Using Genetic Algorithm and Tabu Search in DEA

Gabriel Villa[1], Sebastián Lozano[1] and Fernando Guerrero[1]
[1] Escuela Superior de Ingenieros. Universidad de Sevilla. Camino de los Descubrimientos s.n., 41092 – Sevilla, Spain
gvilla@esi.us.es

Abstract. Data Envelopment Analysis (DEA) is a well-known methodology for the measurement of the relative efficiency of comparable processing units. Generally, DEA models are based on linear programming . Although there is a family of DEA models known as Free Disposal Hull (FDH) which uses binary variable, in the conventional approach those models are decomposable and easily solved. However, when a centralized approach is used, as in this paper, the problem develops a combinatorial structure which makes appropriate the use of metaheuristic approaches such as genetic algorithms and tabu search. For benchmarking, although its computational inefficiency precludes its use in large scale problems, an optimal branch and bound approach will also be presented. The results show that the metaheuristics (especially, the genetic algorithm) can get very good results with very low computing requirements.

1 Introduction

Data Envelopment Analysis, known by its initials DEA, is a mathematical tool based on linear programming (LP) to assess the relative efficiency of different processing units using as input data just the amounts of each resource that are consumed by each unit (inputs), and the amount of each of the products that it generates (outputs). Because of this, units must be processes or organizations that produce in a similar way (units must use the same resources and output the same products). The processing units being assessed are usually called Decision Making Units (DMU) in recognition of the assumption that each unit has freedom to decide its level of inputs and outputs. DEA measures the units' efficiency through the comparison of a unit with any linear combinations of the existing units.

Two main features must be defined in this kind of problems: technology and orientation. Determining the technology consists in defining the feasible region where the units can operate. There are three most common alternatives: Constant Return to Scale (CRS), Variable Return to Scale (VRS) and Free Disposal Hull (FDH) technologies. CRS technology considers as feasible any linear combination of the observed DMUs while in VRS technology only convex linear combinations are feasible. On the other hand, FDH technology considers as feasible the following set:

$$T_{FDH} = \left\{ (\bar{x}, \bar{y}) : \exists \vec{\lambda} \geq 0 : \vec{\lambda} X \leq \bar{x}, \vec{\lambda} Y \geq \bar{y}, \text{con} \quad \bar{e}^t \cdot \vec{\lambda} = 1, \quad \lambda_j \in \{0,1\} \right\} \qquad \textbf{(1)}$$

where X and Y are the matrices of inputs and outputs of the given units.

Defining the problem orientation consists in determining what search direction will be used to project processing units on efficiency ones. There are two orientations: input and output orientation. The first one consists in reducing as much as possible all inputs equi-proportionally without reducing outputs. Therefore, units will be projected on another efficient DMU whose resources consumption is lower. On the other hand, in the output orientation, an equi-proportional increase of products is realized in the search of an efficient unit with bigger level of outputs and with not greater resource consumption. Once technology and orientation have been chosen, the resulting DEA model can be formulated. Thus, for CRS technology the models are known as CCR-I (for input orientation) and CCR-O (for output orientation) while for VRS technology there are two types of models BCC-I/BCC-O and the additive model. All these models can be found in any book on DEA, for example Cooper et al (2000).

In this paper we are concerned with the FDH technology. The traditional approach leads to FDH-I and FDH-O models which are decomposable and can be efficiently solved through an specialized algorithm. However, the model proposed in this paper differs from the conventional FDH models in the fact that the DMUs are not projected on the efficient frontier in an independent way, but in an aggregated way. Therefore the aim will not be the individual improvement of every DMU but the improve of the whole set. A centralized resource allocation approach involves taking into account total input consumption and total output production. This view makes sense whenever the different units belong to the same organization (for example, branches of a bank) which supplies the inputs to them all and consequently appropriates of their outputs. Such an organization has a global objective even when measuring individual performance.

In the next section a brief review of the literature on centralized resource allocation is presented. In section 3 the proposed FDH model is introduced. In section 4 the different solution methods are presented. In section 5 the computational experiences carried out are described and their results analyzed. Finally, in the last section, a summary and the conclusions are presented.

