T.C. Dokuz Eylül Üniversitesi İktisadi ve İdari
Bilimler Fakültesi Dergisi, Cilt: 2, Sayı: 1, Yıl: 1987

THREE ALTERNATIVE HEURISTIC SOLUTION PROCEDURES FOR CONCAVE SIMPLE PLANT LOCATION PROBLEM

Nilgün MORALI (*)

SUMMARY

Facilities location problem involves the determination of the number, location, and size of sources that will, most economically, supply the given set of destinations with some commodity, material or service. The most classical examples are the determination of the location and the capacity of warehouses, distribution centers, communication centers or production facilities. If bach possible facility is assumed to be capable of satisfying all of the demand, the problem is called Simple Plant Location Problem (SPLP). In addition, if the concavity of the cost function is also assumed, then it is called Concave Cost Simple Plant Location Problem (CCSPLP).

In this work, the mathematical formulation of CCSPLP is based on a piece-wise linear concave cost function, and it is assumed that there is a possible facility associated with each linear segment. Then three heuristic solution procedures for CCSPLP are developed which are the extensions of heuristics developed for SPLP. These are compared both for the resulting objective values and computing times, on the basis of 44 numerical examples.

(*) Dr., D.E.Ü.İ.İ.B.F., Ekonometri Bölümü

1. INTRODUCTION

Facilities location problem involves with the determination of the number, location, and size of the sources that will, most economically, supply the given set of destinations with some commodity, material, or service. The most classical examples are the determination of the location and the capacity of warehouses, distribution centers, communication centers, or production facilities.

Facilities location problems received a great deal of attention in the past decade, resulting in the development of various models and algorithms. These models and algorithms may be categorized according to various criteria. Some of these critera are as follows:

1— Objective: Cost minimization or return-on-asset
2— Cost function: Linear or concave cost function
3— Potential locations: Anywhere (infinite set; i.e. location on a plane) or particular sites (feasible set; i.e. location on a network)
4— Solution procedure: Optimizing or heuristic (approximate)
5— Planning horizon: One period or multi-period
6— Present sites: To be included or not
7— Capacities: To be included or not

These also represent the basic choices in a location study that face the analyst.

This work handles the facilities location problem as follows: Given a number of demand areas for a certain product, each with a demand b_i and a number alternative sites Where facilities may be built to satisty these demands, it is to be determined where the facilities should be placed and which demand areas are to be served by a given facility. The objective is that the sum of the transportation cost and the amortized facility cost is minimized. This problem is called the Simple Plant Location Problem (SPLP) if no capacity constraints exist; i.e. one facility can satisfy all of the demand. If the assumption of the concavity of the cost function is also added, then it is called the Concave Cost Simple Plant Location Problem (CCSPLP).

2. THE GENERAL MODEL

In the mathematical formulation of CCSPLP the cost function is approximated by a piece-wise linear concave function, and it is assumed that there is a facility associated with each linear segment, the fixed cost

being determined by extending the segment until it intersects the cost axis (Figure 1). For example facility j is associated with three facilities, namely (j,1), (j,2), (j,3), since there are three segments in the cost function shown on Figure 1.

