77	6,006	0.267	11.085	126.0%	
Uniform	Medium	77	77	6,006	0.388	9.071	2.9%	
2-centroid	Low	77	77	6,006	0.355	7.632	77.2%	
1-centroid	Medium	63	63	4,032	0.266	7.138	2.3%	
Uniform	Medium	63	63	4,032	0.252	6.949	3.4%	
2-centroid	Low	63	63	4,032	0.293	4.257	66.5%	
1-centroid	Low	63	63	4,032	0.161	2.829	43.9%	
Uniform	Low	89	89	8,010	0.319	1.291	14.4%	
1-centroid	High	77	77	6,006	0.454	1.127	0.0%	
Uniform	Low	77	77	6,006	0.300	0.766	13.8%	
Uniform	Low	63	63	4,032	0.172	0.527	15.0%	

A.2. Full results of experiments run for Section 3.3

Table 10. Results for all 60 test cases conducted for additional experiments.

Location distribution scenario	Fixed cost level scenario	|*I*|	|*J*|	Problem size	CPU (seconds) Algorithm	CPU (seconds) CPLEX-MIP	Optimality Gap (%)	"Timeout" occurrence (*)
Uniform	High	55	37	2,072	0.034	12.177	0.5%	
Uniform	High	34	10	350	0.022	0.161	91.2%	
Uniform	High	16	83	1,411	0.068	1.786	0.0%	
Uniform	High	73	32	2,368	0.047	3.308	0.6%	
Uniform	High	32	67	2,211	0.070	3.215	0.1%	
Uniform	High	64	27	1,755	0.028	0.926	2.7%	
Uniform	High	71	31	2,232	0.048	2.815	0.3%	
Uniform	High	23	97	2,328	0.150	0.287	0.0%	
Uniform	High	84	73	6,205	0.247	3601.490	0.2%	*
Uniform	High	20	53	1,113	0.037	0.954	0.1%	
Uniform	High	88	79	7,031	0.471	42.268	0.0%	
Uniform	High	80	32	2,592	0.034	2.112	0.9%	
Uniform	High	37	52	1,976	0.042	3.359	0.2%	
Uniform	High	61	30	1,860	0.031	0.899	0.5%	
Uniform	High	36	30	1,110	0.021	0.565	0.6%	
Uniform	High	47	72	3,456	0.290	0.419	0.0%	
Uniform	High	64	94	6,110	0.541	27.983	0.0%	
Uniform	High	99	45	4,500	0.214	3.089	0.2%	
Uniform	High	51	28	1,456	0.026	0.680	0.3%	
Uniform	High	29	12	360	0.012	0.536	0.8%	
Uniform	Low	55	37	2,072	0.094	0.212	12.3%	
Uniform	Low	34	10	350	0.025	0.069	2.9%	
Uniform	Low	16	83	1,411	0.033	0.586	6.7%	
Uniform	Low	73	32	2,368	0.098	0.135	8.7%	
Uniform	Low	32	67	2,211	0.054	3603.950	4.4%	*
Uniform	Low	64	27	1,755	0.067	0.217	12.0%	
Uniform	Low	71	31	2,232	0.094	0.309	4.9%	
Uniform	Low	23	97	2,328	0.042	2.694	0.6%	
Uniform	Low	84	73	6,205	1.149	0.646	15.6%	
Uniform	Low	20	53	1,113	0.030	4.109	5.6%	
Uniform	Low	88	79	7,031	0.992	0.748	15.7%	
Uniform	Low	80	32	2,592	0.088	0.323	8.6%	
Uniform	Low	37	52	1,976	0.035	3617.030	14.3%	*
Uniform	Low	61	30	1,860	0.132	0.160	14.9%	
Uniform	Low	36	30	1,110	0.073	0.183	20.2%	
Uniform	Low	47	72	3,456	0.190	3617.020	7.5%	*
Uniform	Low	64	94	6,110	0.279	3665.820	11.3%	*
Uniform	Low	99	45	4,500	0.597	0.388	15.5%	
Uniform	Low	51	28	1,456	0.074	0.111	12.1%	
Uniform	Low	29	12	360	0.021	0.087	31.3%	
Uniform	Medium	55	37	2,072	0.075	1.782	68.1%	
Uniform	Medium	34	10	350	0.018	0.150	8.6%	
Uniform	Medium	16	83	1,411	0.050	0.927	0.8%	

