THE FORMAL MOULDING IN ECONOMICS

MIRCEA IOSIF NEAMȚU
Department of Computer Science
Lucian Blaga University
5-7 Ioan Ratiu Str, Sibiu
ROMANIA

EMIL MARIN POPA
Department of Computer Science
Lucian Blaga University
5-7 Ioan Ratiu Str, Sibiu
ROMANIA

Abstract: Within the most diverse branches of economical sciences, problems occur which can be solved through different alternatives. To differentiate these solving modes, a certain purpose needs to be reached and the best solution is the one in which the purpose is best fulfilled.

The result of the economic analyses leads to choosing some values for the variables that describe the process and that can be economical and physical sizes (goods, financial values, distances etc.). The actual conditions of the study insert limitations in formulating the problem, the optimal solution being the one that leads to the best choice in choosing the variable values within the establishing the imposed restrictions.

For solving this kind of problem, the purpose of mathematics in economics is to add a value to an economic function within a just analyses of an actual situation, to differentiate the secondary aspects from the main aspects and to apply the correct economical politics for actual situations.

These aspects are being proved in this article with the help of actual problems treated from the point of view of mathematical programming and from an formal point of view.

Key-Words: models, process, finite automats, regular expression, grammars

1. Introduction

In order to solve a mathematical programming problem, the following step need to be followed:

- Establishing the values x_1, ..., x_n, in fact the vector $X = (x_1, \dots ,x_n)^T \in R^n$;
- Establishing the set objective, a function x_1,..., x_n, as in $f(X)$, for which a maximization or minimization process will be applied
- Establishing the restrictions, the connections between the variables and their limitations, limitations without which the problem doesn't have a practical value, and that can have the form $g_i(X) \le 0$, $i = 1..m$. the restrictions with the form $x_j \ge 0$, $j = 1..n$, called conditions of nonnegative, occur when the variables represent quantities that have a real interpretations only when negative.

In conclusion, the general problem of mathematical programming is searched under the form of determining the minimal or maximal value of the function:

$f(X) = f(x_1, \dots ,x_n)$,

under the conditions:

$g_i(X) = g_i(x_1, \dots ,x_n) \le 0$, $i = 1..m$,
where $f, g_i : R^n \rightarrow R$.

The function f is being optimized and it is called *objective or efficiency function.*

If the functions f and g_i, $i = 1..m$, are linear functions, we say that the problem is of linear programming.

E.g.: we announce a problem of linear programming which we want to analyze by modeling it from an mathematic an a formal point of view.
The resources R_i, $i = 1..m$, wich can in fact be money investments, labourforce, row materials, etc, affordable in quatities b_i, $i = 1..m$ (conventional unites) will be used for producing the activities A_j, $j = 1..n$ that can be projects, pices, products etc. By knowing the inncom unites on activities $c_{j.,}$ as well as coeficienții tehnologici a_{ij}, respectiv the quantities for each resources R_i necesary to acomplish one unit from $A_{j,}$ we hope to determine the level of x_j of the activities A_j, $i = 1..m$, $j = 1..n$, for which the total venit should be maximezed.

We represent the data of the problem in the following table:

$R_i \backslash A_j$	$A_1 \dots A_j \dots A_n$	Disponibil
R_1	$a_{11} \dots a_{1j} \dots a_{1n}$	b_1
...	... ...	...
R_3	$a_{i1} \dots a_{ij} \dots a_{in}$	b_i
...	... ...	...
R_m	$a_{m1} \dots a_{mj} \dots a_{mn}$	b_m
Venitul unitar	$c_1 \dots c_j \dots c_n$	

The mathematic model is being written in the following mode:

	under vectorial form:
$[\max]f(X) = \sum_{j=1}^{n} c_j x_j$ (1)	$[\max]f(X) = c_1x_1+\dots+c_nx_n$ (1')
$\sum_{j=1}^{n} a_{ij}b_j$, $i = 1..m$ (2)	or: $a_1x_1+\dots+a_nx_n \le b$ (2')
$x_j \ge 0$, $j = 1..n$ (3)	$x_1, \dots, x_n \ge 0$ (3')