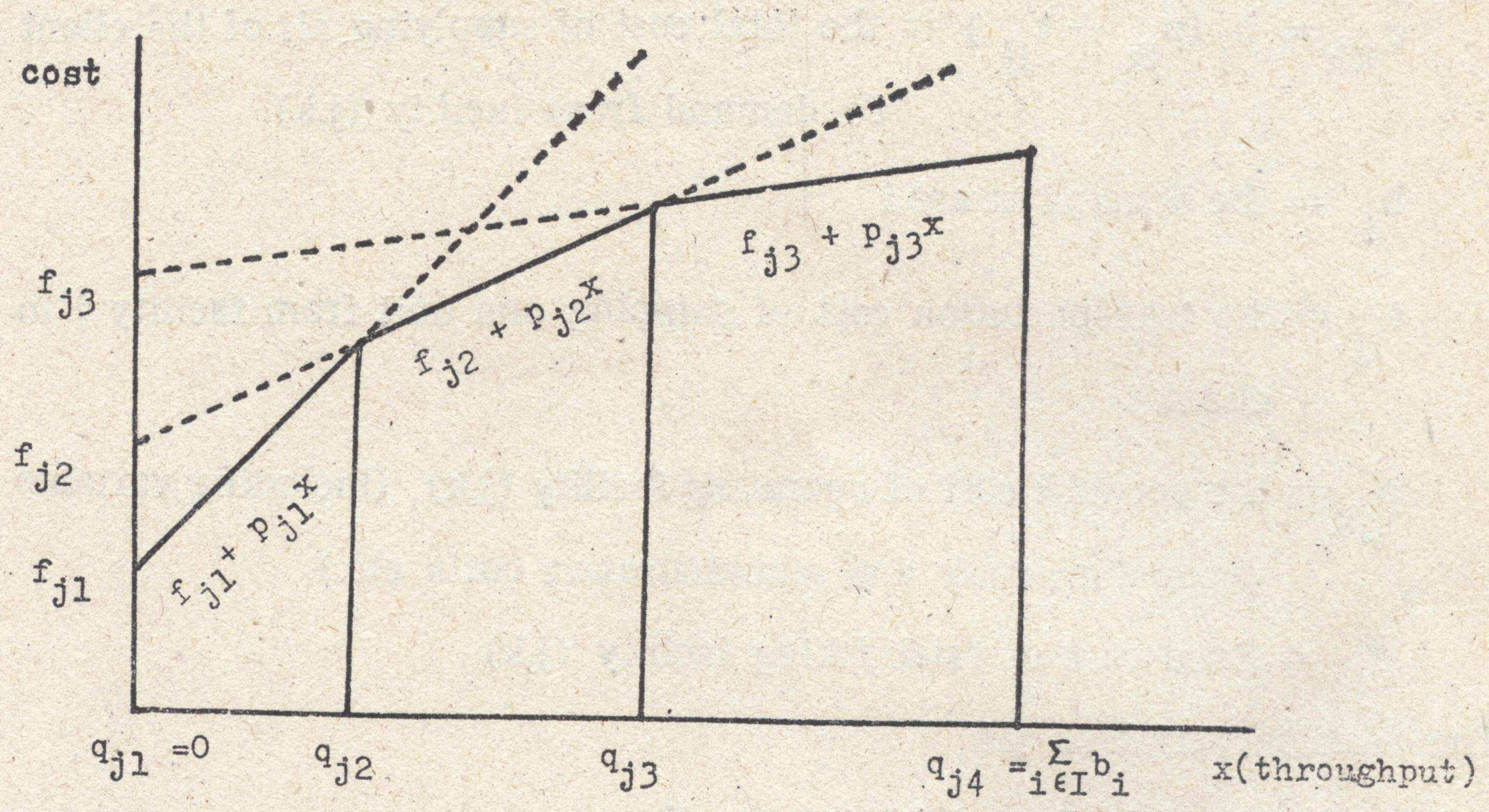

FIGURE 1. The extended production (or warehousing) cost curve for facility j with three segments

With the above assumption, the mathematical formulation of CCSPLP may be stated as:

$$\text{minimize} = \sum_{i\varepsilon I} \sum_{j\varepsilon J} \sum_{k\varepsilon K_j} c_{ijk} X_{ijk} + \sum_{j\varepsilon J} \sum_{k\varepsilon K_j} f_{jk} y_{jk} \qquad (1)$$

subject to

$$\sum_{j\varepsilon J} \sum_{k\varepsilon K_j} X_{ijk} = 1 \qquad i\varepsilon I \qquad (2)$$

$$0 \leq x_{ijk} \leq y_{jk} \leq 1 \qquad i\varepsilon I,\ j\varepsilon J,\ k\varepsilon K_j \qquad (3)$$

