Acta et Commentationes, Sciences of Education, nr. 2(36), 2024
p. 107-114

ISSN 1857-0623
E-ISSN 2587-3636

CZU: 004.4:37.0

DOI: 10.36120/2587-3636.v36i2.107-114

# IMPLEMENTAREA PACHETELOR SIMPLEX ȘI OPTIMISATION ÎN SOLUȚIONAREA JOCURILOR MATRICEALE

**Natalia JOSU**, dr., conf. univ.
https://orcid.org/0000-0002-3687-5437
Catedra Informatică și Tehnologii Informaționale
Universitatea Pedagogică de Stat ”Ion Creangă” din Chișinău

**Rezumat**. În acest articol este prezentat modul de implementare a pachetelor *simplex* și *Optimisation* a software-ului Maple în procesul de soluționare a jocurilor matriceale.

**Cuvinte cheie**: teoria jocurilor, joc matriceal, strategie, funcție de câștig, Maple, simplex, Optimisation, programarea liniară.

## THE IMPLEMENTATION OF THE SIMPLEX AND OPTIMIZATION PACKAGES IN THE PROCESS OF SOLVING MATRIX GAMES

**Abstract**. The following article presents the implementation of the simplex and optimization packages in Maple software in the process of solving matrix games.

**Keywords**: game theory, matrix game, strategy, payoff function, Maple, simplex, Optimization, linear programming.

## Noțiuni introductive

Există diverse formulări privind definiția *Teoriei jocurilor* sau noțiunea de *Joc strategic*. Teoria jocurilor are drept scop determinarea metodelor de alegere a celor mai bune decizii în situaţii de conflict, în care acţionează mai mulţi factori raţionali ce urmăresc interese opuse [1] sau teoria jocurilor reprezintă un ansamblu de instrumente analitice proiectate pentru a ne ajuta să înţelegem fenomenele pe care le observăm la interacţiunea factorilor de decizie în situaţii de conflict şi cooperare [2].

John von Neumann şi Oskar Morgenstern au definit *jocul/jocul strategic* ca „*orice interacțiune între diverși agenți, guvernat de un set de reguli specifice, care stabilesc mutările posibile ale fiecărui participant şi câștigurile pentru fiecare combinație de mutări*”. Mutările posibile ale fiecărui participant sunt stabilite de anumite reguli. Le fel aceste reguli precizează şi condiţiile în care se termină jocul, precum şi recompensa (câştigul) pentru fiecare jucător [3].

## Clasificarea jocurilor

În cadrul Teoriei jocurilor se disting diverse tipuri de clasificări. Aceastea pot varia în funcție de diferite criterii, cum ar fi: numărul de jucători, numărul de strategii, tipul funcției de câștig, natura interacțiunii dintre jucători, natura câștigului, numărul de mutări, starea informației etc.

- După **numărul jucătorilor** pot exista jocuri cu *2* sau *n* jucători și jocuri contra naturii. Jocurile cu doi jucători se mai numesc *jocuri pare*, iar cele cu *n* jucători - *jocuri multiple*.
- După **numărul strategiilor** deosebim *jocuri finite* și *jocuri infinite*.
- După **structura câștigului** deosebim *jocuri cu sumă nulă* (sau sumă zero) și *jocuri cu sumă nenulă*.
- Conform **tipului strategiilor** aplicate se disting jocuri în *strategii pure* și jocuri în *strategii mixte*.
- În dependență de **tipul funcției de câștig**, jocurile pot fi clasificate în jocuri *matriciale*, *bimatriceale*, *convexe* etc.
- În **funcție de natura cooperării** sau interacțiunii jucătorilor deosebim *jocuri fără coaliție* (necooperative) și *jocuri de coaliție* (cooperative).
- În funcție de **natura informației** se deosebesc jocuri cu *informație completă* și jocuri cu *informație incompletă*.
- În raport cu **desfășurarea în timp** a jocurilor se deosebesc *jocuri statice* și *jocuri dinamice*.
- În cazul jocurilor dinamice, în raport cu **natura informației** avem jocuri cu *informație perfectă* și jocuri cu *informație imperfectă*.
- După **tipul mutărilor** deosebim *jocuri strategice* și *jocuri de noroc*. Teoria jocurilor nu are ca obiect de studiu jocurile de noroc.
- După **forma de reprezentare a jocului** se numără *forma normală* sau *strategică* (caz particular forma matriceală) și *forma extinsă/extensivă/pozițională* (descrierea jocului sub forma unui arbore). *Forma normală* reprezintă jocul printr-o matrice, în care sunt indicate strategiile și câștigurile posibile pentru fiecare jucător. *Forma extinsă* reprezintă jocul printr-un arbore de decizie, evidențiind secvențele de mișcări, alegerile disponibile și informațiile disponibile în fiecare nod de decizie.