Noting that $A = (a_{ij})$, $i = 1..m$, $j = 1..n$; or:

$$A = (a_1, \dots, a_n),\ b = (b_1, \dots, b_m)^T,$$

$$C = (c_1, \dots, c_n),\ X = (x_1, \dots, x_n)^T,$$

where $a_j = (a_{1j}, \dots, a_{mj})^T$ is the coloumn j from the matrix A.
The problem is being written under teh form of a matrix as followed:

$$[\max]f(X) = CX,\ AX \le b,\ X \ge 0.$$

The problem has the following standard form:

$[\text{optim}]f(X) = CX$ (4)		$(\text{opt})f(X)=c_1x_1+\dots+c_nx_n$ (4')
$AX = b$	(5) or:	$a_1x_1+\dots+a_nx_n=b$ (5')
$X \ge 0$	(6)	$x_1, \dots, x_n \ge 0$ (6')

Any maximum problem can be transposed for minimum and vice versa, thanks to the obvious reraltion;

$$[\max]f(X) = -[\min][-f(X)]$$

and

$$[\min]f(X) = -[\max][-f(X)]$$

Considering the linear programming problem under a standard form, given by the relations (4),(5),(6), the solutions of the problems can be announced in the following mode:

Definition 1. The vector $X \in R^n$, $X = (x_1, \dots, x_n)^T$ is called *possible solution* (admissible, programm) if it satisfies the relations (5) and (6). We note with P the multitude of posible solutions.

Definition 2. If $X \in P$ also satisfies tha condition (4), X is called *optimal solution.* We note with O the multitude of optimal solutions.
Observation: the condition is imposed that: rang$A = m$, for the compatibility of system (5), and for the undetermination of this sourt of system we need to have $m < n$.

Definition 3. The m variables associated to the coloumns B are called *base varaiables*. They form a subvector of X and are called X_B. The Rest of *n-m* variables are called *secondary variables* and form the sebvector of X noted with Xs. If $X_S = 0$ the system of (5) becomes: $BX_B = b$, from where $X_B = B^{-1}b$.
Observation: If B is a canonic base, than $X_B = b$ (coloumn of the free terms).

Definition 4. A base is $B = a_1, \dots, a_m$. The vector $X = (X_B, X\text{s})$, cu $X_B = B^{-1}b$, $X_S = 0$, $X \in P$ and it is called *possible base solution.* If X_B has m positive components, X will be the *possible undegenerated base* solution, in the contrary it will be *degenerated.*
We note with P_B the multitude of possible base solutions.

2. Case study

There is teh system:

$$\begin{cases} 3x_1 - 6x_3 + x_4 = 6 \\ -x_1 + x_2 + 2x_3 = 4 \end{cases},$$

the matrix of the system A

$$A = \begin{pmatrix} 3 & 0 & -6 & 1 \\ -1 & 1 & 2 & 0 \end{pmatrix},$$

with the vectores:

$$a_1 = \begin{pmatrix} 3 \\ -1 \end{pmatrix},\ a_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix},\ a_3 = \begin{pmatrix} -6 \\ 2 \end{pmatrix},\ a_4 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

Because rang$A = 2$, the system is double compatible undetermined. A solution can be found as followed:

$$\begin{cases} 3x_1 = 6 + 6x_3 - x_4 \\ -x_1 + x_2 = 4 - 2x_3 \end{cases},$$

where for:

$x_3 = 1$; $x_4 = 3$, we find: $x_1 = 3$ and $x_2 = 5$.

Therefore, if we follow the definition 1, the vector $X = (3, 5, 1, 3)^T$ is a possible solution. The vectors a_4, a_2, form a canonic base, therefore the variables x_4 and x_2 will be base variables and x_1 and x_3 will be secondary values. If $x_1 = x_3 = 0$, we obtain $x_2 = 4$ and $x_4 = 6$, meaning that the vector $X = (0, 4, 0, 6)^T$ is a possible base solution, according to definition nr. 5.