$$\sum_{k\varepsilon K_j} y_{jk} \leq 1 \qquad j\varepsilon J \qquad (4)$$

$$y_{jk} \varepsilon \{0\ 1\} \qquad j\varepsilon J,\ k\varepsilon K_j \qquad (5)$$

$$x_{ijk} \geq 0 \qquad i\varepsilon I,\ j\varepsilon J,\ k\varepsilon K_j \qquad (6)$$

where

$I = \{1,2,\ldots,m\}$ = the set of indices of demand areas

$J = \{1\,2\,\ldots,n\}$ = the set of indices of proposed facility sites

$K_j = \{1;2,\ldots,k_j\}$ = the set of indices of the linear segments in the cost function for facility j

$c_{ijk} = b_i (p_{jk} + t_{ij})$ = the total cost of supplying all of the client i's demand from facility (j,k)

b_i = the demand at are i

t_{ij} = the transportation cost of shipping one unit from facility j to client i

p_{jk} = the per unit cost of operating facility (j,k), (including variable production costs and administrative costs etc.)

f_{jk} = fixed cost of establishing facility (j,k)

The decision variables are

$$y_{jk} = \begin{cases} 1 & \text{if facility (j,k) is opened} \\ 0 & \text{otherwise} \end{cases}$$

x_{ijk} = the fraction of client i's demand to be supplied from facility (j,k)

3. SOLUTION PROCEDURES

Three heuristic programs are developed for solving this particular class of problems, CCSPLP, which are extensions of heuristics developed for SPLP. The reason why it is preferred to work with heuristics instead of exact solution procedures is that both SPLP and COSPLP are integer programming problems and the efficiency of exact solution procedures decrease rapidly as the problem size increases. A more mundane reason for using heuristic methods is that in solving location problems of this type, management is rarely interested in the single optimal solution. Management is rather interested in several «good» alternative solutions with an assurance that they are not very far from the optimum. Moreover, as the input data to such problems can never be precise, sensitivity analysis becomes imperative, which requires that the problem be solved repeatedly. It is therefore necessary that a solution method be computationally efficient both in terms of computation time and computer storage.

The first one of the heuristics proposed is an extension Cornuejols, Fisher, and Nemhauser (1977) greedy heuristic (CFN greedy heuristic), the second one is the drop heuristic, and the third one is an extension of Kuehn and Hamburger (1963) add heuristic (KH add heuristic).

3.1. Extension of CFN Greedy Heuristic

The greedy heuristic determines to open the facilities one after the other as long as it reduces the total cost by making use of the Langrangian dual variables.

The Langrangian relaxation for CCSPLP, based on the m equations given in (2) which are weighted by multipliers u_i and inserted into the objective function, may be expressed by

$$L(x,y,u) = \sum_{i\varepsilon I} \sum_{j\varepsilon J} \sum_{k\varepsilon K_j} c_{ijk} x_{ijk} + \sum_{j\varepsilon J} \sum_{k\varepsilon K_j} f_{jk} y_{jk} - \sum_{i\varepsilon I} u_i \left(\sum_{j\varepsilon J} \sum_{k\varepsilon K_j} x_{ijk} - 1 \right)$$

Then the problem becomes

$$\min z_L = \sum_{j\varepsilon J} \sum_{k\varepsilon K_j} \left[\sum_{i\varepsilon I} (c_{ijk} - u_i) x_{ijk} - f_{jk} y_{jk} \right] + \sum_{i\varepsilon I} u_i \qquad (7)$$

subject to the constraints given in (3), (4), (5), (6).

