A TOOL FOR HELPING OPERATIONS RESEARCH TEACHING

Edson Luiz Franca Senne
Antonio Fernando Branco Costa
Fernando Augusto Silva Marins

A TOOL FOR HELPING OPERATIONS RESEARCH TEACHING

Edson Luiz Franca Senne[1,2]
e-mail: edson.senne@bruneLac.uk

Antonio Fernando Branco Costa[3]
e-mail: fbranco@grtooo.uesp.ansp.br

Fernando Augusto Silva Marins[1,3]
e-mail: feniando.marins@bruneLac.uk

ABSTRACT

This paper describes an integrated environment in which a student can acquire and test knowledge about Operations Research techniques. In this environment all the interactions with the users are performed by means of an appropriated graphical interface related to the problem and the algorithm chosen to solve it During a step-by-step problem solving session, the user makes the decisions and lets the system process the necessary computations. Moreover, this environment is able: (i) to check the correctness of the user's replies, (ii) to display warnings every time a wrong decision is taken, and (iii) to return to any previous step in order to give more flexibility to the teacher's exposition. The system is designed to include: Primal Simplex, Dual Simplex, Revised Simplex, Stepping-Stone Method for Transportation Problems, Hungarian Method for Assignment Problems, and Sensitivity Analysis.

INTRODUCTION

In the last decade many authors have proposed to change the traditional method of teaching Operations Research (OR) using microcomputers to make it more interesting. Many papers describing experiences of the role of microcomputers in the OR teaching and reviews of educational OR softwares have been published.

Jensen (1983), one of the pioneers in this subject, discusses the advantages and disadvantages of mainframes versus microcomputers in the learning process and comments on the desirable characteristics for educational software packages. He also presents several programs which have been implemented on microcomputers, such as: Regression Demonstration, Markov Chain Demonstration, Economic Evaluation of Projects, Revised Simplex, Two-Dimensional Linear Programming, Pure Network Flow Optimization and Dynamic Programming. Other authors have also presented their experiences in OR teaching. Wheeler (1985) has emphasize the teaching as a combination of lectures, tutorials and suggested readings, where the computer is an important tool for ingenuity and discovery. On the other hand, King (1988) proposes a five-stage approach for teaching mathematical

---

[1]Brunel University, Department of Mathematics and Statistics, Uxbridge, UB8 3PH, England, UK
[2]Universidade Estadual Paulista, Departamento de Matemática, 12500-000 Guaratinguetá, SP, Brazil
[3]Universidade Estadual Paulista, Departamento de Produção, 12500-000 Guaratinguetá, SP, Brazil

modeling. These stages are described as: conceptualization, verbalization, symbolization, manipulation and representation. The author suggests that confining mathematics to the last three stages and reducing the manipulation to a minimum by the use of friendly computer packages, the major emphasis shifts to the first two stages, pointed out as the most important ones. King also presents some areas where this methodology has been applied, such as: Simple Stock Control and Distribution, Financial Modeling and Planning, Linear Programming, Decision Trees, Network Analysis and Discrete-Event Simulation.

A different kind of work is presented by Sharda (1988,1992), Llewellyn and Sharda (1990), Janczyk and Beasly (1988) and Yurkiewicz (1988). These authors have presented in-depth analyses of programs devised for educational use in OR courses and designed to be run on microcomputers. Although some software packages evaluated are designed to show all tables produced during the solution process, there is no conversational features that help the users to learn the techniques involved.

In this work, a new kind of educational OR software is discussed. The basic idea is to use the computer as a teaching aid. In order to achieve the goal that students should improve their understanding of OR algorithms it is necessary to ask for student input in the iterative process. Such student input is then compared with the computations carried out according to the underlying algorithmic procedure. A comparison is made and if the students make mistakes, error messages will be displayed and the students may revise the input. Moreover, in this environment all the interactions with the users are performed in an appropriated graphical interface. At the end of each session a performance report is presented.

THE PROPOSED TOOL

The software environment proposed in this work should include the main OR techniques normally taught at both undergraduate and graduate courses such as Industrial Engineering and Applied and Computational Mathematics (Costa et alii, 1991; Serine et alii, 1993). Each technique has been implemented as an independent module of the system so that new modules can easily be incorporated. Each module is related to an appropriate graphical interface where the problem is defined and the solution process is shown.