Table 10. (Continued)

Location distribution scenario	Fixed cost level scenario	$\|I\|$	$\|J\|$	Problem size	CPU (seconds) Algorithm	CPU (seconds) CPLEX-MIP	Optimality Gap (%)	"Timeout" occurrence (*)
Uniform	Medium	73	32	2,368	0.071	1.332	70.1%	
Uniform	Medium	32	67	2,211	0.060	2.861	3.7%	
Uniform	Medium	64	27	1,755	0.072	1.016	107.1%	
Uniform	Medium	71	31	2,232	0.085	1.310	85.0%	
Uniform	Medium	23	97	2,328	0.121	15.738	0.2%	
Uniform	Medium	84	73	6,205	0.389	24.032	52.9%	
Uniform	Medium	20	53	1,113	0.056	1.203	3.2%	
Uniform	Medium	88	79	7,031	0.433	79.057	1.2%	
Uniform	Medium	80	32	2,592	0.093	2.005	73.7%	
Uniform	Medium	37	52	1,976	0.059	4.380	110.5%	
Uniform	Medium	61	30	1,860	0.061	1.601	55.3%	
Uniform	Medium	36	30	1,110	0.041	0.691	151.9%	
Uniform	Medium	47	72	3,456	0.315	16.959	1.0%	
Uniform	Medium	64	94	6,110	0.550	156.280	1.1%	
Uniform	Medium	99	45	4,500	0.427	4.825	38.1%	
Uniform	Medium	51	28	1,456	0.084	0.883	107.3%	
Uniform	Medium	29	12	360	0.022	0.176	20.5%	

APPENDIX B.

PARAMETERS AND RESULTS OF REAL-LIFE PROBLEM CONSIDERED IN SECTION 4

B.1. Input parameters

Table 11. Warehouse location input parameters.

Warehouse location	Zipcode 2-digit prefix	Capacity (tons)	Fixed cost ($) Standard	Fixed cost ($) High	Fixed cost ($) Low
Boston, MA	01	10,000	$1,000,000	$1,500,000	$666,666
Phoenix, AZ	85	6,000	$300,000	$450,000	$200,000
Atlanta, GA	30	15,000	$500,000	$750,000	$333,333
Philadelphia, PA	19	10,000	$500,000	$750,000	$333,333
Los Angeles, CA	91	10,000	$1,200,000	$1,800,000	$800,000
Chicago, IL	60	8,000	$500,000	$750,000	$333,333
Cincinnati, OH	45	15,000	$500,000	$750,000	$333,333
Houston, TX	77	10,000	$400,000	$600,000	$266,666
Dallas, TX	75	10,000	$400,000	$600,000	$266,666
Denver, CO	80	14,000	$700,000	$1,050,000	$466,666
Miami, FL	33	12,000	$800,000	$1,200,000	$533,333
Detroit, MI	48	7,000	$200,000	$300,000	$133,333
Minneapolis, MN	55	12,000	$500,000	$750,000	$333,333
St. Louis, MO	63	10,000	$400,000	$600,000	$266,666
Charlotte, NC	28	10,000	$500,000	$750,000	$333,333
Portland, OR	97	5,000	$400,000	$600,000	$266,666
Seattle, WA	98	8,000	$700,000	$1,050,000	$466,666
Salt Lake City, UT	84	7,000	$500,000	$750,000	$333,333
Cleveland, OH	44	9,000	$300,000	$450,000	$200,000
Kansas City, MO	64	14,000	$600,000	$900,000	$400,000
San Antonio, TX	78	9,000	$400,000	$600,000	$266,666
Austin, TX	78	7,000	$400,000	$600,000	$266,666
Tucson, AZ	85	8,000	$300,000	$450,000	$200,000
Las Vegas, NV	89	15,000	$700,000	$1,050,000	$466,666
Oklahoma City, OK	73	11,000	$600,000	$900,000	$400,000
San Francisco, CA	94	7,000	$1,300,000	$1,950,000	$866,666
Albuquerque, NM	87	15,000	$400,000	$600,000	$266,666
San Diego, CA	92	8,000	$600,000	$900,000	$400,000

Table 12. Customer demand location input parameters.