2 Relevant resource allocation literature

There have been some previous approaches in the literature that handle the DMUs in a joint manner. Golany et al (1993) presented a resource allocation model (with input orientation) based on the DEA methodology. In a later paper, Golany and Tamir (1995) proposed a DEA model for resource reallocation (with output orientation) which includes constraints that impose upper bounds on the total consumption of the units. Athanassopoulos (1995) presents a goal programming DEA model (GoDEA) for centralized resource planning. Athanassopoulos (1999) proposes a new model, this time not based on the envelopment form but on the multiplier form. Likewise, Färe et al (1997) and Beasley (2001) present models where the DMUs are treated in a joint form.

3 Proposed FDH resource allocation models using DEA

In this section a model for resource allocation with FDH technology is presented: centralized FDH-I model. It is a radial model.

3.1 Centralized FDH input orientation model

There are two essential differences which distinguish this model from the conventional DEA:

- Instead of solving a linear programming model for every existing DMU in the problem, all the units are projected simultaneously.
- Instead of reducing the inputs of every DMU, it is the total input consumption of DMUs what is tried to reduce. Besides what is important is maintaining the same level of the total amount of outputs but it is not forbidden that a certain DMU reduces its outputs.

Let:

j,r=1,2,...,n	indexes for DMUs
i=1, 2,...,m	index for inputs
k=1, 2,...,p	index for outputs
x_{ij}	amount of input i consumed by DMU j
y_{kj}	amount of output k produced by DMU j
θ	radial reduction of total input
$(\lambda_{1r}, \lambda_{2r},..., \lambda_{nr})$	vector for projecting DMU r

The model is:

$$\text{Minimise} \quad \theta$$

s.t.

$$\sum_{r=1}^{n}\sum_{j=1}^{n}\lambda_{jr}\, x_{ij} \le \theta \sum_{j=1}^{n} x_{ij} \qquad \forall i$$

$$\sum_{r=1}^{n}\sum_{j=1}^{n}\lambda_{jr}\, y_{kj} \ge \sum_{r=1}^{n} y_{kr} \qquad \forall k \qquad (2)$$

$$\sum_{j=1}^{n}\lambda_{jr} = 1 \qquad \forall r \qquad \lambda_{jr} \in \{0,1\} \quad \theta \text{ free}$$

This is a linear programming problem that contains n^2+1 variables and $m+(p+1)\times n$ constraints. Once the model is resulted, the corresponding vector $(\lambda_{1r}, \lambda_{2r},..., \lambda_{nr})$ defines for each DMU r the operating point at which it should aim. Inputs and outputs of each such point can be computed as

$$\hat{x}_{ir} = \sum_{j=1}^{n} \lambda^{*}_{jr}\, x_{ij} \qquad \forall i \tag{3}$$

$$\hat{y}_{kr} = \sum_{j=1}^{n} \lambda^{*}_{jr}\, y_{kj} \qquad \forall k \tag{4}$$

4 Resolution of the centralized FDH problem

In this section we will consider the different solution methods that can be used for the proposed model.

4.1 Solving the traditional FDH model.

The first approach is to solve the traditional FDH model:

As we mentioned in the introduction, this problem can be solved using a simple algorithm. Let

$$\begin{aligned} &\text{Minimise} \quad \theta_r \\ &\text{s.t.} \\ &\sum_{j=1}^{n} \lambda_j\, x_{ij} \le \theta_r\, x_{ir} \qquad \forall i \\ &\sum_{j=1}^{n} \lambda_{jr}\, y_{kj} \ge y_{kr} \qquad \forall k \\ &\sum_{j=1}^{n} \lambda_j = 1 \quad \lambda_j \in \{0,1\} \quad \theta_r \text{ free} \end{aligned} \tag{6}$$

This is the set of existing DMUs that dominate the DMU r. The optimal solution to the traditional FDH model can be obtained as

$$\theta_r = \min_{j \in D(r)} \left\{ \max_{i=1,\ldots,m} \left\{ \frac{x_{ij}}{x_{ir}} \right\} \right\} \tag{7}$$

For that solution, the radial reduction of total input can be computed as

$$\theta_{FDH} = \min_{i=1,\ldots,m} \left\{ \frac{\sum_{j(r)\in D(r)} x_{ij(r)}}{\sum_{j=1}^{n} x_{ij}} \right\} \quad (8)$$

In this way, the saving in total input consumption achieved by solving the traditional FDH model is obtained.