In addition to the educational characteristics of the softwares presented by Yurkiewicz (1988), the proposed environment allows a new interactive step-by-step mode for problem solving. These characteristics aim at two educational goals:

- to encourage the students to solve academic problems, allowing them to check their own knowledge about the chosen algorithm. This is possible because during a problem solving session the student should answer questions about important decisions which are formulated by the system. In the case of wrong answers, the student is advised and has a chance to make a new decision. The system will present the correct decision if the limit of errors is exceeded (the limit of errors by step is specified at the beginning of the session). Furthermore, at the end of the session a report will be available containing the student identification, the number of errors by step, the initial data and the optimal solution.

- to support the teaching of each implemented technique. It is easy to visualize and to follow each stage of the algorithm application by means of the graphical interface. It is

2

Important to observe that the user can <u>return to a previous stage</u> in order to revise decisions using the `PgUp` option.

The following modules have been implemented: Primal Simplex, Revised Simplex, Stepping-Stone Method for Transportation Problems, and Hungarian Method for Assignment Problems. In addition, the module is being implemented. The system also intends to include Sensitivity Analysis techniques.

It is presented below, some stages of the Stepping Stone method application in order to ilustrate the user-friendly characteristics of the proposed software environment, for the following problem:

"A company has three Plants (origins) producing a certain product that is shipped to two Distribution Centers (sinks). Plants Ol, 02 and O3 produce 100,50 and 200 units per month, respectively. Distribution Centers SI and S2 need to receive 150 and 100 units per month, repectivefy. The freight costs (Cij)for shipments from Oi to Sj are given by: C11 = 5, C12 = 2, C21 = 4, C31 = 1 and C32 - 4. It is known that any shipment from 02 to S2 is impossible. How much should be shipped from each Plant to each Distribution Center in order to minimize the total shipping cost?"

This problem was solved by **SYMON** - System for **Monitoring the Transportation Model Resolution,** one module of the proposed environment. It is presented below some computer screens displayed with some comments about them. In each screen, the last rows are reserved for the user menu.

Initially, the characteristics of the above problem are defined (see Figure 1). In this screen the user inputs the number of origins (3) and the number of sinks (2), selects the type of the problem (minimization) and especifies the allowable number of errors by stage (1) and then, should press `PgDn` to continue.

SYMON
System for Monitoring the Transportation Model Resolution

Number of origins: [3] (Maximum = 6)

Number of sinks: [2] (Maximum = 5)

Type of the problem: [Minimization] Maximization

Allowable number of user's errors/stage: [1] (Maximum = 5)

↔↑↓ Move	**ENTER** Next cell	**F9** Clear Cell
`PgUp` Return		`PgDn` Advance

Fig. 1 - Problem Characterization

Figure 2 presents the cost coefficients (freight cost) given by the user in the right part of each cell.

STAGE 1- Input the Cost Coefficients			
	S1	S2	SUPPLY
01	5	2	
02	4		
03	1	4	
DEMAND			

↔↑↓ Move	**ENTER** Next cell	**F9** Clear Cell
PgUp Return		**PgDn** Advance

Fig. 2 - Freight Costs for Shipments

Figures 3 and 4 present the supply and demand values given by the user in the last column and last row, respectively.

STAGE 2- Input the Supplies			
	S1	S2	SUPPLY
O1	5	2	100
O2	4		50
O3	1	4	200
DEMAND			

↑↓ Move	**ENTER** Next cell	**F9** Clear Cell
PgUp Return		**PgDn** Advance

Fig. 3 - Origins' Supplies

STAGE 3- Input the Demands			
	S1	S2	SUPPLY
O1	5	2	100
O2	4		50
O3	1	4	200
DEMAND	150	100	

↔ Move	**ENTER** Next cell	**F9** Clear Cell
PgUp Return		**PgDn** Advance

Fig. 4 - Sinks' Demands

After input these values the user should verify the problem balancing. Figure 5 shows the screen where the user has identified die necessity of a dummy sink (selecting the answer "yes" for the question). In the next figure is presented the user's answer about the dummy demand value (100). Note, in this stage, the availability of a calculator for the necessary computations (key **F5** in the menu). Figure 7 presents the screen to input the dummy shipments cost coefficients. Observe that the user has given null values of all cells (right part) in DS column.