Customer Zipcode 2-digit Prefix	Demand (tons)	Customer Zipcode 2-digit Prefix	Demand (tons)	Customer Zipcode 2-digit Prefix	Demand (tons)
01	988	**35**	213	**68**	44
02	773	**36**	135	**69**	1
03	253	**37**	1,010	**70**	238
04	71	**38**	123	**71**	235
05	76	**39**	112	**72**	587
06	582	**40**	404	**73**	459
07	1,042	**41**	139	**74**	493
08	921	**42**	93	**75**	582
10	294	**43**	681	**76**	405
11	450	**44**	2,144	**77**	2,956
12	260	**45**	3,931	**78**	502
13	210	**46**	277	**79**	186
14	796	**47**	723	**80**	209
15	377	**48**	861	**81**	53
16	381	**49**	1,216	**82**	16
17	150	**50**	422	**83**	262
18	600	**51**	127	**84**	148
19	893	**52**	21	**85**	3,238
20	46	**53**	381	**86**	70
21	764	**54**	291	**87**	79
22	886	**55**	1,190	**88**	8
23	177	**56**	82	**89**	279
24	545	**57**	14	**90**	590
25	168	**58**	6	**91**	756
26	43	**59**	11	**92**	1,040
27	360	**60**	2,160	**93**	215
28	938	**61**	232	**94**	449
29	1,384	**62**	91	**95**	374
30	1,758	**63**	1,382	**96**	12
31	283	**64**	749	**97**	285
32	833	**65**	88	**98**	772
33	883	**66**	561	**99**	13
34	216	**67**	69		

Table 13. Transportation cost matrix for all warehouse-customer combinations ($/ton).

	Boston, MA	Phoenix, AZ	Atlanta, GA	Philadelphia, PA	Los Angeles, CA	Chicago, IL	Cincinnati, OH	Houston, TX	Dallas, TX	Denver, CO	Miami, FL	Detroit, MI	Minneapolis, MN	St. Louis, MO	Charlotte, NC	Portland, OR	Seattle, WA	Salt Lake City, UT	Cleveland, OH	Kansas City, MO	San Antonio, TX	Austin, TX	Tucson, AZ	Las Vegas, NV	Oklahoma City, OK	San Francisco, CA	Albuquerque, NM	San Diego, CA
01	0	222	87	25	254	82	65	156	146	170	119	60	105	103	68	246	243	205	50	119	173	173	222	236	152	263	194	250
02	6	226	89	28	259	87	69	159	150	176	119	65	111	107	70	252	249	211	55	124	176	176	226	241	156	269	198	256
03	7	224	93	31	255	84	69	161	150	172	126	61	105	106	74	245	242	206	53	122	177	177	224	236	155	264	196	252
04	20	236	106	45	266	95	81	174	163	182	138	72	114	118	88	253	249	216	66	134	191	191	236	246	167	274	207	263
05	10	220	92	31	251	79	66	158	148	167	127	57	101	103	74	241	237	202	50	118	175	175	220	232	151	260	192	248
06	8	217	80	18	250	78	59	149	140	167	112	56	104	97	61	244	242	202	45	114	166	166	217	232	146	260	189	246
07	16	209	71	9	243	71	51	141	131	160	104	51	99	90	52	239	238	196	38	107	157	157	209	226	138	254	181	239
08	22	207	65	4	241	70	49	136	127	159	98	51	100	87	46	239	238	195	37	105	153	153	207	225	135	253	179	237
10	14	211	73	11	244	73	53	142	133	162	106	52	100	91	54	240	239	197	40	109	159	159	211	227	140	256	183	241
11	14	213	73	12	246	75	55	144	135	163	105	54	102	93	54	242	241	199	42	111	161	161	213	229	141	257	185	243
12	10	211	81	20	244	71	55	148	137	160	116	49	95	93	63	236	233	195	40	109	164	164	211	225	142	253	183	240
13	19	203	77	21	235	63	48	141	130	151	115	40	86	85	60	227	224	186	33	101	158	158	203	216	134	244	175	232
14	31	191	69	24	223	51	37	131	119	139	111	28	76	73	54	216	214	174	21	89	147	147	191	205	122	232	163	220
15	41	182	51	21	215	45	24	115	105	133	94	28	76	62	36	214	214	169	12	80	132	132	182	199	111	228	154	212
16	37	184	58	22	217	45	28	121	110	134	101	26	74	65	43	214	213	170	13	82	137	137	184	200	114	228	156	214
17	29	196	59	8	229	59	38	127	118	147	97	40	88	76	41	228	227	183	25	94	144	144	196	213	124	241	168	226
18	21	202	67	8	235	64	45	135	125	153	103	43	91	83	49	231	230	188	31	100	152	152	202	218	131	247	174	232
19	25	202	62	0	236	66	44	132	123	154	96	47	96	82	43	235	235	190	33	101	149	149	202	221	131	249	175	233
20	36	194	51	11	229	61	37	122	113	147	87	45	93	74	32	230	231	184	28	93	139	139	194	214	122	242	166	225
21	33	196	54	8	230	61	38	124	116	149	89	45	93	76	35	231	231	185	29	95	141	141	196	215	124	244	168	227
22	38	193	49	13	227	60	36	119	111	146	84	45	93	73	30	230	231	183	28	92	137	137	193	213	120	241	165	224
23	44	195	45	20	231	67	41	118	111	151	76	54	101	76	26	236	238	188	37	96	135	135	195	218	122	246	168	227
24	58	174	30	33	210	49	23	99	91	130	72	42	85	55	14	217	219	168	26	75	116	116	174	197	101	225	147	206
25	57	171	34	33	206	42	16	99	90	126	79	35	78	51	21	211	213	163	19	71	116	116	171	192	98	221	143	202
26	48	177	42	25	212	44	20	108	98	130	86	32	78	57	28	214	215	167	15	76	125	125	177	197	105	225	150	208
27	58	184	31	33	221	62	36	104	98	142	65	54	98	67	11	230	232	180	38	87	122	122	184	209	110	237	157	216
28	68	174	19	43	211	59	34	93	88	135	58	56	95	59	0	223	227	172	41	79	111	111	174	201	101	229	148	207
29	76	173	15	51	211	66	42	89	85	137	49	64	102	62	9	226	231	174	49	82	106	106	173	202	100	230	148	207
30	87	159	0	62	197	60	40	74	70	124	50	64	95	51	19	214	219	161	52	70	92	92	159	189	86	217	133	192
31	94	162	11	69	201	71	51	73	72	130	39	74	106	60	26	222	228	167	62	78	91	91	162	195	90	222	138	196
32	107	169	29	82	209	89	69	76	79	143	21	92	123	75	41	234	241	179	79	93	93	93	169	205	99	232	146	204
33	119	184	50	96	224	110	89	89	95	162	0	112	144	96	58	253	261	197	98	113	104	104	184	223	117	249	163	219
34	115	178	43	91	218	103	82	83	89	155	7	106	137	89	52	247	254	191	92	106	99	99	178	216	110	243	156	213
35	99	140	19	75	178	54	42	58	52	106	60	64	86	37	36	197	204	143	57	54	75	75	140	171	67	198	115	174
36	109	142	22	84	182	68	55	53	52	114	48	78	100	49	41	206	213	150	69	65	71	71	142	177	71	204	119	177
37	85	147	18	61	184	43	27	71	63	108	67	50	77	34	27	198	202	146	42	53	88	88	147	174	74	202	121	180
38	104	128	31	81	166	47	42	52	43	93	72	62	76	25	46	184	190	130	58	41	69	69	128	158	55	185	102	162
39	115	132	31	91	171	67	58	43	41	104	57	80	96	44	50	196	204	140	73	58	61	61	132	166	61	193	108	166
40	74	153	28	51	188	33	14	82	72	109	77	37	69	34	27	196	200	146	28	54	99	99	153	176	80	204	125	184
41	67	161	28	43	197	38	13	89	80	117	76	36	74	42	22	204	206	154	24	62	106	106	161	184	88	212	134	193
42	85	142	28	62	178	33	22	72	61	101	77	43	67	25	34	189	193	138	38	45	88	88	142	167	69	195	115	174
43	57	165	47	38	198	28	9	102	91	116	95	17	62	46	38	198	199	152	7	63	118	118	165	182	94	210	137	195
44	50	171	52	33	204	33	16	110	98	122	98	17	64	53	41	203	203	158	0	69	125	125	171	188	101	216	143	201
45	65	158	40	44	192	26	0	94	82	111	89	24	62	38	34	195	197	148	16	56	110	110	158	177	87	206	130	188

