THEORIE TABLA REVISEE

Andriamanohisoa Hery Zo [1]**, Randimbindrainibe F.** [2]**, Randriamitantsoa A.A.** [3]
Ecole Doctorale Science et Technique de l'Ingénierie et de l'Innovation (ED –STII)
Laboratoire Sciences Cognitives et Applications (LR-SCA)
Ecole Supérieure Polytechnique – Université d'Antananarivo
[1]***aheryzo@gmail.com***, [2] ***falimanana@mail.ru***, [3] ***andriau23@gmail.com***

Résumé : Dans cet article, nous corrigeons la confusion causée par l'adoption du terme "tabulaire". Nous révisons également de fond en comble notre modèle mathématique de programmation linéaire, sur lequel repose toute notre théorie.

Mots-clés : Tabulaire, Tabla, tabla-num, tabla-var, tabla-const, représentation ensembliste, conforme-linéaire, conforme-affine, Nouveau Modèle de Programmation Tabla-linéaire.

Abstract: In the paper, we correct the confusion caused by the adoption of the term "tabular". We also completely revamp our mathematical model of linear programming, which is the basis of our whole theory.

Keywords: Tabular, Tabla, tabla-num, tabla-var, tabla-const, set representation, conform-linear, conform-affine, New Tabla-linear Programming Model.

1. Introduction et Motivation

Cet article rentre dans le cadre de nos recherches dans le domaine de la modélisation d'optimisation linéaire sur tableurs. Il est donc une suite logique de l'article [1] et de la thèse [2].

1.1 Notations

Les notations et symboles utilisés dans cet article sont exactement les mêmes que dans [1] et [2].

1.2 Contexte de l'article

Dans l'article [1], nous avons posé à l'aide du Système Algébrique Tabla (TAS) les premiers fondements d'une algèbre qui est entièrement inspirée des langages algébriques de modélisation (AML), tout en étant orientée vers une intégration logique et naturelle avec la structure des tableurs. Puis, dans la thèse [2], nous avons développé en détails cette structure et conçu une nouvelle algèbre que nous avons baptisé Algèbre Tabulaire. Dans le cadre de cette nouvelle algèbre, nous avons conçu et défini un nouveau modèle mathématique de programmation linéaire appelé Modèle Théorique de Programmation

Tabulo-linéaire. Ce dernier repose entièrement sur les concepts de constantes tabulaires, variables tabulaires et uitlise exclusivement la famille de fonctions tabulaires SAM.

Cependant, conformément à la philosophie de George B. Dantzig qui affirme dans [3] que le test final d'une théorie est sa capacité à résoudre les problèmes qui on été à son origine, nous avons utilisé notre théorie pour modéliser de très nombreux problèmes d'optimisation linéaire réputés dans le monde scientifique et professionnel comme très difficiles à résoudre sur les tableurs, tels que les grands problèmes de transport avec transbordement, de réaffectation de ressources, de réseaux de télécommunication etc..., et même les problèmes purement mathématiques comme le fameux problème des huit reines ainsi que sa généralisation à n reines. Suite à ces nombreuses expérimentations, nous avons découvert que notre modèle théorique est trop rigide, en contradiction avec le concept de souplesse qui est l'une des caractéristiques principales faisant la puissance des AMLs. Nous pensons donc qu'il est grand temps de corriger ces défauts qui, à notre avis, sont fondationnels.

Pour cela, notre article est divisé en trois parties. Dans la Section 2, nous apporterons quelques modifications dans la terminologie, suite à des confusions causées par l'adjectif *"tabulaire"*, ainsi que sur les concepts de constantes et variables. Dans la Section 3, nous allons corriger la rigidité de notre modèle théorique et dans la Section 4, nous conclurons en précisant les futures orientations de nos recherches dans ce domaine.

2. Modification de terminologie. Nature ensembliste d'un d'objet tabla et fonction de tablas.

Précisons dès le départ que dans cet article, le terme "problème" se réfère toujours à un problème d'optimisation linéaire.