4

STAGE 4.2 – Verify the Supplies and Demands Balancing

	S1		S2		SUPPLY
O1		5		2	100
O2		4			50
O3		1		4	200
DEMAND		150		100	

Is a dummy sink needed?		yes	no

↔ Move	**ENTER** Next cell	**F9** Clear Cell
PgUp Return		**PgDn** Advance

Fig. 5 - Dummy Sink Necessity

STAGE 4.3 – Input the Dummy Supply/Demand Value

	S1		S2		SUPPLY
O1		5		2	100
O2		4			50
O3		1		4	200
DEMAND		150		100	

How much is the dummy demand value?	100

F5 Calculator	**F9** Clear Cell
PgUp Return	**PgDn** Advance

Fig. 6 - Dummy Demand Value

STAGE 4.4 - Input the Cost Coefficients for Dummy Shipments

	S1		S2	DS		SUPPLY
O1		5	2		0	100
O2		4			0	50
O3		1	4		0	200
DEMAND		150	100	100		350

↔ Move	**ENTER** Next cell	**F9** Clear Cell
PgUp Return	**F5** Calculator	**PgDn** Advance

Fig 7-Dummy Shipments Cost coefficients

In the considered problem there is an impossible route from 02 to S2. Figure 8 shows when the user identified the existence of prohibited shipments and in Figure 9, pressing the key **F10**, the user has blocked the associated cell.

STAGE 5.1 - Verify the Prohibited Shipments				
	S1	S2	DS	SUPPLY
O1	5	2	0	100
O2	4		0	50
O3	1	4	0	200
DEMAND	150	100	100	350

Are there prohibited shipments?   yes no

↔Move   **PgUp** Return   **PgDn** Advance

Fig8 - Prohibited Shipments

STAGE 5.2 - Block the Prohibited Shipments					
	S1	S2		DS	SUPPLY
O1	5	2		0	100
O2	4	XX	X	0	50
O3	1	4		0	200
DEMAND	150	100		100	350

↔↓↑ Move	**ENTER** Next Cell	F9 Clear Cell
PgUp Return	F10 Block Cell	**PgDn** Advance

Fig. 9 - Prohibited Shipment Blocking

Figure 10 shows the available initialization rules and the user's selection (northwest corner rule). In Figure 11 the user filled the left part of the cells with the suitable flow values according to the selected rule.

STAGE 6.1 – Select the Initialization Rule					
	S1	S2		DS	SUPPLY
Ol	5	2		0	100
O2	4	XX	X	0	50
O3	1	4		0	200
DEMAND	150	100		100	350

Select the Rule to Determine the Initial Feasible Solution

Northwest Corner   Minimum Cost   Other

↔ Move   **PgUp** Return   **PgDn** Advance

Fig. 10 - Initialization Rule

	STAGE 6.2-Input the Initial Feasible Solution						

		S1		S2		DS		SUPPLY
	O1	100	5		2		0	100
	O2	50	4	XX	X		0	50
	O3		1	100	4	100	0	200
DEMAND		150		100		100		350

↔↓↑ Move	**ENTER** Next Cell	F9 Clear Cell
PgUp Return	**F5** Block Cell	**PgDn** Advance

Fig. 11 - Initial Feasible Flows

Figure 12 shows the screen where the user has correctly identified that there is not an underlying spanning tree for the initial feasible solution. Then, in die next screen (see Figure 13) the user inputs additional null flow (O1-S2 cell) in order to obtain a feasible spanning tree. In die sequence (Figure 14), the user answer die question about die initial total cost (1100).

STAGE 6.3 - Verify Whether the Initial Feasible Solution is a Spanning Tree							

	S1		S2		DS		SUPPLY
O1	100	5		2		0	100
O2	50	4	XX	X		0	50
O3		1	100	4	100	0	200
DEMAND	150		100		100		350

Is there a spanning tree underlying the solution?		yes		no	

↔ Move	**Pgup** Return	**PgDn** Advance

Fig. 12 - Spanning Tree Test

STAGE 6.4 - Input the Remaining Basic Cells							

		S1		S2		DS		SUPPLY
	O1	100	5	0	2		0	100
	O2	50	4	XX	X		0	50
	O3		1	100	4	100	0	200
DEMAND		150		100		100		350

Input additional cells (Null Flows) in order to obtain a Feasible Spanning Tree

↔↓↑Move	**ENTER** Next Cell	**F9** Clear Cell
PgUp Return		**PgDn** Advance

Fig. 13 - Spanning Tree Complement

7

<table>
<tr><td><</td><td colspan="7">STAGE 6.5-Evaluate the Initial Total Cost</td></tr>
</table>