4.2 Solving centralized FDH model

The centralized FDH-I model performs a projection of each processing unit with an existing DMU belonging to the efficient frontier, in order to find the largest feasible reduction of the total of inputs. Therefore, if one wants to find the solution that realizes the maximum reduction of resource consumption, all the possible projections of the DMUs onto every efficient unit must be considered. This is a combinatorial problem whose difficulty will depend on the number of existing units and on the number of them that are efficient. If there are *'n'* units in the problem of which *'e'* they are efficient ($e \leq n$), the number of possible combinations will be e^n.

Even for problems with a number of DMUs not very large, the number of combinations can be very high, which makes necessary to use some metaheuristic for exploring the solution space. In this paper two of the most successful metaheuristics have been tried: genetic and tabu search algorithms. Also, in order to benchmark this approximate methods, the optimal solution has also been computed through a branch and bound algorithm that is also described below.

Tabu search heuristic. The algorithm starts its exploration with the solution provided by the traditional FDH model. Let E be the set of efficient units set in the problem. This set corresponds to the nondominated units, i.e. those units r whose set $D(r)=\{r\}$. A solution corresponds to projecting each nonefficient unit onto an efficient one. A move will consist in the projection of a DMU *r* over another efficient unit $e' \in E$, different from the current one.

Therefore, the size of neighborhood is $O[n*(|E|-1)]$. However, since total output levels must be maintained, not all the neighbors of a certain solution are feasible. Before evaluating a move it is necessary to checked its feasibility. Once verified its feasibility, the increase in the objective function due to a move is

$$\Delta F.O. = \theta^{after} - \theta^{previous} \quad (9)$$

where:

$$\theta^{after} = \max_{i=1..m} \left\{ \frac{\text{current total consumption} - x_{ie} + x_{ie'}}{\text{initial total consumption}} \right\} \quad (10)$$

As adaptive memory mechanism two tabu lists are proposed. In one of them the unit whose projection has been changed is recorded so that for a number of periods T that unit is prohibited from changing its projection again. The second tabu list records the efficient unit onto which the unit is now projected. For a number of periods T', none of the units projected onto such efficient unit can change its projection. The tabu restriction applied consists in that a move is tabu when it is contained in any of both lists.

Additionally, a diversification strategy has been used consisting in that after a certain number of iterations (parameter REP) in which the objective function has not been improved, the search is reinitialized to a solution where each DMU is projected onto the efficient unit onto which it has been projected more fewer times so far. The tabu search algorithm method ends when, for a certain number of iterations ITER, the best solution found does not improve.

Genetic algorithm heuristic. The codification used consists in a vector with so many components as units in the problem. The value of component *r* is the efficient unit on which it is projected, *e(r)*.

The size of the population is POP and the number of generations GEN. The initial population consists of POP-1 admissible solutions which generated in a random way and the solution of the traditional FDH model, which is a feasible solution of the centralized model.

A steady state (a.k.a. as incremental) genetic algorithm has been implemented so that in each iteration a new individual is generated (either through crossover or mutation) and an old individual is deleted from the population. The uniform crossover operator is suggested. Crossover is performed with probability 1-MUT and parents are chosen randomly. For mutation, an individual is selected randomly and the value of one of its components *r* is modified from e(r) to e´(r)≠e(r). The mutation operator is applied with probability MUT.

In every generation, the population is kept ordered according to the fitness value of the individuals. Such fitness includes a specific term for penalizing the unfeasibility that occurs whenever the total sum of a certain output is lower than the initial one. An infeasible solution can be generated due to a mutation or crossover. The fitness used in this paper is

$$\text{O.F.} = \theta + \text{ALPHA} \times \frac{\sum_j y_{kj}^{initial}}{\min_k \left\{ \sum_j y_{kj}^{actual} \right\}}; \text{ if } \frac{\sum_j y_{kj}^{initial}}{\min_k \left\{ \sum_j y_{kj}^{actual} \right\}} > 1 \qquad \textbf{(11)}$$

Clearly, the fitness value for feasible solutions is the radial reduction of the sum of inputs θ.