2.1 Tabulaire et tabla

Dans [2], nous utilisons le terme "tabulaire" (ou "tabulo-" quand il est composé avec un autre adjectif) comme adjectif se rapportant au substantif tabla. Cependant, nous avons remarqué que dans la littérature, cet adjectif est abondamment utilisé et que sa signification est très vague dans plusieurs situations. Par conséquent, après avoir testé différents termes, là où il y a un risque de confusion, nous avons décidé de le remplacer par le terme tabla. Ainsi, nous parleons d'Algèbre Tabla et de Modèle Théorique de Programmation Tabla-Linéaire.

Pour désigner un objet tabla, nous utiliseron le même terme "tabla", suivi d'un suffixe pour indiquer son type. Ainsi, tabla-num désigne un objet tabla de type ℝ. Pour éviter les confusions de notation dans les cas des opérateurs arithmétiques et opérateurs de comparaison, nous avons décidé de noter un objet tabla-opérateur arihmétique par tabla-oa, de type noté OA, et un objet tabla-opérateur de comparaison par tabla-oc, de type noté OC.

2.2 Nature ensembliste d'un objet tabla

Dans ce paragraphe, nous allons considérer un objet tabla en général, et plus particulièrement un tabla-num, comme un ensemble particulier, dont les éléments sont tous de même type et peuvent être accédés par un couple unique de nombre entiers finis (i, j), où i désigne le numéro de ligne et j le numéro de colonne. Ainsi, le tabla-num A de dimensions $m \times n$ peut être représenté par l'ensemble suivant :

$$A = \{A_{i,j} \in \mathbb{R}, (i, j) \in [1, m] \times [1, n]\}$$

Remarquons qu'en posant

$k = n.(i - 1) + j$ et $a_k = A_{i,j}$,

on peut donner la définition suivante :

Définition 2.1

On appelle représentation ensembliste du tabla-num A de dimensions $m \times n$, l'ensemble noté $S(A)$, égal à

$$\{a_k,\ k = 1, \cdots, m \times n\} \qquad (1)$$

Plus généralement, soient $A^{(1)}, \cdots, A^{(l)}$ un système de l tabla-nums, de représentations ensemblistes respectives $S(A^{(1)})$, ..., $S(A^{(l)})$, on définit la représentation ensembliste d'un système de tabla-nums :

Définition 2.2

On appelle représentation ensembliste du système de tabla-nums $A^{(1)}, \cdots, A^{(l)}$ l'ensemble noté $S(A^{(1)} \cup A^{(2)} \cup \cdots \cup A^{(l)})$ égal à $S(A^{(1)}) \cup S(A^{(2)}) \cup \cdots \cup S(A^{(l)})$ au sens de l'algèbre des ensembles.

2.3 Fonction tabla.

Conceptuellement, une fonction est un mécanisme qui permet de calculer ou de créer un objet appelé résultat à l'aide d'autres objets de même nature appelés arguments. Et c'est l'un concept fondamental dans toutes les théories mathématiques connues. Cependant, ce concept est très peu développé dans [2]. Nous allons corriger cette lacune dans cet article. Nous traiterons uniquement le cas des fonctions de tabla-nums.

Définition 2.3

On appelle fonction tabla toute fonction f permettant de créer ou de calculer un tabla-num B à partir des tabla-nums $A^{(1)}, \cdots, A^{(l)}$, et telle que :

1. Les dimensions $p \times q$ de B sont déterminées en fonction des dimensions respectives

   $$p_1 \times q_1, \cdots, p_l \times q_l$$

   de $A^{(1)}, \cdots, A^{(l)}$, et

2. chaque élément $B_{i,j}$ de B est fonction – au sens classique du terme – des élément de

$$E = S(A^{(1)} \cup A^{(2)} \cup \cdots \cup A^{(l)})$$

c'est-à-dire

Formellement, on écrit :

$$B = f(A^{(1)}, \cdots, A^{(l)}) \qquad (2)$$

La définition d'une fonction tabla f comprend deux aspects, un aspect dimensionnel et un aspect fonctionnel scalaire, ce qui rend son étude théorique abstraite très complexe.

Posons

$$B = \{B_{i,j} \in \mathbb{R}, (i,j) \in [1,p] \times [1,q]\},$$

Du point de vue formel, il y a deux possibilités de spécifier la façon de calculer les $B_{i,j}$:

- soit par une seule fonction numérique φ qui admet comme arguments tous les éléments de l'ensemble E plus i et j, ce qui peut s'écrire :

$$B_{i,j} = \varphi(a \mid a \in E, i, j) \qquad (3)$$

- soit par $p \times q$ fonctions numériques notées $\varphi_{i,j}$ qui admet comme seuls arguments les éléments de l'ensemble E, à savoir :

$$B_{i,j} = \varphi_{i,j}(a \mid a \in E) \qquad (4)$$

C'est cette dernière que nous adoptons pour la suite de notre exposé.

2.4 Constante et variable tablas

Précisons que dans cet article le terme "problème" se réfère toujours à un problème d'optimisation linéaire donné.

Définition 2.4

On appelle *"tabla-const"* tout tabla-num, noté A, B, etc..., de dimensions $p \times q$, dont tous les éléments sont des constantes au sens mathématique du terme dans le contexte du problème à résoudre.

Définition 2.5

On appelle *"tabla-var"* tout tabla-num, noté X, Y, etc..., dont tous les éléments sont des variables au sens mathématique du terme dans le contexte du problème à résoudre.

Définition 2.6

On appelle *"tabla-semi-var"* tout tabla-num, noté X, Y, etc..., dont certains éléments sont des variables au sens mathématique du terme dans le contexte du problème à résoudre.

De plus, nous savons que mathématiquement parlant, quelle que soit l'algèbre utilisée (y compris tout le symbolisme qui lui est associé), un problème donné comporte toujours une ou plusieurs inconnues. De plus, toute la littérature abondante à laquelle nous avons pu avoir accès et qui traite de la méthodologie de modélisation s'accorde sur un point très important, à savoir l'identification exhaustive de toutes les inconnues quantitatives pertinentes correspondant aux décisions à prendre, et qui sont appelées "variables de décision" du problème. Ces variables de décision, appelées également variables primitives, se différencient des autres variables du problème, également appelées variables dérivées, par le fait que ces dernières peuvent être formellement exprimées en fonction d'autres variables et éventuellement de constantes.

En se basant sur le même principe, on peut définir dans la Théorie Tabla le concept de tabla-var de décision ou primitive, et celui de tabla-var dérivée.

Définition 2.7

Une tabla-var ou tabla-semi-var Y est dite dérivée dans un problème si elle dépend (ou est définie en fonction) d'autres tabla-vars et/ou d'autres tabla-semi-vars.

Cette catégorie de tabla-vars occupe une place très importante dans la Théorie Tabla, car elles interviennent dans la majorité des contraintes d'optimisation d'un problème.

Définition 2.8

Une tabla-var ou tabla-semi-var X est dite primitive dans un problème si elle ne dépend d'aucun autre tabla-num, variable ou constante.

Notons par $PTV = \{X^{(j)}, j \in [1, l]\}$ l'ensemble de de toutes les tabla-vars de décision pour un problème donné. Comme PTV est un système de tabla-nums, il possède une représentation ensembliste $S(PTV)$, que nous noterons par PV. Concrètement, cela signifie que l'ensemble PV contient toutes les *"inconnues algébriques scalaires"* du problème. De plus, en notant par N_{PV} le cardinal de PV, on peut écrire :

$$PV = \{x^{(k)}, k = 1, \cdots, N_{PV}\} \qquad (5)$$

Remarquons que dans cette notation, nous utilisons la méthode d'indexation $X^{(k)}$ pour les tabla-nums et $x^{(k)}$ (où l'indice k est écrit entre parenthèses en haut), et ce parce qu'on aura à utiliser trois indices à la fois.

Définition 2.9

Dans un problème, une tabla-var dérivée Y de dimensions $p \times q$ est dite linéaire si, quel que soit le couple (i, j), $Y_{i,j}$ est de la forme

$$Y_{i,j} = \sum_{k=1}^{N_{PV}} a_{i,j}^{(k)} \cdot x^{(k)} \qquad (6)$$

où les $a_{i,j}^{(k)}$ sont des coefficients constants (déterminés par les trois indices $i, j\ et\ k$) dans $\mathbb{R}$, et les $x^{(k)}$ les éléments de PV.

Cela signifie tout simplement que chaque élément d'une tabla-var dérivée linéaire est une forme linéaire par rapport aux variables primitives (de décision) du problème.

Définition 2.10

Dans un problème, une tabla-var dérivée Y de dimensions $p \times q$ est dite affine si, quel que soit le couple (i, j), $Y_{i,j}$ est de la forme

$$Y_{i,j} = (\sum_{k=1}^{N_{PV}} a_{i,j}^{(k)} \cdot x^{(k)}) + b_{i,j} \qquad (7)$$

où les $a_{i,j}^{(k)}$ sont des coefficients constants (déterminés par les trois indices $i, j\ et\ k$) dans $\mathbb{R}$, et les $x^{(k)}$ les éléments de PV.

Cela signifie tout simplement que chaque élément d'une tabla-var dérivée linéaire est une forme linéaire par rapport aux variables primitives (de décision) du problème.

2.5 Fonction conforme-linéaire et fonction conforme-affine

2.5.1 Distinction entre formule et équation

Remarquons tout d'abord que malgé sa grande précision en tant que langage, l'algèbre présente quelquefois des confusions. Parmi celles-ci, la confusion entre une formule, une affectation et une équation est la plus fréquente, et la plupart du temps les mathématiciens s'en soucient très peu car ils pensent toujours faire la différence entre les deux à travers le contexte. Dans l'algèbre Tabla, les notations utilisées sont différentes, comme le montre l'exemple suivant :

Soit X, A, B et C des tabla-nums appartenant à STx$\mathbb{R}$[m×n], où X est une tabla-var tandis que A, B et C sont des tabla-const. Supposons que Y est une autre tabla-var appartenant à STx$\mathbb{R}$[m×n].

- La phrase "$Y = A(\times)X\ (+)\ B$" est syntaxiquement valable car elle exprime le fait que la tabla-var Y varie en fonction de X selon cette formule.
- L'expression "$Y = A(\times)C\ (+)\ B$" est correcte : elle représente une affectation.
- L'expression "$A(\times)X\ (+)\ B\ = Y$" est syntaxiquement invalide. Il est évident qu'on a en tête une équation. L'expression correcte est donc la suivante : "$A(\times)X\ (+)\ B\ (=)\ Y$".

Dans la pratique, une formule n'est autre que l'expression de la définition d'une tabla-var qui, de ce fait, est une tabla-var dérivée.

2.5.2 Fonction conforme-linéaire et fonction conforme-affine

Soit f une fonction ayant pour arguments l tabla-vars $X^{(1)}, X^{(2)}, \cdots, X^{(l)}$ de dimensions respectives $p_1 \times q_1$, $p_2 \times q_2, \cdots, p_l \times q_l$ et de représentation ensembliste

$V = S(X^{(1)} \cup X^{(2)} \cup \cdots \cup X^{(l)})$ dans un problème donné.

Pour faciliter le développement du raisonnement, posons

$$N = \sum_{t=1}^{l} p_t . q_t \qquad (8)$$

On peut donc exprimer V comme suit :

$$V = \left\{x^{(i)}, i \in [1, N]\right\} \qquad (9)$$

Soit Y la tabla-var définie par :

$$Y = f(X^{(1)}, X^{(2)}, \cdots, X^{(l)}),$$

dont les dimensions $p \times q$ sont déterminées par les dimensions respectives

$$p_1 \times q_1, p_2 \times q_2, \cdots, p_l \times q_l$$

de $X^{(1)}, X^{(2)}, \cdots, X^{(l)}$.

D'après la relation (4), on peut caractériser les éléments de Y analytiquement par $p \times q$ fonctions numériques notées $\varphi_{i,j}$ qui admettent chacune comme seuls arguments les éléments de l'ensemble V, c'est-à-dire

$$Y_{i,j} = \varphi_{i,j}\left(x^{(k)} \,\middle|\, x^{(k)} \in V\right) \qquad (10)$$

Définition 2.11

On dit que la fonction f est conforme-linéaire si quel que soit (i, j), $Y_{i,j}$ est de la forme

$$Y_{i,j} = \sum_{k=1}^{N} a_{i,j}^{(k)} \cdot x^{(k)} \qquad (11)$$

où les $a_{i,j}^{(k)}$ sont des coefficients constants (déterminés par les trois indices $i, j\ et\ k$) dans $\mathbb{R}$, et les $x^{(k)}$ les éléments de V.

Définition 2.12

On dit que la fonction g est conforme-affine si, quel que soit (i, j), $Y_{i,j}$ est de la forme

$$Y_{i,j} = \left(\sum_{k=1}^{N} a_{i,j}^{(k)} \cdot x^{(k)}\right) + b_{i,j} \qquad (12)$$

où les $a_{i,j}^{(k)}$ (déterminés par les trois indices $i, j\ et\ k$) et les $b_{i,j}$ (déterminés par les deux indices $i\ et\ j$) sont des coefficients constants dans $\mathbb{R}$, et les $x^{(k)}$ les éléments de V.

2.5.3 Relation entre fonction conforme-linéaire et fonction conforme-affine

Théorème 2.1

Dans un problème, pour toute fonction conforme-affine g, il existe une fonction f conforme-linéaire et une tabla-const B telle que

$$g\left(X^{(1)}, \cdots, X^{(l)}\right) = f\left(X^{(1)}, \cdots, X^{(l)}\right) + B \qquad (13)$$

Cette propriété résulte de (8) et (9).

3. Révision du Modèle Théorique de Programmation Tabla-Linéaire

3.1 Ancienne définition [2]

On appelle *"Modèle Théorique de Programmation Tabulo-Linéaire"*, noté TLP, tout système $(\mathrm{X}, \mathrm{C}, \mathrm{A}, \mathrm{A}', \mathrm{B}, \mathrm{B}', \mathrm{D}, \mathrm{D}')$ d'éléments de STx$\mathbb{R}$[m x n] satisfaisant les conditions suivantes :

1. *X est une variable tabulaire, tandis que les autres sont des paramètres tabulaires*
2. *X et C sont isodimensionnels, de dimensions* $\mathrm{p} \times \mathrm{q}$,
3. *SAMH(A, X) et B et B' sont isodimensionnels, de dimensions* $\mathrm{p}' \times 1$,
4. *SAMV(A', X) et D et D' sont isodimensionnels, de dimensions* $1 \times \mathrm{q}'$,
5. *X est solution du problème suivant :*

$$\begin{cases} \text{(Maximiser ou Minimiser) } \mathrm{Z} = \mathrm{SAMT}(\mathrm{C}, \mathrm{X}) \\ \text{sachant que} \\ \qquad \mathrm{B}\ (\le)\ \mathrm{SAMH}(\mathrm{A}, \mathrm{X}) \text{ et } \mathrm{SAMH}(\mathrm{A}, \mathrm{X})\ (\le)\ \mathrm{B}' \\ \qquad \mathrm{D}\ (\le)\ \mathrm{SAMV}(\mathrm{A}', \mathrm{X}) \text{ et } \mathrm{SAMV}(\mathrm{A}', \mathrm{X})\ (\le)\ \mathrm{D}' \\ \qquad \mathrm{X}\ (\ge)\ 0 \end{cases}$$

Remarquons que conformément à la Section 2, X est une tabla-var, C une tabla-const, et Z une tabla-var

3.2 Critiques sur ce modèle

Nous ne reprenons plus ici ce que nous avons déjà amplement développé dans le paragraphe

2.1 concernant le caratère confus du terme "tabulaire".

3.2.1 Structuration des variables

Dans toute activié de modélisation d'un problème d'optimisation, tout repose sur l'identification exhaustive des inconnues ou variables algébriques classiques de décision, qui sont bien entendu des éléments de l'ensemble des nombres réels. Le regroupement de ces variables à l'aide d'ensembles d'indexation (index set) est l'une des grandes contributions des AMLs dans la science de la modélisation [4], [5], [6]. La Théorie Tabla reprend ce concept, ce qui permet d'identifier de façon presque automatique les tablas dont ces variables algébriques classiques deviendront des éléments.

Dans ce modèle, les tabla-vars prévues sont :

- Z exprimant la fonction objectif du problème,
- X désignant la tabla-var de décision du problème,
- Les deux tabla-vars dérivées Y_1 et Y_2 définies par :

$$\begin{cases} Y_1 = SAMH(A, X) \\ \quad\quad et \\ Y_2 = SAMV(A', X) \end{cases}$$

  Exprimant les contraintes du problème.

Le modèle impose donc que toutes les variables du problème soient structurées (ou empaquetées) dans seulement quatre tabla-vars, indépendamment de la complexité du problème.

Nos nombreux travaux de recherches nous ont montré qu'il est très rare que toutes les variables classiques de décision soient regroupées dans une seule tabla-var. Les tabla-vars que nous avons identifiées sont de natures et strucures (dimensions) différentes. Bien entendu, ces tabla-vars ne se comptent pas par centaines, encore moins par milliers, mais dans certaines situations, on en a identifié une bonne dizaine. Enfin il faut noter que la structure bi-dimensionnelle même de l'Algèbre Tabla impose automatiquement l'utilisation de plusieurs tabla-vars et tabla-consts quand le problème à résoudre présente une structure multi-dimensionnelle évidente.

3.2.2 Formulation des contraintes

Toutes les contraintes sont exprimées avec uniquement l'utilisation des fonctions SAM. Bien entendu, nous avons réussi à modéliser valablement un grand nombre de problèmes rien qu'en utilisant les fonctions SAM. Cependant, nous avons rencontré beaucoup de situations où elles étaient tout simplement incapables de traduire la logique du problème. Elles sont très puissantes, mais elles ne sont pas une sorte de couteau (de l'armée) suisse.

De plus, les contraintes d'égalité ne sont pas prévues dans le modèle.

Les deux critiques précédentes peuvent être résumées en trois mots : rigidité structurelle, rigidité fonctionnelle et enfin contraintes incomplètes.

3.3 Formulation revisée

Cette nouvelle formulation tient compte des deux critiques précédentes. Mais il nous faut d'abord donner les définitions suivantes.

3.3.2 Nouveau Modèle Théorique de Programmation Tabla-linéaire

On appelle Nouveau Modèle Théorique de Programmation Tabla-Linéaire tout problème dont les inconnues sont les tabla-vars primitives $X^{(1)}, X^{(2)}, \cdots, X^{(l)}$, ayant pour objectif de minimiser ou maximiser la tabla-var dérivée

$$Z = f\left(X^{(1)}, X^{(2)}, \cdots, X^{(l)}\right)$$

de dimensions 1×1,

et satisfaisant le système des contraintes

$$\begin{cases} Y_1 = g_1\left(X^{(1)}, X^{(2)}, \cdots, X^{(l)}\right) (\geq) \ ou\ (=)\ 0 \\ \vdots \\ Y_l = g_l\left(X^{(1)}, X^{(2)}, \cdots, X^{(l)}\right) (\geq) \ ou\ (=)\ 0 \end{cases}$$

et les contraintes de non négativié

$X^{(i)} (\geq) 0, \forall i \in [1, l]$.

où :

- Z est linéaire (cf Définition 2.8), et
- chaque tabla-var dérivée
  $Y_j = g_j\left(X^{(1)}, X^{(2)}, \cdots, X^{(l)}\right)$
  est affine (cf Définition 2.9)

3.3.2 Propriétés du nouveau modèle théorique de programmation tabla-linéaire

- Ni le nombre des inconnues, ni leurs dimensions respectives ne sont imposées par le modèle : le modèle est structurellement très souple.
- Il ne contient aucune fonction explicite imposée à priori : le modèle est fonctionnellement souple.
- Tous les types de contraintes, équations et inéquations, sont possibles dans le modèle : le modèle est complet.

Démontrons cette dernière propriété.

Les notations utilisées dans cette démonstration sont celles du paragraphe 2.5.2.

Considérons la relation

$$Y_j = g_j\left(X^{(1)}, X^{(2)}, \cdots, X^{(l)}\right) (=)\ 0$$

Comme Y_j est conforme-affine, cette contrainte signifie que chaque élément de Y_j est de la forme

$$\sum_{k=1}^{N} a^{(k)} \cdot x^{(k)} + b = 0$$

d'où la relation

$$\sum_{k=1}^{N} a^{(k)} \cdot x^{(k)} = -b$$

De même, la relation Tabla

$$Y_j = g_j\left(X^{(1)}, X^{(2)}, \cdots, X^{(l)}\right) (\geq)\ 0$$

devient

$$\sum_{k=1}^{N} a^{(k)} \cdot x^{(k)} \geq -b$$

4. Conclusion

Dans cet article, nous présentons et analysons tous les problèmes et difficultés que nous avons rencontré dans nos recherches pour vérifier et valider la Théorie Tabulaire qui a fait l'objet de notre thèse[2]. Les deux principaux problèmes que nous avons analysés sont le caractère confus du terme *"tabulaire"* et le caractère rigide et incomplet du modèle théorique appelé *"Modèle Théorique de Programmation Tabulo-Linéaire"* et noté TLP. Suite à cette analyse détaillée, nous proposons deux changements radicaux. D'abord, sur le plan de la terminologie, nous préconisons l'utilisation du terme Tabla au majuscule pour tout adjectif se référant à la Théorie Tabla, et le terme tabla au minuscule pour désigner tout objet tabla. Ensuite, nous proposons une révision totale du modèle que baptisons *"Nouveau Modèle Théorique de Programmation Tabla-linéaire"* qui est un modèle général à la fois souple et complet. Au vu des résultats prometteurs que nous avons obtenus avec ce modèle, nous pensons que nous avons maintenant une "théorie" dans le vrai sens du terme. Cependant, notre n'avons pas encore atteint notre but initial, à savoir la création d'un système modélisation de problèmes d'optimisation linéaire pour les tableurs. Un tel système exige plus qu'une algèbre et un modèle théorique de modélisation : il faut un concevoir le langage de programmation et créer le compilateur. Et nous y consacrons une grande partie de nos efforts.

BIBLIOGRAPHIE

[1] H.Z. Andriamanohisoa, F. Randimbindrainibe, P. A. Randriamitantsoa, *« Système Tabla Algébrique »*, Mada-ETI, Volume 3, 2012, pp1-18

[2] H.Z. Andriamanohisoa, « *Système de Modélisation sur Tableur*», Th. Doctorat, 2012.

[3] George B. Dantzig, « *Linear Programming and Extensions* », Rand Corporation Project – Princeton University Press, 1963.

[4] C.AC Kuipp, *«Algebraic Languages for Mathematical Programming»*. European Journal of Operations Research 67(1993)25-51, North Holland.

[5] K. Davies, O. Pishko, N.F. Shawawa, D. Pearl *«Modeling Languages»*. CAS 737 / CES 735, 2007.

[6] A. M. Geoffrion *«Indexing in Modeling Languages for Mathematical Programming»*. Management Science, Vol38, No 3(Mars1992)325-344.