Formă de reprezentare a unui joc în teoria jocurilor depinde de natura și contextul jocului analizat. *Forma normală* este indicată atunci când jocul este simultan, iar jucătorii își aleg strategiile în același timp fără a cunoaște alegerile celorlalți. *Forma extinsă* este utilă atunci când jocul are o structură secvențială, cu mai multe etape. În cazul dat este esențial să se indice ordinea mișcărilor, informațiile disponibile și alegerile la fiecare nod de decizie.

**Soluționarea jocurilor matriceale**

Jocurile matriceale reprezintă un subiect de interes special în teoria jocurilor. Motivul principal ar fi simplitatea reprezentării relațiilor strategice între jucători, determinarea cu ușurință a echilibrului Nash, reducerea numărului de strategii a unei matrice complexe utililizând metoda dominării.

Un joc *finit* cu *doi* jucători *A* și *B*, în care *A* posedă *m* strategii pure, iar *B* – *n* strategii pure, poate fi reprezentat cu ajutorul unei matrice. Din acest motiv se mai numește *joc matriceal*. În acest caz, funcția de câștig se reprezintă sub forma unui tabel de plăți, adică o matrice de dimensiune $m \times n$, $A = a_{ij}, i = \overline{1, m}, j = \overline{1, n}$.

| | **Strategiile jucătorului B** | | | | |
|---|---|---|---|---|---|
| **Strategiile jucătorului A** | $A_i/B_j$ | $B_1$ | $B_2$ | $\cdots$ | $B_n$ |
| | $A_1$ | $a_{11}$ | $a_{12}$ | $\cdots$ | $a_{1n}$ |
| | $A_2$ | $a_{21}$ | $a_{22}$ | $\cdots$ | $a_{2n}$ |
| | $\vdots$ | $\vdots$ | $\vdots$ | $\cdots$ | $\vdots$ |
| | $A_m$ | $a_{m1}$ | $a_{m2}$ | $\cdots$ | $a_{mn}$ |

Elementul $a_{ij}$ reprezintă câștigul primului jucător la alegerea strategiilor $A_i$ și $B_j$. Fiind un joc cu sumă nulă, matricea *A* se scrie în raport cu un singur jucător și anume *A*. Pentru jucătorul *B* toate elementele matricei de plăți au semn contrar. Jucătorul *A* se mai numește jucător *maximizant*, iar *B* – jucător *minimizant*. Dacă acest joc este cu *sumă nulă*, atunci mai este numit și *joc antagonist*.

- Jocurile cu sumă nulă *cu punct șa* pot fi soluționate cu ușurință prin determinarea valorii inferioare și superioare a jocului. În acest caz, se spune că soluția jocului se va determina în *strategii pure*.
- Jocurile cu sumă nulă *fără punct șa* pot fi soluționate în *strategii mixte* și pot fi clasificate în: jocuri de dimensiunea $2 \times 2$, $2 \times n$, $m \times 2$ și $m \times n$.