Now, the greedy heuristic algorithm may be stated formally as follows :

Step 1 : Let $s=1$ $J^*=\varnothing$ $J^*=\{ (j,k) ; j\varepsilon J, k\varepsilon K_j \}$

and $u_i^1 = \max_{(j,k)\varepsilon J^*} c_{ijk}$ $i\varepsilon I$

Step 2 : Let $\rho_{jk}(u^s) = \sum_{i\varepsilon I} \max(0, u_i^s - c_{ijk}) - f_{jk}$

for $(j,k)\varepsilon J^*$. Find $(j_s, k_s^*)\varepsilon J^*$ such that

$$\rho_s = \rho_{j_s k_s^*}(u^s) = \max_{(j,k)\varepsilon J^*} \rho_{jk}(u^s)$$

$|J^*| =$

If $\rho_s < o$ and $|J^*| \geqslant o$, set $s=s-1$ and go to Step4. Otherwise set

$$J^* = J^* \cup \{ (j_s, k^*_s) \} \quad \text{and} \quad J^* = J^* - \{ (j_s; k_s) \; ; \; k_s \in K_{j_s} \}$$

If $|J^*| = n$ go to Step 4, otherwise go to Step 3.

Step 3 : Set $s = s+1$ For $i \in I$, set

$$u_i^s = \min_{(j,k) \in J^*} c_{ijk} = u_i^{s-1} - \max (0, u_i^{s-1} - c_{ij_{s-1}k^*_{s-1}})$$

Go to Step 2.

Step 4 : The greedy solution is given by

$$y_{jk} = \begin{cases} 1 & \text{if } (j,k) \in J^* \\ 0 & \text{otherwise} \end{cases}$$

$$x_{ijk} = \begin{cases} y_{jk} & \text{if } c_{ijk} - u_i^s \geqslant 0 \\ 0 & \text{otherwise} \end{cases}$$

$|J^*| = s$ and $z_g = \sum_{i \in I} u_i^l - \sum_{j=1}^{s} \rho_j$

Here it should be noted that, although the set of constraints $\sum_{k \in K_j} y_{jk} \leqslant 1 \quad j \in J$ is not considered in the algorithm, the solution satisfies that condition. (See Morali (1981) for proof)

3.2. Drop Heuristic

Although the idea is got from the drop heuristic of Feldman, Lehrer, and Ray (1966), the algorithm proposed here can not be supposed to be an extension of theirs. In the beginning it is assumed that all the facilities associated with each segment of the cost function are open: i.e., the algorithm is initialized with $\sum_{j \in J} k_j$ open facilities. Then the uneconomical facilities are dropped one by one as long as the total cost is reduced. At each iteration for the choice of closing a facility the smallest delta rule of Khumawala (1972) is utilized.

In the general step, some of the facilities are determined not to be

opened and the others are undetermined. λ_{ijk} is defined to measure the minimum cost savings for the customer i that can be made if facility (j,k) is opened when considered over all undetermined facilities at that iteration. The sum of such minimum savings for facility (j,k) over all customers minus the fixed establishment cost for facility (j,k) is defined by delta Δ_{jk}

If the obtained delta value corresponding to a facility is positive, it is decided to keep that facility opened in the terminal solution. Then the minimum delta among the negative ones is selected to determine the one to close at that iteration. When the index set of facilities corresponding to the undetermined ones is empty, it is stopped.

The drop heuristic algorithm may be formalized as follows :

Step 1 : Set $J0 = \emptyset$ $JK = \{ (j,k) ; j \varepsilon J , k \varepsilon K_j \}$

Step 2 : Compute for all $(j,k) \varepsilon JK$

$$\lambda_{ijk} = \min_{(s,t) \varepsilon JK - \{(j,k)\}} \{ \max(0, c_{ist} - c_{ijk}) \}$$

$$\Delta_{jk} = \sum_{i \varepsilon I} \lambda_{ijk} - f_{jk}$$

Step 3 : Set $JK = JK - \{ (j,k) ; \Delta_{jk} > 0 \}$

$$JO = JO \cup \{ (j,1),(j,2),\ldots,(j,k-1),(j,k+1),\ldots,(j,k_j) ; \Delta_{jk} > 0 \}$$

If $JK = \emptyset$ go to Step 4: Otherwise choose $\Delta_{st} = \min_{(j,k) \varepsilon JK} \Delta_{jk}$

Set $JO = JO \cup \{ (s,t) \}$. Go to Step 2.