3. Models, finite automats, regular expression and grammars

By using the technology of regular expressions, we will give a formal solution to these economic modelling problems.

Theorem: if r is a regularly expression, than there is an AFN that accepts the language represented by this regular expression and vice versa.

On the other hand, if a finite determinist automat M is given, the determination of the regular expression, that is the language accepted by the automat, can be solved by using the following method:

There is $M = (\{q_1, q_2, ..., q_n\}, \Sigma, \delta, q_1, F)$

Noted as followed:

$$R_{ij}^k = \{\alpha \mid (q_i, \alpha) \overset{*}{\mid\!\rightarrow} (q_j, \varepsilon) \wedge [\forall\beta, \alpha = \beta\gamma,$$

$$0 < |\beta| < |\alpha|, (q_i, \beta) \overset{*}{\mid\!\rightarrow} (q_s, \varepsilon) \Rightarrow s \le k]\}$$

for which we have the property:

$$R_{ij}^k = R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1} \cup R_{ij}^{k-1}, k \in \{1,2, \dots ,n\}$$

and :

$$R_{ij}^0 = \{ a \mid q_j \in \delta(q_i, a) \} \cup A \quad \text{where:}$$

$$\text{if } i = j \text{ then } A = \{\varepsilon\}, \text{ else } A = \varnothing.$$

In all these relations: $i,j = 1, \dots ,n$.

R_{ij}^k is formed out of all the words over the alphabet Σ that lead the automat M from the state q_i to the state q_j without passing through any index-state greater than k. Particularly, R_{ij}^n is formed from all the words that lead the automat from the state q_i to the state q_j; R_{1j}^n while $q_j \in F$, is formed out of all the words accepted by the automat in the final state q_j.

There is: $T(M) = \bigcup_{q_j \in F} R_{ij}^n$.

To each regulated set R_{ij}^k corresponds a regular expression r_{ij}^k.

These regulated expressions can be calculated by using the recursive definition of the set R_{ij}^k by using the following formula:

$$r_{ij}^0 = a_1 + a_2 + \dots + a_s + x,$$

where:

$$q_j \in \delta(q_j, a_m), m \in \{1, 2, ... ,s\},$$

$a_1, a_2, ... ,a_s$ are all the symbols from Σ which lead the automat from the state q_i to the state q_j, and x takes the following values only in the next conditions:

$$\text{if } i = j \text{ then } x = \{\varepsilon\}, \text{ else } x = \varnothing.$$

For $i,j,k > 0$, we can write the following relations:

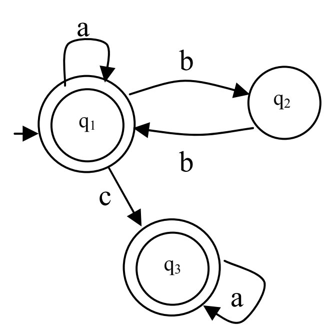
$$r_{ij}^k = r_{ik}^{k-1} (r_{kk}^{k-1})^* r_{kj}^{k-1} + r_{ij}^{k-1}, \forall i,j,k \in \{1, 2, ..., n\},$$

The language accepted by the automat M can be specified with the help of the *regulated expression:*

$$r_{1j_1}^n + r_{1j_2}^n + ... + r_{1j_p}^n ,$$

the obtained regulated expression R_{ij}^k does not depend of the reckoning of the automata's states, but a single request is needed: that it's initial state is q_1.