Jocurile de dimensiunea $2 \times n$, $m \times 2$ pot fi rezolvate cu ușurință prin aplicarea metodei grafice sau pot fi reduse la un joc de dimensiunea $2 \times 2$, care poate fi soluționat atât aplicând metoda grafică cât și metoda analitică. În ceea ce privește jocurile matriciale de dimensiunea $m \times n$, acestea pot fi reduse la rezolvarea unei probleme de programare liniară. În continuare se vor utiliza pachetele *Optimization* și *simplex* ale software-ului Maple pentru soluționarea unei probleme concrete [4].

**Aplicație practică**

Un olar produce oale și ulcioare din ceramică. Pentru producerea unei oale, olarul are nevoie de 2 ore pentru modelare materiei prime, 1 oră pentru decorare şi 3 ore pentru uscare. Pentru un ulcior este nevoie de 2 ore pentru modelare, 2 ore pentru decorare şi 1 oră pentru uscare. Olarul are la dispoziţie 160 ore pentru modelarea materiei prime, 140 ore pentru decorare şi 170 ore pentru uscare. Câte oale și ulcioare ar trebui să producă olarul pentru a-şi maximiza profitul, având în vedere că o oală aduce un profit de 12 u.m., iar un ulcior 8 u.m.

*Soluție:*

Se va nota cu *x* – numărul de oale produse, iar cu *y* – numărul de ulcioare. Inițial se va scrie problema într-o formă tabelară, după care se va formula setul de restricții/constrângeri și funcția obiectiv.

| | Oale | Ulcioare | Numărul de ore |
|---|---|---|---|
| **Modelare** | 2 | 2 | 160 |
| **Decorare** | 1 | 2 | 140 |
| **Uscare** | 3 | 1 | 170 |
| **Funcția obiectiv** | 12 | 8 | $F = 12x + 8y$ |

Problema de programare liniară va avea forma: $\begin{cases} F = 12x + 8y; F \rightarrow max \\ 2x + 2y \leq 160 \\ x + 2y \leq 140 \\ 3x + y \leq 170 \\ x \geq 0, y \geq 0 \end{cases}$.

Inițial se va utiliza instrucțiunea *Maximize* a pachetului *Optimization*:

*with(Optimization):* - inițializarea instrucțiunilor pachetului

*Maximize(12*x+8*y, {0<=x, 0<=y, x+2*y<=140, 2*x+2*y<=160, 3*x+y<=170})*

$$[820., [x = 45., y = 35.]]$$

Astfel soluția oferită de pachetul *Optimization*, ne indică că olarul va trebui să producă 45 oale și 35 ulcioare pentru a avea un profit maxim și anume 820 u.m. În continuare se va lucra cu opțiunea *maximize* a pachetului *simplex*:

*with(simplex):* - inițializarea instrucțiunilor pachetului

*maximize(12*x+8*y, {0<=x, 0<=y, x+2*y <=140, 2*x+2*y<=160, 3*x+y<=170}, NONNEGATIVE);*

$$\{x = 45, y = 35\}$$

Pachetul *simplex* afișează ca rezultat doar valorile pentru *x* și *y*. În acest caz se va substitui în funcția obiectiv aceste valori obținute și se va determina profitul maxim: $F = 12x + 8y = 12 * 45 + 8 * 35 = 820$.

Utilizând pachetul *plots* și instrucțiunea *inequal* se poate ilustra grafic domeniul soluțiilor posibile:

*with(plots):* - inițializarea instrucțiunilor pachetului

*inequal({0 <= x, 0 <= y, x + 2*y <= 140, 2*x + 2*y <= 160, 3*x + y <= 170}, x = -1 .. 60, y = -5 .. 70, optionsfeasible = (color = yellow), optionsopen = (color = red, thickness = 2), optionsclosed = (color = black, thickness = 1), optionsexcluded = (color = gray));*

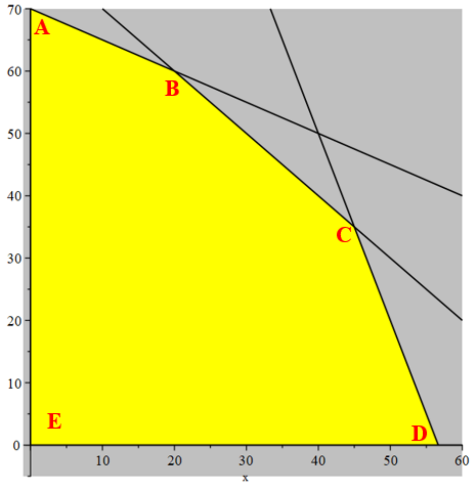


Figura *ABCDE* reprezintă domeniul (poligonul) soluțiilor posibile. La următorul pas se vor determina coordonatele vârfurilor acestui poligon:

A: *solve({x = 0, x + 2*y = 140}, {x, y})* $\{x = 0, y = 70\}$;

B: *solve({x + 2*y = 140, 2*x + 2*y = 160}, {x, y})* $\{x = 20, y = 60\}$;

C: *solve({2*x + 2*y = 160, 3*x + y = 170}, {x, y})* $\{x = 45, y = 35\}$;

D: *solve({y = 0, 3*x + y = 170}, {x, y})* $\left\{x = \frac{170}{3}, y = 0\right\}$;

*E: (0,0).*

În continuare se vor substitui coordonatele vârfurilor *A*, *B*, *C*, *D* și *E* în funcția obiectiv și din valorile obținute se va alege valoarea maximală, ceea ce va reprezenta și profitul maxim.

$F(A) = 12x + 8y = 12 \cdot 0 + 8 \cdot 70 = 560$; $F(B) = 12x + 8y = 12 \cdot 20 + 8 \cdot 60 = 720$; $F(C) = 12x + 8y = 12 \cdot 45 + 8 \cdot 35 = 820$; $F(D) = 12x + 8y = 12 \cdot \frac{170}{3} + 8 \cdot 0 = 680$; $F(D) = 12x + 8y = 12 \cdot 0 + 8 \cdot 0 = 0$.

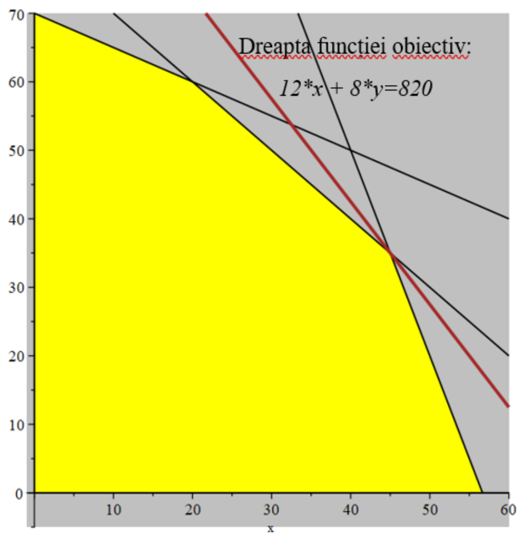
Astfel observăm că în punctul *C(45;35)* funcția obiectiv atinge valoarea maximă și anume *820*. În continuare se va completa graficul ce exprimă domeniul soluțiilor posibile cu dreapta funcţiei obiectiv pentru a ne convinge că aceasta nu intersectează poligonul soluțiilor posibile.

*with(plots):*

*g1 := inequal({0<=x, 0<=y, x+2*y<=140, 2*x+2*y<=160, 3*x+y<=170}, x=-1 .. 60, y= -5 .. 70, optionsfeasible = (color = yellow), optionsopen = (color = red, thickness = 2), optionsclosed = (color = black, thickness = 1), optionsexcluded = (color = gray)):*

*g2 := inequal({820 < 12*x + 8*y}, x = -1 .. 60, y = -5 .. 70, optionsfeasible = (color = blue), optionsopen = (color = brown, thickness = 3), optionsclosed = (color = brown, thickness = 1), optionsexcluded = (color = gray)):*

*display(g1, g2);*



Din imagine se observă că semiplanul *12*x + 8*y>820* nu intersectează poligonul soluțiilor, deci se poate afirma că funcția obiectiv $F = 12x + 8y$ atinge valoarea maximă în punctul *C(45;35)* și valoarea acesteia este *820*.