Step 4 : The solution is

$$y_{jk} = \begin{cases} 1 & \text{if } (j,k) \notin JO \\ 0 & \text{otherwise} \end{cases}$$

$$x_{ijk} = \begin{cases} 1 & \text{if } y_{jk} = 1 \text{ and } c_{ijk} = \min_{(s,t) \notin JO} c_{ist} \\ 0 & \text{otherwise} \end{cases}$$

$$z_d = \sum_{(j,k) \notin JO} \left(\sum_{i \varepsilon I} c_{ijk} x_{ijk} + f_{jk} \right)$$

3.3. Extension of Kuehn-Hamburger Add Heuristic

As it is done in the algorithm developed by Kuehn and Hamburger [4] for SPLP, initialization is made with all facilities closed and then it is decided to open one after the other depending on the greatest cost savings until no additional facility can be opened without increasing the total cost. At each iteration the largest omega rule of Khumawala [3] is used for the choice of opening a facility.

In the general step, some of the facilities are determined to be opened and some are not to be opened, and the others are undetermined. Similar to the Δ_{ijk} in the delta rule, w_{ijk} is defined to measure the

minimum savings for supplying customer i that can be made if facility (j,k) is opened, when considered over all the fixed open facilities at that iteration. The sum of such savings for facility (j,k) over all customers minus the fixed establihment cost for facility (j,k) is defined by omega, Ω_{jk}

If the obtained omega value corresponding to facility is negative, then such a facility is decided to be kept closed and eliminated from further consideration. Then the maximum omega among the positive ones is selected to determine the facility to be opened at that iteration. When the index set of facilities corresponding to the undetermined ones is empty, it is stopped.

The algorithm may be formally stated as follows:

Step 1 : Set $J1 = JK = \{ (j.k) ; j \varepsilon J ; k \varepsilon K_j \}$

Step 2 : Compute for all $(j,k) \varepsilon JK$

$$\propto_{jk} = f_{jk} + \sum_{i \varepsilon I} c_{ijk}$$

Choose $\min_{(j,k) \varepsilon JK} \propto_{jk} = \propto_{st}$ and set

$J1 = J1 \cup \{ (s,t) \}$ $JK = JK - \{ (s;k) \cdot k \varepsilon K_s \}$

Step 3 : Compute for all $(j,k) \varepsilon JK$

$$w_{ijk} = \min_{(s,\,t)\,\varepsilon Jl} \max\,(O,\; c_{ist} - c_{ijk}) \qquad i\varepsilon I$$

$$\Omega_{jk} = \sum_{i\varepsilon I} w_{ijk} - f_{jk}$$

Step 4 : Set $JK = JK - \{\,(j,k) : \Omega_{jk} < O\,\}$

If $JK = \varnothing$ go to Step 5. Otherwise choose

$$\Omega_{st} = \max_{(j,k)\,\varepsilon JK} \Omega_{jk}$$

Set $Jl = Jl \cup \{\,(s,\,t)\,\}$ $\qquad JK = JK - \{\,(s,k)\,;\, k\varepsilon K_s\,\}$

Go to Step 3.

Step 5 : The terminal solution is

$$Y_{jk} = \begin{cases} 1 & (j,k)\varepsilon Jl \\ o & \text{otherwise} \end{cases} \qquad x_{ijk} = \begin{cases} 1 & \text{if } y_{jk}=1 \text{ and } c_{ijk} = \min\limits_{(s,t)\,\varepsilon Jl} c_{ist} \\ o & \text{otherwise} \end{cases}$$

4. COMPUTATIONAL RESULTS AND CONCLUSION

The three algorithms given above are programmed in FORTRAN IV and run on an IBM 370/138. The computational experiments made on the test problems may be classified in three groups:

(1) The transportation cost table and the demand figures are kept fixed; assuming that the warehousing cost functions for every possible location are the same; nineteen different cost functions are tested with $m=50$, $n=24$.