	S1		S2		DS		SUPPLY
O1	100	5	0	2		0	100
O2	50	4	XX	X		0	50
O3		1	100	4	100	0	200
DEMAND	150		100		100		350

How much is the initial total cost?	1100

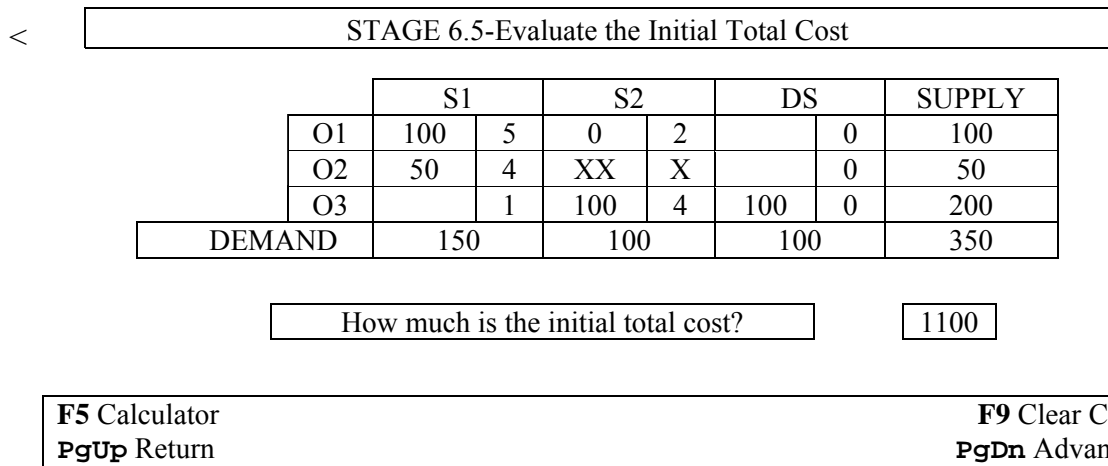|	
F5 Calculator	**F9** Clear Cell
PgUp Return	**PgDn** Advance

Fig. 14 - The Initial Total Cost Evaluation

In the next stage of SYMON's procedure a new row and column is added to the tableau. These new cells must be filled by the user with the associated column (Kj) and row (Li) prices. During this SYMON application the user committed one error in some Kj or Li evaluation. So, an error message is displayed and since the allowable number of errors by stage was selected (see Figure 1) as equal to 1, all the correct values are presented by the system (see Figure 15). Pressing the key F1 the user can access a help with the apropriated mathematical formula for these computations.

STAGE 7 - Determine the Row and Column Prices

		K1=5		K2 = 2		K3 = -2		
		S1		S2		DS		SUPPLY
L1= 0	O1	100	5	0	2		0	100
L2 = -l	O2	50	4	XX	X		0	50
L3 = 2	O3		1	100	4	100	0	200
DEMAND		150		100		100		350

Total Cost =1100	<u>1</u>

↔↓↑ Move	**ENTER** Next Cell	**F9** Clear Cell
PgUp Return	**F5** Calculator	**PgDn** Advance
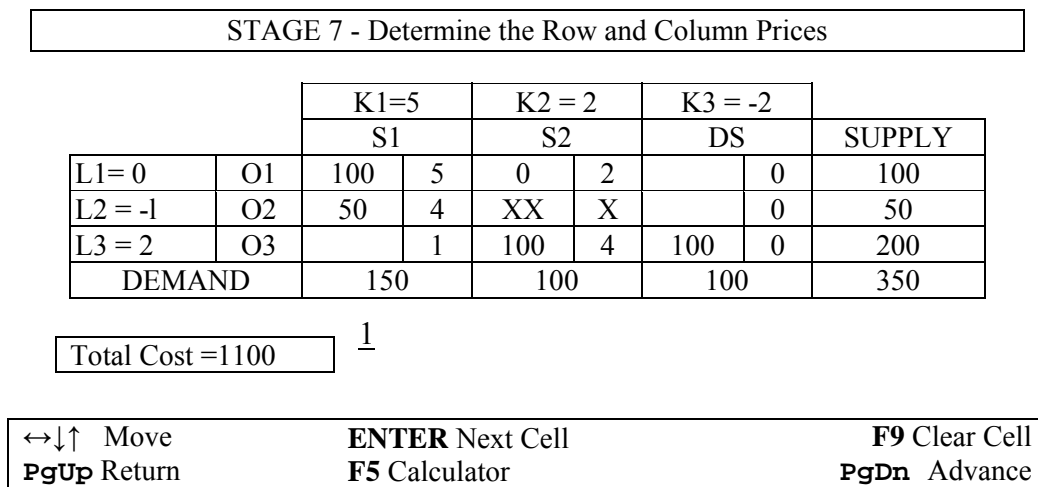
Fig. 15 - Row and Column Prices Determination

Figure 16 shows the correct reduced cost values for non-basic cells (-6 in O3-S1 cell, 2 in Ol-DS cell and 3 in O2-DS cell). Again, one error was committed by the user and after an error message, the system has provided the above values. As above, a help for these computations is available pressing the key **F1**.