Și procesorul tabelar Excel permite soluționarea problemelor de optimizare cu ajutorul comenzii *Solver* din meniul *Data*. Inițial se vor introduce datele cunoscute pe o foaie de calcul Excel (figura 1).

| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | **Date de intrare** | | | | | | |
| 2 | | **Coeficienţii restricţiilor** | | | | | |
| 3 | **Tipul lucrării** | **Oale** | **Ulcioare** | **Total utilizat** | | **Numărul total de ore** | |
| 4 | Modelare | 2 | 2 | 0 | <= | 160 | |
| 5 | Decorare | 1 | 2 | 0 | <= | 140 | |
| 6 | Uscare | 3 | 1 | 0 | <= | 170 | |
| 7 | | **x** | **y** | **Total profit** | | | |
| 8 | **Profit** | 12 | 8 | 0 | | | |
| 9 | | | | | | | |
| 10 | **Planul de producţie** | | | | | | |
| 11 | | **x** | **y** | | | | |
| 12 | Produse | 0 | 0 | | | | |

**Figura 1. Date de intrare**

Așa cum *x* și *y* nu sunt cunoscute, le atribuim inițial valoarea 0. Completăm celulele D4:D6 utilizând formulele:

D4: =SUMPRODUCT(B4:C4;B12:C12);

D5: =SUMPRODUCT(B5:C5;B12:C12);

D6: =SUMPRODUCT(B6:C6;B12:C12).

În mod analog completăm și celula ce exprimă profitul realizat:

D8: =SUMPRODUCT(B8:C8;B12:C12).

La următorul pas se va apela comanda *Solver* și se va completa fereastra comenzii *Solver* după cum urmează în figura 2:

- *Set Objective* – se va introduce adresa celulei, în care se va determina profitul maximal, în cazul dat *D8*;
- *To* – se va alege opțiunea *max*, deoarece se dorește a maximiza profitul;
- *By Changing Variable Cells* – se va introduce adresa celulelor ce își vor modifica valoarea, adică *x*, *y*;
- *Subject to the Constraints* – în câmpul respectiv se vor introduce restricțiile sistemului cu ajutorul butonului *Add*. La fel, în această fereastră există opțiunea de a modifica restricțiile introduse (*Change*), a șterge careva restricție, a reseta din nou opțiunile ferestrei date;
- Butonul *Options* – include opțiuni avansate privind metoda aleasă pentru rezolvare;
- *Select a Solving Method* – se va alege metoda de soluționare a problemei;

- După selectarea butonului *Solve*, utilizatorul are posibilitatea să aleagă unul din cele 3 rapoarte.