(2) The transportation cost table and the demand figures are kept fixed, assuming that the warehousing cost functions for some locations are different from the others; seventeen different combinations of cost functions are tested with $m=50$, $n=25$

(3) The warehousing cost functions for every possible location are assumed to be equal and they are fixed; changing the number of demand centers and the number of possible facility sites, and relatedly, the transportation cost tables and demand figures, eight problems are solved.

TABLE 1

Heuristic Objective Values for The Test Problems in Group 1

Segments in the cost func.	Extension of CFN greedy heuristic	Drop heuiristic	Extension of KH add heuristic
1	933 757.1	933 757.1	1 100 888.1
	884 572.1	881 662.6	998 352.2
	875 359.9	686 621.0	954 584.2
	851 660.1	852 091.8	903 624.2
2	899 993.0	991 610.6	991 468.2
	865 664.6	883 218.0	925 816.2
	959 020.8	977 448.6	999 584.2
	968 550.8	974 926.8	999 447.2
3	743 575.7	742 173.0	751 380.7
	801 838.1	799 543.3.	892 816.2
	864 672.1	868 959.0	919 816.2
	924 636.8	955 248.1	955 816.1
	801 893.1	906 957.3	939 497.7
4	736 886.6	735 713.5	736 896.4
	833 721.4	839 063.6	913 179.2
	890 598.0	906 538.8	941 179.2
	901 598.0	947 335.5	950 179.1
	914 474.2	922 905.5	950 179.1
	872 940.5	874 313.8	909 262.5

TABLE 2

Heuristic Computation Times for Test Problems in Group 1

Segments in the cost func.	Ext. of CFN heuristic CPU time	iterations		Drop heuristic CPU time	iterations	Ext. of KH heuristic CPU time	iterations
1	23.76	16		175.61	9	42.01	3
	22.04	12		198.17	14	32.23	3
	20.52	8		202.55	17	36.62	3
	19.44	6		218.26	18	34.28	3
2	31.60	10		422.64	35	84.03	3
	27.52	7		436.72	40	75.39	3
	26.53	6		450.67	42	75.63	3
	26.32	6		451.22	43	75.42	3
3	47.88	13	1	162.37	58	294.52	10
	40.14	9	1	204.45	63	143.63	3
	34.22	6	1	258.72	66	137.51	3
	34.35	6	1	291.01	65	133.23	3
	34.23	6	1	308.62	67	140.60	3
4	66.24	14	2	488.86	82	425.10	14
	49.24	9	2	327.49	85	209.39	3
	44.80	7	2	446.19	89	210.13	3
	44.18	7	2	318.70	84	210.49	3
	41.99	6	2	542.63	90	208.62	3
	43.49	6	2	673.81	91	212.20	3

TABLE 3

Heuristic Objective Values for The Test Problems in Group 2

Number of Different cost funcs.	Segments in the cost funcs	Extension of CFN greedy heuristic	Drop heuristic	Extension of KH add heuristic
2	2,3	890 567.8	895 785.5	1 107 692.0
		927 489.3	1 110 493.0	1 117 559.0
	2,4	869 970.8	872 462.8	1 075 617.0
		869 628.85	875 292.5	1 075 617.0
	3,3	772 730.8	775 851.7	1 065 560.0
	3,4	883 867.5	875 279.3	1 075 617.0
	4,4	861 060.05	861 704.5	1 073 187.0
		877 206.5	884 294.9	1 075 616.0
3	2,3,4	897 435.75	897 637.8	1 082 187.0
4	2,3,3,4	903 489.3	1 097 149.0	934 179.2
	2,3,4,4	793 675.8	825 212.9	889 179.2
		855 440.4	954 646.6	910 179.2
	3,3,3,4	768 422.6	764 378.5	773 696.0
6	2,3,3,3,4,4	856 106.6	823 911.4	1 061 187.0
	2,3,3,4,4,4	848 606.5	850 234.4	1 083 274.0
	2,3,4,4,4,4	910 693.3	933 989.9	1 115 559.0
8	2,2,3,3,3,3,3,4	849 838.3	739 340.9	1 066 187.0

TABLE 4

Heuristic Computation Times for The Test Problems in Group 2.