By implying in our example also the automat's theory, we put up a particular regulated expression of a given automat, shown in the next graphic:



Taking the above into consideration, the language accepted by the automat can be specified with the help of a regulated expression in which the *objective function f(X)* or the *efficiency function* can be of the following form:

$$r_{11}^{3} + r_{13}^{3}.$$

Solving-method:

By completing the calculus, using the given algorithm and by taking into consideration the characteristics of the regulated quantum, we reach the following results:

$r_{11}^{0} = a + \varepsilon$, $\quad r_{12}^{0} = b$, $\quad r_{13}^{0} = c$,

$r_{21}^{0} = b$, $\quad r_{22}^{0} = \varepsilon$, $\quad r_{23}^{0} = \varnothing$,

$r_{31}^{0} = \varnothing$, $\quad r_{32}^{0} = \varnothing$, $\quad r_{33}^{0} = a + \varepsilon$.

$r_{11}^{1} = r^0_{11}(r^0_{11})^* + r^0_{11} = (a + \varepsilon)(a + \varepsilon)^*(a + \varepsilon) + (a + \varepsilon) =$
$= (a + \varepsilon)((a + \varepsilon)^+ + \varepsilon) = (a + \varepsilon)(a + \varepsilon)^* = (a + \varepsilon)a^* =$
$= a^+ + a^* = \mathbf{a^*}$;

$r_{12}^{1} = r^0_{11}(r^0_{11})^* r^0_{12} + r^0_{12} = (a + \varepsilon)(a + \varepsilon)^* b + b =$
$= (a + \varepsilon)^+ b + b = ((a + \varepsilon)^+ + \varepsilon)b = (a + \varepsilon)^* b = \mathbf{a^* b}$;

$r_{13}^{1} = r^0_{11}(r^0_{11})^* r^0_{13} + r^0_{13} = (a + \varepsilon)(a + \varepsilon)^* c + c =$
$= (a + \varepsilon)^+ c + c = ((a + \varepsilon)^+ + \varepsilon)c = \mathbf{a^* c}$;

$r_{21}^{1} = r^0_{21}(r^0_{11})^* + r^0_{21} = b(a + \varepsilon)^*(a + \varepsilon) + b =$
$= b(a + \varepsilon)^+ + b = b((a + \varepsilon)^+ + \varepsilon) = b(a + \varepsilon)^* = \mathbf{ba^*}$;

$r_{22}^{1} = r^0_{21}(r^0_{11})^* r^0_{12} + r^0_{22} = b(a + \varepsilon)^* b + \varepsilon = \mathbf{ba^* b + \varepsilon}$;

$r_{23}^{1} = r^0_{21}(r^0_{11})^* r^0_{13} + r^0_{23} = b(a + \varepsilon)^* c + \varnothing = \mathbf{ba^* c}$;

$r_{31}^{1} = r^0_{31}(r^0_{11})^* r^0_{11} + r^0_{31} = \varnothing(a + \varepsilon)^* c + \varnothing = \varnothing + \varnothing = \varnothing$;

$r_{32}^{1} = r^0_{31}(r^0_{11})^* r^0_{12} + r^0_{32} = \varnothing(a + \varepsilon)^* b + \varnothing = \varnothing + \varnothing = \varnothing$;

$r_{33}^{1} = r^0_{31}(r^0_{11})^* r^0_{13} + r^0_{33} = \varnothing(a + \varepsilon)^* b + a + \varepsilon =$
$= \varnothing + a + \varepsilon = \mathbf{a + \varepsilon}$;

$r_{11}^{2} = r^1_{12}(r^1_{22})^* r^1_{21} + r^1_{11} = a^* b(ba^* b + \varepsilon)^* ba^* + a^* =$
$= a^* b(ba^* b)^* ba^* + a^* = a^* ba^* b\,(ba^* b)^* + a^* =$
$= a^*(bba^*)^+ + a^* = a^*((bba^*)^+ + \varepsilon) = a^*(bba^*)^* =$
$= \mathbf{(a + bb)^*}$;