The selection of the individual to be deleted from the population is done in a random but biased way according to a geometric probability distribution

$$f(x) = \frac{p_0}{(POP-1)^2}(x-1)^2; \qquad x = 1,2,..., POP \tag{12}$$

where x is the ranking of the individual in the population (x=1 is the best individual and x=POP is the worst), f(x) is the probability that an individual x is selected and p_0 is a constant that is determined by the following condition

$$\sum_{x=1}^{POP} f(x) = 1 \tag{13}$$

Note the elitism in equation (10), i.e. the best individual (x=1) is never selected for deletion.

Branch and bound. An optimal B&B solution method has also been implemented. The solution tree has as many levels as inefficient DMUs exist in the problem and for node *r* there exist so many branches as efficient units on which they can be projected, i.e. |E|. Thus each node can be codified by the DMU index *r* and the index e(r) of the unit onto which it is projected. As initial upper bound, and in order to accelerate the fathoming of nodes, the solution provided by tabu search has been used. A depth-first strategy is proposed. A lower bound associated with a node can be computed as

$$CI\left[e(r)\right] = \max_{i=1..m}\left\{\frac{\sum_{r'\le r} x_{ie(r')} + \sum_{r''>r} \min_{e\in E} x_{ie}}{\sum_{r} x_{ir}}\right\} \tag{14}$$

If this lower bound is larger than the upper bound then the node can be pruned. It is also necessary to verify the feasibility of every node with regards to the total outputs.

5. Computational experience.

In order to benchmark the two heuristic approaches that have been proposed, a total of 50 random problems have been generated. All of them have two inputs and one output. The number of DMUs have been kept low (10 DMUs). Although these can be considered small problems (and the heuristic approaches can solve much larger problems) we have had to use these size in order to be able to solve them optimally with the branch and bound method which is notoriously inefficient. Both the test data and the computer program that generates them are available from the authors upon request.

The parameter values for Tabu search and genetic algorithm that were used are:

ITER=8000; REP=150; T=1; T ' =2
GEN=1000; MUT=0.2; POP=50; ALPHA=2

The results presented correspond to a PC with 500 MHz AMD-K6 processor and are presented in table 2 in the appendix. It can be seen that both Tabu Search and Genetic Algorithms have running times of the order of 10 seconds while B&B can require several hours. Add to this that the relative error of the heuristic methods are rather small (1.71% in average for tabu search and 0.77% for the genetic algorithm).

Figure 1 shows the average relative error as a function of the number of efficient units. The larger this number, the larger the solution space and consequently the more difficult the problem (at least in principle). The relative error for Tabu Search but specially for the Genetic Algorithm is always very low.

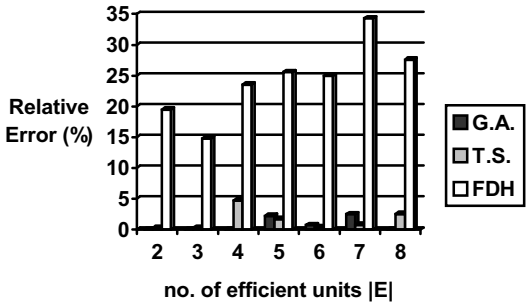

Fig. 1. Average relative error of FDH, genetic algorithm and tabu search.

Four additional test instances with 15 DMUs have been generated and the results of the tabu search, genetic algorithm and branch and bound algorithms are shown below, in table 1. The parameter values for Tabu search and genetic algorithm that were used are:

ITER=10000; REP=150; T=1; T ' =5
GEN=2000; MUT=0.2; POP=50; ALPHA=2

Table 1. Results for 15 DMUs problems

Prob.	|E|	TRADITIONAL FDH		GENETIC ALGORITHM		TABU SEARCH		BRANCH & BOUND	
		Solution	*Time*	*Solution*	*Time*	*Solution*	*Time*	*Solution*	*Time*
1	5	0.5025	0.05s	0.3404	20.22s	0.3404	13.12s	0.3366	8h 29min 24s
2	4	0.4844	0.05s	0.3608	17.38s	0.3792	11.13s	0.3607	1h 46min 32s
3	7	0.6154	0.05s	0.3624	11.32s	0.3462	10.82s	0.3337	3d 22h 20min 8s
4	5	0.5094	0.05s	0.4651	18.56s	0.4651	9.73s	0.4651	17h 39min 1s

The large computing times of the exact method demonstrate that it is not suited for relatively small problems. Note also that, the same as for the smaller problems, the conventional FDH solution is not very good and that both Tabu Search and Genetic Algorithms, which start with that solution, are able to significantly improve upon it.