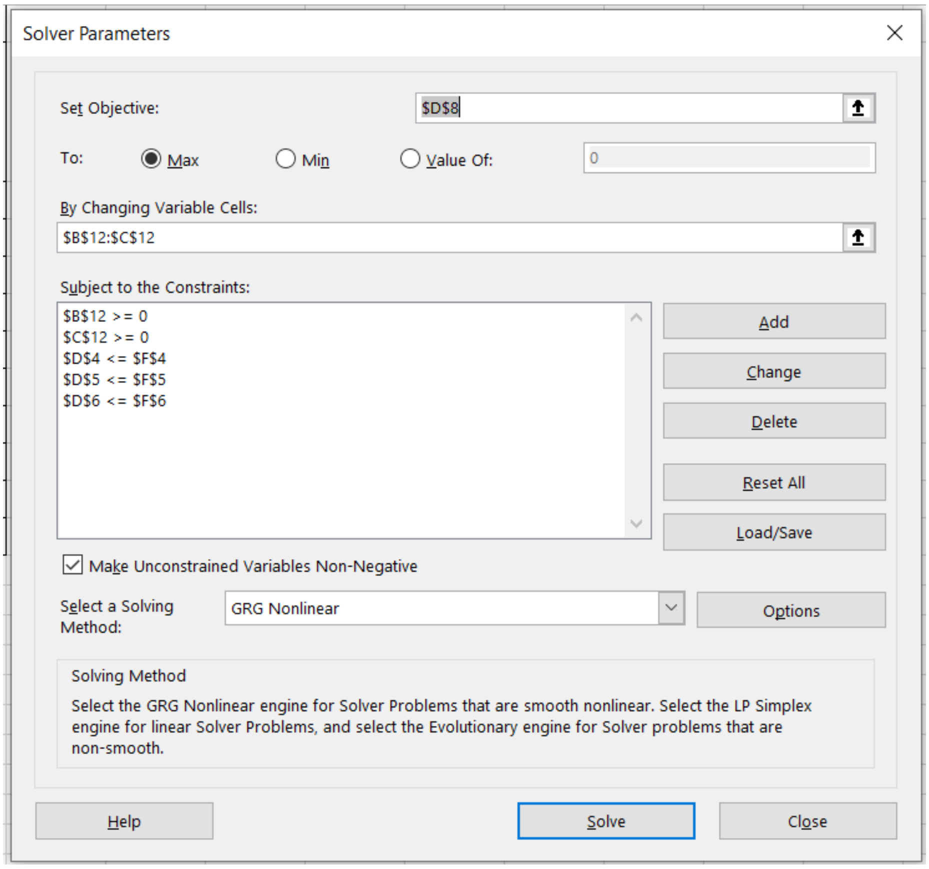



**Figura 2. Fereastra *Solver***

Prin urmare, putem constata că s-a obținut aceeași soluție (figura 3).

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | **Date de intrare** | | | | | | |
| 2 | | **Coeficienții restricțiilor** | | | | | |
| 3 | **Tipul lucrării** | **Oale** | **Ulcioare** | **Total utilizat** | | **Numărul total de ore** | |
| 4 | Modelare | 2 | 2 | 160 | <= | 160 | |
| 5 | Decorare | 1 | 2 | 115 | <= | 140 | |
| 6 | Uscare | 3 | 1 | 170 | <= | 170 | |
| 7 | | x | y | **Total profit** | | | |
| 8 | **Profit** | 12 | 8 | 820 | | | |
| 9 | | | | | | | |
| 10 | **Planul de producție** | | | | | | |
| 11 | | x | y | | | | |
| 12 | Produse | 45 | 35 | | | | |

**Figura 3. Soluția problemei**

## Concluzii

Instrumentele informatice actuale permit facilitarea procesului de soluționare a problemelor de optimizare. Atât procesorul tabelar Excel cât și software-ul Maple sunt ușor de utilizat și oferă utilizatorului atât soluția numerică cât și ilustrează grafic rezultatul obținut.

*Articol realizat în cadrul proiectului de cercetări științifice „Metodologia implementării TIC în procesul de studiere a științelor reale din perspectiva conceptului STEAM și Inteligenței Artificiale", codul 040101, din cadrul Programului instituțional de cercetare (2024-2027), aprobat prin Ordin MEC nr. 102 din 01.02.2024*

## Bibliografie

1. ENICOV, I. Aplicarea teoriei jocurilor şi a teoriei deciziei în managementul firmei. În: *Materialele conferinței „Aspecte ale dezvoltării potenţialului economico-managerial în contextul asigurării securităţii naţionale"*, secțiunea 1-3, Bălți, Moldova, 6-7 iulie 2015. pp. 153-158. ISBN 978-9975-132-35-0.
2. LOZAN, V. Teoria jocurilor şi relaţiile sociale. In: *Studia Universitatis (Seria Ştiinţe Exacte şi Economice).* 2013, nr. 2(69), pp. 23-27. ISSN 1857-2073.
3. ROMAN, M., MARIN, D., STANCU, S. *Teoria jocurilor pentru economiști*. București: Edutura ASE, 2005. ISBN 973-594-598-3.
4. АЛАДЬЕВ, В. З., БОЙКО, В. К., РОВБА, Е. А. *Программирование и разработка приложений в maple*. Таллинн, 2007. ISBN 978-985-417-891-2.