$r_{12}^{2} = r^1_{12}(r^1_{22})^* r^1_{21} + r^1_{12} = a^* b(ba^* b + \varepsilon)^*(ba^* b + \varepsilon) + a^* b =$
$= a^* b((ba^* b + \varepsilon)^+ + \varepsilon) = a^* b(ba^* b + \varepsilon)^* = a^* b(ba^* b)^* =$
$= \mathbf{(a + bb)^* b}$;

$r_{13}^{2} = r^1_{12}(r^1_{22})^* r^1_{13} + r^1_{12} = a^* b(ba^* b + \varepsilon)^* ba^* c + a^* c =$
$= a^* c(a^* b(ba^* b + \varepsilon)^* b + \varepsilon) = a^* c(a^* b\,(ba^* b)^* b + \varepsilon) =$
$= a^* c\,((ba^* b)^+ + \varepsilon) = a^* c((ba^* b)^+ + \varepsilon) = a^* c\,(ba^* b)^* =$
$= \mathbf{(a + bb)^* c}$;

$r_{21}^{2} = r^1_{22}(r^1_{22})^* r^1_{21} + r^1_{21} =$
$= (ba^* b + \varepsilon)(ba^* b + \varepsilon)^* ba^* + ba^* =$
$= ba^*((ba^* b + \varepsilon)(ba^* b + \varepsilon)^* + \varepsilon) =$
$= ba^*((ba^* b + \varepsilon)^+ + \varepsilon) = ba^*(ba^* b + \varepsilon)^* =$
$= ba^*(ba^* b)^* = \mathbf{b(a + bb)^*}$;

$r_{22}^{2} = r^1_{22}(r^1_{22})^* r^1_{22} + r^1_{22} =$
$= (ba^* b + \varepsilon)(ba^* b + \varepsilon)^*(ba^* b + \varepsilon) + (ba^* b + \varepsilon) =$
$= (ba^* b + \varepsilon)((ba^* b + \varepsilon)^+ + \varepsilon) =$
$= (ba^* b + \varepsilon)(ba^* b + \varepsilon)^* = \mathbf{(ba^* b)^+ + \varepsilon}$;

$r_{23}^{2} = r^1_{22}(r^1_{22})^* r^1_{23} + r^1_{23} =$
$= (ba^* b + \varepsilon)(ba^* b + \varepsilon)^* ba^* c + ba^* c =$
$= ba^* c\,((ba^* b + \varepsilon)(ba^* b + \varepsilon)^* + \varepsilon) =$
$= ba^* c((ba^* b + \varepsilon)^+ + \varepsilon) = ba^* c\,(ba^* b + \varepsilon)^* =$
$= (ba^* b)^* ba^* c = \mathbf{b(a + bb)^* c}$;

$r_{31}^{2} = r^1_{32}(r^1_{22})^* r^1_{12} + r^1_{31} = \varnothing(ba^* b + \varepsilon)^* ba^* + \varnothing = \varnothing + \varnothing = \varnothing$;

$r_{33}^{2} = r^1_{32}(r^1_{22})^* r^1_{23} + r^1_{33} = \varnothing(ba^* b + \varepsilon)^* ba^* c + a + \varepsilon =$
$= \varnothing + a + \varepsilon = \mathbf{a + \varepsilon}$;

$r_{11}^{3} = r^2_{13}(r^2_{33})^* r^2_{31} + r^2_{11} =$
$= (a + bb)^* c(a + \varepsilon)^* \varnothing + (a + bb)^* = \mathbf{(a + bb)^*}$;

$r_{13}^{3} = r^2_{13}(r^2_{33})^* r^2_{33} + r^2_{13} =$
$= (a + bb)^* c(a + \varepsilon)^*(a + \varepsilon) + (a + bb)^* c =$
$= (a + bb)^* c((a + \varepsilon)^*(a + \varepsilon) + \varepsilon) =$
$= (a + bb)^* c((a + \varepsilon)^+ + \varepsilon) = (a + bb)^* c(a + \varepsilon)^* =$
$= (a + bb)^* ca^* = \mathbf{(a + bb)^* ca^*}$;