6. Summary and conclusions

Though conventional DEA models project each DMU in an independent manner, the methodology also allows the consideration of a centralized resource allocation point of view. This paper has presented such a centralized model for resource allocation with FDH technology. The problem has a combinatorial structure and for solving it two different metaheuristics have been implemented. To benchmark their performance, an optimal B&B method has also been developed. Computational experiences show that the optimal B&B is not feasible for large problems and that the metaheuristic, specially the Genetic Algorithm, can give very good solutions with minimal computing requirements.

References

1. Athanassopoulos, A.D.: Decision Support for Target-Based Resource Allocation of Public Services in Multiunit and Multilevel Systems. Management Science, 44, 2 (1998) 173-187
2. Athanassopoulos, A.D.: Goal programming & data envelopment analysis (GoDEA) for target-based multi-level planning: Allocating central grants to the Greek local authorities. European Journal of Operational Research. 87 (1995) 535-550
3. Beasley, J.E. (2001) http://mscmga.ms.ic.ac.uk/jeb/deafixed.html
4. Cooper, W.W., Seiford, L.M. y Tone, K.: Data Envelopment Analysis. A Comprehensive Text with Models, Applications, References and DEA-Solver Software. Kluwer Academic Publishers, Boston, MA (2000)
5. Golany, B., Phillips, F.Y. y Rousseau, J.J.: Models for improved effectiveness based on DEA efficiency results. IIE Transactions. 25, 6 (1993) 2-10
6. Golany, B. y Tamir, E.: Evaluating Efficiency-Effectiveness-Equality Trade-offs: A Data Envelopment Analysis Approach. Management Science. 41, 7 (1995) 1172-1184
7. Färe, R., Grabowski, R., Grosskopf, S. y Kraft, S.: Efficiency of a fixed but allocatable input: A non-parametric approach. Economic Letters. 56 (1997) 187-193

Appendix

Table 2. Results for 10 DMUs problems

Prob.	|E|	TRAD. FDH		GEN. ALG.		TABU S.		BRANCH & BOUND	
		Solution	*Time*	*Solution*	*Time*	*Solution*	*Time*	*Solution*	*Time*
1	4	0.8580	0.01s	0.6561	9.09s	0.6561	7.09s	0.6561	17s
2	3	0.7070	0.06s	0.6567	8.90s	0.6716	1.04s	0.6567	5s
3	4	0.7753	0.01s	0.6598	8.89s	0.6598	4.26s	0.6598	23min 30s
4	4	0.4765	0.01s	0.4199	8.84s	0.4433	4.56s	0.4199	11min 42s
5	6	0.8192	0.01s	0.7269	9.06s	0.7306	4.12s	0.7269	7h 1min 25s
6	4	0.5150	0.05s	0.3728	8.95s	0.3728	1.21s	0.3728	2min 1s

