# Mise en place de la commutativité

## Présentation

### Document concerné

Ce document traite des opérations algébriques de commutativité.

Je rédige ce court document pour préciser le travail de programmation nécessaire.

### Versions

13/12/2016 : version initiale

### Auteur

Invisible Media

## Projet traité

### Cadre du projet

Le présent projet est un logiciel de calcul numérique et algébrique.  
Ce projet est un moyen de former des équations très longues et fastidieuses et, de les factoriser selon une approche commune.

But : résoudre des équations, trouver l’ensemble des solutions algébriquement et obtenir une équation en fonction de paramètres et de données numériques fixées.

### Ensemble des fonctionnalités

L’ensemble des fonctionnalités est détaillé dans ce document :

[Toutes les fonctionnalités (dégroupé).xlsx](Toutes%20les%20fonctionnalités%20(dégroupé).xlsx)

### Nom de la fonctionnalité traitée ici

Neurones de commutativité

## Besoin relatif

Les neurones sont des éléments de graphe qu’on nomme également nœud. En chaque nœud du graphe, il peut y avoir de 0 à N branches ou arc où l’autre extrémité est le nœud suivant.

Les opérations arithmétiques sont un ensemble récurrent qui s’adapte parfaitement à la notion de graphe. En particulier, une équation complète est un ensemble arborescent d’opérations. On parle plutôt de graphe lorsqu’un nœud a plusieurs parents.

Une équation où certains termes sont répétés formera un graphe où un terme répété est associé à un seul nœud et, chaque nœud parent ira sur ce nœud.

Les neurones de distribution algébrique concernent toutes les multiplications de termes.

## Relation avec les autres fonctionnalités

### Utilisation

Les neurones sont des objets qui peuvent communiquer entre eux. Du point de vue programmation, les neurones forment un graphe orienté. Puis, chaque neurone ayant une application qui lui est propre, l’application parcourt le graphe en passant par les neurones qui réagissent différemment selon les paramètres.

On note chaque nœud sur un quadrillage à deux dimensions où est le nombre de nœuds. Chaque case du quadrillage héberge un nombre positif ou nul. Ce nombre correspond au nombre de liens qui relient deux nœuds différents (dans les deux sens). L’axe diagonal du quadrillage correspond à une diagonale de nombres nuls, étant entendu qu’il n’existe pas de nœud en lien avec lui-même.

### Explications

La commutativité est une propriété des opérations arithmétiques. L’addition et la multiplication possède cette propriété car les deux opérandes peuvent être échangée sans que le résultat de l’opération en soit changée.

Au contraire, la soustraction et la division ne sont pas commutatives ; si l’on échange les deux opérandes l’une avec l’autre alors le résultat de l’opération change.

Mais, cette définition est insuffisante pour expliquer le vrai problème. La commutativité pose la difficulté d’obtenir une forme canonique d’une équation quelconque. Le fait de la commutativité signifie que l’organisation d’une équation n’a pas d’impact sur le résultat ; l’organisation d’une équation a un impact sur sa représentation schématique et sémantique.

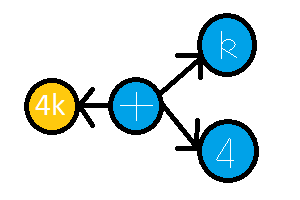
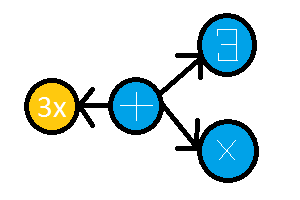
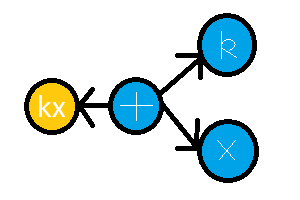
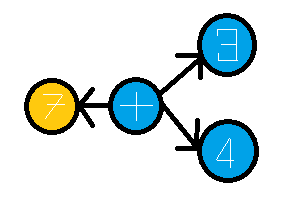
C’est pourquoi, la commutativité doit être appliquée pour former l’équation selon l’organisation la plus claire possible. La forme canonique souhaitée a été présentée dans le document [SP-Heuristique](SP-Heuristique.docx).

Pour obtenir à chaque équation donnée, la forme canonique de l’équation, il faut utiliser la commutativité des opérations pour passer d’une équation non canonique à la même équation organisée selon la forme canonique attendue.

Je fais remarquer que la transitivité n’est pas abordée. En effet, le graphe logique composé après la lecture d’une équation est un graphe orienté de largeur 0 à l’infini ; cela signifie qu’un nœud du graphe aura autant de liens qu’il y a d’opérandes avec la même opération.

### Point de vue graphique

Voici le point de vue graphique d’un neurone où comment on représente graphiquement un neurone.



Les deux branches à droite sont les opérandes de l’opération (ici l’addition). Pour chaque opération, il existe 6 configurations.

### Fabrication des neurones

Je fais remarquer qu’ici ce sont des neurones préfabriqués. Pour une équation donnée, il faut décomposer l’équation en termes d’opérations et construire le graphe de cette équation en utilisant les gabarits de neurones mis à disposition.

Il y a donc un bagage initial où chaque neurone a sa propre fonction et sa propre forme. Je fais également remarquer qu’il n’y a qu’une seule commande principale pour une équation : calculer le résultat de l’équation en donnant à chaque variable une valeur numérique. Pour des variables liées, il faudra toujours exprimer les valeurs à l’aide de l’équation de cette variable. Pour des variables libres, toute valeur numérique est correcte. Enfin, pour les variables inconnues il s’agit de chercher son équation en fonction des connaissances des équations sur les variables liées et la valeur des variables libres.

D’autres opérateurs sont nécessaires pour permettre un ensemble de séquences mathématiques. Quelques instructions sont nécessaires :

1. Déclaration d’une variable libre
2. Déclaration d’une variable liée
3. Déclaration d’une variable inconnue
4. Opérateur d’égalité à vérifier
5. Opérateur d’égalité propre
6. Opérateurs de comparaison ()
7. Opérateur conditionnel :   
   un test de comparaison, une nouvelle séquence si le résultat du test est vérifié, une nouvelle séquence (optionnelle) si le résultat du test est réfuté.

Ces instructions sont définies par des gabarits de neurones. La construction des neurones s’effectue pendant une analyse syntaxique de ces instructions.

## Programmation

L’algorithme de recherche de la forme canonique considère les états dans l’ordre décroissant et commence la recherche en profondeur par les états les plus grands, c’est-à-dire les formes d’équations non souhaitées. Pour évaluer le premier état comme plus grand, il suffit de calculer le plus petit état possible par le calcul du nombre d’opérandes. La commutativité dénombre permutations pour opérandes sur une même opération ; autant dire que si l’algorithme n’est pas optimisé alors le temps de traitement sera trop grand et la disponibilité en mémoire sera insuffisante.