Table 13. (Continued)

	Boston, MA	Phoenix, AZ	Atlanta, GA	Philadelphia, PA	Los Angeles, CA	Chicago, IL	Cincinnati, OH	Houston, TX	Dallas, TX	Denver, CO	Miami, FL	Detroit, MI	Minneapolis, MN	St. Louis, MO	Charlotte, NC	Portland, OR	Seattle, WA	Salt Lake City, UT	Cleveland, OH	Kansas City, MO	San Antonio, TX	Austin, TX	Tucson, AZ	Las Vegas, NV	Oklahoma City, OK	San Francisco, CA	Albuquerque, NM	San Diego, CA
46	74	148	51	56	180	11	15	92	78	98	101	19	47	30	49	180	182	134	24	45	107	107	148	164	79	192	120	177
47	79	143	40	59	178	20	15	81	68	97	89	31	56	24	41	183	186	134	30	42	96	96	143	164	72	192	116	174
48	60	164	64	47	195	23	24	111	97	111	112	0	49	49	56	188	187	146	17	62	126	126	164	176	97	204	136	191
49	71	154	71	60	184	17	32	110	94	100	120	13	36	46	66	175	174	134	30	55	123	123	154	164	91	192	126	181
50	110	114	78	95	144	29	53	85	67	60	126	50	25	30	83	142	144	96	62	21	94	94	114	126	56	154	87	141
51	120	106	88	106	134	40	64	86	67	50	134	61	27	38	94	131	133	85	73	24	93	93	106	115	53	143	78	131
52	100	124	70	84	154	18	43	86	68	71	119	40	27	25	74	152	154	106	51	25	97	97	124	137	61	165	96	151
53	86	139	70	73	169	10	36	98	81	85	120	27	26	34	70	162	163	119	40	40	110	110	139	150	76	177	111	165
54	91	140	84	81	167	24	48	108	91	83	133	34	15	46	82	155	154	116	50	47	119	119	140	146	82	173	112	164
55	105	130	95	96	155	36	62	110	92	72	144	49	0	52	95	140	139	103	64	46	119	119	130	133	79	160	103	153
56	107	128	94	97	153	36	62	107	89	70	143	50	3	50	95	139	138	101	65	44	116	116	128	131	76	158	100	150
57	136	101	110	124	124	59	85	102	84	42	156	78	32	60	116	111	112	71	92	44	107	107	101	101	65	128	76	122
58	140	116	127	132	133	71	97	124	106	58	175	85	36	78	130	107	103	78	100	65	129	129	116	106	87	131	92	131
59	188	96	166	179	99	115	141	142	126	57	210	132	83	115	172	58	56	48	147	97	141	141	96	67	102	88	83	99
60	82	141	60	66	172	0	26	92	77	89	110	23	36	28	59	170	171	125	33	39	106	106	141	155	74	183	113	169
61	90	132	59	73	164	9	30	84	68	81	108	32	36	20	61	165	167	118	40	30	97	97	132	148	65	176	104	161
62	94	128	49	75	162	20	30	72	57	81	97	40	48	9	55	168	172	118	44	26	86	86	128	148	57	176	100	158
63	103	120	51	82	155	28	38	64	48	76	96	49	52	0	59	164	169	113	53	20	77	77	120	142	49	170	92	151
64	119	102	70	101	136	39	56	63	45	56	113	62	46	20	79	144	150	93	69	0	73	73	102	122	36	150	74	132
65	113	110	58	93	145	36	48	58	41	67	101	58	53	10	68	156	162	104	62	12	70	70	110	133	39	161	82	141