Prob.	|E|	TRAD. FDH		GEN. ALG.		TABU S.		BRANCH & BOUND	
		Solution	*Time*	*Solution*	*Time*	*Solution*	*Time*	*Solution*	*Time*
7	5	0.6968	0.05s	0.4113	9.34s	0.4113	5.11s	0.4113	28s
8	3	0.5218	0.05s	0.4570	8.90s	0.4588	0.88s	0.4570	1min 34s
9	7	0.9158	0.01s	0.6691	9.06s	0.6245	3.35s	0.6245	54min 30s
10	7	0.8673	0.05s	0.7326	8.35s	0.7448	6.09s	0.7269	6h 29min 36s
11	7	0.7750	0.01s	0.3496	8.85s	0.3496	6.15s	0.3496	6min 40s
12	8	0.9194	0.06s	0.7196	9.44s	0.7444	2.86s	0.7196	3h 9min 21s
13	4	0.6381	0.01s	0.4708	8.46s	0.5050	2.15s	0.4708	1min 24s
14	5	0.8948	0.01s	0.7844	9.45s	0.7844	2.25s	0.7844	1h 10min 8s
15	5	0.5462	0.01s	0.2902	9.28s	0.2902	3.68s	0.2902	54s
16	5	0.8168	0.01s	0.6542	10.08s	0.6542	1.54s	0.6542	3h 45min 12s
17	6	0.8583	0.01s	0.6870	9.29s	0.6960	2.31s	0.6870	2h 49min 2s
18	8	0.8103	0.05s	0.5395	9.78s	0.5490	15.34s	0.5395	2h 50min 41s
19	5	0.6649	0.06s	0.5525	9.54s	0.5330	4.17s	0.5330	2h 8min 1s
20	3	0.5647	0.01s	0.5598	9.56s	0.5647	0.66s	0.5598	9s
21	6	0.7687	0.01s	0.6576	8.96s	0.6321	1.92s	0.6321	50min 54s
22	5	0.6700	0.01s	0.5927	9.89s	0.5927	0.87s	0.5927	24min 10s
23	6	0.7566	0.05s	0.6050	9.72s	0.6050	1.75s	0.6050	57min 14s
24	4	0.7754	0.06s	0.6526	10.11s	0.7002	1.54s	0.6526	13min 6s
25	3	0.6894	0.01s	0.6506	9.78s	0.6552	0.70s	0.6506	4s
26	3	0.3261	0.01s	0.2296	9.73s	0.2296	1.26s	0.2296	1s
27	3	0.5406	0.01s	0.3220	9.99s	0.3220	1.26s	0.3220	1s
28	5	0.8321	0.01s	0.7328	10.00s	0.7328	6.27s	0.7328	35min 26s
29	3	0.6281	0.01s	0.5892	9.72s	0.5892	9.77s	0.5892	2s
30	3	0.7129	0.01s	0.6157	9.84s	0.6157	10.98s	0.6157	2min 47s
31	5	0.6754	0.01s	0.5039	9.89s	0.5039	4.50s	0.5039	9min 22s
32	4	0.7710	0.01s	0.5752	10.16s	0.6522	1.59s	0.5752	43s
33	3	0.6194	0.01s	0.5298	9.94s	0.5298	10.82s	0.5298	32s
34	5	0.7179	0.01s	0.6068	10.38s	0.5912	2.42s	0.5912	4h 57min 17s
35	5	0.8108	0.01s	0.6384	10.11s	0.6384	1.21s	0.6384	1h 25min 1s
36	4	0.7002	0.06s	0.6638	10.16s	0.6638	7.09s	0.6638	37min 19s
37	4	0.5304	0.05s	0.3584	10.27s	0.4050	2.09s	0.3584	2min 5s
38	6	0.7484	0.01s	0.4781	11.04s	0.4697	6.76s	0.4697	4s
39	6	0.8224	0.01s	0.6148	10.16s	0.6196	2.14s	0.6148	5h 58min 27s
40	5	0.4816	0.01s	0.2755	10.28s	0.2755	11.20s	0.2244	1s
41	3	0.5057	0.05s	0.4522	10.16s	0.4522	0.98s	0.4522	3min 39s
42	4	0.5952	0.01s	0.3811	10.32s	0.4233	3.13s	0.3811	2s
43	3	0.5493	0.01s	0.4129	10.32s	0.4129	1.81s	0.4129	6min 38s
44	6	0.6536	0.06s	0.3444	10.80s	0.3444	5.87s	0.3444	3h 28min 51s
45	3	0.2703	0.05s	0.2559	10.71s	0.2559	1.54s	0.2559	2min 7s
46	3	0.5747	0.01s	0.4505	10.38s	0.4505	2.31s	0.4505	10s
47	2	0.3925	0.05s	0.2697	10.64s	0.2697	0.88s	0.2697	0.5s
48	4	0.7783	0.05s	0.4703	11.21s	0.4703	11.48s	0.4703	57min 59s
49	2	0.3365	0.01s	0.3107	10.49s	0.3125	1.98s	0.3107	9s
50	6	0.5165	0.01s	0.4103	10.65s	0.4103	0.98s	0.4103	2h 42min 5s