Number of Different cost func.	Segments in the cost functions	Ext. of CFN heuristic CPU time	Iterations	Drop heuristic CPU time	Drop heuristic Iterations	Ext. of KH heuristic CPU time	Ext. of KH heuristic Iterations
2	2,3,	32.50	7	786.42	53	134.16	2
		29.40	5	703.81	40	138.21	2
	2,4	37.26	7	1 298.26	65	174.79	2
		35.59	6	1 314.09	66	171.06	2
	3,3	43,76	11	1 322.56	60	157.39	2
	3,4	40.29	6	1 916.37	79	202.55	2
	4,4	45.20	7	2 322.08	189	362.51	2
		44.47	6	2 536.72	91	381.26	2
3	2,3,4,	34.12	6	1 258.07	67	143.78	2
4	2,3,3,4	31.93	5	1 381.60	51	124.48	3
	2,3,4,4	49.87	13	1 596.76	57	157.71	3
		38.22	7	1 638.47	58	156.66	3
	3,3,3,4	46.40.	11	1 903.61	66	275.43	2
6	2,3,3,3,4,4	46.65	11	2 454.02	66	234.12	2
	2,3,3,4,4,4	43.79	9	2 673.51	73	255.61	2
	2,3,4,4,4,4	48.63	10	2 303.45	67	276.23	2
8	2,2,3,3,3,3,3,4	39.04	9	1 820.16	36	196.96	2

TABLE 5

Heuristic Objective Values for The Test Problems in Group 3

Possible facility sites (n)	Demand centers (m)	Extension of CFN greedy heuristic	Drop heuristic	Extension of KH add heuristic
10	20	108 658 855.1	108 533 900.	108 533 900
	50	282 770 201.2	282 580 700.	282 765 500.
	70	404 298 688.6	404 066 300.	404 280 800.
	100	608 814 300.0	608 525 500	608 799 900
20	20	67 255 677.2	67 172 280.	67 254 750.
	50	179 401 110.0	179 143 200.	179 395 600.
	70	257 122 724.6	256 803 000.	257 112 900.
	100	378 157 443.8	377 749 500.	378 149 100.

TABLE 6

Heuristic Computation Times for The Test Problems in Group 3

Possible facilitiy sites (n)	Demand centers (m)	Ext. of CFN heuristic		Drop heuristic		Ext. of KH heuristic	
		CPU time	itera-tions	CPU time	itera-tions	CPU time	itera-tions
10	20	3.93	10	15.97	10	13.12	10
	50	10.18.	10	37.09	10	31.21	10
	70	13.50	10	35.04	10	49.78	10
	100	18.89	10	35.29	10	70.52	10
20	20	8.63	11	85.11	20	36.01	11
	50	25.11	19	169.00	22	169.98	19
	70	34.25	20	242.34	25	254.07	20
	100	47.90	20	314.57	27	417.89	20

In all these experiments, the result could not be compared with exact optimum solutions, because obtaining exact solutions were almost imposible with the available techniques. In the first two groups of experiments, the transportation cost table which Kuehn and Hamburger (1963) have used in their original work is employed, since their test problems have been commonly used by several researchers (see Sa' (1969), Khumawala (1972), Morali (1981)). In the last group of experiments, the necessary data is generated by. random number generation from uniform distribution.