66	126	95	76	108	129	46	64	63	44	49	118	69	48	27	86	138	144	86	76	7	71	71	95	115	32	143	67	125
67	144	78	90	125	112	63	81	60	42	34	128	86	60	43	102	126	133	71	94	24	63	63	78	100	21	128	50	108
68	130	94	90	114	124	49	71	78	60	40	134	71	38	40	98	125	129	76	82	21	84	84	94	106	43	134	67	121
69	151	79	112	136	104	70	94	88	71	21	154	91	51	61	120	103	108	54	103	42	90	90	79	85	49	113	53	102
70	135	120	50	110	160	82	76	26	32	102	64	97	107	56	69	192	202	135	92	63	43	43	120	159	55	185	99	155
71	132	106	54	108	145	67	68	25	16	81	81	86	88	39	71	173	182	116	84	43	41	41	106	141	36	168	82	140
72	119	108	51	97	146	50	55	43	28	74	89	70	70	21	66	165	172	110	70	27	56	56	108	138	35	165	82	141
73	152	73	86	131	111	74	87	40	24	47	117	97	79	49	101	138	148	81	101	36	42	42	73	106	0	133	48	107
74	136	89	70	115	127	60	71	40	21	57	105	82	71	33	85	148	157	92	86	25	48	48	89	119	16	147	63	122
75	146	90	70	123	130	77	82	19	0	71	95	97	92	48	88	161	171	104	98	45	29	29	90	128	24	154	67	125
76	159	77	84	136	116	86	94	23	14	64	108	108	97	59	102	151	163	94	110	51	23	23	77	116	19	142	55	111
77	156	96	74	132	136	92	94	0	19	87	89	111	110	64	93	174	186	117	110	63	18	18	96	138	40	163	77	131
78	173	83	92	149	123	106	110	18	29	84	104	126	119	77	111	167	180	110	125	73	0	0	83	128	42	152	68	117
79	176	52	107	155	92	98	111	46	38	48	132	121	99	73	123	129	142	72	126	59	38	38	52	92	24	117	31	87
80	170	59	124	154	84	89	111	87	71	0	162	111	72	76	135	92	100	37	122	56	84	84	59	67	47	95	34	81
81	185	43	134	168	69	103	124	88	75	16	169	126	88	88	146	86	98	29	136	68	83	83	43	55	52	83	22	66
82	178	72	143	165	86	99	123	113	96	26	184	119	73	92	151	72	77	31	132	73	110	110	72	60	73	87	55	85
83	209	77	177	198	73	132	157	142	127	56	217	151	104	127	186	38	45	29	165	107	137	137	77	41	103	61	72	74
84	205	49	161	190	55	125	148	117	104	37	197	146	103	113	172	58	71	0	158	93	110	110	49	30	81	58	43	54
85	222	0	159	202	40	141	158	96	90	59	184	164	130	120	174	95	113	49	171	102	83	83	0	51	73	70	28	35
86	217	16	160	199	37	136	155	102	94	49	189	158	122	118	174	80	98	33	168	99	91	91	16	38	74	60	27	33
87	194	28	133	175	64	113	130	77	67	34	163	136	103	92	148	99	114	43	143	74	68	68	28	61	48	86	0	59
88	196	27	131	176	66	116	132	71	63	43	158	139	109	94	147	108	123	53	146	77	60	60	27	68	46	92	10	61
89	236	51	189	221	33	155	177	138	128	67	223	176	133	142	201	44	64	30	188	122	128	128	51	0	106	28	61	35
90	255	41	198	237	1	173	193	137	131	85	225	196	156	156	212	74	94	56	206	137	124	124	41	34	112	35	65	6