Figure 17 shows the screen where the user identifed correctly the solution as non-optimal (selecting the answer "no" for the question).

8

STAGE 8 - Determine the Reduced Costs for Non-Basic Cells

		K1=5		K2 = 2		K3 = -2		
		S1		S2		DS		SUPPLY
L1= 0	O1	100	5	0	2	2	0	100
L2 = -l	O2	50	4	XX	X	3	0	50
L3 = 2	O3	-6	1	100	4	100	0	200
	DEMAND	150		100		100		350

Total Cost =1100

Fig. 16 - Reduced Costs for Non-Basic Cells Determination

STAGE 9.1 - Test Optimality

	S1		S2		DS		SUPPLY
O1	100	5	0	2	2	0	100
O2	50	4	XX	X	3	0	50
O3	-6	1	100	4	100	0	200
DEMAND	150		100		100		350

Total Cost = 1100   Iteration no.01

Is this an optimal solution?   yes   no

Fig.17OptimalityTest

In the next screen (Figure 18) the user has correctly selected the entering non-basic cell (O3-S1). In Figure 19 the entering selected cell is displayed as a blinking "IN" and the user should select die leaving basic cell in the associated cycle. In this case the user has correctly selected O3-S2 as the leaving basic cell (observe that Ol-Sl selection would be also a correct alternative). In the next screen (Figure 20) the user has performed the pivoting in the associated cycle (O3-S1, Ol-Sl, O1-S2 and O3-S2).

STAGE 9.2 - Select the Entering Non-Basic Cell

	S1		S2		DS		SUPPLY
Ol	100	5	0	2	2	0	100
02	50	4	XX	X	3	0	50
O3	-6	1	100	4	100	0	200
DEMAND	150		100		100		350

Select the Entering Non-Basic Cell and press <**PgDn**>

Fig.18- Entering Non-Basic Cell Selection

	S1		S2		DS		SUPPLY
O1	100	5	0	2	2	0	100
O2	50	4	XX	X	3	0	50
O3	**IN**	1	**100**	4	100	0	200
DEMAND	150		100		100		350

Find the cycle associated with the Entering Non-Basic Cell
Select the Leaving Basic Cell and press **<PgDn>**

↔↓↑Move   **PgUp** Return   **PgDn** Advance

Fig. 19 - Leaving Basic Cell Selection

STAGE 9.4 – Perform the Pivoting

	S1		S2		DS		SUPPLY
O1	0	5	100	2	2	0	100
O2	50	4	XX	X	3	0	50
O3	100	1		4	100	0	200
DEMAND	150		100		100		350

Entering Cell:31   Leaving Cell:32

Update in the cycle the flow values of the New Basic Solution

↔↓↑Move   **ENTER** Next Cell   **F9** Clear Cell
PgUp Return   **F5** Calculator   **PgDn** Advance

Fig. 20 - Flow Updating

The Figure 21 shows a screen which contains: (a) the previous solution cost (1100); (b) the flow in the entering cell ($X31 = 100$); and (c) the reduced cost of the entering cell in the previous solution ($AC31 = -6$). Based on these values the user has updated the total cost of the present solution as 500. A help for these computations is available pressing the key **F1**.

STAGE 9.5 – Update the Total Cost

	S1		S2		DS		SUPPLY
O1	0	5	100	2	2	0	100
O2	50	4	XX	X	3	0	50
O3	100	1		4	100	0	200
DEMAND	150		100		100		350

Previous Cost: 1100   Flow in Entering Cell: X31=100
                                       Reduced Cost X31:ΔC31=-6

How much is the new total cost?   500

F5 Calculator   **F9** Clear Cell
PgUp Return   **PgDn** Advance

Fig. 21 - Total Cost Updating

The above process is repeated and in the stage 9.1 of iteration number 3 (see Figure 22) the user identifed correctly the solution as optimal.