Haveing all date, we can edit the following table:

	k=0	k=1	k=2	k=3
r_{11}^{k}	$a + \varepsilon$	a^*	$(a + bb)^*$	$\mathbf{(a+bb)^*}$
r_{12}^{k}	b	$a^* b$	$(a + bb)^* b$	
r_{13}^{k}	c	$a^* c$	$(a + bb)^* c$	$\mathbf{(a+bb)^* ca^*}$
r_{21}^{k}	b	ba^*	$b(a + bb)^*$	
r_{22}^{k}	ε	$ba^* b+\varepsilon$	$(ba^* b)^+ + \varepsilon$	
r_{23}^{k}	$\varnothing$	$ba^* c$	$b(a+bb)^* c$	
r_{31}^{k}	$\varnothing$	$\varnothing$	$\varnothing$	
r_{32}^{k}	$\varnothing$	$\varnothing$	$\varnothing$	
r_{33}^{k}	$a + \varepsilon$	$a + \varepsilon$	$a + \varepsilon$	

The regulated expression corresponding to the given automat is:

$$r_{11}^{3} + r_{13}^{3} = (a + bb)^* + (a + bb)^* ca^* = (a + bb)^* (\varepsilon + ca^*);$$

Assuming that we now have a regulated expression of the following form:

$$\mathbf{(a|b)^*a(a|b)(a|b)}$$

we build a grammar that generates the language described by this expression.

Solving-method:

We make the following notations:

$$S = (a|b)^*a(a|b)(a|b) ,$$

and:

$$A' = a(a|b)(a|b).$$

We can write:

$$S = (a|b)^*A',$$

where S is the solution of the equation:

$$S = (a+b)S + A' ,$$

That can be equivalently written:

$$S = aS + bS + A' , \quad (1)$$

but:

$$A' = aA,$$

where:

$$A = (a|b)\ (a|b).$$

Therefore, the relation (1) becomes:

$$S = aS + bS + aA \quad (2)$$

If we note:

$B = (a|b)$, unde:

$$B = a + b, \quad (3)$$

than A becomes:

$$A = (a + b)B = aB + bB. \quad (4)$$

Corresponding to the relations (2) - (4), the set of the grammar productions G, that generates the described regulated expression, is:

$$S \rightarrow aS \mid bS \mid aA$$
$$A \rightarrow aB \mid bB$$
$$B \rightarrow a \mid b$$

The grammar that generates the language described by the given regulated expression is:

$$G = (\{S, A, B\}, \{a, b\}, P, S)$$

where P containes the productions:

$$\{S \rightarrow aS \mid bS \mid aA, A \rightarrow aB \mid bB, B \rightarrow a \mid b\}$$

4. Conclusion

The formal practice can be used within a certain economical problem when we mould an economic or production process, a spare part etc. ... so that *a* and *b* are actions with well set tenure and the pairs of actions are from the string: (a|b)*a(a|b)(a|b), built so that no other restrictions occur.

References

[1]. Algebraic linquistics. Analytical Models Marcus, S. – New York end London Academic Press, 1967;
[2]. Tratat de programarea calculatoarelor. Algoritmi fundamentali, Knuth, D. E. Bucuresti Ed. Tehnica 1974 (translation),
[3]. Modelling the dialogue by means of formal language theory, Gh. Paun, C.L.T.A. 25, 1, 1980,
[4]. Pattern Matching in Strings. D. E. Knuth, J. H. Morris and V. R. Pratt. Jun 1997
[5]. Algorithms on strings, trees, and sequences, D. Gusfield.. Cambridge University Press, 1997
[6]. Introduzione agli algoritmi, T.H. Cormen, C.E. Leirson, L.R. Rivest, Jakson Libri, seconda edizione, 2000
[7]. Introducere in algoritmi, R. Rivest, C.E. Leiserson, T.H. Cormen, 2000,
[8]. Use the Gordon-Melham axioms which model bindinds and substitution, Michael Norrish, 2003.