Table 13. (Continued)

	Boston, MA	Phoenix, AZ	Atlanta, GA	Philadelphia, PA	Los Angeles, CA	Chicago, IL	Cincinnati, OH	Houston, TX	Dallas, TX	Denver, CO	Miami, FL	Detroit, MI	Minneapolis, MN	St. Louis, MO	Charlotte, NC	Portland, OR	Seattle, WA	Salt Lake City, UT	Cleveland, OH	Kansas City, MO	San Antonio, TX	Austin, TX	Tucson, AZ	Las Vegas, NV	Oklahoma City, OK	San Francisco, CA	Albuquerque, NM	San Diego, CA
91	254	40	197	236	0	172	192	136	130	84	224	195	155	155	211	73	94	55	204	136	123	123	40	33	111	35	64	6
92	250	35	192	233	6	169	188	131	125	81	219	191	153	151	207	77	97	54	201	132	117	117	35	35	107	40	59	0
93	256	51	204	240	15	175	196	146	138	86	234	197	155	160	217	60	81	53	208	140	134	134	51	25	119	20	71	20
94	263	70	217	249	35	183	206	163	154	95	249	204	160	170	229	44	65	58	216	150	152	152	70	28	133	0	86	40
95	259	70	214	245	37	179	202	161	152	92	247	200	155	167	226	40	60	55	212	147	151	151	70	25	130	5	85	42
96	252	75	211	239	47	173	197	162	151	87	246	193	147	163	222	27	48	50	206	143	153	153	75	25	129	18	85	51
97	246	95	214	235	73	170	195	174	161	92	253	188	140	164	223	0	21	58	203	144	167	167	95	44	138	44	99	77
98	243	113	219	235	94	171	197	186	171	100	261	187	139	169	227	21	0	71	203	150	180	180	113	64	148	65	114	97
99	224	103	200	215	91	151	177	169	154	83	242	168	119	149	207	27	20	57	183	130	164	164	103	58	130	68	99	93

B.2. Full results

Table 14. Opened warehouse and handled volumes for each solution.

Warehouse location	**Capacity (tons)**	**Standard**			**High**			**Low**		
		Fixed cost ($)	***Volume handled in Algorithm solution (tons)***	***Volume handled in CPLEX-MIP solution (tons)***	***Fixed cost ($)***	***Volume handled in Algorithm solution (tons)***	***Volume handled in CPLEX-MIP solution (tons)***	***Fixed cost ($)***	***Volume handled in Algorithm solution (tons)***	***Volume handled in CPLEX-MIP solution (tons)***
Atlanta, GA	15,000	$500,000	8,248	9,896	$750,000	8,248	13,896	$333,333	6,316	9,896
Cincinnati, OH	15,000	$500,000	14,375		$750,000	13,506	15,000	$333,333	14,375	
Cleveland, OH	9,000	$300,000		9,000	$450,000	9,000		$200,000		9,000
Dallas, TX	10,000	$400,000		10,000	$600,000		10,000	$266,666		10,000
Detroit, MI	7,000	$200,000		7,000	$300,000	7,000	7,000	$133,333		7,000
Houston, TX	10,000	$400,000	5,104		$600,000	8,142		$266,666	5,104	
Kansas City, MO	14,000	$600,000	5,604					$400,000	5,535	
Las Vegas, NV	15,000							$466,666	4,182	
Miami, FL	12,000							$533,333	1,932	
Minneapolis, MN	12,000	$500,000	2,565					$333,333	1,964	
Philadelphia, PA	10,000	$500,000	10,000	10,000				$333,333	10,000	10,000
Tucson, AZ	8,000	$300,000	8,000	8,000	$450,000	8,000	8,000	$200,000	4,488	8,000