STAGE 9.1 – Test Optimality							
		S1		S2		DS		SUPPLY
O1	4	5	100	2	0	0	100
O2	3	4	XX	X	50	0	50
O3	150	1	2	4	50	0	200
DEMAND	150		100		100		350

Total Cost = 350   Iteration no. 03

Is this an optimal solution ?   yes   no

↔ Move   PgUp Return   PgDn Advance

Fig. 22 - Optimality Test (Iteration no. 3)

Finally, SYMON presents the user's performance report (Figure 23) with an option to print the original data, the optimal solution and the report itself.

User's performance Report

Allowable Number of User's Errors/Stage Selected: 1
Total User's Errors in Each Stage:
• Balancing Supplies and Demands: 0
• Obtaining Initial Feasible Basic Solution: 0
• Obtaining Row and Column Prices: 1
• Obtaining Reduced Costs: 1
• Testing Optimality: 0
• Obtaining the Entering Non-Basic Cell: 0
• Obtaining the Leaving Basic Celt 0
• Updating the Flow Values: 0
• Obtaining the Total Costs: 0
Total of User's Errors: 2

Do you wish to print tins report and the optimal solution?   yes   no

↔ Move   PgUp Return   PgDn Advance

Fig.23-User's performance Repot

CONCLUSION
Classically, OR provides support for decision making on situations where there is a need to allocate scarce resources, while having art as a scientific feature. The scientific feature is characterized when theories and algorithms are developed. Yet, during the modeling process, a creative effort by the OR-analysts is necessary.

Basically, OR teaching has three stages:
- the presentation of a methodology for the mathematical system modeling process,
- the development, of general (for large model classes) or specialized (for specific models) algorithms, and
- the discussion about the advantages and disadvantages of known algorithms for a given model class.

Usually, the teachers are concerned with the second stage, that is, they introduce the students in the theory and in the study of known algorithms. In this way, the teaching fails exactly in its creative and intuitive aspects. This work proposes a different teaching methodology, which includes case studies as suggested by Scott and Buchanan (1992), as well as theoretical development and utilization of user-friendly educational softwares in order to contribute to the improvement of the student's knowledge on OR techniques.

Our experience of OR teaching has shown that the computational effort necessary for manual algorithm application is responsible for the deviation of the student's attention about relevant points, which involves the modeling process and the search for the best available algorithm for each model. So, the importance of having an appropriate software tool becomes clear.

The software environment proposed in this work is being implemented both in Pascal and C languages for microcomputers. Even though partially developed, it has been tested at Unesp (Sao Paulo State University) and Unicamp (University of Campinas) OR courses.

The improvement of communication with the user through a multimedia system, so that the students can dispose of on-line classes, is intended as a future research.

REFERENCES

Costa, A.F.B.; Senne, E.L.F. and Matins, F.A.S. "An Educational Linear Programming Software Environment". In: *International Symposium on Mathematical Programming, 14*. Amsterdam, 1991.

Janczyk, WJC and Beasly, J.E. "Multiple Model OR Packages". *Journal of Operational Research Society* (1988) 39(5):487-509.

Jensen, P. "Microcomputer Forum: Microcomputers in the Instruction of Operations Research". *Interfaces* (1983) 13(5):18-24.

King, M. "A Framework for Teaching OR Techniques". *Journal of Operations Research Society* (1988) 39(12): 1087-1093.

Llewellyn, J. and Sharda, R. "Linear Programming Software: 1990 Survey". *OR/MS Today* (1990) 17(5):35-47.

Scott, J. and Buchanan, J. "Teaching Management Science: Hold the Lectures". *OR/MS Today* (1992) 19(5):46-50.

Senne, E.L.F.; Marins, F.A.S.; Costa, A.F.B. "A Graphical Environment for Operations Research Teaching". In: ***World Conference on Operations Research, 13.*** Lisboa, Portugal, Jul. 1993. Abstracts, p. 46.

Sharda, R. "The State of the Art of Linear Programming on Personal Computers". ***Interfaces*** (1988) 18(4):49-58.

Sharda, R. "Linear Programming Software for Personal Computers: 1992 Survey". ***OR/MS Today*** (1992) 19(3):44-60.

Wheeler, F.P. "A Starter Kit for Micro-Based LP Solvers". ***Journal of Operations Research Society*** (1985) 36(7):637-641.

Yurkievicz, J. "Educational Operations Research Software: A Review", ***Interfaces*** (1988) 18(4):59-71.

KEYWORDS:
Education, Learning, Linear Programming, Microcomputers, Operations Research.

ACKNOWLEDGEMENTS: