Large-Scale supply chain optimization via nested partitions

Leyuan Shi
leyuan@ie.engr.wisc.edu
Industrial Engineering Department
University of Wisconsin–Madison

Robert R. Meyer
rrm@cs.wisc.edu
Computer Sciences Department
University of Wisconsin–Madison

Mehmet Bozbay
bozbay@cae.wisc.edu
Industrial Engineering Department
University of Wisconsin–Madison

Winston C. Yang
winston@cs.wisc.edu
Computer Sciences Department
University of Wisconsin–Madison

October 14, 2002

1 Introduction

We consider large-scale supply chain problems with the following properties:

- A set of plants and customers are geographically dispersed in a region.
- Each customer experiences a demand for a variety of products which are manufactured at the plants.
- A given number of warehouses must be located in the distribution network from a list of potential sites.

The goal is to *identify the W best locations (for specified values W* for warehouses that allow satisfaction of each customer's *demand* for each product from

a single warehouse, *minimizing total cost*. This particular format for supply chain optimization problems was motivated by data for a very large real world application furnished to us by Rockwell Automation (Milwaukee, Wisconsin). The actual test problems considered below were generated via suitable approximations to the data distributions provided by Rockwell.

Large-scale problems of this type are intractable when they are input as single problems to general purpose branch-and-bound solvers such as CPLEX. In this research we demonstrate that the nested partition (NP) method is capable of efficiently producing very high quality solutions to large-scale supply chain problems by taking advantage of their structure to generate an appropriate set of restricted subproblems. The implementation of NP that we describe here first employs an optimization-based heuristic to generate an efficiency ranking of the available warehouses and then employs biased sampling (from the ranked warehouse set) guided by the NP global optimization framework. The overall process is implemented using the looping constructs available in the AMPL modeling language and employs the CPLEX mixed-integer programming (MIP) code to solve the optimization subproblems that arise. This implementation represents a novel use of CPLEX as an evaluator of samples generated via a global view that takes advantage of the stucture of the original problem. Our computational results demonstrate that NP also outperforms specialized approaches such as the Lagrange relaxation methods that previously have been regarded as the most effective techniques for large-scale supply chain optimization.

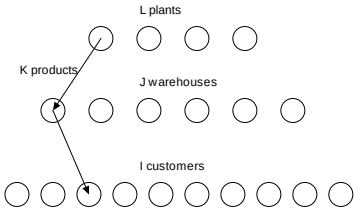

Figure 1: Products go from plants to warehouses to customers.

2 The supply chain model

We formulate the problem as a mixed integer program. The indices and sets are as follows:

- i (I) customer(s)
- j (J) warehouse(s)
- k (K) product(s)
- ℓ (L) plant(s)
- W warehouses to locate

The problem data are as follows:

parameter	description
$c_{\ell jk}$	unit shipping cost from plant ℓ to warehouse j of product k
d_{jik}	unit shipping cost from warehouse j to customer i of product k
f_j	fixed cost of opening and operating a warehouse at site j
w_{ik}	demand of customer i for product k
s_k	volume of unit of product k
q_j	capacity (in volume) of warehouse located at site j
$v_{\ell k}$	supply at plant ℓ of product k

The problem variables are as follows:

decision variables	description
$X_{jik} \in \{0,1\}$	is 1 iff warehouse j supplies customer i with product k
$Y_j \in \{0,1\}$	is 1 iff warehouse is opened at location j
$U_{\ell jk} \geq 0$	shipment from plant ℓ to warehouse j of product k

The overall problem may now be stated:

$$
\left[\begin{array}{lllll}
\min & \displaystyle\sum_{j,k,\ell} c_{\ell jk} U_{\ell jk} & & & \text{plant-warehouse shipping costs} \\
+ & \displaystyle\sum_{i,j,k} d_{jik} w_{ik} X_{jik} & & & \text{warehouse-customer shipping costs} \\
+ & \displaystyle\sum_{j} f_j Y_j & & & \text{warehouse fixed costs} \\
\text{s.t.} & \displaystyle\sum_{j} X_{jik} & = & 1 & \text{single supplier} \\
& \displaystyle\sum_{i,k} s_k w_{ik} X_{jik} & \leq & q_j Y_j & \text{warehouse capacity} \\
& \displaystyle\sum_{i} w_{ik} X_{jik} & = & \displaystyle\sum_{\ell} U_{\ell jk} & \text{conservation at warehouses} \\
& \displaystyle\sum_{j} U_{\ell jk} & \leq & v_{\ell k} & \text{plant capacity} \\
& \displaystyle\sum_{j} Y_j & = & W & \text{fixed number of warehouses} \\
& X_{jik} & \in & \{0,1\} & \\
& Y_j & \in & \{0,1\} & \\
& U_{\ell jk} & \geq & 0 &
\end{array}\right]
$$

3 Problem sets

Using a *factorial design* approach, we set min and max values (see Table **??**) for the 5 design parameters: plants, warehouses, open warehouses, customers, and products. This resulted in $2^5 = 32$ problem sets. After randomly generating the data for each problem, we froze the data so that various solution strategies could be compared for each problem.

	plants	warehouses	open warehouses	customers	products
max	10	30	10	50	3
min	5	100	30	200	10

Table 1: Max and min values of design parameters.

4 Lagrangian relaxation

Many hard integer programming problems can be viewed as easy problems complicated by a relatively small set of *side constraints*. Dualizing the side constraints produces a Lagrangian problem that is easy to solve and whose optimal value is a *lower bound* (for minimization problems) on the optimal value of the original problem. The Lagrangian problem can thus be used in place of a linear programming relaxation to provide bounds in a branch and bound (BB) algorithm (Fisher 1981).The computational experience below demonstrates that a Lagrangian approach specialized to the supply chain problem yields significantly better lower and upper bounds than BB. In particular, we relax single supplier constraints (with multipliers λ_{ik}) and conservation at warehouse constraints (with multipliers θ_{jk}). The resulting problem is

$$\begin{aligned} & \min \left(\sum_{j,k,\ell} c_{\ell jk} U_{\ell jk} + \sum_{i,j,k} d_{jik} w_{ik} X_{jik} + \sum_{j} f_j Y_j \right. \\ + \quad & \left. \sum_{k,\ell} \theta_{jk} \left(\sum_{i} w_{ik} X_{jik} - \sum_{\ell} U_{\ell jk} \right) + \sum_{i,k} \lambda_{ik} \left(1 - \sum_{j} X_{jik} \right) \right), \end{aligned}$$

subject to the following constraints: warehouse capacity (X, Y variables), plant capacity (U variables), fixed number of warehouses (Y variables), and X, Y, binary variables.

Because the relaxation of conservation of flow decouples warehouses from plants, this problem can be decomposed into two separate problems P_1 and P_2. They are the following:

- Let Z_1 be the optimal value of the *plant-warehouse subproblem* P_1:

$$\left[\begin{array}{lll} \min & \displaystyle\sum_{j,k,\ell} (c_{\ell jk} - \theta_{jk}) U_{\ell jk} & \\ & \displaystyle\sum_{j} U_{\ell jk} & \leq \quad v_{\ell k} \\ & U_{\ell jk} & \geq \quad 0 \end{array} \right]$$

- Let Z_2 be the optimal value of the *warehouse-customer subproblem* P_2:

$$\left[\begin{array}{lllll} \min & \sum_{i,j,k}(d_{jik}w_{ik} - \lambda_{ik} + \theta_{jk}w_{ik})X_{jik} & + & \sum_j f_j Y_j \\ \text{s.t.} & \sum_{i,k} s_k w_{ik} X_{jik} & \leq & q_j Y_j \\ & \sum_j Y_j & = & W \\ & X_{jik} & \in & \{0,1\} \\ & Y_j & \in & \{0,1\} \end{array}\right]$$

Let $Z_{\lambda,\theta}$ be the optimal solution to this problem. Then $Z_{\lambda,\theta} = Z_1 + Z_2 + \sum_{i,k} \lambda_{ik}$ is a lower bound to the original problem. To maximize $Z_{\lambda,\theta}$, we use a subgradient algorithm.

5 CPLEX results

For comparison with the Lagrangian approach discussed above and the nested partitions approach to be discussed below, we applied CPLEX to each problem in our test set. Table **??** shows the results of 1- and 2-hour CPLEX runs with default settings. These results (and the results for additional problems of this type that we also considered) show that most of these problems are quite difficult for CPLEX, in the sense that it often produces no feasible solution within two hours.

- NFSTL means CPLEX found no feasible solution within the time limit.
- OP means optimality after the stated number of seconds.
- ** means that the objective value did not improve in hour 2 of the run.
- CPLEX calculates % gap as $100(1 - \text{LB/UB})$, where UB is the value of the best feasible (integer) solution generated .

6 Lagrange/CPLEX results

Table ??: Combining the Lagrangian relaxation process described above with a feasibility process (submission of "best" W warehouses as ranked by the Lagrangian process to CPLEX for completion of best feasible solution using exactly those W warehouses), we obtained results that in many cases reflected significant improvements relative to "pure" CPLEX runs. These are summarized in Table **??**. Note, however, that a number of problems are difficult for both pure CPLEX and our L-CPLEX hybrid approach. For these problems (such as 12, 16, 28, 32), we would like to demonstrate that an NP approach provides a vehicle for obtaining even better feasible solutions and also provides evidence that

prob	plants	warehouses	open	customers	products	% gap (1 hr – 2 hr)
1	5	30	10	50	3	720 OP
2	5	30	10	50	10	16.72 % – 15.05 %**
3	5	30	10	200	3	7.64 % – 5.45%
4	5	30	10	200	10	NFSTL
5	5	30	20	50	3	28 OP
6	5	30	20	50	10	0.12 % – 0.04 %
7	5	30	20	200	3	1200 OP
8	5	30	20	200	10	8.29 %–7.61 %**
9	5	100	10	50	3	6.89 % – 3.52 %
10	5	100	10	50	10	NFSTL – 25.42 %
11	5	100	10	200	3	NFSTL
12	5	100	10	200	10	NFSTL
13	5	100	20	50	3	2900 OP
14	5	100	20	50	10	15.29 % – 15.29 %**
15	5	100	20	200	3	11.69 % – 11.11 %**
16	5	100	20	200	10	NFSTL
17	10	30	10	50	3	1100 OP
18	10	30	10	50	10	22.03 % – 18.47 %**
19	10	30	10	200	3	18.30 % – 16.82 %**
20	10	30	10	200	10	NFSTL
21	10	30	20	50	3	160 OP
22	10	30	20	50	10	3.09 % – 0.09 %
23	10	30	20	200	3	0.75 % – 0.12 %
24	10	30	20	200	10	7.02 % – 6.57 %**
25	10	100	10	50	3	8.27 % – 6.64 %**
26	10	100	10	50	10	NFSTL
27	10	100	10	200	3	NFSTL
28	10	100	10	200	10	NFSTL
29	10	100	20	50	3	6.99 % – 0.21 %
30	10	100	20	50	10	19.64 % – 19.32 %**
31	10	100	20	200	3	11.63 % – 11.35 %**
32	10	100	20	200	10	NFSTL

Table 2: Gaps for 1- and 2-hour CPLEX runs (using default CPLEX settings).

the lower bounds generated by pure CPLEX and by the L-CPLEX hybrid are unrealistic in the sense that the distribution of objective values of randomly generated solutions suggests that those lower bounds are well below any objective value that is likely to obtained.

Table ?? and Table ??: We also tried "starting" CPLEX with the Lagrange LB and UB information, using somewhat different CPLEX strategy options (see Table **??** and Table **??**). CPLEX was not able to produce any significant improvements relative to the best Lagrange solution. An alternative approach worth trying is to consider an NP-based approach using sampling based on the Lagrangian; that is, introducing some randomness in the updating of the multipliers and then using the resulting warehouse set (in conjunction with CPLEX) to construct a feasible solution (see notes below regarding the promising region and its complement).

7 Nested partitions

Insert general NP description here

We use procedures described below to develop rankings for warehouses as starting information and to develop biased sampling approaches for an NP approach that uses AMPL and CPLEX to set up problems for the promising region and complementary region and to evaluate the samples for these regions.

To complete the NP approach, we constructed the *complementary region* by adding instead the complementary constraint (which states that the set of open warehouses contains at most $W-t-1$ of the selected warehouses), and we sampled this region by constructing warehouse sets meeting that complementary constraint. .

Experience with our problem set indicates that the ranking process can generate warehouse sets that lead to good feasible solutions when these warehouses are fixed as "open" in a CPLEX run. In our initial trials of an NP strategy, we specified the promising region by adding the constraint that a subset (as opposed to all) of warehouses from this open set (the subset is designated as OPEN below) must be open:

$$\sum_{j \in \mathrm{OPEN}} Y_j = W - t.$$

In this case, the complementary region is defined by the following constraint:

$$\sum_{j \in \mathrm{OPEN}} Y_j \leq W - t - 1.$$

Our current strategy is

1. Let CPLEX sample both the promising region and complementary region for a fixed amount of time (say, 5 minutes) with its `uppercutoff` parameter set to the objective value of the best Lagrange solution and with a depth-first search strategy.

   The uppercutoff strategy ensures that branch and bound nodes with worse lower bounds worse are closed; they cannot contain a better solution.

   The depth-first search strategy makes CPLEX focus on generating better solutions, as opposed to increasing the lower bound.

2. If CPLEX finds a better solution in either the promising region or complementary region within the time alloted, we define a new fixed set based on that solution (the size of the new fixed set is 1 more than the size of the old fixed set), and we repeat Step 1.

3. If CPLEX does not find a better solution, we construct a new fixed set by selecting an arbitrary element of the open set, adding it to the fixed

set, and repeating Step 1. Note that after a few applications of this of this process, the fixed set equals the open set, and CPLEX solves the subproblem in a few seconds.

4. If neither Step 2 or 3 has produced a better solution, and the size of the fixed set equals the number of required warehouses, then we choose a different subset of the open warehouses to be the fixed set, and we repeat Step 1.

	Lagrange/CPLEX			CPLEX		
prob	LB	objective value	% gap	LB	objective value	% gap
1	351,773	366,541	3.51%	340,180	364,588	0%
2	1,267,118	1,355,928	6.55%	1,015,490	1,355,928	25.11%
3	1,545,949	1,590,121	2.62%	1,314,379	1,587,551	17.21%
4	4,244,436	5,468,330	22.38%	3,959,420	5,468,330	27.59%
5	322,986	337,099	4.09%	322,322	336,749	4.28%
6	1,070,914	1,104,525	2.98%	1,019,990	1,103,766	7.59%
7	1,341,723	1,386,138	3.19%	1,320,117	1,385,964	4.75%
8	3,228,681	4,439,958	27.14%	3,984,895	4,431,639	10.08%
9	296,468	325,675	8.97%	216,196	325,675	33.62%
10	1,083,576	1,276,077	15.09%	738,298	1,276,077	42.14%
11	1,153,796	1,357,120	14.98%	869,286	1,357,120	35.95%
12	2,974,974	5,518,485	45.94%	2,941,679	5,503,354	46.55%
13	243,339	260,485	6.32%	216,349	259,753	16.71%
14	827,694	1,013,161	18.31%	739,594	1,013,161	27%
15	877,432	1,093,835	19.44%	869,260	1,089,151	20.19%
16	1,759,122	4,474,570	59%	2,940,637	4,290,237	31.46%
17	306,293	330,743	5.79%	325,068	325,109	0%
18	1,088,941	1,158,005	5.96%	867,714	1,158,005	25.07%
19	1,295,622	1,380,124	5.94%	1,056,061	1,377,479	23.33%
20	3,854,825	5,013,557	21.15%	3,472,365	4,888,939	28.98%
21	263,175	272,737	3.48%	252,973	272,658	7.22%
22	920,158	951,563	3.21%	878,393	950,696	7.61%
23	1,116,495	1,156,966	3.5%	1,067,217	1,156,966	7.76%
24	3,013,531	3,917,006	23.07%	3,477,645	3,917,006	11.22%
25	292,049	329,536	11.38%	212,751	329,536	35.44%
26	868,815	1,126,558	22.88%	645,700	1,126,558	42.68%
27	1,039,545	1,282,424	18.94%	764,004	1,282,424	40.42%
28	2,219,316	4,739,747	53.18%	2,668,318	4,739,747	43.7%
29	244,325	268,113	8.87%	211,999	268,113	20.93%
30	577,921	864,222	33.13%	644,588	864,222	25.41%
31	752,219	975,645	22.9%	763,491	975,645	21.75%
32	1,234,746	3,646,808	65.88%	2,668,207	3,618,478	26.26%

Table 3: Using 1-hour Lagrange/CPLEX run LB and objective solution in 1-hour CPLEX run with no cuts and with depth-first search.

8 Appendix

These additional problems also demonstrate that our problem generator produces problems that are hard for CPLEX.

cost ratios of the solution be in the ranges given by the 1996 paper "Production, transportation, and distribution planning in a multi-commodity tri-echelon system" by H. Pirkul and V. Jayaraman).

Table **??** shows the results of 1-hour CPLEX runs on problems with various numbers of plants, warehouses, open warehouses, customers, and products. The data was randomly generated by AMPL code different from that in Table **??**.

plants	warehouses	customers	products	open warehouses	% gap
5	30	50	3	10	5.71%
5	30	100	3	10	16.08%
5	30	150	3	10	15.95%
5	30	200	3	10	21.43%
5	30	100	1	10	0%
5	30	100	2	10	9.68%
5	30	100	3	10	16.08%
5	30	100	4	10	18.54%
5	30	100	5	10	22.91%
5	30	100	6	10	30.74%
5	30	100	7	10	31.16%
5	30	100	8	10	33.51%
5	30	100	9	10	16.08%
5	30	100	10	10	35.93%
5	30	10	10	10	0%
5	30	20	10	10	7.00%
5	30	30	10	10	22.20%
5	30	40	10	10	24.68%
5	30	50	10	10	33.81%
10	100	10	10	20	0%
10	100	20	10	20	3.57%
10	100	30	10	20	17.97%
10	100	40	10	20	29.12%
10	100	50	10	20	30.82%
10	100	100	10	20	NFSTL
10	100	200	1	20	4.68%
10	100	200	2	20	25.33%
10	100	200	10	10	NFSTL
10	100	200	10	20	NFSTL
10	100	200	10	30	NFSTL
10	100	200	10	40	NFSTL
10	100	200	10	50	23.62%
10	100	200	10	60	16.53%
10	100	200	10	70	12.76%
10	100	200	10	80	7.55%
10	100	200	10	90	1.47%
10	100	200	10	100	0%

Table 4: 1-hour CPLEX runs on some randomly generated data. Note that % gap is calculated as $100((\text{best objective value})/(\text{linear relaxation value}) - 1)$, which is different from the way CPLEX calculates % gap.