The result obtained from the computational experiments are shown Tables 1 to 6. The observations from these tables may be stated as follows:

1) The objective values obtained by the extension of CFN greedy heuristic and the drop heuristic are very close to each other one being less than the other or vice verse, but 30 in 44 problems the former gives smaller objective values than the latter (see tables 1,3,5)

2) The objective values obtained by the extension of the KH add heuristic is always the greatest (see tables 1,3,5)

3) There is always a certain order in CPU times; drop heuristic has the longest, the extension of CFN greedy heuristic has the smallest and the extension of KH add heuristic has in between the others for m=50, n=24 .

4) Since the drop heuristic assumes that all the facilities are open in the beginning and drop one by one until it reaches the solution, the number of iterations and relatedly the CPU time is an increasing function of the number of possible facility sites (see table 6).

5) When the number of iterations are aqual for all three heuristics, extension of CFN greedy heuristic again has the smallest CPU time, but the order changes for the others (see table 6).

6) When the warehousing cost functions and the number of possible facility sites are fixed; CPU time is an increasing function of the number of demand centers for all three heuristics (see table 6).

7) The CPU time for all three heuristics is an increasing function of the number of possible facility sites when cost functons and the number of demand centers are kept fixed (see table 6).

REFERENCES

Cornuejols, G., Fisher, M., Nemhauser, G.L. (1977): «Location of Bank Acounts to Optimize Float: An Analytic Study of Exact and Approximate Algorithms», **Management Science,** V. 23, No. 8, pp. 789-810.

Feldman, E., Lehrer, F.A., Ray, T.L. (1966): «Warehouse Location Under Continuous Economies of Scale», **Management Science,** V.12, No. 9, pp. 670-684.

Khumawala, B.M. (1972); «An Efficient Branch and Bound Algorithm For the Warehouse Location Problem», **Management Science,** V. 18, No. 12, pp. B718-B731.

Kuehn, A.A., Hamburger, M.J. (1963); «A Heuristic Program for Locating Warehouses», **Management Science,** V.9, No. 9, pp. 643-666.

Moralı, N. (1981); «**Simple Plant Location Problem with Economies of Scale**», **Master** Thesis, Dep. of Operational Research and Statistics, Middle East Technical University.

Sa', G. (1969); «Branch-and-Bound and Approximate Solutions to the Capacitated Plant-Location Problem», **Operations Research,** V. 17, No. 6, pp. 1005-1016.

KONKAV MALİYETLİ YALIN KURULUŞ YERİ PROBLEMİ İÇİN ÜÇ ALTERNATİF YAKLAŞIK ÇÖZÜM YÖNTEMİ

Kuruluş yeri seçimi problemi, verilen bir hedefler setinin mal veya hizmet taleplerini en ekonomik biçimde karşılamak üzere, kaynakların sayı, yer ve büyüklüklerinin belirlenmesi ile ilgilidir. En klasik örnekler toptancıların, dağıtım merkezlerinin veya üretim merkezlerinin kuruluş yerlerinin ve kapasitelerinin belirlenmesidir. Eğer her olabilir kuruluş için bütün talebi karşılayabileceği varsayımı yapılırsa, problem Yalın Kuruluş Yeri Problemi (YKYP) olarak adlandırılmaktadır. Ayrıca, maliyet fonksiyonunun konkavlığı varsayımı eklenirse, probleme Konkav Maliyetli Yalın Kuruluş Yeri Problemi (KMYKYP) adı verilmektedir.

Bu çalışmada, KMYKYP'nin matematiksel formülasyonu parçalı doğrusal konkav maliyet fonksiyonuna dayandırılmıştır ve her doğrusal parçaya bir kuruluşun karşılık geldiği varsayılmıştır. Daha sonra YKYP için geliştirilmiş yaklaşık çözüm algoritmaları genişletilerek KMYKYP için üç yaklaşık çözüm algoritması geliştirilmiştir. Bunlar 44 sayısal örnek üzerinde denenerek amaç fonksiyonlarının aldığı değerler ve hesaplama süresine bağlı olarak karşılaştırılmıştır.